Completeness Theorems for Adaptively Secure Broadcast

Ran Cohen* Juan Garay† Vassilis Zikas‡

Abstract

The advent of blockchain protocols has reignited the interest in adaptively secure broadcast, as 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., that this task is impossible against an adaptive adversary corrupting a strict 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. Nonetheless, 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.

*Efi Arazi School of Computer Science, Reichman University. E-mail: cohenran@runi.ac.il.

†Texas A&M University. E-mail: garay@cse.tamu.edu.

‡Purdue University. E-mail: vzikas@cs.purdue.edu.

Contents

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 [76, 65]) 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 $t < n/3$ [76, 43], a larger corruption threshold cannot be tolerated [65, 39, 16]. 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 [34] (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 any number of $t \leq n$ corruptions.[1]

Simulation-based vs. property-based definitions. Broadcast can be thought of as a concrete instance of secure multi-party computation (MPC) [86, 52]. 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, 51]. 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 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, 53]) 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.

[1] For the related *consensus* problem (aka Byzantine *agreement*), where all parties have an input, the best achievable bound is $t < n/2$ [40].

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 [57], this is no longer true 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 [34] (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 [57] 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]

Hirt and Zikas [57] 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 [57] 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* [57] *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]

[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* [54, 59, 63, 55].

[3] Although in this work we focus on broadcast, corruption fairness can easily be defined—and is a natural

Atomic vs. non-atomic multisend. The attack from [57] 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., [38, 34, 39]), where the adversary could corrupt a party and make it "crash" (or change its input [37]) 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., [24, 20, 21, 27]). The ability of the adversary to corrupt a party in such a manner has also been referred to as *strongly rushing* [2, 1, 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 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 [46], 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 [34] (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., [29, 1, 28, 84, 13, 82]), 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 to work with as it admits more powerful adversaries; indeed, certain impossibility results in the non-atomic multisend model do not translate to the atomic-multisend regime [57, 17, 1]. 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. The point we are making 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 [65, 39, 16] without using "private-state setup assumptions"[7] (such as a PKI) is the *resource-restricted cryptography* (RRC) paradigm [49], where instead of considering arbitrary

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.

[5] We stress that the above is orthogonal to the synchrony assumption: Consider for example a synchronous setting, where a round takes 60 seconds (i.e., any message sent by an honest party is delivered after at most 60 seconds) and corrupting a party takes 30 seconds. 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 the non-atomic multisend model 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 [42].

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 [47, 75] employs proofs of work (PoWs) [35] 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 [49] that can tolerate any dishonest minority, given only a "public-state setup."

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) [79, 11] 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" [44], non-interactive non-malleable commitments [66], 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* [57] *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 [57] 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 [57]). 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 the *right* property-based definition of adaptively secure broadcast?"[8]; rather, any such definition that

[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.

	property-based	simulation-based
PKI	✗* **Thm 1**	✗ HZ [57]
PKI+RO	✗* **Cor 16**	✗ **Cor 15**
PKI+TLP	✓ **Thm 2**	✗ **Thm 4**
PKI+TLP+RO	✓ **Thm 2**	✓ **Thm 5**

Table 1: Feasibility of adaptively secure broadcast with non-atomic multisend, synchronous communication. The left column considers the property-based definition and the right one the simulation-based definition. 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 10. 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 stands for the programmable random-oracle model.

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* [71], 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 [57] 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 property:*

- ***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

[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.

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 additional power of this definition compared to the weaker definition (that guarantees *agreement* and *validity*, but not *corruption-fairness*) 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 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 9.

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 [57]? The attack from [57] 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 [57] 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, 83, 84, 82]. 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

[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.

[11] In most broadcast protocols from the literature (e.g., [34, 45, 28, 83, 84]), the sender starts by sending its input to all parties, meaning that $R = 1$.

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. We include a more detailed discussion on the breadth of the impossibility result in Appendix A.

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 1 (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 be used to overcome the impossibility of adaptively secure broadcast. We use time-lock puzzles [79, 11] 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., [34, 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] the TLP was hidden for a duration of a round, whereas we hide it for the duration of the entire protocol; see Section 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 [68, 11] 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 2 (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 Section 5.1 we extend this argument to rule out *any* adaptively secure broadcast protocol even facing an (R, T)-bounded adversary. In turn, this impossibility result implies that the TLP assumption is not sufficient for realizing simulation-based broadcast.

Theorem 4 (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 [72]. 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 [9, 5] 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 proven realizable from computational hardness assumptions, rather than requiring, as [9, 5] 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. Such a treatment turns out to be non-trivial and we believe can be of independent interest.

Theorem 5 (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 impossibility of simulation-based security 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 4 to hold when we replace TLPs with an (even programable) RO (see Corollary 15). 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 1 (see Corollary 16).

Composition in resource-restricted settings. The protocols in our positive results (Theorems 2 and 5) 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 [34] 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 3 and 6) 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 [34], 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 [57]. 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 polylogarithmic, 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 [34].

We note that although the protocol in [83] relies on similar assumptions as the ones in this work, it **does not** answer the question posed in this paper as it is vulnerable to the attack from [46], showing that the Dolev-Strong protocol (DS) [34] 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 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

[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 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 [46].

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 [79, 11] (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. [70] 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 [70] 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 [72] 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, 49].

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.

Finally, we note that some of our ideas are reminiscent of the complexity leveraging technique as used in [74]; specifically, in [74] super-polynomial simulators extracted from non-interactive commitments by a brute force computation, somewhat similarly to the way the environment extracts from the TLP in our lower bounds.

Organization of the paper. Section 2 presents the model and the cryptographic primitives that are used in this paper. In Section 3 we present the property-based and simulation-based definitions of broadcast and of corruption-unfair broadcast. In Section 4 we analyze the property-based definition, presenting the impossibility result and the protocol construction from time-lock puzzles. Finally, Section 5 treats the simulation-based definition, separating it from the property-based definition, and showing how to realize it in the programmable random-oracle model. The composition theorems for RRC are presented in Sections 4.2 and 5.2.

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 [46] 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., [29, 1, 28, 84, 13, 82]).
- 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., [38, 34, 39, 37]) and cryptographic protocols [24, 20, 21, 27]. This models has also been referred to as *strongly adaptive* [54, 59, 63, 55] and *strongly rushing* [2, 1, 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 network. 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 [57, 17, 1]. In fact, as proven by Katz et al. [60], 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 [24, 20] 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., [72, 50, 56, 48]), other impossibility results are stronger and do not rely on the absence of secure erasures (e.g., [57, 61, 17, 33]).

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 Section 3.2).

We present our results in a *synchronous* model with an online distinguisher (aka, the *environment*); this is the prevalent model in many frameworks for cryptographic protocols; see, e.g., [21, 73, 58, 64, 7, 67, 8, 9]. 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 [60, 64, 8].

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., [44]. 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., [14, 44, 68, 11, 66, 36, 69, 18, 80, 62, 83, 41, 9, 3, 81]. 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 [68, 11]. 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* $s_0, s_1 \in \{0,1\}^\kappa$*:*

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

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. [68] 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 5. *[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 the sender's message. This is traditionally captured via the *agreement* and *validity* properties. However, as observed in Hirt and Zikas [57], 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 [57] 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 [57], 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 Figure 1.

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.

Figure 1: The corruption-fairness experiment for adaptively secure broadcast

Note that, as observed in [57], the protocol of Dolev and Strong [34] (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 [46]). The broadcast protocols from [57, 31, 32] satisfy this property for $t \leq n/2$ in the standard (non-atomic-multisend) model, and similarly, the protocol from [46] for $t < n$ in the atomic-multisend model.

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 Figure 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 [57, 46]. 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 [57], 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 $m_{\mathsf{out}} := \bot$ 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 $\mathsf{isOutputLocked} := \mathsf{true}$.

Figure 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, defined in Figure 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.

Figure 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).

Lemma 9. *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.*

Proof. Assume that π satisfies Definition 7 but does not satisfy Definition 5. If *termination*, *agreement*, or *validity* are not satisfied, then we immediately derive a contradiction; therefore, we assume that the *corruption-fairness* property is not satisfied. This means that there exists a PPT adversary

$\mathcal{A}'$ that can win $\mathsf{Expt}^{\mathsf{fair\text{-}bcast}}_{\pi,\mathcal{A}'}(\kappa)$ (i.e., bias the output to 0^κ while keeping the sender honest given 0^κ as input) with probability $1/2 + \mu(\kappa)$ for a non-negligible μ.

Consider the environment $\mathcal{Z}$ and adversary $\mathcal{A}$ defined as follows: First, $\mathcal{Z}$ chooses a random bit $b \leftarrow \{0,1\}$ and activates the sender with input b^κ. Next, $\mathcal{A}$ attacks the protocol execution by proceeding according to the operations of $\mathcal{A}'$. Finally, the environment checks whether when $b = 0$ the sender remained honest, and when $b = 1$ the output of an arbitrary honest party is 0^κ; if so, the environment outputs 1 (real) and otherwise 0 (ideal). By construction, the environment outputs 1 when interacting with the real-world protocol with probability $1/2 + \mu(\kappa)$.

By Definition 7, there exists a PPT simulator $\mathcal{S}$ that can simulate this attack. However, by definition of $\mathcal{F}_{\mathsf{bc}}$, the simulator can exactly simulate one of the two actions below, but not both.

- Learn the input without corrupting the sender; in this case if the input is 1^κ the adversary can no longer influence the output to be 0^k.
- Corrupt the sender without first learning the input value and set the output to be 0^κ; in this case if the input is 0^κ the sender does not remain honest.

It follows that for any PPT simulator, the environment will output 1 when interacting with the ideal-world computation with probability $1/2 + \mathsf{negl}(\kappa)$. This leads to a contradiction to the assumption that π satisfies Definition 7. □

4 Property-based Adaptively Secure Broadcast

In this section we analyze the property-based definition of adaptively secure broadcast. In Section 4.1 we extend the impossibility result of Hirt and Zikas [57] to this regime, and in Section 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 [57, 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., [34, 43]), the first round is actually this public round. We denote the class of protocols with such a step-release structure as $\Pi_{\mathsf{step\text{-}rel}}$ (see Definition 10 below).

In our treatment of simulation-based security in Section 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 4). We note in passing that this argument can easily be adapted to complete the argument of Hirt and Zikas [57, Lemma 8]. However, it turns out that the class $\Pi_{\mathsf{step\text{-}rel}}$ is even more relevant in the property-based setting.[13] 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 $\textsc{view}^{\rho}_{\pi,\hat{\mathcal{P}}}(x,\kappa)$ 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, $\textsc{view}^{1}_{\pi,\hat{\mathcal{P}}}(\cdot)$ consists of the inputs and the setup (including

[13] For the interested reader, the source of this discrepancy is discussed in Appendix A.

