Completeness Theorems for Adaptively Secure Broadcast

Ran Cohen[1(✉)], Juan Garay[2], and Vassilis Zikas[3]

[1] Efi Arazi School of Computer Science, Reichman University, Herzliya, Israel
cohenran@runi.ac.il
[2] Texas A&M University, College Station, USA
garay@cse.tamu.edu
[3] Purdue University, West Lafayette, USA
vzikas@cs.purdue.edu

Abstract. The advent of blockchain protocols has reignited the interest in adaptively secure broadcast; it is by now well understood that broadcasting over a diffusion network allows an adaptive adversary to corrupt the sender depending on the message it attempts to send and change it. Hirt and Zikas [Eurocrypt '10] proved that this is an inherent limitation of broadcast in the simulation-based setting—i.e., this task is impossible against an adaptive adversary corrupting a majority of the parties (a task that is achievable against a static adversary).

The contributions of this paper are two-fold. First, we show that, contrary to previous perception, the above limitation of adaptively secure broadcast is **not** an artifact of simulation-based security, but rather an inherent issue of adaptive security. In particular, we show that: (1) it also applies to the property-based broadcast definition adapted for adaptive adversaries, and (2) unlike other impossibilities in adaptive security, this impossibility cannot be circumvented by adding a programmable random oracle, in neither setting, property-based or simulation-based.

Second, we turn to the resource-restricted cryptography (RRC) paradigm [Garay *et al.*, Eurocrypt '20], which has proven useful in circumventing impossibility results, and ask whether it also affects the above negative result. We answer this question in the affirmative, by showing that time-lock puzzles (TLPs)—which can be viewed as an instance of RRC—indeed allow for achieving the property-based definition and circumvent the impossibility of adaptively secure broadcast. The natural question is then, do TLPs also allow for simulation-based adaptively secure broadcast against corrupted majorities? We answer this question in the negative. However, we show that a positive result can be achieved via a *non-committing* analogue of TLPs in the programmable random-oracle model.

Importantly, and as a contribution of independent interest, we also present the first (limited) composition theorem in the resource-restricted setting, which is needed for the complexity-based, non-idealized treatment of TLPs in the context of other protocols.

H. Handschuh and A. Lysyanskaya (Eds.): CRYPTO 2023, LNCS 14081, pp. 3–38, 2023.
https://doi.org/10.1007/978-3-031-38557-5_1

1 Introduction

A physical broadcast channel enables a set of n parties to communicate as if talking via a megaphone: Once a party speaks, all other parties are guaranteed to hear its message. In a *broadcast protocol* (aka Byzantine Generals [66,76]) the parties are asked to realize this "megaphone" capability over point-to-point channels, even when a subset of them collude and actively disrupt the protocol's execution. The standard formulation of a broadcast protocol requires two core properties: *agreement* (all honest parties output the same value, even if the sender is cheating) and *validity* (if the sender is honest, then all honest parties output its message). A broadcast protocol is t-resilient if both properties hold facing any set of (up to) t misbehaving and colluding parties.

Broadcast is one of the most studied problems in the context of fault-tolerant distributed computing and cryptographic protocols, leading to numerous breakthrough results. For example, classical results show that while t-resilient broadcast protocols can be constructed in the plain model for $t < n/3$ [44,76], a larger corruption threshold cannot be tolerated [16,40,66]. Overcoming this lower bound requires working in weaker models. A common approach is to assume a setup assumption in the form of a *public-key infrastructure* (PKI) for digital signatures [35] (where every party generates a pair of signing/verification keys, and publishes its verification key during the setup phase), or more involved *correlated randomness* (where a trusted party generates correlated secrets to the parties before the protocol begins; e.g., an "information-theoretic PKI" [77]); this approach enables broadcast protocols tolerating $t \leq n$ corruptions.[1]

Simulation-Based vs. Property-Based Definitions. Broadcast can be thought of as a concrete instance of secure multi-party computation (MPC) [53,86]. MPC protocols enable a set of mutually distrusting parties to compute a function on their private inputs, while guaranteeing various properties such as correctness, privacy, independence of inputs, and more. While the original security definitions had the above property-based flavor, nowadays standard definitions formalize the above requirements (and others) in a simulation-based manner [20,21,52]. Informally, in the simulation paradigm for security, the protocol execution is compared to an ideal world where the parties have access to a trusted party (the "ideal functionality") that captures the security properties the protocol is required to achieve. The trusted party takes the parties' inputs and performs the computation on their behalf. A protocol is then regarded secure if for any adversary attacking it, there exists an ideal-world adversary (the "simulator") attacking the execution in the ideal world, such that no external distinguisher (environment) can tell the real and the ideal executions apart.

Simulation-based definitions provide several advantages compared to the property-based approach. First, in a property-based definition, it may be the case that an important property is missed (e.g., one may require privacy of the inputs but neglect to require input independence); this may be subtle to

[1] For the related *consensus* problem (aka Byzantine *agreement*), where all parties have an input, the best achievable bound is $t < n/2$ [41].

notice since the properties should capture both the guarantees towards the honest parties as well as the influence the adversary may have over the computation. Second, the holistic approach provides a simple and clear definition that can be applied in complex settings, such as adaptive corruptions and concurrent executions. Third, many simulation-based security definitions guarantee security under composition, which enables analyzing a complex task where sub-protocols are modeled as ideal functionalities, and later replaced by protocols securely realizing them.

For the specific case of broadcast, the commonly used ideal functionality (e.g., [25,54]) mimics an ideal megaphone in a rather simple way: First, the sender provides its message to the ideal functionality, which later hands it out to the adversary and to all other parties.

Adaptively Secure Broadcast. It is not hard to see that a broadcast protocol which is secure according to the property-based definition (requiring *agreement* and *validity*) also realizes the ideal megaphone functionality when the set of corrupted parties is defined at the onset of the protocol (i.e., when the adversary is *static*). However, as observed by Hirt and Zikas [58], this no longer holds in the adaptive-corruption setting. The issue is that a *rushing* adversary may be the first to learn the sender's input message, in which case it can corrupt the sender and replace its message (i.e., "bias" the content of the message), or simply crash it, in case of fail-stop adversaries. For example, the protocol of Dolev and Strong [35] (and the vast majority of the protocols in the literature) begins by having the sender send its message to all other parties, who then proceed to make sure they all agree on the output value. In case the first party receiving this message is corrupted, the adversary can decide whether to corrupt the sender (thus preventing all other parties from learning it) as a function of that message.

Hirt and Zikas [58] defined a weaker functionality that captures this capability of the adversary to influence the output. In this *corruption-unfair broadcast* functionality, once the functionality receives the input from the sender, it first hands it to the adversary, who can now corrupt the sender and replace its input *before* the functionality sends the output to the remaining honest parties. Such a broadcast protocol is *corruption-unfair* because the adversary gets a "double dipping" capability to both learn the sender's input before the other parties *and* to change it. This is in contrast to the megaphone functionality that allows the adversary to *either* be the first to learn the message *or* to corrupt the sender (without first learning the message) and choose the output, but not both. This difference is best illustrated when each party broadcasts a random bit: if corruption-unfair broadcast is used the adversary has the capability to bias the agreed-upon bits towards 0 by corrupting only senders that broadcast 1 and flipping their bit, whereas if the ideal megaphone is used the adversary does not get any advantage over simply randomly guessing which parties broadcast 1.[2]

[2] In the context of collective coin tossing, the capability of the adversary to first learn the sender's message and later to corrupt the sender and change its input has been referred to as *strongly adaptive* [55,56,60,64].

Hirt and Zikas [58] further showed that the megaphone functionality can be realized for $t \leq n/2$ (i.e., when the adversary cannot corrupt a majority of the parties); the idea is for the sender to "commit" its message into the system using verifiable secret sharing (VSS), and later use corruption-unfair broadcast to reconstruct the original message (as observed in [31,32], robust secret sharing can be used instead of VSS). But for the dishonest-majority setting, as mentioned above, Hirt and Zikas showed that realizing the megaphone functionality in the case of adaptive adversaries is impossible.

The fact that the problem statement (and proof) of the impossibility of adaptively secure broadcast in [58] were both given only with respect to a simulation-based definition, created the (as we prove, inaccurate) perception that this impossibility is an artifact of simulation-based security, and would not carry-over to property-based definitions—see, for example, [12,83,84]. In particular, the first central question of our work, which we answer in the affirmative, is:

Does the impossibility of adaptively secure broadcast in [58] *also apply to the property-based setting?*

In fact, the quest to answer the above question reveals a deeper issue one needs to account for when addressing adaptive security of a protocol using a property-based definition. In particular, as discussed below, in order to answer this question, we distill a natural property that secure computation protocols—not just broadcast protocols—tolerating adaptive adversaries should satisfy, which we term *corruption fairness*.[3]

Atomic vs. Non-atomic Multisend. The attack from [58] applies in the so-called *non-atomic multisend* model, where sending multiple messages to the network are considered as separate operations. This is the classical model considered in the distributed-computing literature since the '80s (e.g., [35,39,40]), where the adversary could corrupt a party and make it "crash" (or change its input [38]) right after the party sends its messages to some of the parties, but before it completed sending to all parties. This is also the standard model for capturing adaptive corruptions in the MPC literature (e.g., [20,21,24,27]). The ability of the adversary to corrupt a party in such a manner has also been referred to as *strongly rushing* [1,2,83].[4]

In passing, we note that one can also motivate the non-atomic multisend model by modern message diffusion protocols, such as the one used in distributed-ledger constructions: for example, consider the setting where a party's outgoing communication goes through a router (e.g., an ISP) that may

[3] Although in this work we focus on broadcast, corruption fairness can easily be defined—and is a natural requirement—for any adaptively secure MPC task.

[4] In this work we refrain from using the term "strongly rushing," because we believe it creates the misconception of an assumption on the adversary. We view the non-atomic multisend model as the *"plain"* model for a rushing adversary, a view which is consistent with the literature [20,21], and atomic multisend as an assumption on the network which limits the adversary's adaptivity.

queue (or even block) some (or all) outgoing messages. If we view such diffusion protocols as emulating send-to-many communication (i.e., multisend), then by corrupting the router the adversary can achieve the same message delivery patterns as in a non-atomic multisend scenario.[5]

In [47], Garay *et al.* noticed that this attack does not carry over to the *atomic multisend* model where the sender is guaranteed not to be corrupted in the time between sending its first message for a given round and the time it completes sending all messages for that round, and, further, a message that has been sent is guaranteed to arrive at its destination. Interestingly, Garay *et al.* showed another variation of this attack illustrating that the protocol of Dolev and Strong [35] (and all other protocols in the literature) does not realize the megaphone functionality even in the atomic-multisend model. Complementarily, they presented an adaptively secure broadcast protocol tolerating $t < n$ corruptions in this model.

Even though the atomic-multisend model has recently gained popularity with many consensus protocols that seek security against adaptive corruptions (e.g., [1,13,28,29,82,84]), the non-atomic-multisend model is the one that corresponds to the "plain" network model, as it makes less assumptions on the underlying communication network.[6] This model is more challenging as it admits more powerful adversaries; indeed, certain impossibility results in the non-atomic multisend model do not translate to the atomic-multisend regime [1,17,58]. Let us stress that it is neither the goal nor the intention of this work to dismiss the atomic multisend model, which is frequently used in the distributed-computing literature. Our point here is that atomic multisend is a network assumption limiting the rushing power of the adversary (and, therefore, its adaptivity), by effectively posing a restriction on the adversary's ability (speed) to corrupt.

The Resource-Restricted Paradigm. A more recent approach to overcome the impossibility results of broadcast [16,40,66] without using "private-state setup assumptions"[7] (such as a PKI) is the *resource-restricted cryptography* (RRC) paradigm [50], where instead of considering arbitrary adversaries that run in probabilistic polynomial time (PPT), additional restrictions are assumed on their capabilities. For example, when the computational power of the adversary is assumed to be smaller than the combined computational power of the honest parties, Nakamoto-style consensus [48,75] employs proofs of work (PoWs) [36] to overcome the aforementioned lower bounds without relying on PKI-like setup assumptions. This is a fruitful promising approach that has led to broadcast protocols [4] and secure multi-party computation protocols [50] that can tolerate any dishonest minority, given only a "public-state setup."

[5] We stress that the above is orthogonal to the synchrony assumption: Consider for example a synchronous setting, where a round takes 60 s (i.e., any message sent by an honest party is delivered within 60 s) and corrupting a party takes 30 s. Then delaying messages at the router gives the adversary time to corrupt the sender and crash it based on messages it sends, dropping all pending messages.

[6] We view non-atomic multisend as the *"plain"* model for a rushing adversary, a view which is consistent with the literature on security models for MPC [20,21].

[7] Terminology taken from [43].

