Communication Lower Bounds for Cryptographic Broadcast Protocols*

Erica Blum† Elette Boyle‡ Ran Cohen§ Chen-Da Liu-Zhang¶

September 4, 2023

Abstract

Broadcast protocols enable a set of n parties to agree on the input of a designated sender, even in the face of malicious parties who collude to attack the protocol. In the honest-majority setting, a fruitful line of work harnessed randomization and cryptography to achieve low-communication broadcast protocols with sub-quadratic total communication and with "balanced" sub-linear communication cost per party.

However, comparatively little is known in the *dishonest-majority* setting. Here, the most communication-efficient constructions are based on the protocol of Dolev and Strong (SICOMP '83), and sub-quadratic broadcast has not been achieved even using randomization and cryptography. On the other hand, the only nontrivial $\omega(n)$ communication lower bounds are restricted to *deterministic* protocols, or against *strong adaptive* adversaries that can perform "after the fact" removal of messages.

We provide new communication lower bounds in this space, which hold against arbitrary cryptography and setup assumptions, as well as a simple protocol showing near tightness of our first bound.

- **Static adversary.** We demonstrate a tradeoff between *resiliency* and *communication* for randomized protocols secure against $n - o(n)$ *static* corruptions. For example, $\Omega(n \cdot \mathsf{polylog}(n))$ messages are needed when the number of honest parties is $n/\mathsf{polylog}(n)$; $\Omega(n\sqrt{n})$ messages are needed for $O(\sqrt{n})$ honest parties; and $\Omega(n^2)$ messages are needed for $O(1)$ honest parties.

  Complementarily, we demonstrate broadcast with $O(n \cdot \mathsf{polylog}(n))$ total communication and balanced $\mathsf{polylog}(n)$ per-party cost, facing any constant fraction of static corruptions.
- **Weakly adaptive adversary.** Our second bound considers $n/2 + k$ corruptions and a *weakly adaptive* adversary that cannot remove messages "after the fact." We show that any broadcast protocol within this setting can be attacked to force an arbitrary party to send messages to k other parties.

  Our bound implies limitations on the feasibility of *balanced* low-communication protocols: For example, ruling out broadcast facing 51% corruptions, in which all non-sender parties have sublinear communication locality.

*A preliminary version of this work appeared in *DISC 2023*.

†University of Maryland. E-mail: erblum@umd.edu.

‡Reichman University and NTT Research. E-mail: eboyle@alum.mit.edu.

§Reichman University. E-mail: cohenran@runi.ac.il.

¶HSLU and Web3 Foundation. E-mail: chen-da.liuzhang@hslu.ch.

Contents

1 Introduction

In a *broadcast* protocol (a.k.a. Byzantine generals [PSL80, LSP82]) a designated party (the sender) distributes its input message in a way that all honest parties agree on a common output, equal to the sender's message if the sender is honest. Broadcast is amongst the most widely studied problems in the context of distributed computing, and forms a fundamental building block in virtually any distributed system requiring reliability in the presence of faults. The focus of this work is on synchronous protocols that proceed in a round-by-round manner.

Understanding the required communication complexity of broadcast is the subject of a rich line of research. Most centrally, this is measured as the total number of bits communicated within the protocol, as a function of the number of parties n and corrupted parties t. Other metrics have also been studied, such as *message complexity* (number of actual messages sent), *communication locality* (defined as the *maximal degree* of a party in the induced communication graph of the protocol execution [BGT13]), and *per-party* communication requirements (measuring how communication is split across parties).

The classical lower bound of Dolev and Reischuk [DR85] showed that $\Omega(n + t^2)$ messages are necessary for *deterministic* protocols (a cubic message complexity is also sufficient facing any linear number of corruptions [DR85, DS83, MR21][1]). This result came as part of several seminal impossibility results for deterministic protocols presented in the '80s, concerning feasibility [FLP83], resiliency [LSP82, FLM86], round complexity [FL82, DS83], and connectivity [Dol82, FLM86]. Those lower bounds came hand in hand with feasibility results initiated by Ben-Or [Ben83] and Rabin [Rab83], as well as Dolev and Strong [DS83], showing that randomization and cryptography are invaluable tools in achieving strong properties within broadcast protocols.