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 will 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 to exist 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 will also assume for simplicity (again without loss of generality) that for any such party, its view includes the party's output.

The definition of the class $\Pi_{\mathsf{step\text{-}rel}}$ 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 10 (The protocol class $\Pi_{\mathsf{step\text{-}rel}}$)**.** *For any protocol π in the class $\Pi_{\mathsf{step\text{-}rel}}$, 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 broadcast 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 uniformly distributed 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 $\Pi_{\mathsf{step\text{-}rel}}$. 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 $\Pi_{\mathsf{step\text{-}rel}}$ 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 [70]—such broadcast protocols are *not* adaptively secure, not even according to the property-based definition. We include a more detailed discussion on the choice and generality of $\Pi_{\mathsf{step\text{-}rel}}$ in Appendix A.

4.1 Impossibility of Property-based Adaptively Secure Broadcast

We start by adapting the impossibility result of Hirt and Zikas [57] to work with the property-based definition. In particular, we present a simpler argument than [57] that extends the impossibility to: (1) capture a smaller, Boolean input domain (as opposed to exponential-size domain in [57]), and (2)

we show the impossibility with respect to a property-based definition (as opposed to the simulation-based definition in [57]). 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 9 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 1. *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.*

Proof. Without loss of generality, we prove this statement for Boolean broadcast. Assume towards a contradiction that π is a Boolean broadcast protocol (according to the property-based definition) with sender P_s, tolerating an adaptive PPT t-adversary. By classical impossibility results [65, 39, 16], if $t \geq n/3$ then π cannot be defined in the plain model (even assuming standard cryptographic hardness assumptions), and some form of setup is required. That is, we consider a trusted dealer that samples correlated randomness $(\mathtt{r}_1, \ldots, \mathtt{r}_n) \leftarrow D_\pi$ from some efficiently sampleable distribution D_π, and privately hands each P_i the string $\mathtt{r}_i$. Without loss of generality, assume that the random coins used by each party are defined within $\mathtt{r}_i$, so the transcript and the view of each party are random variables over the probability space defined by the random coins used for sampling from D_π and by the random choice of the input bit $x \leftarrow \{0,1\}$.

By Definition 10 there exist a round number $\hat{r}_\pi$, a set $\hat{\mathcal{P}}_\pi$, and an algorithm $\hat{B}_\pi$ for the protocol π. The adversary $\mathcal{A}$ that breaks the corruption-fairness property of Definition 5 is defined as follows:

1. For rounds $1, \ldots, \hat{r}_\pi - 1$ the adversary corrupts no party.
2. At the beginning of round $\hat{r}_\pi$: $\mathcal{A}$ corrupts all the parties in $\hat{\mathcal{P}}_\pi$, and uses its rushing ability to deliver all the messages that parties outside of $\hat{\mathcal{P}}_\pi$ send to the corrupted parties. This way, $\text{VIEW}^{\hat{r}_\pi}_{\pi,\hat{\mathcal{P}}_\pi}$ includes all messages that are in those parties' view of an honest execution at round $\hat{r}_\pi$.
3. $\mathcal{A}$ computes $\hat{y} \leftarrow \hat{B}_\pi(\text{VIEW}^{\hat{r}_\pi}_{\pi,\hat{\mathcal{P}}_\pi})$ and proceeds as follows:
   - **if** $\hat{y} = 1$, then $\mathcal{A}$ corrupts also the sender P_s and crashes all corrupted parties—so that the last messages received by parties in $\mathcal{P} \setminus (\hat{\mathcal{P}}_\pi \cup \{P_s\})$ were the ones received at round $\hat{r}_\pi - 1$.
   - **else** (i.e., if $\hat{y} = 0$) the adversary does not corrupt P_s and allows all parties to continue playing their protocol.

Next, we show that adversary $\mathcal{A}$ violates the corruption-fairness property of Definition 5. In slight abuse of notation, but without loss of generality, we will use $\mathcal{P}$ to denote the broadcast-protocol parties simulated by the Challenger in $\mathsf{Expt}^{\mathsf{fair\text{-}bcast}}_{\pi,\mathcal{A}}(\kappa)$, and denote the sender by P_s. First we observe that the input b that the Challenger uses in its simulation of broadcast towards $\mathcal{A}$ is distributed uniformly. Hence, the views that $\mathcal{A}$ and Challenger witness in the corruption-fairness experiment up to round $\hat{r}_\pi$ are distributed identically as in an honest execution of π, and, therefore, the properties of the class $\Pi_{\mathsf{step\text{-}rel}}$ hold for the interaction between them.

We consider the following events in the simulated execution of π between the Challenger and $\mathcal{A}$:

- $\mathcal{E}_{\mathtt{b=0}}$ occurs when the Challenger chooses $b = 0$.
- $\mathcal{E}_{\mathtt{b=1}}$ occurs when the Challenger chooses $b = 1$.
- $\mathcal{E}_{\mathtt{h}}$ occurs if the sender P_s is honest at the end of the simulated protocol execution.
- $\mathcal{E}_{\mathtt{c}}$ occurs if the sender P_s gets corrupted before the end of the simulated protocol execution.

– $\mathcal{E}_{\texttt{flip}}$ occurs if on sender input x, some party that is honest until it terminates, outputs $y = 1-x$ in the simulated execution. (Note that by *agreement* this implies that all honest parties will output y except with negligible probability.)

Note that the events $\mathcal{E}_{\texttt{b=0}} \wedge \mathcal{E}_{\texttt{h}}$ and $\mathcal{E}_{\texttt{b=1}} \wedge \mathcal{E}_{\texttt{c}} \wedge \mathcal{E}_{\texttt{flip}}$ are disjoint, therefore:

$$\begin{aligned}\Pr\left[\mathsf{Expt}^{\mathsf{fair\text{-}bcast}}_{\pi,\mathcal{A}}(\kappa) = 1\right] &= \Pr\Big[(\mathcal{E}_{\texttt{b=0}} \wedge \mathcal{E}_{\texttt{h}}) \vee (\mathcal{E}_{\texttt{b=1}} \wedge \mathcal{E}_{\texttt{c}} \wedge \mathcal{E}_{\texttt{flip}})\Big] \\ &= \Pr\Big[\mathcal{E}_{\texttt{b=0}} \wedge \mathcal{E}_{\texttt{h}}\Big] + \Pr\Big[\mathcal{E}_{\texttt{b=1}} \wedge \mathcal{E}_{\texttt{c}} \wedge \mathcal{E}_{\texttt{flip}}\Big]. \end{aligned} \tag{1}$$

Next, we observe that when the input bit is $b = 0$, then the adversary $\mathcal{A}$ corrupts the sender and flips the output bit only with negligible probability (i.e., the probability that $\hat{B}_\pi$ outputs the wrong value 1, which is negligible for the protocols in the class we are considering). Hence:

$$\Pr\Big[\mathcal{E}_{\texttt{b=0}} \wedge \mathcal{E}_{\texttt{h}}\Big] = \Pr\Big[\mathcal{E}_{\texttt{h}} \mid \mathcal{E}_{\texttt{b=0}}\Big] \cdot \Pr\Big[\mathcal{E}_{\texttt{b=0}}\Big] = \frac{1}{2} \cdot \Pr\Big[\mathcal{E}_{\texttt{h}} \mid \mathcal{E}_{\texttt{b=0}}\Big] \geq \frac{1}{2} \cdot (1 - \nu(\kappa)), \tag{2}$$

for some negligible function $\nu(\cdot)$.

Similarly, by the second property of the protocol class $\Pi_{\mathsf{step\text{-}rel}}$, the above adversary makes the honest parties output $1 - b$ with noticeable probability. Hence:

$$\begin{aligned}\Pr\Big[\mathcal{E}_{\texttt{b=1}} \wedge \mathcal{E}_{\texttt{c}} \wedge \mathcal{E}_{\texttt{flip}}\Big] &= \Pr\Big[\mathcal{E}_{\texttt{c}} \wedge \mathcal{E}_{\texttt{flip}} \mid \mathcal{E}_{\texttt{b=1}}\Big] \cdot \Pr\Big[\mathcal{E}_{\texttt{b=1}}\Big] \\ &= \frac{1}{2} \cdot \Pr\Big[\mathcal{E}_{\texttt{c}} \wedge \mathcal{E}_{\texttt{flip}}\Big] \\ &\geq \frac{1}{2} \cdot \Big(1 - \mu(\kappa)\Big) \cdot q(\kappa), \end{aligned} \tag{3}$$

for some noticeable function $q : \mathbb{N} \to [0, 1]$ and some negligible function $\mu(\cdot)$.

Putting the above together we get that:

$$\begin{aligned}\Pr\left[\mathsf{Expt}^{\mathsf{fair\text{-}bcast}}_{\pi,\mathcal{A}}(\kappa) = 1\right] &= \Pr\Big[(\mathcal{E}_{\texttt{b=0}} \wedge \mathcal{E}_{\texttt{h}})] + \Pr[(\mathcal{E}_{\texttt{b=1}} \wedge \mathcal{E}_{\texttt{c}} \wedge \mathcal{E}_{\texttt{flip}})\Big] \\ &\geq \frac{1}{2} \cdot (1 - \nu(\kappa)) + \frac{1}{2} \cdot \Big(1 - \mu(\kappa)\Big) \cdot q(\kappa) \\ &= \frac{1}{2} + \frac{q(\kappa)}{2} - \frac{\nu(\kappa) + \mu(\kappa) \cdot q(\kappa)}{2}. \end{aligned} \tag{4}$$

Since $q(\kappa)$ is not negligible, nor is $q(\kappa)/2$. Furthermore, since $\mu(\kappa)$ and $\nu(\kappa)$ are negligible and $q(\cdot) \leq 1$, it holds that $\mu(\kappa) \cdot q(\kappa)$ is also negligible; hence, so is $\nu(\kappa) + \mu(\kappa) \cdot q(\kappa)$. Therefore,

$$\frac{q(\kappa)}{2} - \frac{\nu(\kappa) + \mu(\kappa) \cdot q(\kappa)}{2}$$

is a noticeable function, which means that Equation 4 contradicts the corruption-fairness property of the broadcast definition. □

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 [34]) 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.