Another example of resource-restricted cryptography is *time-based hardness.* Here there is no restriction on the overall computational power of the adversary (other than being PPT); instead, there is an assumed bound on the number of parallel steps that the adversary can take within a given time interval. This assumption enables the usage of *time-lock puzzles* (TLPs) [11,79] and has been used for example by Boneh and Naor [14] to overcome the lower bound by Cleve [30] and construct a fair coin-tossing protocol between two parties. This approach has led to several interesting results, such as "resource fairness" [45], non-interactive non-malleable commitments [67], and round-efficient randomized broadcast [83]. Another use case of time-based hardness which has been shown to be sufficiently strong to overcome Cleve's impossibility is *verifiable delay functions* [15,78,85].

Thus, the second main question we ask in this paper is:

Can the impossibility of adaptively secure broadcast [58] *be circumvented in the resource-restricted cryptography paradigm?*

Intriguingly, the answer to the above seemingly innocent question is different depending on the definition of (adaptively secure) broadcast one adopts—property-based vs. simulation-based—and/or on how strong a setup we are willing to assume. In particular, we answer this question in the affirmative in the case of property-based definition via TLPs, which can be viewed as an instance of RRC. However, in the case of simulation-based security, it turns out that TLPs do not suffice. Nonetheless, we show that a positive result—i.e., simulation-based adaptively secure broadcast against corrupted majorities—can be achieved based on *non-committing* TLPs, which use access to a programmable random oracle.

1.1 Our Contributions

In this paper we carry out a thorough investigation of adaptively secure broadcast for a wide class of common setups, both in the property-based and in the simulation-based security settings. In the property-based setting, we devise a characterization of the feasibility landscape covering a broad class of protocols—effectively, all known broadcast protocols from the literature; for simulation-based security, our characterization is *complete*—i.e., it covers all possible protocols. Our results are summarized in Table 1.

We proceed to describe our contributions in more detail.

A Property-Based Definition of Adaptively Secure Broadcast. Our first contribution towards investigating the applicability of the impossibility results of Hirt and Zikas [58] to the property-based setting, is to come up with a property-based definition of secure broadcast that captures the essence of an adaptive attack (like the one from [58]). We stress that one might be able to come up with several variants of such a definition, that capture different aspects of corrupting a party in an adaptive fashion. Our goal, however, is not to answer the question "What is

Table 1. Feasibility of adaptively secure broadcast with non-atomic multisend, synchronous communication. All negative results (lower bounds) hold for any dishonest majority of fail-stop corruptions and any correlated-randomness setup; (∗) negative results for property-based broadcast are for protocols in the class $\Pi_{\text{step-rel}}$ (that includes all known broadcast protocols), see Definition 9. All positive results (protocol constructions) tolerate an arbitrary number of malicious corruptions and require a PKI for signatures. TLP stands for a weak time-lock puzzle and RO for programmable random-oracle model.

	property-based	simulation-based
PKI	✗* **Theorem** 2	✗ HZ [58]
PKI+RO	✗* **Corollary** 3	✗ **Corollary** 2
PKI+TLP	✓ **Theorem** 3	✗ **Theorem** 5
PKI+TLP+RO	✓ **Theorem** 3	✓**Theorem** 6

the *right* property-based definition of adaptively secure broadcast?"[8]; rather, any such definition that extends the standard property-based definition to capture natural effects of adversarial adaptivity is well suited for understanding applicability of lower bounds to the property-based setting, because they highlight different attack surfaces that might be exploited by an adaptive adversary.

In a nutshell, the new definition aims to capture the following natural property of adaptively secure protocols (which has thus far not been made explicit in the analysis of adaptive security): The (adaptive) adversary should not be able to corrupt a party—in our case the sender—and influence this party's input value based on this value.[9] One can easily see the importance of such a property for randomized tasks beyond just broadcast, such as for example leader election. In fact, as it will become apparent from our property-based impossibility proof, the corruption fairness property is related to the existence of a *committal round* [72], i.e., a fixed round in which all inputs to the protocol are committed. As proven by Canetti et al. [26], a committal round is necessary for boosting static security to adaptive security, generically, in perfectly secure MPC.

We name the new property *corruption fairness with respect to inputs* (*corruption fairness* for short). In more detail, following [58] and with the illustrating example of broadcasting a random bit in mind (where the adversary's goal is to corrupt only parties who broadcast a given value, say 1, and flip their bit), our definition goes as follows (see Definition 5 for a formal version).

Definition (Broadcast, property-based definition, informal)**.** An n-party protocol is an *adaptively secure t-resilient broadcast protocol* according to the property-based definition if, in addition to agreement and validity, it satisfies the following:

[8] In fact, we conjecture that it might be impossible to capture *all* natural properties of adaptive security in *one* property-based definition, i.e., without effectively resorting to the simulation-based paradigm.

[9] It might be useful to make a distinction here between *corruption fairness* and *input independence*: The latter requires that the adversary cannot bias corrupted parties' input based on the honest parties' input, and, unlike corruption fairness, applies both to static and adaptive adversaries.

- **Corruption fairness with respect to inputs:** The probability of any PPT adversary to win the following game is bounded by $1/2 + \mathsf{negl}(\kappa)$ (where κ denotes the security parameter). When attacking an execution of the protocol where the sender begins with a random bit $b \leftarrow \{0,1\}$ as its input, we say that the adversary wins the game if one of the following events occurs:
  - $b = 0$ and the sender remained honest at the end of the protocol;
  - $b = 1$ and the common output of the honest parties is 0.

We emphasize that the definition can easily be generalized to deal with arbitrary, polynomial-length messages, and to any message x_0 that the adversary wishes to bias towards. That is, where the goal of the adversary is to keep the sender honest whenever sending the message x_0, but corrupt the sender when sending a message $x \neq x_0$ and force the output to be x_0.

We illustrate the power of this definition compared to the weaker definition (that guarantees only *agreement* and *validity*) via the following use cases:

- The first is *collective coin flipping* where each party broadcasts a random bit. When corrupting an arbitrary set of t parties the adversary can set their inputs to 1, but on expectation $t/2$ of them already started with 1, so on expectation $(n + t)/2$ values will be 1 and $(n - t)/2$ values will be 0. Using a broadcast protocol satisfying the definition above, the adversary gains no more power. However, using the weaker definition, the adversary can dynamically choose to corrupt t parties who broadcast 0, thus on expectation $n/2+t$ values will be 1 and $n/2 - t$ values will be 0.
- The second is *hiding a small number of senders in a large population.* In many settings a small set of initially unpredictable parties should reliably broadcast their messages. Using a broadcast protocol satisfying the weaker notion, the adversary can monitor the system and immediately corrupt any party who sends a message, thus executing a DoS attack. This can be overcome using a broadcast protocol satisfying the definition above, where each sender broadcasts its message while adding '1' as a prefix, whereas all other parties broadcast the zero string; messages starting with '0' are later discarded.[10] A similar approach was used in the broadcast protocol of Wan et al. [83] to achieve a single-round reliable communication by a small set of unpredictable senders; however, as we explain below, their construction still does not satisfy the *corruption-fairness* property.

Impossibility of Property-Based Adaptively Secure Broadcast. It is not hard to verify that any broadcast protocol that is secure according to the simulation-based definition (i.e., realizes the ideal megaphone functionality) is also secure according to the property-based definition of (adaptively secure) broadcast. The intuition is that a simulator that interacts with the megaphone functionality can win the corruption-fairness game only with probability 1/2 (by guessing

[10] Note that in our model the adversary can corrupt a party after sending a message and drop the message from the network, but this is done *independently* of the content of the message; therefore, we require all other parties to broadcast dummy messages.

the input), and therefore any adversary that can win the corruption-fairness game with a noticeable probability over 1/2 can be translated to a distinguisher between the real and ideal computations. We formally prove this result in Lemma 1.

However, one may ask whether the property-based definition is actually weaker than the simulation-based definition, or if it is equivalent. Stated differently, does the property-based definition above capture the attack from [58]? The attack from [58] rules out the simulation-based definition, but that may perhaps be due to another feature of the megaphone functionality.

Our second contribution is extending the impossibility result from [58] to rule out the property-based definition for a large class of protocols that includes all published approaches to construct broadcast protocols, in particular recent ones explicitly targeting adaptive security [28,82–84]. Intuitively, this covers all protocols that define an *a-priori*-known round R such that prior to round R it is guaranteed that no set of size $\lfloor n/2 \rfloor - 1$ "knows" the sender's input (in the sense that if this set emulates in its head a continuation of the protocol where all other parties crash, it has a noticeable error probability), and at round R there exists a set of size $\lfloor n/2 \rfloor - 1$ that "knows" the sender's input (i.e., by emulating the continuation, the set errs only with negligible probability).[11] In the sequel, we will denote this class of "step-release" protocols by $\Pi_{\mathsf{step\text{-}rel}}$. It is worth noting that existence (but not *a-priori* public knowledge) of such a round is guaranteed in any *execution* of any broadcast protocol, which follows from the fact that at the beginning of the protocol only the sender knows his input, whereas at the end everyone learns it.

This means that in term of *feasibility*, the simulation-based definition and the property-based definition are *equivalent* for all protocols from the class $\Pi_{\mathsf{step\text{-}rel}}$: i.e., for $t \leq n/2$ both definitions can be satisfied, and for $t > n/2$ both definitions cannot be satisfied. Note that this does not imply that any protocol that satisfies the property-based definition also satisfies the simulation-based definition.

Theorem 2 (Impossibility of property-based broadcast, informal)**.** *Let $t > n/2$. Then, there is no adaptively secure broadcast protocol (from the class $\Pi_{\mathsf{step\text{-}rel}}$) tolerating a fail-stop, PPT t -adversary that satisfies the property-based definition of (adaptively secure) broadcast.*

We note that the impossibility result holds even assuming any correlated-randomness setup and/or secure data erasures.

Overcoming the Property-Based Impossibility via TLPs. Next, we study whether the RRC paradigm can overcome the impossibility of adaptively secure broadcast. We use TLPs [11,79] for this task. The idea is quite simple: the sender "hides" its message inside a TLP and uses a protocol for corruption-unfair broadcast (e.g., [35,83]) to send the puzzle to all parties; every recipient can open the puzzle after investing a polynomial amount of computation and obtain the output. We note that our usage of TLPs is similar to Wan et al. [83] with the difference that in [83]

[11] In most broadcast protocols from the literature (e.g., [28,35,46,83,84]), the sender starts by sending its input to all parties, meaning that $R = 1$.

the TLP was hidden for a duration of a round, whereas we hide it for the duration of the entire protocol; see Appendix 1.2 for a detailed comparison.

The guarantee provided by a TLP with gap $\varepsilon < 1$ (see Definition 2) is that when setting the puzzle with difficulty parameter $T^{1/\varepsilon}$, any adversary that can evaluate circuits of polynomial size, but of depth bounded by $T(\kappa)$, cannot solve the puzzle with better than negligible probability. We will say that an adversary is (R, T)-*bounded* if the number of parallel steps it can take within R rounds is bounded by $T(\kappa)$. Therefore, if the corruption-unfair broadcast protocol takes R rounds, we are guaranteed that any (R, T)-bounded adversary cannot win the corruption-fairness game with more than $1/2 + \mathsf{negl}(\kappa)$ probability.

In fact, our protocol does not require a "lightweight" generation of the puzzle, and can use a puzzle generation that is as computationally expensive as solving the puzzle. Therefore, we only require the *weak* variant of time-lock puzzles [11,69] that allows for parallelizable, yet computationally expensive puzzle generation, and can be based on one-way functions and the existence of non-parallelizing languages [11]. We show:

Theorem 3 (Feasibility of property-based broadcast via TLPs, informal)**.** *Let $t \leq n$, let T be a polynomial, assume that weak time-lock puzzles exist, and that corruption-unfair broadcast can be computed in R rounds. Then, there is an adaptively secure broadcast protocol tolerating an (R, T) -bounded t -adversary that satisfies the property-based definition of (adaptively secure) broadcast.*

TLP Barriers for Simulation-Based Broadcast. Next, we ask whether TLPs are also sufficient to satisfy the simulation-based definition of broadcast. Somewhat surprisingly, the answer to this question is negative, thus posing a separation between the two definitions. The main reason is illustrated when trying to simulate the protocol that satisfies the property-based definition. When the sender is honest and a simulator tries to simulate the TLP without knowing the message, it gets stuck, since the TLP is a committing object: Once the puzzle is generated it can only be opened to a unique value. Therefore, the simulator's success probability is again restricted to correctly guessing the sender's input, which results in a noticeable distinguishing probability between the real and ideal executions.

In Sect. 5.1 we extend this argument to rule out *any* adaptively secure broadcast protocol even facing an (R, T)-bounded adversary. In turn, this implies that TLPs are not sufficient for realizing simulation-based broadcast.

Theorem 5 (Impossibility for simulation-based broadcast from TLPs, informal)**.** *Let $t > n/2$, and let R and T be polynomials. Then there is no adaptively secure broadcast protocol tolerating an (R, T) -bounded, fail-stop, PPT t -adversary that satisfies the simulation-based definition of broadcast.*

The impossibility result can be extended to hold even assuming any correlated-randomness setup, secure data erasures, and/or a non-programmable random oracle, in addition to TLPs.