As opposed to bounds on feasibility, resiliency, and round complexity, the impossibility of [DR85] held for over 20 years, in both the honest- and dishonest-majority settings. Recently, there has been progress in the honest-majority setting, with several works demonstrating how randomization and cryptography can be used to bypass the classical communication complexity bound and achieve sub-quadratic communication: with information-theoretic security [KSSV06, KS09, KS11, BGH13] or with computational security under cryptographic assumptions [CM19, ACD+19, CKS20, BKLL20, BCG21]; some of these protocols even achieve poly-logarithmic locality and "balanced" sub-linear communication cost per party. While the security of some of these constructions holds against a *static* adversary that specifies corruptions before the protocol's execution begins, some of these protocols are even secure against a *weakly adaptive* adversary; that is, an adversary that cannot retract messages sent by a party upon corrupting that party. Abraham et al. [ACD+19] showed that this relaxation is inherent for sub-quadratic broadcast, even for randomized protocols, by demonstrating an $\Omega(t^2)$ communication lower bound in the presence of a *strongly rushing* adversary; that is, an adversary that can "drop" messages by corrupting the sender after the message is sent but before it is delivered — this ability is known as *after-the-fact removal.*

Focusing on the *dishonest-majority* setting, however, comparatively little is known about communication complexity. Here, the most communication-efficient broadcast constructions are based on the protocol of Dolev and Strong [DS83], and broadcast with $o(nt)$ messages has not been achieved even using randomization and cryptography. The state-of-the-art protocols, for a constant fraction $t = \Theta(n)$ of corruptions, are due to Chan et al. [CPS20] in the weakly adaptive

[1] We note that [DR85, DS83, MR21] rely on cryptography, so they are not deterministic per se; however, these protocols make a black-box use of the cryptographic primitives and are deterministic otherwise.

setting under a trusted setup assumption, and to Tsimos et al. [TLP22] in the static setting under a weaker setup assumption; however, both works require $\Omega(nt)$ communication, namely $\tilde{O}(n^2)$.[2] On the other hand, the only nontrivial $\omega(n)$ communication lower bounds are those discussed above, restricted to deterministic protocols, or against strong adaptive adversaries.

1.1 Our Contributions

In this work, we explore the achievable communication complexity of broadcast in the dishonest-majority setting. We provide new communication lower bounds in this space, which hold against arbitrary cryptographic and setup assumptions, as well as a simple protocol showing near tightness of our first bound. Our results consider a *synchronous* communication model: lower bounds in this model immediately translate into lower bounds in the *asynchronous* and *partially synchronous* models, whereas protocols in the latter models can only tolerate $t < n/3$ corruptions [DLS88] implying that synchrony is inherently needed for our protocol construction.

Static adversary. We begin with the setting of static corruptions. We demonstrate a simple modification to the protocol of Chan et al. [CPS20], incorporating techniques from Tsimos et al. [TLP22], which obtains a new protocol with essentially optimal $\tilde{O}(n)$ communication. The resulting protocol relies on the same assumptions as [CPS20]: namely, cryptographic verifiable random functions (VRFs)[3] and a trusted public-key infrastructure (PKI) setup, where the keys for each party are honestly generated and distributed. Further, the protocol is resilient against any constant fraction of static corruptions as in [TLP22], and achieves balanced $\tilde{O}(1)$ cost per party.

Proposition 1.1 (sub-quadratic broadcast facing a constant fraction of static corruptions)**.** *Let $0 < \epsilon < 1$ be any constant. Assuming a trusted-PKI for VRFs and signatures, it is possible to compute broadcast with $\tilde{O}(n)$ total communication ($\tilde{O}(1)$ per party) facing a static adversary corrupting $(1-\epsilon) \cdot n$ parties.*

Perhaps more interestingly, in the regime of $n - o(n)$ static corruptions, we demonstrate a feasibility tradeoff between *resiliency* and *communication* that nearly tightly complements the above upper bound. We show that resilience in the face of only $\epsilon(n) \cdot n$ honest parties, for $\epsilon(n) \in o(1)$, demands message complexity scaling as $\Omega(n/\epsilon(n))$. Note that a lower bound on message complexity is stronger than for communication complexity, directly implying the latter. Our lower bound holds for randomized protocols, given any cryptographic assumption and any setup information that is generated by an external trusted party and given to the parties before the beginning of the protocol, including the assumptions of the above upper bound.