In Section 4.2.1 we define the protocol in a hybrid model where a trusted party is in charge of executing corruption-unfair broadcast. Later, in Section 4.2.2 we prove a composition theorem that enables securely replacing the trusted party with a corruption-unfair broadcast protocol, e.g., Dolev and Strong [34].

4.2.1 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 Section 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 11 (**(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 $T(\kappa)$.*

Theorem 2. *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 $\pi_{\mathsf{bc\text{-}prop}}$ (Figure 4) is a broadcast protocol (according to Definition 5) that is secure against an (R,T)-bounded adaptive PPT t-adversary.*

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.

Figure 4: Adaptively secure, property-based broadcast protocol

Proof. We prove each property separately.

Termination. Every honest party is guaranteed to receive a puzzle from $\mathcal{F}_{\mathsf{ubc}}$ within R rounds, and therefore is guaranteed to produce an output and terminate after a polynomial number of steps needed to open the puzzle.

Agreement. By definition of $\mathcal{F}_{\mathsf{ubc}}$, all honest parties are guaranteed to receive the same Z, and therefore, by the correctness of the time-lock puzzle, are guaranteed to output the same message with overwhelming probability.

Validity. Assume the sender is honest at the end of the execution and is set with input m. By definition of $\mathcal{F}_{\mathsf{ubc}}$, all honest parties are guaranteed to receive Z. Therefore, by the correctness of the time-lock puzzle, all honest parties will compute $m = \mathsf{PSol}(Z)$ with overwhelming probability.

Corruption-fairness. Let $\mathcal{A}$ be an (R,T)-bounded PPT adversary that wins $\mathsf{Expt}^{\mathsf{fair\text{-}bcast}}_{\pi,\mathcal{A}}(\kappa)$ with probability $1/2+\mu(\kappa)$ for a non-negligible μ. Note that the only information that $\mathcal{A}$ receives during the execution of the protocol is the puzzle Z (since the views of non-sender parties that get corrupted contain no information about the input bit), and based on this information $\mathcal{A}$ has to decide whether to corrupt the sender and flip its bit, i.e., decide whether the puzzle Z contains 1^κ. Since $\mathcal{A}$ is (R,T)-bounded and has to decide whether to corrupt the sender within R rounds, the algorithm that $\mathcal{A}$ computes can be represented as a depth $T(\kappa)$ circuit. However, since the difficulty parameter of the puzzle is set to $T^{1/\varepsilon}$, it follows that $\mathcal{A}$ can be used to break the security of the time-lock puzzle, according to Definition 2. □

4.2.2 Realizing Ideal Corruption-Unfair Broadcast

Next, we would like to instantiate $\mathcal{F}_{\mathsf{ubc}}$ with the protocol of Dolev and Strong [34]. 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 [34] 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 12 (**(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 13 (**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.

Theorem 3. *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.

Proof. Let $\mathcal{A}$ be an $(R,(1-\alpha)\cdot T)$-bounded PPT t-adversary, and let $\varepsilon(\kappa)$ denote the maximal probability with which $\pi^{\mathcal{F}\to\rho}$ does *not* satisfy agreement, validity, or corruption-fairness (where the probability is over the random coins of the adversary, the protocol, and potentially the challenger (in

the case of corruption-fairness)). That is, agreement holds except for probability $\varepsilon_{\mathsf{agree}}(\kappa) \leq \varepsilon(\kappa)$, validity holds except for probability $\varepsilon_{\mathsf{valid}}(\kappa) \leq \varepsilon(\kappa)$, and $\mathcal{A}$ wins the experiment $\mathsf{Expt}^{\mathsf{fair\text{-}bcast}}_{\pi,\mathcal{A}}(\kappa)$ with probability at most $1/2 + \varepsilon_{\mathsf{fair}}(\kappa)$ with $\varepsilon_{\mathsf{fair}}(\kappa) \leq \varepsilon(\kappa)$. We will prove that $\varepsilon(\kappa)$ is negligible.

Since the only communication in π is via $\mathcal{F}$, and there is only one invocation of $\mathcal{F}$, it holds that $\mathcal{A}$ can be split into the part of $\mathcal{A}$ interacting with ρ, denoted $\mathcal{A}_\rho$, and the part of $\mathcal{A}$ interacting with the caller part of ρ (with respect to $\pi^{\mathcal{F}\to\rho}$), denoted $\mathcal{A}_{\pi^{\mathcal{F}\to\rho}\backslash\rho}$. Further, there exists $0 \leq \beta \leq 1-\alpha$ such that $\mathcal{A}_\rho$ is $(R, \beta \cdot T)$-bounded.

First, by the assumption that ρ is $(R, \alpha \cdot T)$-bounded and realizes $\mathcal{F}$ against PPT t-adversaries with resource-respecting simulation, and since $\mathcal{A}_\rho$ is $(R, \beta \cdot T)$-bounded, there exists a PPT $(R, (\alpha+\beta) \cdot T)$-bounded simulator $\mathcal{S}_\rho$ such that for any PPT environment $\mathcal{Z}$, and in particular $\mathcal{A}_{\pi^{\mathcal{F}\to\rho}\backslash\rho}$, the distinguishing probability between the execution of ρ with the adversary $\mathcal{A}_\rho$ and the ideal computation of $\mathcal{F}$ with the simulator $\mathcal{S}_\rho$ is negligible. Let $\varepsilon_\rho(\kappa)$ denote the negligible distinguishing probability of $\mathcal{A}_{\pi^{\mathcal{F}\to\rho}\backslash\rho}$.

Second, note that $\mathcal{A}_{\pi^{\mathcal{F}\to\rho}\backslash\rho}$ acts identically on the caller part of ρ (with respect to $\pi^{\mathcal{F}\to\rho}$) and on the caller part of $\mathcal{F}$ (with respect to π). Therefore, the adversary $\mathcal{A}$ induces an adversary $\mathcal{A}_\pi$ for the protocol π (in the $\mathcal{F}$-hybrid model) by replacing the interface of the protocol ρ by the interface of the ideal functionality $\mathcal{F}$, and by invoking the simulator $\mathcal{S}_\rho$ for interacting with $\mathcal{A}_{\pi^{\mathcal{F}\to\rho}\backslash\rho}$. Recall that $\mathcal{A}_\rho$ is $(R, \beta \cdot T)$-bounded and $\mathcal{A}_{\pi^{\mathcal{F}\to\rho}\backslash\rho}$ together with $\mathcal{A}_\rho$ is $(R, (1-\alpha) \cdot T)$-bounded; since $\mathcal{S}_\rho$ is $(R, (\alpha+\beta) \cdot T)$-bounded, it holds that the maximal depth that $\mathcal{A}_\pi$ can evaluate in R rounds is

$$(1-\alpha)T(\kappa) - \beta \cdot T(\kappa) + (\alpha+\beta) \cdot T(\kappa) = T(\kappa);$$

that is, $\mathcal{A}_\pi$ is (R, T)-bounded. Let $\varepsilon_\pi(\kappa)$ denote the maximal probability with which π does *not* satisfy agreement, validity, or corruption-fairness when interacting with $\mathcal{A}_\pi$; by the assumed security of π it holds that $\varepsilon_\pi(\kappa)$ is negligible.

The theorem now follows since $\varepsilon(\kappa) \leq \varepsilon_\pi(\kappa) + \varepsilon_\rho(\kappa)$, which is negligible. □

Note that in the corruption-unfair broadcast protocol π_{ubc} of Dolev and Strong [34], 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 2, 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 3, 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 therefore derive the following corollary.

Corollary 14. *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 Section 5.1 we show that the assumptions used in Section 4.2 that satisfy the property-based definition are not sufficient

to realize the simulation-based definition, and in Section 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 14 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 [72].

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 1, 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 $\hat{B}_\pi$;[14] otherwise, if the input is 1 then it checks if the simulator managed to flip

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

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 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 and proof.

Theorem 4. *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[15]) correlated-randomness setup and/or secure data erasures, and holds even for (R, T)-bounded environments and adversaries and assuming time-lock puzzles.*

Proof. To make the exposition easier to follow, we will prove the statement in two steps: First we prove it for protocols from the class $\Pi_{\mathsf{step\text{-}rel}}$ from Definition 10, and then we show how to extend this argument to an arbitrary protocol.

Step 1. Impossibility for protocols from $\Pi_{\mathsf{step\text{-}rel}}$. Let π be a broadcast protocol from the class $\Pi_{\mathsf{step\text{-}rel}}$. As in the proof of Theorem 1 we consider a (trusted dealer that samples a) correlated-randomness setup $(\mathtt{r}_1, \ldots, \mathtt{r}_n) \leftarrow D_\pi$ from some distribution D_π, and privately hands each P_i the string $\mathtt{r}_i$. Without loss of generality, assume that the random coins used by each party are defined within $\mathtt{r}_i$, so the transcript and the view of each party are random variables over the probability space defined by the random coins used for sampling from D_π and by the random choice of the input bit $x \leftarrow \{0, 1\}$.

By Definition 10 there exist a round index $\hat{r}_\pi$, a set $\hat{\mathcal{P}}_\pi$, and an algorithm $\hat{B}_\pi$ for the protocol π. One might be tempted to play the same strategy as in the proof of Theorem 1—i.e., at round $\hat{r}_\pi$, the adversary who corrupts the parties in $\hat{\mathcal{P}}_\pi$ evaluates $\hat{B}_\pi$ 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. Unfortunately, this attack might not work in this setting, as the definition of $\hat{B}_\pi$ makes no restriction on the (parallel) time that it takes to compute its output other than that this time is polynomial. Thus, an (R, T)-bounded adversary/environment pair might not be able to evaluate $\hat{B}_\pi$ before round $\hat{r}_\pi$ finishes. This is, for example, the case if the view of the parties in $\hat{\mathcal{P}}_\pi$ includes a freshly generated time-lock puzzle, as in the protocol $\pi_{\mathsf{bc\text{-}prop}}$ (Figure 4), that requires multiple rounds to be solved by an (R, T)-bounded adversary.