Overcoming the Simulation-Based Impossibility of Adaptively Secure Broadcast. We note that the above "barrier" resembles other barriers in achieving adaptive

security of committing cryptographic primitives, such as commitments [23,26] and public-key encryption [73]. Next, we show that a programmable random oracle can be used to construct a non-committing variant of TLPs, which in turn allows us to overcome the above barrier. Namely, instead of hiding the message m inside the puzzle, the sender samples a random one-time pad key x, hides x inside the puzzle, and corruption-unfairly broadcasts the puzzle along with $c = m \oplus H(x)$. Now the simulator can simulate a puzzle when the sender is honest, and upon a corruption request of the sender (or after R rounds have elapsed, and it can safely ask the megaphone functionality for the output), the simulator can program the random oracle appropriately.

This approach is similar to the one in [5,9] who used a programmable RO to model composable TLPs. What **substantially differentiates** our treatment is that we rely on the *complexity-based* definition of TLPs [11], which is realizable from computational hardness assumptions, rather than requiring, as [5,9] do, access to an ideal functionality which is **not** (and arguably *cannot* be) implemented from such assumptions in the plain model. This forces us to explicitly treat the composability issues of (complexity-based) TLP constructions. This turns out to be non-trivial and may be of independent interest.

Theorem 6 (Feasibility of simulation-based broadcast via TLPs in the RO model, informal)**.** *Let $t < n$, let T be a polynomial, assume that weak TLPs exist, and that corruption-unfair broadcast can be executed in R rounds. Then, there is an adaptively secure broadcast protocol, according to the simulation-based definition, tolerating an (R, T) -bounded t -adversary in the programmable random-oracle model.*

Random Oracle (RO) Barriers. Given the simulation-based impossibility even assuming TLPs and the above possibility when assuming TLP in tandem with a programable RO, one might wonder whether just assuming a programable RO would do the trick. We answer this question in the negative, by showing how to adapt the impossibility of Theorem 5 to hold when we replace TLPs with an (even programable) RO (see Corollary 2). To complete the picture, we also show how to derive the impossibility of property-based adaptively secure broadcast even assuming an RO, as a simple corollary of Theorem 2 (see Corollary 3).

Composition in Resource-Restricted Settings. The protocols in our positive results (Theorems 3 and 6) rely on delivering a TLP to all parties via an ideal corruption-unfair broadcast functionality $\mathcal{F}_{\mathsf{ubc}}$; indeed, one can later use the protocol of Dolev and Strong [35] as a concrete instantiation of corruption-unfair broadcast in the PKI model. It might be tempting to use an off-the-shelf composition theorem for claiming security of the derived protocol. However, it turns out that standard composition theorems no longer apply in the RRC setting since the adversary may take advantage of the honest parties' resources in the sub-protocol when attacking the higher-level execution. For example, given a corruption-unfair broadcast protocol π, consider a new protocol π' where some party P_i sends a TLP to another party P_j who solves the puzzle and returns the solution to P_i; otherwise, all parties proceed according to π. Clearly, π' has

the same security guarantees as π; however, when used to instantiate $\mathcal{F}_{\mathsf{ubc}}$ in our broadcast constructions, the adversary can corrupt P_i and send the sender's TLP to P_j and this way learn the underlying message.[12]

As an additional contribution, we prove a limited composition theorem (Theorems 4 and 7) that is sufficient for instantiating $\mathcal{F}_{\mathsf{ubc}}$ in our setting by protocols that also consider a bound on the parallel computational resources of honest parties; for example, in the case of Dolev and Strong [35], honest parties only sign and verify signatures, but do not perform other computations, so the adversary cannot "outsource" solving the puzzle to honest parties. We leave the quest for a more general composition theorem as an interesting open problem.

Summary of Our Contributions. Taken together, our results distill the essence and extend the reach of the impossibility result from [58]. This establishes that the impossibility of adaptively secure broadcast is not just an artifact of the simulation-based definition, but it also applies to an extension of the property-based broadcast definition to the adaptive-corruptions case. Further, we show how the resource-restricted paradigm separates the property-based definition from the simulation-based definition, which serves as yet another motivation for using simulation-based security, especially when designing adaptively secure protocols. Finally, we prove the first composition theorem in the RRC setting, where UC composition no longer holds.

1.2 Related Work

Recently, Wan et al. [83] used TLPs to construct adaptively secure **corruption-unfair** broadcast protocols (i.e., not the adaptively secure primitive we are after) in the non-atomic multisend model, with the goal of reducing the round complexity of randomized broadcast from linear to poly-logarithmic, facing a constant fraction of corrupted parties. As pointed out by the authors, their goal was *not* to realize the megaphone functionality, but only to satisfy the property-based definition of (corruption-unfair) broadcast. The main idea in [83] is to use TLPs to "hide" the contents of the messages *for one round at a time* in a way that essentially provides atomic-multisend guarantees. Given this, they run the polylogarithmic-round protocol of Chan et al. [28], which in turn is based on Dolev and Strong [35].

We note that although the protocol in [83] relies on similar assumptions as the ones in this work, it **does not** answer our question as it is vulnerable to the attack from [47], showing that the Dolev-Strong protocol (DS) [35] is **not** adaptively secure even in the atomic-multisend model. Specifically, the adversary waits for the completion of the first round of DS, in which the sender sends its input to all parties. Before the second round begins, the adversary (who learns

[12] The UC composition theorem in [22] applies to *balanced* environments, i.e., environments that do not give honest parties much more resources than to the adversary. In [22] the focus is on running time, whereas in this work it is on *parallel* running time; hence, by abusing the terminology from [22], one can say that the environment in our protocol is not balanced with respect to parallel running time.

the content of the message at that point) can decide whether to corrupt the sender and "inject" a signature on a different message to some of the second-round messages (thus forcing the protocol to abort and output a default value), or keep the sender honest and let the protocol successfully complete with the original input message. In contrast, in our construction we hide the message using a TLP for the entire duration of DS protocol (not in a round-by-round way), and this enables overcoming the attack from [47].

Baum et al. [9] study a stronger version of TLPs that provides universal composability. They define an ideal TLP functionality, and prove that realizing it inherently requires a programmable random oracle. Next, they realize the TLP functionality based on generic-group-style formalization of the repeated-squaring technique from [79] as well as a restricted programmable and observable random oracle [19]. In contrast to the weaker, property-based definition of TLP [11,79] (used in this paper), the reliance on a random oracle in [9] enables the TLP functionality to define a fixed and *a priori* known step with the guarantee that the adversary learns nothing about the content of the TLP prior to that step and that once that step is reached, the content of the puzzle is fully revealed. The ideas from [9] that apply to the two-party setting were extended in [10] to capture the multi-party case, as well as verifiable delay functions.

In more detail, Baum et al. [9] give an elegant argument showing that coin-flipping protocols based on TLPs, such as the one of Boneh and Naor [14], cannot be simulated without resorting to a programmable RO, even facing so-called *computationally restricted environments*. Essentially, when simulating a TLP-based coin-flipping protocol, the environment may first get the information needed to learn the output (possibly after the conclusion of the protocol) and then abort with probability 1/2. Next, it can check whether the output learned from its view matches the honest party's output; if so it outputs 'ideal' and if not 'real'. The simulator who receives the honest party's output must simulate the view using this output bit without knowing whether the environment will abort or not; in case of abort, the simulator must equivocate the output obtained from the committed view by the environment to be a random bit—a task that cannot be achieved in the standard model.

Although our proof technique and overall reasoning are very different from those in [9], the source of the impossibility in both cases is the fact that TLPs are non-equivocable. Such equivocality turns out to be essential in both simulation arguments, despite the inherent difference of the primitives and the statements themselves. For example, as the impossibility of [9] relies on Cleve's impossibility [30], the attack applies even with static corruptions; further, when considering the multiparty setting it is oblivious to the underlying network (e.g., it applies even given a broadcast channel). In contrast, in our setting, the attack crucially relies on the adversary's adaptive and rushing capabilities, and is very sensitive to the underlying network assumptions (e.g., the attack no longer holds in the atomic-multisend model).

Matt et al. [71] formalized the notion of delayed adaptive corruptions in UC, where the adversary, who wishes to corrupt a certain party, gets hold of the newly corrupted party only after some time has elapsed. The goal of their paper

is to prove security of various flooding protocols (that inherently require a strong form of atomic multisend capabilities) in this model. In contrast to [71] we do not restrict in the model the time it takes the adversary to corrupt a party, but instead rely on cryptographic assumptions.

Arapinis et al. [5] presented a UC modeling of TLPs in the UC framework. To overcome the non-equivocation barrier of TLPs, they follow Nielsen [73] and use a programmable random oracle to equivocate the content of the TLP; our construction for overcoming the impossibility of simulation-based adaptively secure broadcast from TLPs essentially uses the same technique as [5] for equivocating the TLP using a programmable RO. As opposed to [9], they do not rely on generic-group-style assumptions and rely solely on a programmable random oracle; however, to restrict the computational capabilities of the adversary, the authors use a functionality wrapper that limits the number of evaluation queries that can be done in a round in the spirit of [6,50].

We remark that [5,9,10] use an **ideal functionality** to model TLPs, but it is unclear how to **compose** a realization of the TLP functionality in a protocol that invokes it without resorting to generic-group-style assumptions or a functionality wrapper (as discussed above, standard UC composition does not apply in the resource-restricted setting). In contrast, our composition theorem provides a fine-grained analysis of a limited "plug-and-play" design for TLPs.

2 Preliminaries

In this section we first present the network model, followed by some basics on simulation-based security, and conclude with the definition of time-lock puzzles.

2.1 The Model

An n-party protocol $\pi = (P_1, \ldots, P_n)$ is an n-tuple of PPT interactive Turing machines (ITMs). The term *party* P_i refers to the i^{th} ITM; we denote the set of parties by $\mathcal{P} = \{P_1, \ldots, P_n\}$. Each party P_i starts with input $x_i \in \{0,1\}^*$ and random coins $r_i \in \{0,1\}^*$. Without loss of generality, the input length of each party is assumed to be the security parameter κ. We consider protocols that additionally have a setup phase (used, e.g., to model a public-key infrastructure (PKI)) where a trusted dealer samples (possibly correlated) secret values $(\mathtt{r}_1, \ldots, \mathtt{r}_n) \leftarrow D_\pi$ from some efficiently sampleable distribution D_π, and hands party P_i the secret string $\mathtt{r}_i$ (referred to as the correlated randomness of P_i). While our lower bounds hold with respect to any distribution for correlated randomness, our upper bounds rely on a weaker setup assumption of a PKI for digital signatures, where each party generates a pair of signing/verification keys and publishes its verification key.

An *adversary* $\mathcal{A}$ is another PPT ITM describing the behavior of the corrupted parties. It starts the execution with input that contains the security parameter (in unary) and an additional auxiliary input. At any time during the execution of the protocol the adversary can corrupt one of the honest parties, in which case the adversary can read its internal state (containing its input, random coins,

correlated randomness, and incoming messages) and gains control over it. A t-adversary is limited to corrupt up to t parties.

The parties execute the protocol over a fully connected synchronous network of point-to-point channels. That is, the execution proceeds in rounds: Each round consists of a *send phase* (where parties send their messages from this round) followed by a *receive phase* (where they receive messages from other parties). The adversary is assumed to be *rushing*, which means that it can see the messages the honest parties send in a round before determining the messages that the corrupted parties send in that round. The communication lines between the parties are assumed to be ideally authenticated (and thus the adversary cannot modify messages sent between two honest parties but can read them).

Throughout the execution of the protocol, all the honest parties follow the instructions of the prescribed protocol, whereas the corrupted parties receive their instructions from the adversary. In our positive results, the adversary is considered to be actively malicious, meaning that it can instruct the corrupted parties to deviate from the protocol in any arbitrary way. Our lower bounds, however, only rely on fail-stop adversaries that can crash parties, but not cheat in any other way. At the conclusion of the execution, the honest parties output their prescribed output from the protocol, the corrupted parties do not output anything and the adversary outputs an (arbitrary) function of its view of the computation (containing the views (internal states) of the corrupted parties).

Atomic Multisend. A subtle point that is central to this work is the capabilities of the adversary when corrupting a party that has just sent its messages for the round. Two central models are considered in the literature:

- In the *atomic multisend* model [47] a message that has been sent to the network is guaranteed to be delivered to its recipients even if the sender becomes corrupted shortly after sending; further, the messages are sent to the network as an atomic operation in the sense that once the sender begins sending its messages for the round it cannot become corrupted until it has finished sending all of its messages for the round. This model has gained popularity in many recent consensus protocols (e.g., [1,13,28,29,82,84]).
- In the standard (*non-atomic multisend*) model, the operation of sending messages to the channel is not atomic, and the adversary may corrupt a sender *after* it sent its message to some party P_i and *before* it has sent its message to another party P_j; further, the adversary can drop the message the newly corrupted sender sent to P_i and replace it with another. This is the model that has been used in classical models of distributed computation (e.g., [35,38–40]) and cryptographic protocols [20,21,24,27]. This models has also been referred to as *strongly adaptive* [55,56,60,64] and *strongly rushing* [1,2,83].