Theorem 1.2 (communication lower bound for static corruptions)**.** *Let $\epsilon(n) \in o(1)$. If there exists a broadcast protocol that is secure against $(1-\epsilon(n)) \cdot n$ static corruptions, then the message complexity of the protocol is $\Omega(n \cdot \frac{1}{\epsilon(n)})$.*

For example, for $n - n/\log^d(n)$ corruptions with a constant $d \geq 1$ (i.e., $\epsilon(n) = \log^{-d}(n)$), the message complexity must be $\Omega(n \cdot \log^d(n))$. For $n - \sqrt{n}$ corruptions (i.e., $\epsilon(n) = 1/\sqrt{n}$), the message

[2] As standard in relevant literature, in this work $\tilde{O}$ notation hides polynomial factors in $\log(n)$ as well as in the cryptographic security parameter κ.

[3] A verifiable random function [MRV99] is a pseudorandom function that provides a non-interactively verifiable proof for the correctness of its output.

complexity must be $\Omega(n \cdot \sqrt{n})$. And for $n - c$ corruptions with a constant $c > 1$ (i.e., $\epsilon(n) = c/n$), the message complexity must be $\Omega(n^2)$, in particular meaning that sub-quadratic communication is impossible in this regime.

As noted, Theorem 1.2 holds for any cryptographic and setup assumptions. This captures, in particular, PKI-style setup (such as the VRF-based PKI of Chan et al. [CPS20]) in which the trusted party samples a private/public key-pair for each party and gives each party its private key together with the vector of all public keys. It additionally extends to even stronger, more involved setup assumptions for generating *correlated randomness* beyond a product distribution, e.g., setup for threshold signatures where parties' secret values are nontrivially correlated across one another.

Weakly adaptive adversary. The lower bound of Theorem 1.2 carries over directly to the setting of weakly adaptive adversaries. Shifting back to the regime of a constant fraction of corruptions, one may naturally ask whether a protocol such as that from Proposition 1.1 can also exist within this regime.

Unfortunately, given a few minutes thought one sees that a balanced protocol with polylogarithmic per-party communication as demonstrated by Proposition 1.1 cannot translate to the weakly adaptive setting. The reason is that if the *sender* party speaks to at most t other parties, then the adaptive adversary can simply corrupt each receiving party and drop the message, thus blocking any information of the sender's input from reaching honest parties.

However, this attack applies only to the unique sender party. Indeed, non-sender parties contribute no input to the protocol to be blocked; and, without the ability to perform "after-the-fact" message removal, a weakly adaptive adversary cannot prevent communication from being *received* by a party without a very large number of corruptions.

We therefore consider the locality of *non-sender* parties, and ask whether sublinear locality is achievable. Our third result answers this question in the negative. That is, we show an efficient adversary who can force any party of its choosing to communicate with a large number of neighbors. Note that this in particular lower bounds the per-party communication complexity of non-sender parties.

Theorem 1.3 (non-sender locality facing adaptive corruptions)**.** *Let $0 < k < (n-1)/2$ and let π be an n-party broadcast protocol secure against $t = n/2 + k$ adaptive corruptions. Then, for any non-sender party P_{i^*} there exists a PPT adversary that can force the locality of P_{i^*} to be larger than k, except for negligible probability.*

For example, for $k \in \Theta(n)$, e.g., a constant fraction $t = 0.51 \cdot n$ of corruptions, the locality of P_{i^*} must be $\Theta(n)$, thus forming a separation from Proposition 1.1 for the locality of non-sender parties. Similarly to Theorem 1.2, this bound holds in the presence of any correlated-randomness setup and for any cryptographic assumptions.

We remark that our bound further indicates a design requirement for any protocol attempting to achieve sub-quadratic $o(n^2)$ communication complexity within this setting. To obtain $o(n^2)$ communication, it must of course be that nearly all parties have sublinear communication locality. Our result shows that any such protocol must include instructions causing a party to send out messages to a linear number of other parties upon determining that it is under attack.

Summary. For completeness, Table 1 summarizes our results alongside prior work.