Thus, we need a different strategy for the environment. Consider the following two (R, T)-bounded environments $\mathcal{Z}_0$ and $\mathcal{Z}_1$: $\mathcal{Z}_b$ gives the sender input b with probability 1 and works as follows: It instantiates an execution with a *dummy adversary* [21], namely, an adversary that simply follows the environment's instructions. It then proceeds is as follows:

1. For rounds $1, \ldots, \hat{r}_\pi - 1$ it (instructs the adversary to) corrupts no party.
2. At the beginning of round $\hat{r}_\pi$: The environment tells its dummy adversary $\mathcal{A}$ to corrupt all the parties in $\hat{\mathcal{P}}_\pi$, and use its rushing ability to deliver all the messages that parties outside of $\hat{\mathcal{P}}_\pi$ send to the corrupted parties. This way, $\text{VIEW}^{\hat{r}_\pi}_{\pi, \hat{\mathcal{P}}_\pi}$ includes all messages that are in those parties' view of an honest execution at round $\hat{r}_\pi$.
3. After receiving all $\hat{r}_\pi$-round messages from all honest parties running the protocol (or from the simulator in the ideal world) $\mathcal{Z}_b$ instructs $\mathcal{A}$ to corrupt also the sender P_s and crash

[15] 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.

all corrupted parties, including the sender, so that the last messages received by parties in $\mathcal{P} \setminus (\hat{\mathcal{P}}_\pi \cup \{P_s\})$ were the ones received at round $\hat{r}_\pi - 1$.

4. $\mathcal{Z}_b$ allows the protocol to terminate and records the output y of any honest party.
5. Finally, $\mathcal{Z}_b$ takes its time after the protocol has outputted to evaluate $\hat{b} = \hat{B}_\pi(\text{VIEW}^{\hat{r}_\pi}_{\pi,\hat{\mathcal{P}}_\pi})$.[16]
6. The output of $\mathcal{Z}_b$ is then computed as follows:
   - If $b = 0$: output 0 (real) if $\hat{b} = b$ and output 1 (ideal) otherwise;
   - if $b = 1$: output 0 (real) if $y = 0$ and output 1 (ideal) otherwise.

Next, we show that any simulator that successfully simulates against $\mathcal{Z}_0$ will fail to simulate against $\mathcal{Z}_1$. To make the statement even stronger, we do not even restrict the simulator to be (R, T)-bounded. Indeed, let $\mathcal{S}$ be any simulator in the $\mathcal{F}_{\mathsf{bc}}$-ideal experiment for the dummy adversary.

We consider the following events in the real execution of π with this environment $\mathcal{Z}$ and the dummy adversary$\mathcal{A}$ (recall that b stands for the input bit, $\hat{b}$ for the output of $\hat{B}_\pi$, and y for the common output bit):

- $\mathcal{E}^{\mathcal{Z},\mathcal{A},\pi}_{b=\hat{b}}$ occurs when in the real experiment $b = \hat{b}$.
- $\mathcal{E}^{\mathcal{Z},\mathcal{A},\pi}_{b=1}$ occurs when in the real experiment $b = 1$.
- $\mathcal{E}^{\mathcal{Z},\mathcal{A},\pi}_{b=0}$ occurs when in the real experiment $b = 0$.
- $\mathcal{E}^{\mathcal{Z},\mathcal{A},\pi}_{b=1\neq y}$ occurs when in the real experiment $b = 1$ and $y = 0$.

By definition of the events, for the environments $\mathcal{Z}_0$ and $\mathcal{Z}_1$ we have:

$$\Pr\left[\mathcal{E}^{\mathcal{Z}_0,\mathcal{A},\pi}_{b=0}\right] = \Pr\left[\mathcal{E}^{\mathcal{Z}_1,\mathcal{A},\pi}_{b=1}\right] = 1. \tag{5}$$

Additionally, by correctness of the protocol π and the definition of the class $\Pi_{\mathsf{step\text{-}rel}}$, there exists a negligible function μ such that:

$$\Pr\left[\mathcal{E}^{\mathcal{Z}_0,\mathcal{A},\pi}_{b=\hat{b}}\right] = 1 - \mu(\kappa). \tag{6}$$

Since $\mathcal{Z}_0$ always uses input bit $b = 0$, it always uses the first condition to determine the output, i.e., output 0 (real) if and only if $b = \hat{b}$. Hence, for the output $\text{REAL}_{\mathcal{Z}_0,\mathcal{A},\pi}(\kappa)$ of the environment $\mathcal{Z}_0$ in the real experiment with (dummy) adversary $\mathcal{A}$, the output is 1 (ideal) with negligible probability:

$$\Pr\Big[\text{REAL}_{\mathcal{Z}_0,\mathcal{A},\pi}(\kappa) = 1\Big] = \Pr\Big[\ \overline{\mathcal{E}^{\mathcal{Z}_0,\mathcal{A},\pi}_{b=\hat{b}}}\ \Big] = \mu(\kappa). \tag{7}$$

Additionally, for the environment $\mathcal{Z}_1$, by definition of the class $\Pi_{\mathsf{step\text{-}rel}}$ when every party in $\hat{\mathcal{P}}_\pi$ crashes before sending their $\hat{r}_\pi$-round message, then the output of the honest parties is flipped with noticeable probability. That is, there exists a noticeable function q such that.

$$\Pr\left[\mathcal{E}^{\mathcal{Z}_1,\mathcal{A},\pi}_{b=1\neq y}\right] = q(\kappa). \tag{8}$$

Since $\mathcal{Z}_1$ always uses input bit $b = 1$, it always uses the second condition to determine the output, i.e., output 0 (real) if and only if $y = 0$. Hence:

$$\Pr\Big[\text{REAL}_{\mathcal{Z}_1,\mathcal{A},\pi}(\kappa) = 0\Big] = \Pr\left[\mathcal{E}^{\mathcal{Z}_1,\mathcal{A},\pi}_{b=1\neq y}\right] = q(\kappa). \tag{9}$$

Let us now turn to the ideal experiment. Consider the following events in the ideal execution of π with environment $\mathcal{Z}$ and the simulator $\mathcal{S}$:

[16] Observe that the fact that the environment is (R, T)-bounded restricts how fast it can compute the output of $\hat{B}_\pi$ but since $\hat{B}_\pi$ is a polynomial-time algorithm, a polynomial-time environment will be able to compute it within its runtime.

- $\mathcal{E}^{\mathcal{Z},\mathcal{S},\mathcal{F}_{\mathsf{bc}}}_{b=\hat{b}}$ occurs when in the ideal experiment $b = \hat{b}$.
- $\mathcal{E}^{\mathcal{Z},\mathcal{S},\mathcal{F}_{\mathsf{bc}}}_{b=0}$ occurs when in the ideal experiment $b = 0$.
- $\mathcal{E}^{\mathcal{Z},\mathcal{S},\mathcal{F}_{\mathsf{bc}}}_{b=1}$ occurs when in the ideal experiment $b = 1$.
- $\mathcal{E}^{\mathcal{Z},\mathcal{S},\mathcal{F}_{\mathsf{bc}}}_{b=1\neq y}$ occurs when in the ideal experiment $b = 1$ and $y = 0$.
- $\mathcal{E}^{\mathcal{Z},\mathcal{S},\mathcal{F}_{\mathsf{bc}}}_{\mathsf{rush}}$ which occurs when the simulator asks $\mathcal{F}_{\mathsf{bc}}$ for the output while the sender is still honest and before sending the $\hat{r}_\pi$-round simulated messages to the environment or receiving the output from the functionality $\mathcal{F}_{\mathsf{bc}}$.

As above, we have:

$$\Pr\left[\mathcal{E}^{\mathcal{Z}_0,\mathcal{S},\mathcal{F}_{\mathsf{bc}}}_{b=0}\right] = \Pr\left[\mathcal{E}^{\mathcal{Z}_1,\mathcal{S},\mathcal{F}_{\mathsf{bc}}}_{b=1}\right] = 1. \tag{10}$$

We next observe that the event $\mathcal{E}^{\mathcal{Z},\mathcal{S},\mathcal{F}_{\mathsf{bc}}}_{\mathsf{rush}}$ occurs with the same probability for $\mathcal{Z}_0$ and $\mathcal{Z}_1$. The reason is that this event is triggered by the simulator based on its view of the (ideal) execution before it gets any information from $\mathcal{F}_{\mathsf{bc}}$ and while $\mathcal{Z}_b$ (asks its adversary to) behave according to the protocol. Indeed, up to round $\hat{r}_\pi$, the environment $\mathcal{Z}_b$ behaves identically to $\mathcal{Z}_{\bar{b}}$ and independently of b. Thus, for some function $f_{S,\pi,\mathcal{F}_{\mathsf{bc}}}(\kappa)$ it holds that:

$$\Pr\left[\ \overline{\mathcal{E}^{\mathcal{Z}_1,\mathcal{S},\mathcal{F}_{\mathsf{bc}}}_{\mathsf{rush}}}\ \right] = \Pr\left[\ \overline{\mathcal{E}^{\mathcal{Z}_0,\mathcal{S},\mathcal{F}_{\mathsf{bc}}}_{\mathsf{rush}}}\ \right] = f_{S,\pi,\mathcal{F}_{\mathsf{bc}}}(\kappa). \tag{11}$$

We next observe that when the simulator asks for the value before corrupting the sender (and hence before learning any information about the input), by definition of $\mathcal{F}_{\mathsf{bc}}$ the output gets locked to this (input) value and cannot be flipped even if $\mathcal{S}$ later corrupts the sender. Hence:

$$\Pr\left[\mathcal{E}^{\mathcal{Z}_1,\mathcal{S},\mathcal{F}_{\mathsf{bc}}}_{b=1\neq y} \;\middle|\; \mathcal{E}^{\mathcal{Z}_1,\mathcal{S},\mathcal{F}_{\mathsf{bc}}}_{\mathsf{rush}}\right] = 0. \tag{12}$$