In this work we consider the non-atomic multisend model. Clearly, this is the preferred one as it requires less assumptions on the underlying communication. However, this model is more challenging as it considers more powerful adversaries; indeed, certain impossibility results in the non-atomic-multisend model do not translate to the atomic-multisend realm [1,17,58]. In fact, as proven by Katz et al. [61], atomic multisend is a strictly weaker model facing dishonest-majority

as it cannot be realized from the basic ingredients needed for synchronous communication (bounded-delay channels and a synchronizing clock).

Secure Data Erasures. Two models are normally considered in the adaptive-corruption setting, depending on the ability of honest parties to securely erase certain parts of their memory (i.e., from their internal state) without leaving any trace; see [20,24] for a discussion. While some impossibility results of adaptively secure cryptographic protocols crucially rely on parties *not* being able to erase any information, and completely break otherwise (e.g., [49,51,57,73]), other impossibility results are stronger and do not rely on the absence of secure erasures (e.g., [17,33,58,62]).

In this work we do not assume secure erasures for our protocol constructions; however, our impossibility results hold even in the secure-erasures model. This makes for the strongest statements; to avoid confusion we will state the model explicitly in each section.

2.2 Simulation-Based Security

Some of the results in this work consider a simulation-based definition of broadcast, where security is defined via the real vs. ideal paradigm. Namely, a protocol is considered secure if every attack that can be executed by a PPT adversary in the real-world execution, can be simulated by a PPT simulator in an ideal world, where an incorruptible trusted third party (aka, the *ideal functionality*) receives inputs from the parties and carries out the computation on their behalf. For the specific task of broadcast, the trusted party receives the input from the broadcaster and delivers it to all other parties (see Sect. 3.2).

We consider a *synchronous* model with an online distinguisher (aka, the *environment*); this is the prevalent model in many frameworks for cryptographic protocols; see, e.g., [7–9,21,59,65,68,74]. Such a model requires the simulator to report its view to the distinguisher in every round. We do not rely on any other specific properties of the model, but for concreteness, we state our results in the synchronous model of the UC framework as defined in [8,61,65].

Loosely speaking, we consider protocols that run in a hybrid model where parties have access to a simple "clock" functionality $\mathcal{G}_{\mathsf{clock}}$. This functionality keeps a counter, which is incremented once *all honest parties* request the functionality to do so, i.e., once all honest parties have completed their operations for the current round. In addition, all communication is done over bounded-delay channels, where each party requests the channel to fetch messages that are sent to him, such that the adversary is allowed to delay the message delivery by a bounded and *a priori* known number of fetch requests. Stated differently, once the sender has sent some message, it is guaranteed that the message will be delivered within a known number of activations of the receiver. For simplicity, we assume that every message is delivered within a single fetch request.

We note that when considering online distinguishers, a resource-restricted adversary may bypass its limitations by delegating some of its computation to the environment. It is therefore standard to restrict the resources of the environment as well, see e.g., [45]. In this work, when considering a resource-restricted adversary

in the simulation-based setting, we will consider the pair of an adversary and an environment as resource restricted, in the sense the their joint resource is bounded.

To simplify the presentation we describe the functionalities and protocols in a less technical way than standard UC formulations (e.g., we do not explicitly mention the session id and party id in every message, and somewhat abuse the activation policy by batching several operations together).

2.3 Time-Lock Puzzles

Time-lock puzzles [79] enable a sender to "lock" its message in a way that "unlocking" requires an inherently sequential computation. This is a powerful primitive that has led to many results, and has been extensively studied; see, e.g., [3,9,11,14,18,37,42,45,63,67,69,70,80,81,83]. While the standard definition requires the puzzle generation to be "lightweight" compared to solving the puzzle, our feasibility results can be based on the weaker notion in which puzzle generation is as computationally expensive as solving the puzzle (yet, as opposed to puzzle solving, the puzzle generation is parallelizable). Such weak time-lock puzzles are known from the minimal assumption of one-way functions and the existence of non-parallelizing languages [11,69]. In this paper we follow the formulation by Bitansky et al. [11].

Puzzles. A puzzle is associated with a pair of parameters: A security parameter κ determining the cryptographic security of the puzzle, as well as a difficulty parameter T that determines how difficult it is to solve the puzzle.

Definition 1 (Puzzle). *A* puzzle *is a pair of algorithms* $(\mathsf{PGen}, \mathsf{PSol})$ *satisfying the following requirements.*

- Syntax:
  - $Z \leftarrow \mathsf{PGen}(T, s)$ *is a probabilistic algorithm that takes as input a difficulty parameter* T *and a solution* $s \in \{0,1\}^\kappa$*, where* κ *is a security parameter, and outputs a puzzle* Z*.*
  - $s = \mathsf{PSol}(Z)$ *is a deterministic algorithm that takes as input a puzzle* Z *and outputs a solution* s*.*
- Completeness: *For every security parameter* κ*, difficulty parameter* T*, solution* $s \in \{0,1\}^\kappa$ *and puzzle* Z *in the support of* $\mathsf{PGen}(T, s)$*,* $\mathsf{PSol}(Z)$ *outputs* s*.*
- Efficiency:
  - $Z \leftarrow \mathsf{PGen}(T, s)$ *can be computed in time* $\mathsf{poly}(\log T, \kappa)$*.*
  - $\mathsf{PSol}(Z)$ *can be computed in time* $T \cdot \mathsf{poly}(\kappa)$*.*

Time-Lock Puzzles. In a time-lock puzzle, we require that the parallel time required to solve a puzzle is proportional to the time it takes to solve the puzzle honestly, up to some fixed polynomial loss.

Definition 2 (Time-lock puzzle). *A puzzle* $(\mathsf{PGen}, \mathsf{PSol})$ *is a* time-lock puzzle with gap $\varepsilon < 1$ *if there exists a polynomial* $T_1(\cdot)$*, such that for every polynomial* $T(\cdot) \geq T_1(\cdot)$ *and every polysize adversary* $\mathcal{A} = \{\mathcal{A}_\kappa\}_{\kappa \in \mathbb{N}}$ *of depth* $\mathsf{depth}(\mathcal{A}_\kappa) \leq$

$T^{\varepsilon}(\kappa)$, there exists a negligible function μ, such that for every $\kappa \in \mathbb{N}$, and every pair of solutions $s_0, s_1 \in \{0,1\}^{\kappa}$:

$$\Pr\left[b \leftarrow \mathcal{A}_{\kappa}(Z) \;\middle|\; b \leftarrow \{0,1\}, Z \leftarrow \mathsf{PGen}(T, s_b)\right] \leq 1/2 + \mu(\kappa).$$

Definition 3 (Weak puzzle). *A weak puzzle is a pair of algorithms* $(\mathsf{PGen}, \mathsf{PSol})$ *satisfying the* Syntax *and* Completeness *requirements as per Definition 1, and the following* weak efficiency *requirement.*

- Weak Efficiency:
  - $Z \leftarrow \mathsf{PGen}(T, s)$ *can be computed by a uniform circuit of size* $\mathsf{poly}(T, \kappa)$ *and depth* $\mathsf{poly}(\log T, \kappa)$.
  - $\mathsf{PSol}(Z)$ *can be computed in time* $T \cdot \mathsf{poly}(\kappa)$.

Mahmoody et al. [69] showed how to construct a weak time-lock puzzle in the random-oracle model while Bitansky et al. [11] showed how to construct it from any one-way function and non-parallelizing language.

Definition 4 (Non-parallelizing language). *A language* $\mathcal{L} \in \mathsf{DTime}(T(\cdot))$ *is* non-parallelizing with gap $\varepsilon < 1$ *if for every family of non-uniform polysize circuits* $\mathcal{B} = \{\mathcal{B}_{\kappa}\}_{\kappa \in \mathbb{N}}$ *where* $\mathsf{depth}(\mathcal{B}_{\kappa}) \leq T^{\varepsilon}(\kappa)$ *and every large enough* κ, $\mathcal{B}_{\kappa}$ *fails to decide* $\mathcal{L}_{\kappa} = \mathcal{L} \cap \{0,1\}^{\kappa}$.

Theorem 1 ([11]). *Let* $\varepsilon < 1$. *Assume that one-way functions exist, and that for every polynomially bounded function* $T(\cdot)$ *there exists a non-parallelizing language* $\mathcal{L} \in \mathsf{DTime}(T(\cdot))$ *with gap* ε. *Then, for any* $\varepsilon_1 < \varepsilon$ *there exists a weak time-lock puzzle with gap* ε_1.

3 Broadcast Protocols: Definitions

Intuitively, a broadcast protocol should emulate a "megaphone" functionality in the sense that when the sender speaks, all recipients receive its message. This is traditionally captured via the *agreement* and *validity* properties. However, as observed in Hirt and Zikas [58], such a property-based definition falls short of capturing the ideal megaphone functionality when facing adaptive corruptions. Namely, the ideal megaphone functionality does not allow the adversary to corrupt the sender after learning its input message, and change it retrospectively. Hirt and Zikas [58] further showed that the ideal megaphone functionality cannot be realized in the dishonest-majority setting in the standard (non-atomic-multisend) communication model.

3.1 Property-Based Broadcast

With the goal of distilling the essence of the impossibility result in [58], we provide a weaker, property-based definition that is complete in the presence of adaptive corruptions. In addition to *termination*, *agreement*, and *validity*, this

definition requires another property: *corruption fairness with respect to inputs* (corruption-fairness for short). As discussed in the introduction, even though this definition is weaker than the simulation-based one, it is still stronger than the traditional definition of broadcast and enables realizing tasks for which traditional broadcast is not sufficient.

Recall that when broadcasting a random bit via a "corruption-unfair" broadcast (where only *termination*, *agreement*, and *validity* are guaranteed), the adversary gets to learn the input bit *before* deciding whether to corrupt the sender and change its input; for example, the adversary may corrupt the sender when the input is 1 and flip it to 0, but when the input is 0 the adversary may continue without corrupting the sender. Informally, a broadcast protocol should not concede this capability to the adversary.

Without loss of generality, we consider the message space to be $\{0,1\}^\kappa$. Looking ahead, our lower bounds hold even in the simpler, Boolean case where the message space is $\{0,1\}$, while our upper bounds hold for any polynomial-length messages. The goal of the adversary in the corruption-fairness experiment is to force the output to be some predetermined message $x_0 \in \{0,1\}^\kappa$ but without corrupting the sender in case it begins with input x_0. Again, without loss of generality, we let $x_0 = 0^\kappa$, and to simplify the definition consider two potential messages in the experiment: 0^κ and 1^κ.

Definition 5 (Broadcast, property-based definition). *An n-party protocol π, where a distinguished sender holds an initial input message $m \in \{0,1\}^\kappa$, is a broadcast protocol (according to the property-based definition) tolerating adaptive PPT t-adversaries, if the following conditions are satisfied for any adaptive PPT t-adversary $\mathcal{A}$:*

- ***Termination:*** *There exists an a-priori-known round R such that the protocol is guaranteed to complete (i.e., every so-far honest party produces an output value) within R rounds.*
- ***Agreement:*** *All honest parties (at the end of the protocol) output the same value, with all but negligible probability.*
- ***Validity:*** *If the sender is honest (at the end of the protocol) then all honest parties (at the end of the protocol) output m, with all but negligible probability.*
- ***Corruption fairness with respect to inputs:***

$$\Pr\left[\mathsf{Expt}^{\mathsf{fair\text{-}bcast}}_{\pi,\mathcal{A}}(\kappa) = 1\right] \leq \frac{1}{2} + \mathsf{negl}(\kappa),$$

where the experiment $\mathsf{Expt}^{\mathsf{fair\text{-}bcast}}_{\pi,\mathcal{A}}(\kappa)$ is defined in Fig. 1.

Note that, as observed in [58], the protocol of Dolev and Strong [35] (as well as most broadcast protocols in the literature) allows an adversary to first learn the sender's input message m, and later change the common output as a function of m. Therefore, this protocol does not satisfy the *corruption-fairness* property (even in the atomic-multisend model [47]). The broadcast protocols from [31,32,58] satisfy this property for $t \leq n/2$ in the standard model, and similarly, the protocol from [47] for $t < n$ in the atomic-multisend model.

Experiment $\mathsf{Expt}^{\mathsf{fair\text{-}bcast}}_{\pi,\mathcal{A}}(\kappa)$

1. The challenger samples a uniformly random bit $b \leftarrow \{0,1\}$ and invokes $\mathcal{A}$ on input 1^κ.
2. The challenger samples randomness[a] $(\mathtt{r}_1, \ldots, \mathtt{r}_n) \leftarrow D_\pi$ and simulates the protocol π on sender-input b^κ toward $\mathcal{A}$, who can adaptively corrupt parties throughout the execution. (The challenger simulates all honest parties, and upon a corruption request reveals the internal state of the corrupted party to the adversary, as well as the control over that party.)
3. The output of the experiment is set to 1 if:
   - $b = 0$ and the sender is honest at the end of the protocol;
   - $b = 1$ and the output value of an arbitrary honest party is 0^κ.