	setup	corruptions	total com.	locality (non-sender)	ref.
strongly adaptive	bare pki	$t < n$	$O(n^3)$	n	[DS83]
	any	$t = \Theta(n)$	$\Omega(n^2)$	$\Omega(n)$	[ACD+19]
weakly adaptive	trusted pki	$t = \Theta(n)$	$\tilde{O}(n^2)$	$O(n)$	[CPS20]
	any	$t = \Theta(n)$		$\Omega(n)$	Thm. 1.3
static	any (deterministic)	$t = \Theta(n)$	$\Omega(n^2)$	$\Omega(n)$	[DR85]
	bare pki	$t = \Theta(n)$	$\tilde{O}(n^2)$	$\tilde{O}(1)$	[TLP22]
	trusted pki	$t = \Theta(n)$	$\tilde{O}(n)$	$\tilde{O}(1)$	Prop. 1.1
	any	$t = (1-\epsilon(n)) \cdot n$, $\epsilon(n) \in o(1)$	$\Omega(n \cdot \frac{1}{\epsilon(n)})$		Thm. 1.2
		e.g., $t = n - \frac{n}{\mathsf{polylog}(n)}$	$\Omega(n \cdot \mathsf{polylog}(n))$		
		e.g., $t = n - \sqrt{n}$	$\Omega(n \cdot \sqrt{n})$		
		e.g., $t = n - O(1)$	$\Omega(n^2)$		

Table 1: Communication requirements of dishonest-majority (synchronous) broadcast. We consider the standard, property-based definition of broadcast (see Definition 2.1). For each type of adversary (strongly adaptive, weakly adaptive, and static), we consider the state-of-the-art protocols and lower bounds in terms of setup, number of corruptions, total communication and (non-sender) locality. For setup we distinguish bare PKI, where each party locally generates its key pair, as opposed to trusted PKI, where all keys are generated by a trusted dealer. Reference [DR85] is only for deterministic protocols.

1.2 Technical Overview

The proof of Proposition 1.1 follows almost immediately from [CPS20] and [TLP22]. We therefore focus on our lower bounds.

Communication lower bound for static corruptions. The high-level idea of the attack underlying Theorem 1.2 is to split all parties except for the sender P_s into two equal-size subsets, $\mathcal{A}$ and $\mathcal{B}$, randomly choose a set $\mathcal{S}$ of size $\epsilon(n) - 1$ parties in $\mathcal{A}$ and a party $\mathsf{P}_{i^*} \in \mathcal{B}$, and corrupt all parties but $\mathcal{S} \cup \{\mathsf{P}_{i^*}\}$ (as illustrated in Figure 1). The adversary proceeds by running two independent executions of the protocol. In the first, the sender runs an execution on input 0 towards $\mathcal{A}$, and all corrupted parties in $(\{\mathsf{P}_s\} \cup \mathcal{A}) \setminus \mathcal{S}$ ignore all messages from parties in $\mathcal{B}$ (pretending they all crashed). In the second, the sender runs an execution on input 1 towards $\mathcal{B}$, and all corrupted parties in $(\{\mathsf{P}_s\} \cup \mathcal{B}) \setminus \{\mathsf{P}_{i^*}\}$ ignore all messages from parties in $\mathcal{A}$.

As long as the honest parties in $\mathcal{S}$ and the honest party P_{i^*} do not communicate, the adversary will make them output different values. This holds because, conditioned on no communication between $\mathcal{S}$ and P_{i^*}, the view of honest parties in $\mathcal{S}$ is indistinguishable from a setting where the adversary crashes all parties in $\mathcal{B}$ and an honest sender has input 0; in this case, all parties in $\mathcal{A}$ (and in particular in $\mathcal{S}$) must output 0. Similarly, conditioned on no communication between $\mathcal{S}$ and P_{i^*}, the view of P_{i^*} is indistinguishable from a setting where the adversary crashes all parties in $\mathcal{A}$ and an honest sender has input 1; in this case, all parties in $\mathcal{B}$ (and in particular P_{i^*}) must output 1.

The challenge now is to argue that the honest parties in $\mathcal{S}$ and the honest party P_{i^*} do not communicate with noticeable probability. Note that this does not follow trivially from the overall low communication complexity, as the communication patterns unfold as a function of the adversarial behavior, which in particular depends on the choice of $\mathcal{S}$ and P_{i^*}. The argument instead