Therefore, for the output $\text{IDEAL}_{\mathcal{Z}_1,\mathcal{S},\mathcal{F}_{\mathsf{bc}}}(\kappa)$ of the environment $\mathcal{Z}_1$ in the ideal experiment with simulator $\mathcal{S}$ we have:

$$\begin{aligned}
&\Pr\Big[\text{IDEAL}_{\mathcal{Z}_1,\mathcal{S},\mathcal{F}_{\mathsf{bc}}}(\kappa) = 0\Big] = \Pr\left[\mathcal{E}^{\mathcal{Z}_1,\mathcal{S},\mathcal{F}_{\mathsf{bc}}}_{b=1\neq y}\right] \\
&= \Pr\left[\mathcal{E}^{\mathcal{Z}_1,\mathcal{S},\mathcal{F}_{\mathsf{bc}}}_{b=1\neq y} \mid \mathcal{E}^{\mathcal{Z}_1,\mathcal{S},\mathcal{F}_{\mathsf{bc}}}_{\mathsf{rush}}\right] \cdot \Pr\left[\mathcal{E}^{\mathcal{Z}_1,\mathcal{S},\mathcal{F}_{\mathsf{bc}}}_{\mathsf{rush}}\right] + \Pr\left[\ \mathcal{E}^{\mathcal{Z}_1,\mathcal{S},\mathcal{F}_{\mathsf{bc}}}_{b=1\neq y} \mid \overline{\mathcal{E}^{\mathcal{Z}_1,\mathcal{S},\mathcal{F}_{\mathsf{bc}}}_{\mathsf{rush}}}\ \right] \cdot \Pr\left[\ \overline{\mathcal{E}^{\mathcal{Z}_1,\mathcal{S},\mathcal{F}_{\mathsf{bc}}}_{\mathsf{rush}}}\ \right] \\
&\overset{\text{Eq.12}}{=} \Pr\left[\ \mathcal{E}^{\mathcal{Z}_1,\mathcal{S},\mathcal{F}_{\mathsf{bc}}}_{b=1\neq y} \mid \overline{\mathcal{E}^{\mathcal{Z}_1,\mathcal{S},\mathcal{F}_{\mathsf{bc}}}_{\mathsf{rush}}}\ \right] \cdot \Pr\left[\ \overline{\mathcal{E}^{\mathcal{Z}_1,\mathcal{S},\mathcal{F}_{\mathsf{bc}}}_{\mathsf{rush}}}\ \right] \\
&\overset{\text{Eq.11}}{\leq} f_{S,\pi,\mathcal{F}_{\mathsf{bc}}}(\kappa).
\end{aligned} \tag{13}$$

Moreover, when the simulator does not ask for the output value before corrupting the sender, the $\hat{r}_\pi$-round messages are independent of the actual input value b. Since b is chosen randomly and $\hat{b}$ is the outcome of algorithm $\hat{B}_\pi$ on a view independent of b, it holds that:

$$\max_{\mathcal{Z}\in\{\mathcal{Z}_0,\mathcal{Z}_1\}} \left\{\Pr\left[\ \overline{\mathcal{E}^{\mathcal{Z},\mathcal{S},\mathcal{F}_{\mathsf{bc}}}_{b=\hat{b}}} \mid \overline{\mathcal{E}^{\mathcal{Z},\mathcal{S},\mathcal{F}_{\mathsf{bc}}}_{\mathsf{rush}}}\ \right]\right\} \geq \frac{1}{2}. \tag{14}$$

Without loss of generality assume that the above holds for $\mathcal{Z} = \mathcal{Z}_0$ (otherwise one simply needs to flip the role of the bit in the argument, since the simulator's code before querying the functionality cannot depend on the input bit), i.e.,

$$\Pr\left[\ \overline{\mathcal{E}^{\mathcal{Z}_0,\mathcal{S},\mathcal{F}_{\mathsf{bc}}}_{b=\hat{b}}} \mid \overline{\mathcal{E}^{\mathcal{Z}_0,\mathcal{S},\mathcal{F}_{\mathsf{bc}}}_{\mathsf{rush}}}\ \right] \geq \frac{1}{2}. \tag{15}$$

Thus,

$$\begin{aligned}
&\Pr\left[\text{IDEAL}_{\mathcal{Z}_0,\mathcal{S},\mathcal{F}_{\mathsf{bc}}}(\kappa)=1\right] = \Pr\left[\ \overline{\mathcal{E}^{\mathcal{Z}_0,\mathcal{S},\mathcal{F}_{\mathsf{bc}}}_{b=\hat{b}}}\ \right] \\
&= \Pr\left[\ \overline{\mathcal{E}^{\mathcal{Z}_0,\mathcal{S},\mathcal{F}_{\mathsf{bc}}}_{b=\hat{b}}} \mid \mathcal{E}^{\mathcal{Z}_0,\mathcal{S},\mathcal{F}_{\mathsf{bc}}}_{\mathsf{rush}}\ \right]\cdot\Pr\left[\mathcal{E}^{\mathcal{Z}_0,\mathcal{S},\mathcal{F}_{\mathsf{bc}}}_{\mathsf{rush}}\right] + \Pr\left[\ \overline{\mathcal{E}^{\mathcal{Z}_0,\mathcal{S},\mathcal{F}_{\mathsf{bc}}}_{b=\hat{b}}} \mid \overline{\mathcal{E}^{\mathcal{Z}_0,\mathcal{S},\mathcal{F}_{\mathsf{bc}}}_{\mathsf{rush}}}\ \right]\cdot\Pr\left[\ \overline{\mathcal{E}^{\mathcal{Z}_0,\mathcal{S},\mathcal{F}_{\mathsf{bc}}}_{\mathsf{rush}}}\ \right] \\
&\overset{\text{Eqs.15,11}}{\geq} \Pr\left[\ \overline{\mathcal{E}^{\mathcal{Z}_0,\mathcal{S},\mathcal{F}_{\mathsf{bc}}}_{b=\hat{b}}} \mid \mathcal{E}^{\mathcal{Z}_{-0},\mathcal{S},\mathcal{F}_{\mathsf{bc}}}_{\mathsf{rush}}\ \right]\cdot\Pr\left[\mathcal{E}^{\mathcal{Z}_0,\mathcal{S},\mathcal{F}_{\mathsf{bc}}}_{\mathsf{rush}}\right] + \frac{1}{2}\cdot f_{S,\pi,\mathcal{F}_{\mathsf{bc}}}(\kappa) \\
&\geq \frac{1}{2}\cdot f_{S,\pi,\mathcal{F}_{\mathsf{bc}}}(\kappa). \qquad (16)
\end{aligned}$$

Putting things together, the distinguishing advantage of $\mathcal{Z}_0$ in distinguishing a real execution of π with dummy adversary $\mathcal{A}$ from and an $\mathcal{F}_{\mathsf{bc}}$-ideal execution with simulator $\mathcal{S}$ is:

$$\left|\Pr\left[\text{REAL}_{\mathcal{Z}_0,\mathcal{A},\pi}(\kappa)=1\right]-\Pr\left[\text{IDEAL}_{\mathcal{Z}_0,\mathcal{S},\mathcal{F}_{\mathsf{bc}}}(\kappa)=1\right]\right|. \qquad (17)$$

Since $\Pr[\text{REAL}_{\mathcal{Z}_0,\mathcal{A},\pi}(\kappa)=1]$ is negligible (by Equation 7), the assumed security of π (which demands that the above distinguishing advantage be negligible) implies that $\Pr[\text{IDEAL}_{\mathcal{Z}_0,\mathcal{S},\mathcal{F}_{\mathsf{bc}}}(\kappa)=1]$ should also be negligible, which, by Equation 16, means that $f_{S,\pi,\mathcal{F}_{\mathsf{bc}}}(\kappa)$ is negligible.

However, in this case the distinguishing advantage of $\mathcal{Z}_1$, defined as

$$\left|\Pr\left[\text{REAL}_{\mathcal{Z}_1,\mathcal{A},\pi}(\kappa)=0\right]-\Pr\left[\text{IDEAL}_{\mathcal{Z}_1,\mathcal{S},\mathcal{F}_{\mathsf{bc}}}(\kappa)=0\right]\right|, \qquad (18)$$

will be noticeable since by Equations 9 and 13 it holds that $\Pr[\text{REAL}_{\mathcal{Z}_1,\mathcal{A},\pi}(\kappa)=0]$ is noticeable and $\Pr[\text{IDEAL}_{\mathcal{Z}_1,\mathcal{S},\mathcal{F}_{\mathsf{bc}}}(\kappa)=0]$ is at most $f_{S,\pi,\mathcal{F}_{\mathsf{bc}}}(\kappa)$, hence negligible. This contradicts the assumed security of π.

This concludes the impossibility proof for protocols in $\Pi_{\mathsf{step\text{-}rel}}$.

Step 2: Extending the impossibility to arbitrary protocols. To extend the above argument to an arbitrary R-round protocol, we use an argument inspired by the MPC literature: Consider the following algorithm $\hat{B}^{(1)}_{\pi}$: For any given set $\hat{\mathcal{P}} \subseteq \mathcal{P}\setminus\{P_s\}$, given the joint view $\text{VIEW}^{\rho}_{\pi,\hat{\mathcal{P}}}(x,\kappa)$ 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, the algorithm $\hat{B}^{(1)}_{\pi}$ operates as follows: it simulates a continuation of the protocol execution, with the parties in $\hat{\mathcal{P}}$ having randomness reported in their joint view, and all parties in $\mathcal{P}\setminus\hat{\mathcal{P}}$ crashing before sending their messages in round ρ, and outputs the output of any (simulated) party in $\hat{\mathcal{P}}$. (By definition of broadcast, and the assumed security of π we know that all simulated parties should output the same bit).

Consider an execution of π, with a random sender-input (bit) b. Since $P_s \notin \hat{\mathcal{P}}$ and b is random, at the beginning of the protocol $\text{VIEW}^{1}_{\pi,\hat{\mathcal{P}}}(x,\kappa)$ includes no information on b. This means that $\hat{B}^{(1)}_{\pi}(\text{VIEW}^{1}_{\pi,\hat{\mathcal{P}}}(x,\kappa)) \neq b$ with noticeable probability (=1/2). On the other hand, the correctness of π mandates that in the last round of an honest execution, every party will output b, except with negligible probability. This means, that there must be some round $\rho_{\pi,\hat{\mathcal{P}}}$ such that $\hat{B}^{(1)}_{\pi}(\text{VIEW}^{\rho}_{\pi,\hat{\mathcal{P}}}(b,\kappa)) \neq b$ with noticeable probability, whereas $\hat{B}^{(1)}_{\pi}(\text{VIEW}^{\rho+1}_{\pi,\hat{\mathcal{P}}}(b,\kappa)) \neq b$ is negligible.

Let $\mathcal{P} = \{P_s, P_2, \ldots, P_n\}$ be any reordering of the player set (e.g., where $P_2,\ldots,P_n$ are ordered according to their party IDs). Denote by $\hat{\mathcal{P}}_1$ the first $\lfloor (n-1)/2 \rfloor$ of $\mathcal{P}\setminus\{P_s\}$, i.e., $\hat{\mathcal{P}}_1 = \{P_2,\ldots,P_{\lfloor (n-1)/2\rfloor+1}\}$ and $\hat{\mathcal{P}}_2$ be the remainder, i.e., $\hat{\mathcal{P}}_2 = \hat{\mathcal{P}}\setminus(\{P_s\}\cup\hat{\mathcal{P}}_1)$. Denote by $\rho_{\pi,\hat{\mathcal{P}}_1}$ the first (smallest) round for which the view of $\hat{\mathcal{P}}_1$ has the above property (i.e., the error

of $\hat{B}_\pi^{(1)}$ in the above process switches from noticeable to negligible); and denote by $\rho_{\pi,\hat{\mathcal{P}}_2}$ the first (smallest) round for which the view of $\hat{\mathcal{P}}_2$ has the above property.