   Otherwise, the output of the experiment is set to 0. (If all parties are corrupted, the output is set to be 0.)

[a] Without loss of generality this includes both the protocol's random coins and any potentially correlated randomness (setup) parties might use.

Fig. 1. The corruption-fairness experiment for adaptively secure broadcast

We shall refer to the commonly used property-based definition of broadcast as *corruption-unfair broadcast*.

Definition 6 (Corruption-unfair broadcast, property-based definition). *An n-party protocol π tolerating an adaptive PPT t-adversary, is a* corruption-unfair broadcast protocol *if* agreement, validity *and* termination *hold, but* corruption-fairness *does not necessarily hold.*

3.2 Simulation-Based Broadcast

While the property-based definitions provide the core requirements of broadcast, they are weaker than simulation-based definitions and are therefore more suitable for lower bounds. We next present the stronger simulation-based definitions which are better suited for proving the security of protocol constructions.

Definition 7 (Broadcast, simulation-based definition). *An n-party protocol π, is a* broadcast protocol *(according to the simulation-based definition) tolerating an adaptive PPT t-adversary, if π securely realizes the broadcast functionality, defined in Fig. 2.*

We note that our functionality captures causality of corruption vs. information release—the two events that affect corruption-fairness—in an explicit manner, as opposed to [47,58]. Concretely, we specify the causality of the events that the adversary asks to learn the output and that the output value is locked. In particular, in [58], once the input is handed to the functionality, it is automatically locked (so the adversary is not allowed to corrupt the sender and change it). Although this does not make a difference in a standalone setting with an "offline distinguisher" (as the simulator can decide whether to corrupt the sender before

the sender hands its input to the ideal functionality), in a UC-like setting the simulator might not be informed when the (honest) input is given. This might enable the design of protocols which artificially reduce the simulator's choice to corrupt and erase; e.g., if the sender chooses one of polynomially many rounds to start broadcasting its input.

The functionality $\mathcal{F}_{\mathsf{bc}}$

- **Initialization:** The functionality initializes the output message $m_{\mathsf{out}} := \perp$ and a Boolean flag $\mathsf{isOutputLocked} := \mathsf{false}$.
- **Input:** The sender sends an input message $m \in \{0,1\}^\kappa$. The functionality sets the output message $m_{\mathsf{out}} := m$.
- **Output request:** If the adversary asks to receive the output value and there exists at least one corrupted party, the functionality hands the adversary the message m_{out} and sets $\mathsf{isOutputLocked} := \mathsf{true}$. If all parties are honest, the functionality ignores this request.
- **Corruption request:** If the adversary corrupts the sender, the functionality hands the adversary the message m_{out}. The adversary can provide the functionality a message m' and if $\mathsf{isOutputLocked} = \mathsf{false}$, the functionality sets the output message to be $m_{\mathsf{out}} := m'$.
- **Output:** The functionality sends m_{out} as output to all parties and sets $\mathsf{isOutputLocked} := \mathsf{true}$.

Fig. 2. The broadcast functionality

Next, we provide the simulation-based definition of corruption-unfair broadcast, where the adversary can first learn the message and later corrupt the sender and replace its message.

Definition 8 (Corruption-unfair broadcast, simulation-based definition) *An n-party protocol π, is a corruption-unfair broadcast protocol (according to the simulation-based definition) tolerating an adaptive PPT t-adversary, if π securely realizes the corruption-unfair broadcast functionality, in Fig. 3.*

The functionality $\mathcal{F}_{\mathsf{ubc}}$

- **Input:** The sender sends an input message $m \in \{0,1\}^\kappa$. The functionality sets the output value $m_{\mathsf{out}} := m$ and sends m to the adversary.
- **Corruption request:** If the adversary corrupts the sender, the adversary can provide the functionality a message m' and if no honest party received the output yet, the functionality sets the output message to be $m_{\mathsf{out}} := m'$.
- **Output:** The functionality sends m_{out} as output to all parties.

Fig. 3. The corruption-unfair broadcast functionality

As a sanity check, we prove that a protocol that satisfies the simulation-based definition (Definition 7) also satisfies the property-based definition (Definition 5). The proof can be found in the full version of the paper [34].

Lemma 1. *If an n-party protocol π is a broadcast protocol according to the simulation-based definition tolerating an adaptive PPT t-adversary, then π is a broadcast protocol according to the property-based definition tolerating an adaptive PPT t-adversary.*

4 Property-Based Adaptively Secure Broadcast

In this section we analyze the property-based definition of adaptively secure broadcast. In Sect. 4.1 we extend the impossibility result of Hirt and Zikas [58] to this regime, and in Sect. 4.2 we show how to overcome this impossibility using resource-restricted cryptography; namely, via time-lock puzzles.

First we observe that although the impossibility statement in Hirt and Zikas [58, Lemma 8] is for all protocols, the proof presented there uses an implicit assumption that for an invocation of broadcast with sender P_s, the adversary is aware of the first subset of $\mathcal{P} \setminus \{P_s\}$ of size $t-1$, which receives information about the input that P_s is attempting to broadcast, and the actual round in which this occurs. An analogue of this property can also be defined for computationally secure protocols, where information might be available to a set but *computationally inaccessible*. In fact, all published dishonest-majority broadcast protocols have such a "release" round, which is not only defined, but also publicly known by the protocol structure; e.g., where the sender sends its input to everyone in the first round (e.g., [35,44]), the first round is actually this public round. We denote this class of protocols as $\Pi_{\mathsf{step\text{-}rel}}$ (see Definition 9 below).

In our treatment of simulation-based security in Sect. 5.1, we provide an argument, inspired by the MPC literature, which allows us to extend our simulation-based impossibility to arbitrary protocols, i.e., beyond the class $\Pi_{\mathsf{step\text{-}rel}}$ (see Step 2 in the proof of Theorem 5). We note in passing that this argument can easily be adapted to complete the argument of Hirt and Zikas [58, Lemma 8]. However, it turns out that the class $\Pi_{\mathsf{step\text{-}rel}}$ is even more relevant in the property-based setting. Therefore, we next formally specify this class and prove our impossibility results for all protocols that satisfy it.

For any given protocol π in the correlated-randomness model, any subset of parties $\hat{\mathcal{P}} \subseteq \mathcal{P}$, and any round ρ, let $\text{VIEW}^{\rho}_{\pi,\hat{\mathcal{P}}}(x,\kappa)$ denote the joint view of the parties in $\hat{\mathcal{P}}$ at the beginning of round ρ in an honest execution (i.e., without the adversary corrupting anyone) on sender-input x, where κ is the security parameter. In particular, $\text{VIEW}^{1}_{\pi,\hat{\mathcal{P}}}(\cdot)$ consists of the inputs and the setup (including randomness) of all parties in $\hat{\mathcal{P}}$ at the beginning of the protocol (before any message is exchanged). For simplicity—to capture also randomized protocols with non-simultaneous termination—we allow the view to be defined even after a party terminates: if for some $P \in \hat{\mathcal{P}}$, party P terminates in some round $\rho \leq R$ (where R is the upper bound of the protocol's round complexity guaranteed by the *termination* property of Definition 5), then $\text{VIEW}^{R}_{\pi,\hat{\mathcal{P}}}(\cdot)$ includes the view of this party up to termination (round R). We also assume for simplicity (again wlog) that for any such party, its view includes the party's output.

The definition of the class $\Pi_{\text{step-rel}}$ ensures that a round $\hat{r}_\pi$ and a set $\hat{\mathcal{P}}_\pi \subseteq \mathcal{P} \setminus \{P_s\}$ of size $|\hat{\mathcal{P}}_\pi| < \lfloor n/2 \rfloor$ are defined by the protocol, such that the set $\hat{\mathcal{P}}_\pi$ is the first set of parties that are able to learn the actual input and this happens in round $\hat{r}_\pi$; i.e., no other set of parties (of the same size) is able to output the input of the sender based on its view from rounds $1, \ldots, \hat{r}_\pi - 1$. Formally:

Definition 9 (The protocol class $\Pi_{\text{step-rel}}$). *For any protocol π in the class $\Pi_{\text{step-rel}}$, there exists some round number $\hat{r}_\pi$, a set $\hat{\mathcal{P}}_\pi \subseteq \mathcal{P} \setminus \{P_s\}$ of size $|\hat{\mathcal{P}}_\pi| < \lfloor n/2 \rfloor$, and a PPT algorithm $\hat{B}_\pi$ such that the following properties hold:*

1. *There exists a negligible function ν such that for any input x it holds that*

$$\Pr\left[\hat{B}_\pi\left(\text{VIEW}^{\hat{r}_\pi}_{\pi,\hat{\mathcal{P}}_\pi}(x,\kappa)\right) = x\right] \geq 1 - \nu(\kappa).$$

2. *Let D be the input domain of the protocol (the set of possible inputs). If the input x is chosen uniformly at random from D, then the output of the honest parties in the following experiment is $y \neq x$ with noticeable probability:*
   (a) *Initiate the protocol π with sender P_s receiving a uniform input $x \leftarrow D$, and sample and distribute the correlated randomness according to π.*
   (b) *Consider a fail-stop adversary that corrupts the parties in $\hat{\mathcal{P}}_\pi \cup \{P_s\}$ in round $\hat{r}_\pi$ and crashes them before sending their round-$\hat{r}_\pi$ messages.*
   (c) *Have the honest parties complete their protocol and set y to the output of any honest party (e.g., the one with the smallest index).*

We stress that such a set $\hat{P}_\pi$ and round r_π is well defined in the execution of *any* broadcast protocol, not just protocols in $\Pi_{\text{step-rel}}$. This follows directly from the validity property of broadcast—at the beginning only the sender knows the input and at the end everyone outputs it. What makes $\Pi_{\text{step-rel}}$ a subclass of all protocols, is the assumption that $\hat{P}_\pi$ and r_π are defined by the protocol itself (and not at execution time). This seemingly strong restriction is sufficient to capture all published broadcast protocols and is therefore sufficient for the statement we are making in this section, that without assumptions limiting the adaptive corruption ability of the adversary—e.g., atomic multisend or slow corruption [71]—such broadcast protocols are *not* adaptively secure, not even according to the property-based definition.

4.1 Impossibility of Property-Based Adaptively Secure Broadcast

We start by adapting the impossibility result of Hirt and Zikas [58] to work with the property-based definition. In particular, we present a simpler argument than [58] that extends the impossibility to: (1) capture a smaller, Boolean input domain (as opposed to exponential-size domain in [58]), and (2) we show the impossibility with respect to a property-based definition (as opposed to the simulation-based definition in [58]). We also observe that this proof strategy works both for deterministic and randomized protocols assuming any correlated-randomness setup and/or secure data erasures. We note that by Lemma 1 an impossibility of a broadcast protocol according to the property-based definition also rules out such protocols secure according to the simulation-based definition.

Theorem 2. *Let $t > n/2$. Then, there exists no broadcast protocol in the class $\Pi_{\mathsf{step\text{-}rel}}$ (secure according to the property-based definition) tolerating an adaptive, fail-stop PPT t-adversary. The theorem holds both for deterministic and randomized protocols assuming any correlated-randomness setup and/or secure erasures.*

The proof can be found in the full version of the paper [34].

4.2 Property-Based Adaptively Secure Broadcast Protocol

Next, we proceed to show that the property-based definition of broadcast can be realized assuming a time-lock puzzle. The high-level idea is quite simple. The sender hides its message inside a (weak) time-lock puzzle, and uses a corruption-unfair broadcast protocol (e.g., Dolev and Strong [35]) to deliver the puzzle to all parties. The TLP parameters should guarantee that the adversary cannot solve the puzzle before the corruption-unfair broadcast completes.

We start by defining the protocol in a hybrid model where a trusted party is in charge of executing corruption-unfair broadcast, and later proceed to prove a composition theorem that enables securely replacing the trusted party with a corruption-unfair broadcast protocol, e.g., Dolev and Strong [35].

Adaptively Secure Broadcast Given Ideal Corruption-Unfair Broadcast. In the spirit of resource-restricted cryptography, we will not consider arbitrary PPT adversaries, since otherwise the impossibility results from Sect. 4.1 will kick in. Instead we will assume an upper bound on the number of parallel steps an adversary can perform during the protocol's execution.