Let i be such that $\rho_{\pi,\hat{\mathcal{P}}_i} \leq \rho_{\pi,\hat{\mathcal{P}}_{3-i}}$. It is easy to verify that this means that the properties of Definition 10 are satisfied for $\hat{\mathcal{P}}_\pi = \hat{\mathcal{P}}_i$. Because $t > n/2$, each of the sets $\hat{\mathcal{P}}_1$ and $\hat{\mathcal{P}}_2$ have size at most $t-1$. Hence, if the environment is given as an advise the round $\rho_{\pi,\hat{\mathcal{P}}_i}$ and the index i, it can easily play the strategy described in the previous case using the defined algorithm $\hat{B}_\pi^{(1)}$. One might be tempted to assume that this completes the proof for a non-uniform environment, but this is not the case, as for a protocol outside the class $\Pi_{\mathsf{step\text{-}rel}}$, the above i and $\rho_{\pi,\hat{\mathcal{P}}_i}$ might depend not only on the protocol but also on the parties setup and randomness. Instead, we complete the argument as follows: Consider an environment who takes a random guess for i and $\rho_{\pi,\hat{\mathcal{P}}_i}$ and gives it as an advise to its adversary. With probability $\frac{1}{2} \cdot \frac{1}{R+1}$ this guess will be correct, and conditioned to it being correct, all the events of the real experiment in the above Step 1 of the proof will occur with the probability stated there. This means that Equation 9 will now become

$$\Pr\left[\text{REAL}_{\mathcal{Z}_1,\mathcal{A},\pi}(\kappa) = 0\right] \geq q(\kappa) \cdot \frac{1}{2} \cdot \frac{1}{R+1}, \tag{19}$$

which is also a noticeable function as $q(\kappa)$ is noticeable and R is polynomial in κ (recall that π is polynomial time). Note however, that the above adaptation of the adversary does not (asymptotically) change the probability of $\Pr\left[\text{IDEAL}_{\mathcal{Z}_1,\mathcal{S},\mathcal{F}_{\mathsf{bc}}}(\kappa) = 0\right]$ in the ideal experiment which remains negligible, as it does not affect the ideal events, which means that the distinguishing advantage

$$\left|\Pr\left[\text{REAL}_{\mathcal{Z}_1,\mathcal{A},\pi}(\kappa) = 0\right] - \Pr\left[\text{IDEAL}_{\mathcal{Z}_1,\mathcal{S},\mathcal{F}_{\mathsf{bc}}}(\kappa) = 0\right]\right|, \tag{20}$$

remains noticeable, contradicting the assumed security of π. □

Replacing TLPs with a (Programable) Random Oracle. One can verify that replacing, in Theorem 4, 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 4, where Step 1 is even simpler and uses the simpler attack from Theorem 1—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 15. *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 10, it is straightforward to verify that all the events involved in the proof of Theorem 1 remain intact. Indeed, the probability of these events is derived directly from Definition 5. This proves the following simple corollary of Theorem 1.

Corollary 16. *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 Section 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 (and, in particular, ensure that the puzzle is opened for 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 [9, 5] to model TLPs in the UC framework.

We proceed to state the theorem.

Theorem 5. *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 $\pi_{\mathsf{bc\text{-}sim}}$ (Figure 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.*

Proof. Correctness of the protocol follows from the correctness of the time-lock puzzle and of $\mathcal{F}_{\mathsf{ubc}}$. Let $\mathcal{A}$ be a PPT adversary; we will construct a PPT simulator $\mathcal{S}$ interacting with the ideal functionality $\mathcal{F}_{\mathsf{bc}}$ and with ideal (dummy) parties $\tilde{P}_1, \ldots, \tilde{P}_n$, for which no PPT environment $\mathcal{Z}$, such that $\mathcal{Z}$ and $\mathcal{A}$ are (R, T)-bounded, can distinguish between interacting with the real protocol and $\mathcal{A}$ or with the ideal computation and $\mathcal{S}$, except with negligible probability.

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 $\mathcal{F}_{\mathsf{ubc}}$.
  - **Recover the output:** Upon receiving (Z, c), each party computes $x = \mathsf{PSol}(Z)$ and outputs $c \oplus H(x)$.

Figure 5: Adaptively secure, simulation-based broadcast protocol

The simulator $\mathcal{S}$ initially constructs virtual parties $P_1, \ldots, P_n$, and emulates the corruption-unfair broadcast functionality $\mathcal{F}_{\mathsf{ubc}}$ and the random oracle H towards the adversary $\mathcal{A}$. The simulator forwards every message received from $\mathcal{Z}$ to $\mathcal{A}$, and similarly, forwards to $\mathcal{Z}$ every message sent from $\mathcal{A}$ to the environment. To simulate the protocol, $\mathcal{S}$ proceeds as follows:

- The simulator stores a list L of the random oracle queries made by $\mathcal{A}$. Whenever $\mathcal{A}$ queries the random oracle with a value z, the simulator checks to see if a pair (z, w) (for some w) appears in L, and if so returns w to $\mathcal{A}$; otherwise, $\mathcal{S}$ samples a random $w \leftarrow \{0,1\}^\kappa$, adds (z, w) to L and returns w to $\mathcal{A}$.
- If $\mathcal{A}$ requests to corrupt a party before the first round, $\mathcal{S}$ corrupts the dummy party in the ideal world. In case of corrupting the sender, $\mathcal{S}$ continues the rest of the simulation by honestly running all the remaining honest parties.
- In case the sender is honest in the first round, $\mathcal{S}$ samples random $c, x \leftarrow \{0,1\}^\kappa$ and computes the puzzle $Z \leftarrow \mathsf{PGen}(T, x)$. Next, $\mathcal{S}$ sends (Z, c) to $\mathcal{A}$ as the leakage from $\mathcal{F}_{\mathsf{ubc}}$.
- If $\mathcal{A}$ requests to corrupt a party during the first R rounds of the protocol, $\mathcal{S}$ corrupts the dummy party in the ideal world. In case of corrupting the sender, $\mathcal{S}$ learns the input message m, and adds (x, w) to L for $w = m \oplus c$ (in case $(x, \cdot)$ already appears in L the simulator aborts).
- If after R rounds the sender is still honest, $\mathcal{S}$ requests the output from $\mathcal{F}_{\mathsf{bc}}$, receives back m, and adds (x, w) to L for $w = m \oplus c$ (in case $(x, \cdot)$ already appears in L the simulator aborts).
- If $\mathcal{A}$ requests to corrupt a party after the first R rounds of the protocol, $\mathcal{S}$ corrupts the dummy party in the ideal world.

Let $\mathcal{Z}$ be a PPT environment such that $\mathcal{Z}$ and $\mathcal{A}$ are (R, T)-bounded. Denote by $\mathcal{E}$ the event that $\mathcal{A}$ queried the random oracle on x during the first R rounds. Note that if the event $\mathcal{E}$ does not occur then the simulation is perfect and induces identically distributed views between the real and the ideal computations. If the event $\mathcal{E}$ does occur, then $\mathcal{Z}$ can distinguish since in this case the simulator must decide on $H(x)$ before knowing m. Denote by $q(\kappa)$ an upper bound on the number of queries made by $\mathcal{A}$, then it holds that this event occurs with probability at most $q(\kappa)/2^\kappa + \mu(\kappa)$, where $\mu(\kappa)$ is the probability in which the environment $\mathcal{Z}$ and $\mathcal{A}$ can jointly learn x from the puzzle Z. Since $\mathcal{Z}$ and $\mathcal{A}$ are (R, T)-bounded, they can evaluate circuits of degree at most $T(\kappa)$ within R rounds, and by the security of the TLP it holds that $\mu(\kappa)$ is negligible. Thus, the overall distinguishing advantage of the environment is $q(\kappa)/2^\kappa + \mu(\kappa)$, which is negligible. □

In a similar way to Theorem 3, we will prove a limited composition theorem for instantiating $\mathcal{F}_{\mathsf{ubc}}$ with the protocol of Dolev and Strong [34] in the simulation-based setting.

Theorem 6. *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.

Proof. Given a PPT t-adversary $\mathcal{A}$ attacking $\pi^{\mathcal{F}\to\rho}$, consider the induced adversary $\mathcal{A}_\pi$ as described in the proof of Theorem 3. That is, split $\mathcal{A}$ into the part $\mathcal{A}_\rho$ interacting with ρ and the part $\mathcal{A}_{\pi^{\mathcal{F}\to\rho}\setminus\rho}$ interacting with the calling part of ρ (with respect to $\pi^{\mathcal{F}\to\rho}$); next, replace in $\mathcal{A}_{\pi^{\mathcal{F}\to\rho}\setminus\rho}$ the interface of ρ and $\mathcal{A}_\rho$ with the ideal computation of $\mathcal{F}$ and the simulator $\mathcal{S}_\rho$ that is guaranteed to exist by the assumed security of ρ. Given a PPT environment $\mathcal{Z}$ it holds that

$$\text{REAL}_{\mathcal{Z},\mathcal{A},\pi^{\mathcal{F}\to\rho}}(\kappa,z)\approx_c \text{HYBRID}^{\mathcal{F}}_{\mathcal{Z},\mathcal{A}_\pi,\pi}(\kappa,z),$$

since any distinguishing advantage between $\text{REAL}_{\mathcal{Z},\mathcal{A},\pi^{\mathcal{F}\to\rho}}$ and $\text{HYBRID}^{\mathcal{F}}_{\mathcal{Z},\mathcal{A}_\pi,\pi}$ immediately translates into a distinguishing advantage of the environment consisting of $\mathcal{Z}$ together with $\mathcal{A}_{\pi^{\mathcal{F}\to\rho}\setminus\rho}$ between interacting with ρ and $\mathcal{A}_\rho$ or with $\mathcal{F}$ and $\mathcal{S}_\rho$.

By the assumption that ρ is $(R,\alpha\cdot T)$-bounded and realizes $\mathcal{F}$ against PPT t-adversaries with resource-respecting simulation, it holds that if $\mathcal{A}$ and $\mathcal{Z}$ are $(R,(1-\alpha)\cdot T)$-bounded, then $\mathcal{A}_\pi$ and $\mathcal{Z}$ are (R,T)-bounded. By the assumed security of π there exists a PPT simulator $\mathcal{S}_\pi$ such that for every PPT environment $\mathcal{Z}$ such that $\mathcal{Z}$ and $\mathcal{A}_\pi$ are (R,T)-bounded, it holds that

$$\text{HYBRID}^{\mathcal{F}}_{\mathcal{Z},\mathcal{A}_\pi,\pi}(\kappa,z)\approx_c \text{IDEAL}_{\mathcal{Z},\mathcal{S}_\pi,\mathcal{G}}(\kappa,z).$$

We therefore conclude that for every PPT environment $\mathcal{Z}$ such that $\mathcal{Z}$ and $\mathcal{A}$ are $(R,(1-\alpha)\cdot T)$-bounded, it holds that

$$\text{REAL}_{\mathcal{Z},\mathcal{A},\pi^{\mathcal{F}\to\rho}}(\kappa,z)\approx_c \text{IDEAL}_{\mathcal{Z},\mathcal{S}_\pi,\mathcal{G}}(\kappa,z).$$ □

In a similar way to Corollary 14 we derive the following corollary. Consider an instantiation of the corruption-unfair broadcast protocol π_{ubc} of Dolev and Strong [34] 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 5, 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 6, 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) that is secure against $(n,(1-\alpha)\cdot T)$-bounded PPT t-adversaries. We therefore derive the following corollary.

Corollary 17. *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 $\pi_{\mathsf{bc\text{-}sim}}(T,\kappa)$ is a broadcast protocol (according to Definition 7) 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.

Bibliography

[1] I. Abraham, T.-H. H. Chan, D. Dolev, K. Nayak, R. Pass, L. Ren, and E. Shi. Communication complexity of Byzantine agreement, revisited. In *38th ACM PODC*, pages 317–326, 2019.

[2] I. Abraham, S. Devadas, D. Dolev, K. Nayak, and L. Ren. Synchronous Byzantine agreement with expected $O(1)$ rounds, expected $O(n^2)$ communication, and optimal resilience. In *FC 2019*, volume 11598 of *LNCS*, pages 320–334, 2019.

[3] A. B. Alexandru, J. Loss, C. Papamanthou, and G. Tsimos. Sublinear-round broadcast without trusted setup against dishonest majority. Cryptology ePrint Archive, Report 2022/1383, 2022. https://eprint.iacr.org/2022/1383.

[4] M. Andrychowicz and S. Dziembowski. PoW-based distributed cryptography with no trusted setup. In *CRYPTO 2015, Part II*, volume 9216, pages 379–399, 2015.

[5] M. Arapinis, N. Lamprou, and T. Zacharias. Astrolabous: A universally composable time-lock encryption scheme. In *ASIACRYPT 2021, Part II*, volume 13091 of *LNCS*, pages 398–426. Springer, Heidelberg, 2021.

[6] C. Badertscher, U. Maurer, D. Tschudi, and V. Zikas. Bitcoin as a transaction ledger: A composable treatment. In *CRYPTO 2017, Part I*, volume 10401 of *LNCS*, pages 324–356, 2017.

[7] C. Badertscher, P. Gazi, A. Kiayias, A. Russell, and V. Zikas. Ouroboros genesis: Composable proof-of-stake blockchains with dynamic availability. In *ACM CCS 2018*, pages 913–930, 2018.

[8] C. Badertscher, R. Canetti, J. Hesse, B. Tackmann, and V. Zikas. Universal composition with global subroutines: Capturing global setup within plain UC. In *TCC 2020, Part III*, volume 12552 of *LNCS*, pages 1–30, 2020.

[9] C. Baum, B. David, R. Dowsley, J. B. Nielsen, and S. Oechsner. TARDIS: A foundation of time-lock puzzles in UC. In *EUROCRYPT 2021, Part III*, volume 12698 of *LNCS*, pages 429–459, 2021.

[10] C. Baum, B. David, R. Dowsley, R. Kishore, J. B. Nielsen, and S. Oechsner. CRAFT: composable randomness beacons and output-independent abort MPC from time. In *PKC 2023, Part I*, LNCS, pages 439–470, 2023.

[11] N. Bitansky, S. Goldwasser, A. Jain, O. Paneth, V. Vaikuntanathan, and B. Waters. Time-lock puzzles from randomized encodings. In *ITCS 2016*, pages 345–356, 2016.

[12] E. Blum, J. Katz, and J. Loss. Synchronous consensus with optimal asynchronous fallback guarantees. In *TCC 2019, Part I*, volume 11891, pages 131–150, 2019.

[13] E. Blum, J. Katz, C.-D. Liu-Zhang, and J. Loss. Asynchronous Byzantine agreement with subquadratic communication. In *TCC 2020, Part I*, volume 12550 of *LNCS*, pages 353–380, 2020.

[14] D. Boneh and M. Naor. Timed commitments. In *CRYPTO 2000*, volume 1880 of *LNCS*, pages 236–254, 2000.

[15] D. Boneh, J. Bonneau, B. Bünz, and B. Fisch. Verifiable delay functions. In *CRYPTO 2018, Part I*, volume 10991 of *LNCS*, pages 757–788, 2018.

[16] M. Borderding. Levels of authentication in distributed agreement. In *10th International Workshop on Distributed Algorithms WDAG*, pages 40–55, 1996.

[17] E. Boyle, R. Cohen, D. Data, and P. Hubácek. Must the communication graph of MPC protocols be an expander? In *CRYPTO 2018, Part III*, volume 10993 of *LNCS*, pages 243–272, 2018.

[18] Z. Brakerski, N. Döttling, S. Garg, and G. Malavolta. Leveraging linear decryption: Rate-1 fully-homomorphic encryption and time-lock puzzles. In *TCC 2019, Part II*, volume 11892 of *LNCS*, pages 407–437, 2019.

[19] J. Camenisch, M. Drijvers, T. Gagliardoni, A. Lehmann, and G. Neven. The wonderful world of global random oracles. In *EUROCRYPT 2018, Part I*, volume 10820 of *LNCS*, pages 280–312, 2018.

[20] R. Canetti. Security and composition of multiparty cryptographic protocols. *Journal of Cryptology*, 13 (1):143–202, 2000.

[21] R. Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *42nd FOCS*, pages 136–145, 2001.

[22] R. Canetti. Universally composable security. *Journal of the ACM*, 67(5):28:1–28:94, 2020.

[23] R. Canetti and M. Fischlin. Universally composable commitments. In *CRYPTO 2001*, volume 2139 of *LNCS*, pages 19–40, 2001.

[24] R. Canetti, U. Feige, O. Goldreich, and M. Naor. Adaptively secure multi-party computation. In *28th ACM STOC*, pages 639–648, 1996.

[25] R. Canetti, Y. Lindell, R. Ostrovsky, and A. Sahai. Universally composable two-party and multi-party secure computation. In *34th ACM STOC*, pages 494–503, 2002.

[26] R. Canetti, I. Damgård, S. Dziembowski, Y. Ishai, and T. Malkin. Adaptive versus non-adaptive security of multi-party protocols. *Journal of Cryptology*, 17(3):153–207, 2004.

[27] R. Canetti, A. Cohen, and Y. Lindell. A simpler variant of universally composable security for standard multiparty computation. In *CRYPTO 2015, Part II*, volume 9216 of *LNCS*, pages 3–22, 2015.

[28] T.-H. H. Chan, R. Pass, and E. Shi. Sublinear-round Byzantine agreement under corrupt majority. In *PKC 2020, Part II*, volume 12111, pages 246–265, 2020.

[29] J. Chen and S. Micali. Algorand: A secure and efficient distributed ledger. *Theoretical Computer Science*, 777:155–183, 2019.

[30] R. Cleve. Limits on the security of coin flips when half the processors are faulty (extended abstract). In *18th ACM STOC*, pages 364–369, 1986.

[31] R. Cohen, S. Coretti, J. A. Garay, and V. Zikas. Probabilistic termination and composability of cryptographic protocols. In *CRYPTO 2016, Part III*, volume 9816 of *LNCS*, pages 240–269, 2016.

[32] R. Cohen, S. Coretti, J. A. Garay, and V. Zikas. Round-preserving parallel composition of probabilistic-termination cryptographic protocols. In *ICALP 2017*, volume 80 of *LIPIcs*, pages 37:1–37:15. Schloss Dagstuhl, 2017.

[33] R. Cohen, a. shelat, and D. Wichs. Adaptively secure MPC with sublinear communication complexity. In *CRYPTO 2019, Part II*, volume 11693, pages 30–60, 2019.

[34] D. Dolev and H. R. Strong. Authenticated algorithms for Byzantine agreement. *SIAM Journal on Computing*, 12(4):656–666, 1983.

[35] C. Dwork and M. Naor. Pricing via processing or combatting junk mail. In *CRYPTO'92*, volume 740 of *LNCS*, pages 139–147, 1993.

[36] L. Eckey, S. Faust, and J. Loss. Efficient algorithms for broadcast and consensus based on proofs of work. Cryptology ePrint Archive, Report 2017/915, 2017. http://eprint.iacr.org/2017/915.

[37] P. Feldman. *Optimal Algorithms for Byzantine Agreement*. PhD thesis, Stanford University, 1988. https://dspace.mit.edu/handle/1721.1/14368.

[38] M. J. Fischer and N. A. Lynch. A lower bound for the time to assure interactive consistency. *Information Processing Letters*, 14(4):183–186, 1982.

[39] M. J. Fischer, N. A. Lynch, and M. Merritt. Easy impossibility proofs for distributed consensus problems. *Distributed Computing*, 1(1):26–39, 1986.

[40] M. Fitzi. *Generalized communication and security models in Byzantine agreement*. PhD thesis, ETH Zurich, Zürich, Switzerland, 2003. URL http://d-nb.info/967397375.

[41] C. Freitag, I. Komargodski, R. Pass, and N. Sirkin. Non-malleable time-lock puzzles and applications. In *TCC 2021, Part III*, volume 13044 of *LNCS*, pages 447–479, 2021.

[42] J. A. Garay and A. Kiayias. SoK: A consensus taxonomy in the blockchain era. In *CT-RSA 2020*, volume 12006 of *LNCS*, pages 284–318, 2020.

[43] J. A. Garay and Y. Moses. Fully polynomial Byzantine agreement in t+1 rounds. In *25th ACM STOC*, pages 31–41, 1993.

[44] J. A. Garay, P. D. MacKenzie, M. Prabhakaran, and K. Yang. Resource fairness and composability of cryptographic protocols. In *TCC 2006*, volume 3876 of *LNCS*, pages 404–428, 2006.

[45] J. A. Garay, J. Katz, C.-Y. Koo, and R. Ostrovsky. Round complexity of authenticated broadcast with a dishonest majority. In *48th FOCS*, pages 658–668. IEEE Computer Society Press, 2007.