Definition 10 ((R,T)-bounded adversary). *A PPT adversary $\mathcal{A}$ is (R,T)-bounded if for every $\kappa \in \mathbb{N}$, the maximal depth of a circuit that $\mathcal{A}$ can evaluate within R communication rounds is bounded by $T(\kappa)$.*

Protocol $\pi_{\mathsf{bc\text{-}prop}}(T,\kappa)$

- **Hybrid model:** The protocol is defined in the corruption-unfair broadcast $\mathcal{F}_{\mathsf{ubc}}$-hybrid model, where $\mathcal{F}_{\mathsf{ubc}}$ produces an output within R rounds.
- **Public parameters:** A puzzle $(\mathsf{PGen}, \mathsf{PSol})$ with gap $\varepsilon < 1$, a difficulty parameter T, and the security parameter κ.
- **Private input:** The sender has a private input $m \in \{0,1\}^\kappa$.
- **The protocol:**
  - **Lock:** The sender computes $Z \leftarrow \mathsf{PGen}(T^{1/\varepsilon}, m)$.
  - **Corruption-unfair broadcast:** The sender broadcasts Z via $\mathcal{F}_{\mathsf{ubc}}$.
  - **Recover the output:** Upon receiving Z, each party computes $m = \mathsf{PSol}(Z)$ and outputs m.

Fig. 4. Adaptively secure, property-based broadcast protocol

Theorem 3. *Let $t \leq n$ and let $T(\cdot)$ be a polynomial. Assume that weak time-lock puzzles with gap $\varepsilon < 1$ exist and that corruption-unfair broadcast can be computed in R rounds against an adaptive PPT t-adversary. Then, Protocol $\pi_{\mathsf{bc\text{-}prop}}$ (Fig. 4) is a broadcast protocol (according to Definition 5) that is secure against an (R, T)-bounded adaptive PPT t-adversary.*

The proof of Theorem 3 can be found in the full version of the paper [34].

Realizing Ideal Corruption-Unfair Broadcast. Next, we would like to instantiate $\mathcal{F}_{\mathsf{ubc}}$ with the protocol of Dolev and Strong [35]. However, as discussed in the introduction, standard composition theorems no longer apply in the resource-restricted setting. We therefore prove the following limited composition theorem that is sufficient for instantiating $\mathcal{F}_{\mathsf{ubc}}$ with the protocol of Dolev and Strong [35] in $\pi_{\mathsf{bc\text{-}prop}}$; we leave the quest for a more general composition theorem as an interesting open problem.

Similarly to standard composition theorems (e.g., [21]), given a protocol π in the $\mathcal{F}$-hybrid model and another protocol ρ that realizes $\mathcal{F}$, we wish to argue security for the protocol $\pi^{\mathcal{F}\to\rho}$ where the call to $\mathcal{F}$ is replaced by an invocation of ρ. Given an adversary $\mathcal{A}$ to $\pi^{\mathcal{F}\to\rho}$ we derive an adversary to π by considering the induced adversary to ρ and "replace" the execution of ρ with the induced adversary by an ideal computation of $\mathcal{F}$ with the simulator that is guaranteed to exist by the security of ρ. However, as opposed existing composition theorems, we need to ensure that the simulator does not use too many resources. Many simulation strategies have the simulator run in its head the honest parties along with the adversary; in the following two definitions we capture the requirement that such simulators do not use additional resources.

Definition 11 ((R, T)-bounded protocol). *Let $\rho = (P_1, \ldots, P_n)$ be an n-party protocol. We say that ρ is (R, T)-bounded if for every κ, the maximal depth of a circuit that can be evaluated by any P_i within R communication rounds is bounded by $T(\kappa)$.*

Definition 12 (Resource-respecting simulation). *An (R, T_1)-bounded protocol ρ securely realizes a functionality $\mathcal{F}$ against PPT t-adversaries with* resource-respecting simulation*, if every PPT adversary $\mathcal{A}$ can be simulated by a PPT simulator $\mathcal{S}$, and further, if $\mathcal{A}$ is (R, T_2)-bounded then $\mathcal{S}$ is $(R, T_1 + T_2)$-bounded.*

We are now ready to state the limited composition theorem. The proof can be found in the full version of the paper [34].

Theorem 4. *Let π be a protocol in the $\mathcal{F}$-hybrid model, where $\mathcal{F}$ is invoked exactly once and all communication is conveyed via $\mathcal{F}$ (i.e., the parties do not send any other messages), and assume that π is a broadcast protocol (according to Definition 5) that is secure against (R, T)-bounded adaptive PPT t-adversaries. Let $0 < \alpha < 1$ be a constant and let ρ be an $(R, \alpha \cdot T)$-bounded protocol that realizes $\mathcal{F}$ against PPT t-adversaries with resource-respecting simulation.*

Then, the protocol $\pi^{\mathcal{F}\to\rho}$ that is obtained by replacing the call to $\mathcal{F}$ with an execution of ρ, is a broadcast protocol (according to Definition 5) that is secure against $(R, (1-\alpha)\cdot T)$-bounded PPT t-adversaries.

Note that in the corruption-unfair broadcast protocol π_{ubc} of Dolev and Strong [35], honest parties need only to sign, verify, and send signatures, and further, the simulator essentially runs the code of the honest parties towards the adversary. Consider an instantiation of π_{ubc} with some signature scheme such that the number of sequential steps made by each honest party in the protocol is bounded by $T = T(n, \kappa)$; stated differently, the protocol is (n, T)-bounded (i.e., $R = n$). Let $0 < \alpha < 1$ and denote $T' = \frac{1}{\alpha}T$. By Theorem 3, Protocol $\pi_{\mathsf{bc\text{-}prop}}(T', \kappa)$ is a broadcast protocol (according to Definition 5) that is secure against an (n, T')-bounded adaptive PPT t-adversary. By Theorem 4, the protocol π that is obtained by replacing the call to $\mathcal{F}_{\mathsf{ubc}}$ with an execution of π_{ubc}, is a broadcast protocol (according to Definition 5) that is secure against $(n, (1-\alpha)\cdot T)$-bounded PPT t-adversaries. We derive the following corollary.

Corollary 1. *Assume that weak time-lock puzzles with gap $\varepsilon < 1$ exist, let $t \leq n$, let $0 < \alpha < 1$ be a constant, and let T be a polynomial such that $\pi_{\mathsf{bc\text{-}prop}}(T, \kappa)$ is a broadcast protocol (according to Definition 5) that is secure against an (n, T)-bounded adaptive PPT t-adversary, and that π_{ubc} is an $(n, \alpha T)$-bounded corruption-unfair broadcast protocol.*

Then, the protocol π that is obtained by replacing the call to $\mathcal{F}_{\mathsf{ubc}}$ with an execution of π_{ubc}, is a broadcast protocol (according to Definition 5) given a PKI for digital signatures, that is secure against $(n, (1-\alpha)\cdot T)$-bounded PPT t-adversaries.

5 Simulation-Based Adaptively Secure Broadcast

In this section we analyze the simulation-based definition of broadcast. In Sect. 5.1 we show that the assumptions used in Sect. 4.2 that satisfy the property-based definition are not sufficient to realize the simulation-based definition, and in Sect. 5.2 we show how to overcome the new impossibility via the new notion of non-committing time-lock puzzles.

5.1 Impossibility of Simulation-Based Adaptively Secure Broadcast

We next demonstrate that assuming time-lock puzzles does not help in realizing adaptively secure broadcast according to the simulation-based definition. We remark that our impossibility applies to all (polynomial-time) protocols and not just protocols in the class $\Pi_{\mathsf{step\text{-}rel}}$. This impossibility combined with Corollary 1 demonstrate a separation between the two definitions, property-based and simulation-based, but also the fact that time-lock puzzles are less effective in a simulation-based setting. Intuitively, the reason is that the puzzle is a non-interactive object which has a *binding* property (once handed over, its solution cannot be changed) and a *temporary hiding* property (while the solver works to

solve the puzzle, they cannot distinguish it from a puzzle with another solution). In fact, once one observes these properties, the limits of the strength of TLPs for simulation-based adaptive security becomes less of a surprise, as it resembles analogous issues displayed by primitives with similar properties, such as commitments [23,26] and public-key encryption [73].

Before stating our results we first extend the notion of (R, T)-bounded adversaries to the simulation-based setting, where the adversary can use the computational resources of the environment. We consider the pair of environment $\mathcal{Z}$ and adversary $\mathcal{A}$ to be (R, T)-bounded, meaning that for every $\kappa \in \mathbb{N}$, the maximal depth of a circuit that $\mathcal{Z}$ and $\mathcal{A}$ can jointly evaluate within R communication rounds, is bounded by $T(\kappa)$.

We note that by restricting the joint resources of the environment and the adversary, we actually obtain a stronger impossibility result, since even a weaker distinguisher can distinguish between the real execution and the simulated one. Moreover, the result is in fact even stronger since we do not restrict the simulator to be (R, T)-bounded.

We are now ready to state the impossibility result, showing that even TLPs cannot help circumvent the impossibility of adaptively secure broadcast under simulation-based security. Recall that this impossibility holds for any polynomial-time protocol. Nonetheless, for ease in exposition, we prove the statement in two steps: First we prove it for protocols in the class $\Pi_{\mathsf{step\text{-}rel}}$, and then we extend it to protocols besides this class.

In a nutshell, the first (and most involved) step above is proven by using the fact that, by definition of $\Pi_{\mathsf{step\text{-}rel}}$, in round $\hat{r}_\pi$ the adversary attacking π and corrupting $\hat{\mathcal{P}}_\pi$ has all the information it needs to recover the output (even when the sender is honest). This means that, in order to simulate, the simulator needs to give its adversary this information. But the only way the simulator can ensure this is by asking the functionality $\mathcal{F}_{\mathsf{bc}}$ for the sender's input. This gives rise to the following distinguishing strategy for the environment: Once the environment gets its $\hat{r}_\pi$-round messages, it attempts to flip the output by corrupting the sender and all parties in the set $\hat{\mathcal{P}}_\pi$ defined by class $\Pi_{\mathsf{step\text{-}rel}}$. What complicates things is that, unlike the proof of Theorem 2, the environment cannot set a trap for the simulator by making its choice to corrupt the sender depend on the output of $\hat{B}_\pi$. The reason is that the input (to $\hat{B}_\pi$) view of round $\hat{r}_\pi$ might include TLPs, which the environment cannot quickly solve (within round $\hat{r}_\pi$) by the time it decides whether or not to corrupt the sender and try to flip the output.

Instead the environment does the following: It always, optimistically, corrupts the sender and tries to flip the output; it then uses *input-dependent* check-events to distinguish as follows. If the input is 0 the environment checks that the simulator gave it consistent $\hat{r}_\pi$-round messages by running algorithm $\hat{B}_\pi$;[13] otherwise, if the input is 1 then it checks if the simulator managed to flip the bit by looking at the output of $\mathcal{F}_{\mathsf{bc}}$. As discussed above, the only way the simulator can ensure that the first check succeeds is by asking the functionality $\mathcal{F}_{\mathsf{bc}}$ for the input; however, when this happens, the output of $\mathcal{F}_{\mathsf{bc}}$ gets locked

[13] The environment can take its time running $\hat{B}_\pi$ after the protocol terminates.

which will make it impossible for the simulator to flip the output. Hence, one of the two check events will occur noticeably more frequently in the real than in the ideal world, rendering the protocol insecure. We proceed with formal statement; the proof can be found in the full version of the paper [34].

Theorem 5. *Let $t > n/2$. Then, there exists no broadcast protocol which is secure according to the simulation-based definition and tolerates an adaptive, fail-stop, PPT, t-adversary. The theorem holds both for deterministic and randomized protocols assuming any (even inefficient[14]) correlated-randomness setup and/or secure data erasures, and holds even for (R, T)-bounded environments and adversaries and assuming time-lock puzzles.*

Replacing TLPs with a (Programable) Random Oracle. One can verify that replacing, in Theorem 5, the time-lock puzzle (and the (R,T)-bounded environment assumption) with a random oracle—even a programmable one—does not affect the impossibility. Indeed, the proof of this statement follows the same line of arguments as Theorem 5, where Step 1 is even simpler and uses the simpler attack from Theorem 2—i.e., at round $\hat{r}_\pi$, the environment who corrupts the parties in $\hat{\mathcal{P}}_\pi$ evaluates $\hat{B}_\pi$ (the environment can now do that as it is not (R, T)-bounded) on their view, and depending on the output of $\hat{B}_\pi$, either corrupts the sender and crashes all corrupted parties, or lets the protocol complete. It is easy to verify that the probabilities of the events in the proof will remain (asymptotically) the same, and are not affected by adding a random oracle.

Indeed, the real-world events are defined in a way which does not alter their distribution when a random oracle is assumed; and in the ideal world, programmability cannot help the simulator alter the events, as (1) the environment does not change its behavior depending on the RO, and (2) the ideal-world events depend only on the environment and the behavior of the ideal functionality (which is also independent of the RO). Thus we can derive the following corollary:

Corollary 2. *Let $t > n/2$. Then, there exists no broadcast protocol in the (programable) random-oracle model, which is secure according to the simulation-based definition and tolerates an adaptive, fail-stop, PPT, t-adversary. The theorem holds both for deterministic and randomized protocols assuming any (even inefficient) correlated-randomness setup and/or secure data erasures.*

In passing, we note that a similar corollary can also be derived for the property-based definition. In particular, we can extend the property-based model of execution by allowing all relevant machines (the parties, the adversary, and the challenger from Definition 5) oracle access to a random function. Then, for all such protocols, as long that they satisfy Definition 9, it is straightforward

[14] Classical correlated randomness setup assumes efficient sampling and distribution mechanisms. By removing such restrictions here we can even capture non-programmable random oracle, as an exponential-space correlated randomness functionality that samples the entire random table of the RO.

to verify that all the events involved in the proof of Theorem 2 remain intact. Indeed, the probability of these events is derived directly from Definition 5. This proves the following simple corollary of Theorem 2.

Corollary 3. *Let $t > n/2$. Then, there exists no broadcast protocol in the class $\Pi_{\mathsf{step\text{-}rel}}$ (secure according to the property-based definition) tolerating an adaptive, fail-stop PPT t-adversary in the random-oracle model. The theorem holds both for deterministic and randomized protocols, and assuming any correlated-randomness setup and/or secure erasures.*

5.2 Simulation-Based Adaptively Secure Broadcast Protocol

The main reason why the protocol from Sect. 4.2 does not realize the simulation-based definition is that once the simulator simulates an honest sender broadcasting the puzzle Z (without knowing the real input value), it cannot equivocate the content of the puzzle upon corruption of the sender, or when the protocol completes and the output is revealed. We now proceed to construct an adaptively secure broadcast protocol that satisfies the simulation-based definition in the programmable random-oracle model. First off, we introduce the notion of time-lock puzzles that are *non-committing*.

Non-committing Time-Lock Puzzles. Standard constructions of time-lock puzzles are committing in the sense that once a puzzle is generated, it can be opened into a unique message with all but negligible probability. In contrast, a *non-committing* time-lock puzzle enables a simulator to initially simulate a puzzle, and later, given an arbitrary message m, to "explain" the puzzle as containing m. We show how to achieve this notion given a standard time-lock puzzle and a programmable random oracle, by generating $Z = \mathsf{PGen}(T, x)$ for a random $x \leftarrow \{0,1\}^\kappa$ and attaching $c = H(x) \oplus m$ to the puzzle. Once the simulator is asked to equivocate the new puzzle (Z, c) to the message m, it can program the random oracle to return $H(x) = c \oplus m$. We note that a similar idea was used in [5,9] to model TLPs in the UC framework.

We proceed to state the theorem. The proof can be found in the full version of the paper [34].

Theorem 6. *Assume that weak TLPs with gap $\varepsilon < 1$ exist and that corruption-unfair broadcast can be computed in R rounds against an adaptive, PPT t-adversary, for $t \leq n$. Let $T(\cdot)$ be a polynomial. Then, Protocol $\pi_{\mathsf{bc\text{-}sim}}$ (Fig. 5) is a broadcast protocol according to the simulation-based definition (Definition 7) that is secure against an adaptive t-adversary in the programmable random-oracle model, where the adversary and the environment are PPT and (R, T)-bounded.*

Similarly to Theorem 4, we prove a limited composition theorem for the simulation-based setting. The proof can be found in the full version of the paper [34].

Theorem 7. *Let π be a protocol in the $\mathcal{F}$-hybrid model, where $\mathcal{F}$ is invoked exactly once and all communication is conveyed via $\mathcal{F}$ (i.e., the parties do not send any other messages) that realizes a functionality $\mathcal{G}$ against an adaptive t-adversary, where the adversary and the environment are PPT and (R, T)-bounded. Let $0 < \alpha < 1$ be a constant, and let ρ be an $(R, \alpha \cdot T)$-bounded protocol that realizes $\mathcal{F}$ against PPT t-adversaries with resource-respecting simulation.*

Then, the protocol $\pi^{\mathcal{F} \to \rho}$ that is obtained by replacing the call to $\mathcal{F}$ with an execution of ρ, realizes $\mathcal{G}$ against an adaptive t-adversary, where the adversary and the environment are PPT and $(R, (1-\alpha) \cdot T)$-bounded.

Protocol $\pi_{\mathsf{bc\text{-}sim}}(T, \kappa)$

- **Hybrid model:** The protocol is defined in the corruption-unfair broadcast $\mathcal{F}_{\mathsf{ubc}}$-hybrid model, requiring R rounds. The parties have access to a random oracle $H : \{0,1\}^\kappa \to \{0,1\}^\kappa$.
- **Public parameters:** A puzzle $(\mathsf{PGen}, \mathsf{PSol})$ with gap $\varepsilon < 1$, a difficulty parameter T, and the security parameter κ.
- **Private input:** The sender has a private input $m \in \{0,1\}^\kappa$.
- **The protocol:**
  - **Lock:** The sender samples a random $x \leftarrow \{0,1\}^\kappa$ and computes $Z \leftarrow \mathsf{PGen}(T^{1/\varepsilon}, x)$.
  - **Corruption-unfair broadcast:** The sender sets $c = m \oplus H(x)$ and broadcasts (Z, c) via $\mathcal{F}_{\mathsf{ubc}}$.
  - **Output:** Upon receiving (Z, c), compute $x = \mathsf{PSol}(Z)$ and outputs $c \oplus H(x)$.

Fig. 5. Adaptively secure, simulation-based broadcast protocol

Consider an instantiation of the corruption-unfair broadcast protocol π_{ubc} of Dolev and Strong [35] with some signature scheme such that the protocol is (n, T)-bounded. Let $0 < \alpha < 1$ and denote $T' = \frac{1}{\alpha} T$. By Theorem 6, Protocol $\pi_{\mathsf{bc\text{-}sim}}(T', \kappa)$ is a broadcast protocol (according to Definition 7) that is secure against an (n, T')-bounded adaptive PPT t-adversary. By Theorem 7, the protocol π that is obtained by replacing the call to $\mathcal{F}_{\mathsf{ubc}}$ with an execution of π_{ubc}, is a broadcast protocol (according to Definition 7) secure against $(n, (1-\alpha) \cdot T)$-bounded PPT t-adversaries. We therefore derive the following corollary.

Corollary 4. *Assume that weak TLP with gap $\varepsilon < 1$ exist, let $t \leq n$, let $0 < \alpha < 1$ be a constant, and let T be a polynomial such that $\pi_{\mathsf{bc\text{-}sim}}(T, \kappa)$ is a broadcast protocol that is secure against an (n, T')-bounded adaptive PPT t-adversary, and that π_{ubc} is an $(n, \alpha T)$-bounded corruption-unfair broadcast protocol.*

Then, the protocol π that is obtained by replacing the call to $\mathcal{F}_{\mathsf{ubc}}$ with an execution of π_{ubc}, is a broadcast protocol (according to Definition 7) in the programmable random-oracle model and given a PKI for digital signatures, that is secure against $(n, (1-\alpha) \cdot T)$-bounded PPT t-adversaries.

Acknowledgments. Ran Cohen's research is supported in part by NSF grant no. 2055568. Juan Garay's research is supported in part by NSF grants no. 2001082 and

2055694. Vassilis Zikas's research is supported in part by NSF grant no. 2055599 and by Sunday Group. The authors are also supported by the Algorand Centres of Excellence programme managed by Algorand Foundation. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of Algorand Foundation.

References

1. Abraham, I., et al.: Communication complexity of Byzantine agreement, revisited. In: 38th ACM PODC, pp. 317–326 (2019)
2. Abraham, I., Devadas, S., Dolev, D., Nayak, K., Ren, L.: Synchronous byzantine agreement with expected $O(1)$ rounds, expected $O(n^2)$ communication, and optimal resilience. In: Goldberg, I., Moore, T. (eds.) FC 2019. LNCS, vol. 11598, pp. 320–334. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-32101-7_20
3. Alexandru, A.B., Loss, J., Papamanthou, C., Tsimos, G.: Sublinear-round broadcast without trusted setup against dishonest majority. Cryptology ePrint Archive, Report 2022/1383 (2022). https://eprint.iacr.org/2022/1383
4. Andrychowicz, M., Dziembowski, S.: PoW-based distributed cryptography with no trusted setup. In: Gennaro, R., Robshaw, M. (eds.) CRYPTO 2015, Part II. LNCS, vol. 9216, pp. 379–399. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48000-7_19
5. Arapinis, M., Lamprou, N., Zacharias, T.: Astrolabous: a universally composable time-lock encryption scheme. In: Tibouchi, M., Wang, H. (eds.) ASIACRYPT 2021, Part II. LNCS, vol. 13091, pp. 398–426. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-92075-3_14
6. Badertscher, C., Maurer, U., Tschudi, D., Zikas, V.: Bitcoin as a transaction ledger: a composable treatment. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017, Part I. LNCS, vol. 10401, pp. 324–356. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63688-7_11
7. Badertscher, C., Gazi, P., Kiayias, A., Russell, A., Zikas, V.: Ouroboros genesis: composable proof-of-stake blockchains with dynamic availability. In: ACM CCS 2018, pp. 913–930 (2018)
8. Badertscher, C., Canetti, R., Hesse, J., Tackmann, B., Zikas, V.: Universal composition with global subroutines: capturing global setup within plain UC. In: Pass, R., Pietrzak, K. (eds.) TCC 2020, Part III. LNCS, vol. 12552, pp. 1–30. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-64381-2_1
9. Baum, C., David, B., Dowsley, R., Nielsen, J.B., Oechsner, S.: TARDIS: a foundation of time-lock puzzles in UC. In: Canteaut, A., Standaert, F.-X. (eds.) EUROCRYPT 2021, Part III. LNCS, vol. 12698, pp. 429–459. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-77883-5_15
10. Baum, C., David, B., Dowsley, R., Kishore, R., Nielsen, J.B., Oechsner, S.: CRAFT: composable randomness beacons and output-independent abort MPC from time. In: Boldyreva, A., Kolesnikov, V. (eds.) PKC 2023, Part I. LNCS, vol. 13940, pp. 439–470. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-31368-4_16
11. Bitansky, N., Goldwasser, S., Jain, A., Paneth, O., Vaikuntanathan, V., Waters, B.: Time-lock puzzles from randomized encodings. In: ITCS 2016, pp. 345–356 (2016)
12. Blum, E., Katz, J., Loss, J.: Synchronous consensus with optimal asynchronous fallback guarantees. In: Hofheinz, D., Rosen, A. (eds.) TCC 2019, Part I. LNCS,

vol. 11891, pp. 131–150. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-36030-6_6
13. Blum, E., Katz, J., Liu-Zhang, C.-D., Loss, J.: Asynchronous byzantine agreement with subquadratic communication. In: Pass, R., Pietrzak, K. (eds.) TCC 2020, Part I. LNCS, vol. 12550, pp. 353–380. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-64375-1_13
14. Boneh, D., Naor, M.: Timed commitments. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 236–254. Springer, Heidelberg (2000). https://doi.org/10.1007/3-540-44598-6_15
15. Boneh, D., Bonneau, J., Bünz, B., Fisch, B.: Verifiable delay functions. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018, Part I. LNCS, vol. 10991, pp. 757–788. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-96884-1_25
16. Borderding, M.: Levels of authentication in distributed agreement. In: 10th International Workshop on Distributed Algorithms WDAG, pp. 40–55 (1996)
17. Boyle, E., Cohen, R., Data, D., Hubáček, P.: Must the communication graph of MPC protocols be an expander? In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018, Part III. LNCS, vol. 10993, pp. 243–272. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-96878-0_9
18. Brakerski, Z., Döttling, N., Garg, S., Malavolta, G.: Leveraging linear decryption: rate-1 fully-homomorphic encryption and time-lock puzzles. In: Hofheinz, D., Rosen, A. (eds.) TCC 2019, Part II. LNCS, vol. 11892, pp. 407–437. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-36033-7_16
19. Camenisch, J., Drijvers, M., Gagliardoni, T., Lehmann, A., Neven, G.: The wonderful world of global random oracles. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018, Part I. LNCS, vol. 10820, pp. 280–312. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-78381-9_11
20. Canetti, R.: Security and composition of multiparty cryptographic protocols. J. Cryptol. **13**(1), 143–202 (2000)
21. Canetti, R.: Universally composable security: a new paradigm for cryptographic protocols. In: 42nd FOCS, pp. 136–145 (2001)
22. Canetti, R.: Universally composable security. J. ACM **67**(5), 28:1–28:94 (2020)
23. Canetti, R., Fischlin, M.: Universally composable commitments. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 19–40. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44647-8_2
24. Canetti, R., Feige, U., Goldreich, O., Naor, M.: Adaptively secure multi-party computation. In: 28th ACM STOC, pp. 639–648 (1996)
25. Canetti, R., Lindell, Y., Ostrovsky, R., Sahai, A.: Universally composable two-party and multi-party secure computation. In: 34th ACM STOC, pp. 494–503 (2002)
26. Canetti, R., Damgård, I., Dziembowski, S., Ishai, Y., Malkin, T.: Adaptive versus non-adaptive security of multi-party protocols. J. Cryptol. **17**(3), 153–207 (2004)
27. Canetti, R., Cohen, A., Lindell, Y.: A simpler variant of universally composable security for standard multiparty computation. In: Gennaro, R., Robshaw, M. (eds.) CRYPTO 2015, Part II. LNCS, vol. 9216, pp. 3–22. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48000-7_1
28. Chan, T.-H.H., Pass, R., Shi, E.: Sublinear-round byzantine agreement under corrupt majority. In: Kiayias, A., Kohlweiss, M., Wallden, P., Zikas, V. (eds.) PKC 2020, Part II. LNCS, vol. 12111, pp. 246–265. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-45388-6_9
29. Chen, J., Micali, S.: Algorand: a secure and efficient distributed ledger. Theoret. Comput. Sci. **777**, 155–183 (2019)