[46] J. A. Garay, J. Katz, R. Kumaresan, and H.-S. Zhou. Adaptively secure broadcast, revisited. In *30th ACM PODC*, pages 179–186, 2011.

[47] J. A. Garay, A. Kiayias, and N. Leonardos. The bitcoin backbone protocol: Analysis and applications. In *EUROCRYPT 2015, Part II*, volume 9057 of *LNCS*, pages 281–310, 2015.

[48] J. A. Garay, Y. Ishai, R. Ostrovsky, and V. Zikas. The price of low communication in secure multi-party computation. In *CRYPTO 2017, Part I*, volume 10401 of *LNCS*, pages 420–446, 2017.

[49] J. A. Garay, A. Kiayias, R. M. Ostrovsky, G. Panagiotakos, and V. Zikas. Resource-restricted cryptography: Revisiting MPC bounds in the proof-of-work era. In *EUROCRYPT 2020, Part II*, volume 12106, pages 129–158, 2020.

[50] S. Garg and A. Sahai. Adaptively secure multi-party computation with dishonest majority. In *CRYPTO 2012*, volume 7417 of *LNCS*, pages 105–123, 2012.

[51] O. Goldreich. *Foundations of Cryptography: Basic Applications*, volume 2. Cambridge University Press, Cambridge, UK, 2004.

[52] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game or A completeness theorem for protocols with honest majority. In *19th ACM STOC*, pages 218–229. ACM Press, 1987.

[53] S. Goldwasser and Y. Lindell. Secure multi-party computation without agreement. *Journal of Cryptology*, 18(3):247–287, July 2005.

[54] S. Goldwasser, Y. T. Kalai, and S. Park. Adaptively secure coin-flipping, revisited. In *ICALP 2015, Part II*, volume 9135 of *LNCS*, pages 663–674, 2015.

[55] I. Haitner and Y. Karidi-Heller. A tight lower bound on adaptively secure full-information coin flip. In *61st FOCS*, pages 1268–1276, 2020.

[56] C. Hazay, Y. Lindell, and A. Patra. Adaptively secure computation with partial erasures. In *34th ACM PODC*, pages 291–300, 2015.

[57] M. Hirt and V. Zikas. Adaptively secure broadcast. In *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 466–485, 2010.

[58] D. Hofheinz and J. Müller-Quade. 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.

[59] Y. T. Kalai, I. Komargodski, and R. Raz. A lower bound for adaptively-secure collective coin-flipping protocols. In *DISC*, pages 34:1–34:16, 2018.

[60] J. Katz, U. Maurer, B. Tackmann, and V. Zikas. Universally composable synchronous computation. In *TCC 2013*, volume 7785, pages 477–498, 2013.

[61] J. Katz, A. Thiruvengadam, and H.-S. Zhou. Feasibility and infeasibility of adaptively secure fully homomorphic encryption. In *PKC 2013*, volume 7778 of *LNCS*, pages 14–31, 2013.

[62] J. Katz, J. Loss, and J. Xu. On the security of time-lock puzzles and timed commitments. In *TCC 2020, Part III*, volume 12552, pages 390–413, 2020.

[63] H. A. Khorasgani, H. K. Maji, and T. Mukherjee. Estimating gaps in martingales and applications to coin-tossing: Constructions and hardness. In *TCC 2019, Part II*, volume 11892 of *LNCS*, pages 333–355, 2019.

[64] A. Kiayias, H.-S. Zhou, and V. Zikas. Fair and robust multi-party computation using a global transaction ledger. In *EUROCRYPT 2016, Part II*, volume 9666 of *LNCS*, pages 705–734, 2016.

[65] L. Lamport, R. E. Shostak, and M. C. Pease. The Byzantine generals problem. *ACM Trans. Program. Lang. Syst.*, 4(3):382–401, 1982.

[66] H. Lin, R. Pass, and P. Soni. Two-round and non-interactive concurrent non-malleable commitments from time-lock puzzles. In *58th FOCS*, pages 576–587, 2017.

[67] C.-D. Liu-Zhang and U. Maurer. Synchronous constructive cryptography. In *TCC 2020, Part II*, volume 12551 of *LNCS*, pages 439–472, 2020.

[68] M. Mahmoody, T. Moran, and S. P. Vadhan. Time-lock puzzles in the random oracle model. In *CRYPTO 2011*, volume 6841 of *LNCS*, pages 39–50, 2011.

[69] G. Malavolta and S. A. K. Thyagarajan. Homomorphic time-lock puzzles and applications. In *CRYPTO 2019, Part I*, volume 11692, pages 620–649, 2019.

[70] C. Matt, J. B. Nielsen, and S. E. Thomsen. Formalizing delayed adaptive corruptions and the security of flooding networks. In *CRYPTO 2022, Part II*, volume 13508 of *LNCS*, pages 400–430. Springer, Heidelberg, 2022.

[71] S. Micali and P. Rogaway. Secure computation (abstract). In *CRYPTO'91*, volume 576 of *LNCS*, pages 392–404, 1992.

[72] J. B. Nielsen. Separating random oracle proofs from complexity theoretic proofs: The non-committing encryption case. In *CRYPTO 2002*, volume 2442 of *LNCS*, pages 111–126, 2002.

[73] J. B. Nielsen. *On Protocol Security in the Cryptographic Model*. PhD thesis, University of Aarhus, 2003. https://www.brics.dk/DS/03/8/BRICS-DS-03-8.pdf.

[74] R. Pass. Simulation in quasi-polynomial time, and its application to protocol composition. In *EUROCRYPT 2003*, volume 2656 of *LNCS*, pages 160–176, 2003.

[75] R. Pass, L. Seeman, and a. shelat. Analysis of the blockchain protocol in asynchronous networks. In *EUROCRYPT 2017, Part II*, volume 10211 of *LNCS*, pages 643–673, 2017.

[76] M. C. Pease, R. E. Shostak, and L. Lamport. Reaching agreement in the presence of faults. *Journal of the ACM*, 27(2):228–234, 1980.

[77] B. Pfitzmann and M. Waidner. Unconditional Byzantine agreement for any number of faulty processors. In *STACS*, volume 577 of *LNCS*, pages 339–350, 1992.

[78] K. Pietrzak. Simple verifiable delay functions. In *ITCS 2019*, volume 124, pages 60:1–60:15, 2019.

[79] R. L. Rivest, A. Shamir, and D. A. Wagner. Time-lock puzzles and timed-release crypto. Technical report, Massachusetts Institute of Technology, USA, 1996.

[80] L. Rotem and G. Segev. Generically speeding-up repeated squaring is equivalent to factoring: Sharp thresholds for all generic-ring delay functions. In *CRYPTO 2020, Part III*, volume 12172 of *LNCS*, pages 481–509, 2020.

[81] S. Srinivasan, J. Loss, G. Malavolta, K. Nayak, C. Papamanthou, and S. A. K. Thyagarajan. Transparent batchable time-lock puzzles and applications to byzantine consensus. In *PKC 2023, Part I*, LNCS, pages 554–584, 2023.

[82] G. Tsimos, J. Loss, and C. Papamanthou. Gossiping for communication-efficient broadcast. In *CRYPTO 2022, Part III*, volume 13509 of *LNCS*, pages 439–469. Springer, Heidelberg, 2022.

[83] J. Wan, H. Xiao, S. Devadas, and E. Shi. Round-efficient Byzantine broadcast under strongly adaptive and majority corruptions. In *TCC 2020, Part I*, volume 12550 of *LNCS*, pages 412–456, 2020.

[84] J. Wan, H. Xiao, E. Shi, and S. Devadas. Expected constant round Byzantine broadcast under dishonest majority. In *TCC 2020, Part I*, volume 12550 of *LNCS*, pages 381–411, 2020.

[85] B. Wesolowski. Efficient verifiable delay functions. In *EUROCRYPT 2019, Part III*, volume 11478 of *LNCS*, pages 379–407, 2019.

[86] A. C.-C. Yao. Protocols for secure computations (extended abstract). In *23rd FOCS*, pages 160–164. IEEE Computer Society Press, 1982.

A On the Limits of the Property-Based Definition

We note that one can easily extend the class $\Pi_{\mathsf{step\text{-}rel}}$ (but make it more contrived) to capture also certain artificial protocols (e.g., where sender waits a random number of rounds before doing anything). This can be done by requiring that the round R is not known a priori but becomes known to the adversary at the beginning of that round. We chose to keep the class simpler because it already captures all published dishonest-majority broadcast protocols, which we believe is sufficient to make the point for the unsuitability of property-based definitions for adaptive security.

Providing a complete characterization of broadcast protocols according to the property-based definition is a challenging task. We do not pursue this direction as the purpose of this definition is to illustrate barriers of broadcast protocols rather than to be used for protocol design (the simulation-based definition is more suitable for that). An example of a protocol which falls outside this class that we do not know whether it satisfies the property-based definition or not is as follows: The protocol consists of κ phases, where in each phase the parties run the protocol of Dolev and

Strong [34]. The sender initially samples $i^* \leftarrow [\kappa]$; in the first i^* phases it broadcasts a random bit and in the remaining phases the sender broadcasts its actual input bit. The adversary can guess the right phase with inverse polynomial probability, but for our attack to go through it must know the "right" round with overwhelming probability. We emphasize that even such a protocol cannot realize the simulation-based definition, for which we do provide a complete characterization.

In the simulation-based setting (Section 5.1) we are able to show a broader impossibility, i.e., excluding a wider class of protocols than in the property-based setting (Section 4.1), where the impossibility is restricted to the class $\Pi_{\mathsf{step\text{-}rel}}$. One reason for this is that simulation-based definitions frame the problem in a "tighter" manner, and therefore potentially exclude more protocols. However, there is also a technical reason for that: In a property-based definition, one needs to prove that a specific event occurs with certain high probability—in our case, this event is the adversary being able to flip the input upon corrupting the sender later in the protocol. However, how high this probability is depends heavily on the protocol, and the class $\Pi_{\mathsf{step\text{-}rel}}$ is the class that gives us a clean handle on this probability.

On the other hand, in simulation-based security one just needs to prove that the distance between two (potentially unspecified) probability distributions in the real and the ideal experiments is noticeable. This is a far less protocol-dependent requirement, and is the core of the reason why we are able to extend the impossibility to arbitrary protocols. We view the fact that the simulation-based definition allows us to get such a broader impossibility result as yet another indication of its superiority for the analysis of cryptographic protocols.