30. Cleve, R.: Limits on the security of coin flips when half the processors are faulty (extended abstract). In: 18th ACM STOC, pp. 364–369 (1986)
31. Cohen, R., Coretti, S., Garay, J., Zikas, V.: Probabilistic termination and composability of cryptographic protocols. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016, Part III. LNCS, vol. 9816, pp. 240–269. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53015-3_9
32. Cohen, R., Coretti, S., Garay, J.A., Zikas, V.: Round-preserving parallel composition of probabilistic-termination cryptographic protocols. In: ICALP 2017. LIPIcs, vol. 80, pp. 37:1–37:15. Schloss Dagstuhl (2017)
33. Cohen, R., Shelat, A., Wichs, D.: Adaptively secure MPC with sublinear communication complexity. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019, Part II. LNCS, vol. 11693, pp. 30–60. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26951-7_2
34. Cohen, R., Garay, J., Zikas, V.: Completeness theorems for adaptively secure broadcast. Cryptology ePrint Archive, Report 2021/775 (2021). https://eprint.iacr.org/2021/775
35. Dolev, D., Strong, H.R.: Authenticated algorithms for Byzantine agreement. SIAM J. Comput. **12**(4), 656–666 (1983)
36. Dwork, C., Naor, M.: Pricing via processing or combatting junk mail. In: Brickell, E.F. (ed.) CRYPTO 1992. LNCS, vol. 740, pp. 139–147. Springer, Heidelberg (1993). https://doi.org/10.1007/3-540-48071-4_10
37. Eckey, L., Faust, S., Loss, J.: Efficient algorithms for broadcast and consensus based on proofs of work. Cryptology ePrint Archive, Report 2017/915 (2017). http://eprint.iacr.org/2017/915
38. Feldman, P.: Optimal Algorithms for Byzantine Agreement. Ph.D. thesis, Stanford University (1988). https://dspace.mit.edu/handle/1721.1/14368
39. Fischer, M.J., Lynch, N.A.: A lower bound for the time to assure interactive consistency. Inf. Process. Lett. **14**(4), 183–186 (1982)
40. Fischer, M.J., Lynch, N.A., Merritt, M.: Easy impossibility proofs for distributed consensus problems. Distrib. Comput. **1**(1), 26–39 (1986)
41. Fitzi, M.: Generalized communication and security models in Byzantine agreement. Ph.D. thesis, ETH Zurich, Zürich, Switzerland (2003). http://d-nb.info/967397375
42. Freitag, C., Komargodski, I., Pass, R., Sirkin, N.: Non-malleable time-lock puzzles and applications. In: Nissim, K., Waters, B. (eds.) TCC 2021, Part III. LNCS, vol. 13044, pp. 447–479. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-90456-2_15
43. Garay, J., Kiayias, A.: SoK: a consensus taxonomy in the blockchain era. In: Jarecki, S. (ed.) CT-RSA 2020. LNCS, vol. 12006, pp. 284–318. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-40186-3_13
44. Garay, J.A., Moses, Y.: Fully polynomial Byzantine agreement in t+1 rounds. In: 25th ACM STOC, pp. 31–41 (1993)
45. Garay, J., MacKenzie, P., Prabhakaran, M., Yang, K.: Resource fairness and composability of cryptographic protocols. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 404–428. Springer, Heidelberg (2006). https://doi.org/10.1007/11681878_21
46. Garay, J.A., Katz, J., Koo, C.-Y., Ostrovsky, R.: Round complexity of authenticated broadcast with a dishonest majority. In: 48th FOCS, pp. 658–668. IEEE Computer Society Press (2007)
47. Garay, J.A., Katz, J., Kumaresan, R., Zhou, H.-S.: Adaptively secure broadcast, revisited. In: 30th ACM PODC, pp. 179–186 (2011)

48. Garay, J., Kiayias, A., Leonardos, N.: The bitcoin backbone protocol: analysis and applications. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015, Part II. LNCS, vol. 9057, pp. 281–310. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46803-6_10
49. Garay, J., Ishai, Y., Ostrovsky, R., Zikas, V.: The price of low communication in secure multi-party computation. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017, Part I. LNCS, vol. 10401, pp. 420–446. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63688-7_14
50. Garay, J., Kiayias, A., Ostrovsky, R.M., Panagiotakos, G., Zikas, V.: Resource-restricted cryptography: revisiting MPC bounds in the proof-of-work era. In: Canteaut, A., Ishai, Y. (eds.) EUROCRYPT 2020. LNCS, vol. 12106, pp. 129–158. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-45724-2_5
51. Garg, S., Sahai, A.: Adaptively secure multi-party computation with dishonest majority. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 105–123. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-32009-5_8
52. Goldreich, O.: Foundations of Cryptography: Basic Applications, vol. 2. Cambridge University Press, Cambridge (2004)
53. Goldreich, O., Micali, S., Wigderson, A.: How to play any mental game or A completeness theorem for protocols with honest majority. In: 19th ACM STOC, pp. 218–229. ACM Press (1987)
54. Goldwasser, S., Lindell, Y.: Secure multi-party computation without agreement. J. Cryptol. **18**(3), 247–287 (2005)
55. Goldwasser, S., Kalai, Y.T., Park, S.: Adaptively secure coin-flipping, revisited. In: Halldórsson, M.M., Iwama, K., Kobayashi, N., Speckmann, B. (eds.) ICALP 2015, Part II. LNCS, vol. 9135, pp. 663–674. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-47666-6_53
56. Haitner, I., Karidi-Heller, Y.: A tight lower bound on adaptively secure full-information coin flip. In: 61st FOCS, pp. 1268–1276 (2020)
57. Hazay, C., Lindell, Y., Patra, A.: Adaptively secure computation with partial erasures. In: 34th ACM PODC, pp. 291–300 (2015)
58. Hirt, M., Zikas, V.: Adaptively secure broadcast. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 466–485. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13190-5_24
59. Hofheinz, D., Müller-Quade, J.: A synchronous model for multi-party computation and the incompleteness of oblivious transfer. Cryptology ePrint Archive, Report 2004/016 (2004). http://eprint.iacr.org/2004/016
60. Kalai, Y.T., Komargodski, I., Raz, R.: A lower bound for adaptively-secure collective coin-flipping protocols. In: DISC, pp. 34:1–34:16 (2018)
61. Katz, J., Maurer, U., Tackmann, B., Zikas, V.: Universally composable synchronous computation. In: Sahai, A. (ed.) TCC 2013. LNCS, vol. 7785, pp. 477–498. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-36594-2_27
62. Katz, J., Thiruvengadam, A., Zhou, H.-S.: Feasibility and infeasibility of adaptively secure fully homomorphic encryption. In: Kurosawa, K., Hanaoka, G. (eds.) PKC 2013. LNCS, vol. 7778, pp. 14–31. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-36362-7_2
63. Katz, J., Loss, J., Xu, J.: On the security of time-lock puzzles and timed commitments. In: Pass, R., Pietrzak, K. (eds.) TCC 2020, Part III. LNCS, vol. 12552, pp. 390–413. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-64381-2_14

64. Khorasgani, H.A., Maji, H.K., Mukherjee, T.: Estimating gaps in martingales and applications to coin-tossing: constructions and hardness. In: Hofheinz, D., Rosen, A. (eds.) TCC 2019, Part II. LNCS, vol. 11892, pp. 333–355. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-36033-7_13
65. Kiayias, A., Zhou, H.-S., Zikas, V.: Fair and robust multi-party computation using a global transaction ledger. In: Fischlin, M., Coron, J.-S. (eds.) EUROCRYPT 2016, Part II. LNCS, vol. 9666, pp. 705–734. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49896-5_25
66. Lamport, L., Shostak, R.E., Pease, M.C.: The Byzantine generals problem. ACM Trans. Program. Lang. Syst. **4**(3), 382–401 (1982)
67. Lin, H., Pass, R., Soni, P.: Two-round and non-interactive concurrent non-malleable commitments from time-lock puzzles. In: 58th FOCS, pp. 576–587 (2017)
68. Liu-Zhang, C.-D., Maurer, U.: Synchronous constructive cryptography. In: Pass, R., Pietrzak, K. (eds.) TCC 2020, Part II. LNCS, vol. 12551, pp. 439–472. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-64378-2_16
69. Mahmoody, M., Moran, T., Vadhan, S.: Time-lock puzzles in the random oracle model. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 39–50. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22792-9_3
70. Malavolta, G., Thyagarajan, S.A.K.: Homomorphic time-lock puzzles and applications. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019, Part I. LNCS, vol. 11692, pp. 620–649. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26948-7_22
71. Matt, C., Nielsen, J.B., Thomsen, S.E.: Formalizing delayed adaptive corruptions and the security of flooding networks. In: Dodis, Y., Shrimpton, T. (eds.) CRYPTO 2022, Part II. LNCS, vol. 13508, pp. 400–430. Springer, Heidelberg (2022). https://doi.org/10.1007/978-3-031-15979-4_14
72. Micali, S., Rogaway, P.: Secure computation. In: Feigenbaum, J. (ed.) CRYPTO 1991. LNCS, vol. 576, pp. 392–404. Springer, Heidelberg (1992). https://doi.org/10.1007/3-540-46766-1_32
73. Nielsen, J.B.: Separating random oracle proofs from complexity theoretic proofs: the non-committing encryption case. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 111–126. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45708-9_8
74. Nielsen, J.B.: On protocol security in the cryptographic model. Ph.D. thesis, University of Aarhus (2003). https://www.brics.dk/DS/03/8/BRICS-DS-03-8.pdf
75. Pass, R., Seeman, L., Shelat, A.: Analysis of the blockchain protocol in asynchronous networks. In: Coron, J.-S., Nielsen, J.B. (eds.) EUROCRYPT 2017, Part II. LNCS, vol. 10211, pp. 643–673. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-56614-6_22
76. Pease, M.C., Shostak, R.E., Lamport, L.: Reaching agreement in the presence of faults. J. ACM **27**(2), 228–234 (1980)
77. Pfitzmann, B., Waidner, M.: Unconditional Byzantine agreement for any number of faulty processors. In: Finkel, A., Jantzen, M. (eds.) STACS 1992. LNCS, vol. 577, pp. 337–350. Springer, Heidelberg (1992). https://doi.org/10.1007/3-540-55210-3_195
78. Pietrzak, K.: Simple verifiable delay functions. In: ITCS 2019, vol. 124, pp. 60:1–60:15 (2019)
79. Rivest, R.L., Shamir, A., Wagner, D.A.: Time-lock puzzles and timed-release crypto. Technical report, Massachusetts Institute of Technology, USA (1996)

80. Rotem, L., Segev, G.: Generically speeding-up repeated squaring is equivalent to factoring: sharp thresholds for all generic-ring delay functions. In: Micciancio, D., Ristenpart, T. (eds.) CRYPTO 2020, Part III. LNCS, vol. 12172, pp. 481–509. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-56877-1_17
81. Srinivasan, S., Loss, J., Malavolta, G., Nayak, K., Papamanthou, C., Thyagarajan, S.A.K.: Transparent batchable time-lock puzzles and applications to byzantine consensus. In: Boldyreva, A., Kolesnikov, V. (eds.) PKC 2023, Part I. LNCS, vol. 13940, pp. 554–584. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-31368-4_20
82. Tsimos, G., Loss, J., Papamanthou, C.: Gossiping for communication-efficient broadcast. In: Dodis, Y., Shrimpton, T. (eds.) CRYPTO 2022, Part III. LNCS, vol. 13509, pp. 439–469. Springer, Heidelberg (2022). https://doi.org/10.1007/978-3-031-15982-4_15
83. Wan, J., Xiao, H., Devadas, S., Shi, E.: Round-efficient byzantine broadcast under strongly adaptive and majority corruptions. In: Pass, R., Pietrzak, K. (eds.) TCC 2020, Part I. LNCS, vol. 12550, pp. 412–456. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-64375-1_15
84. Wan, J., Xiao, H., Shi, E., Devadas, S.: Expected constant round byzantine broadcast under dishonest majority. In: Pass, R., Pietrzak, K. (eds.) TCC 2020, Part I. LNCS, vol. 12550, pp. 381–411. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-64375-1_14
85. Wesolowski, B.: Efficient verifiable delay functions. In: Ishai, Y., Rijmen, V. (eds.) EUROCRYPT 2019, Part III. LNCS, vol. 11478, pp. 379–407. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-17659-4_13
86. Yao, A.C.-C.: Protocols for secure computations (extended abstract). In: 23rd FOCS, pp. 160–164. IEEE Computer Society Press (1982)