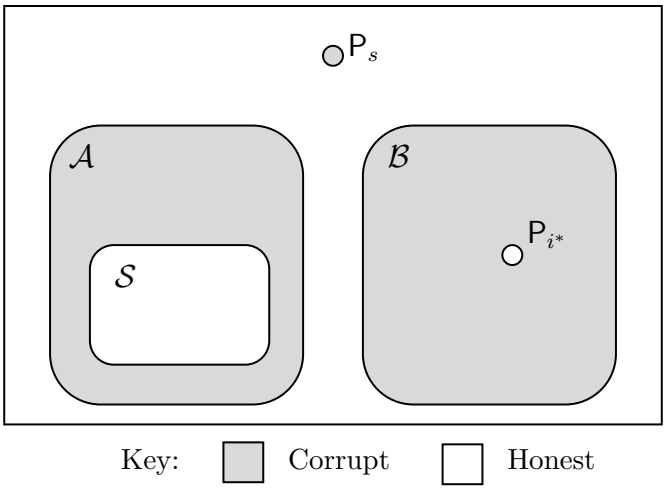

Figure 1: The partition of the parties used in the proof of Theorem 1.2. All the parties except for the sender P_s are partitioned into two equal-size sets, $\mathcal{A}$ and $\mathcal{B}$. A subset $\mathcal{S} \subseteq \mathcal{A}$ of size $\epsilon(n) \cdot n - 1$ and a party $\mathsf{P}_{i^*} \in \mathcal{B}$ are uniformly sampled, and the adversary statically corrupts all parties but $\mathcal{S} \cup \{\mathsf{P}_{i^*}\}$. The goal of the attack is to ensure that $\mathcal{S}$ and P_{i^*} do not communicate, thus forcing them to output different bits.

follows from a series a delicate steps that compare the view of parties in this execution with other adversarial strategies.

The underlying trick is to analyze the event of communication between $\mathcal{S}$ and P_{i^*} by splitting into two sub-cases: when $\mathcal{S}$ speaks to P_{i^*} *before* receiving any message from P_{i^*}, and when P_{i^*} speaks to $\mathcal{S}$ *before* receiving any message from $\mathcal{S}$. (Note, these events are not disjoint.) The important observation is that before any communication is received by the other side, then each side's view in the attack is identically distributed as in a hypothetical execution in which the corresponding set $\mathcal{A}$ or $\mathcal{B}$ crashes from the start. Since these simple crash adversarial strategies are indeed independent of $\mathcal{S}$ and P_{i^*}, then we can easily analyze and upper bound the probability of $\mathcal{S}$ and P_{i^*} communicating within their hypothetical executions. To finalize the argument, we carry this analysis over to show that with noticeable probability P_{i^*} does not communicate with $\mathcal{S}$ in an actual execution with the original adversary.

Locality lower bound for weakly adaptive corruptions. We proceed to consider the setting of *weakly adaptive* corruptions. As mentioned above, in the adaptive setting it is easy to see that the sender must communicate with many parties, since otherwise the adversary may crash every party that the sender communicates with; therefore, the challenging part is to focus on non-sender parties. Further, when considering strong adaptive adversaries that can perform after-the-fact message removal by corrupting the sender, every honest party must communicate with a linear number of parties [ACD+19]. In our setting, we do not consider such capabilities of the adversary. In particular, once the adversary learns that an honest party has chosen to send a message, this message cannot be removed or changed.

Unlike our previous lower bound which assumed $n - o(n)$ corruptions, here we consider $n/2 + k$ corruptions for $k \in O(n)$, so we cannot prevent sets of honest parties from communicating with each other. Our approach, instead, is to keep the targeted party P_{i^*} confused about the output of

other honest parties.

More concretely, our adversarial strategy splits all parties but the sender and P_{i^*} into disjoint equal-size sets $\mathcal{S}_0$ and $\mathcal{S}_1$ of parties, samples a random bit b and corrupts the sender party and the parties in $\mathcal{S}_{1-b}$. The adversary communicates with $\mathcal{S}_0$ as if the sender's input is 0 and all parties in $\mathcal{S}_1$ have crashed, and at the same time plays towards $\mathcal{S}_1$ as if the sender's input is 1 and all parties in $\mathcal{S}_0$ have crashed (as illustrated in Figure 2). Although the adversary cannot prevent honest parties from $\mathcal{S}_b$ from sending messages to the targeted party P_{i^*}, it can corrupt every party that *receives* a message from P_{i^*}. The effect of this attack is that, although P_{i^*} can tell that the sender is cheating, P_{i^*} cannot know whether parties in $\mathcal{S}_0$ or parties in $\mathcal{S}_1$ are honest. And, moreover, P_{i^*} cannot know whether the remaining honest parties *know* that the sender is cheating or if they believe that the sender is honest and other parties crashed—in which case they must output a bit (either 0 if $\mathcal{S}_0$ are honest or 1 if $\mathcal{S}_1$ are honest). To overcome this attack, P_{i^*} must communicate with sufficiently many parties such that the adversary's corruption budget will run out, i.e., with output locality at least k.

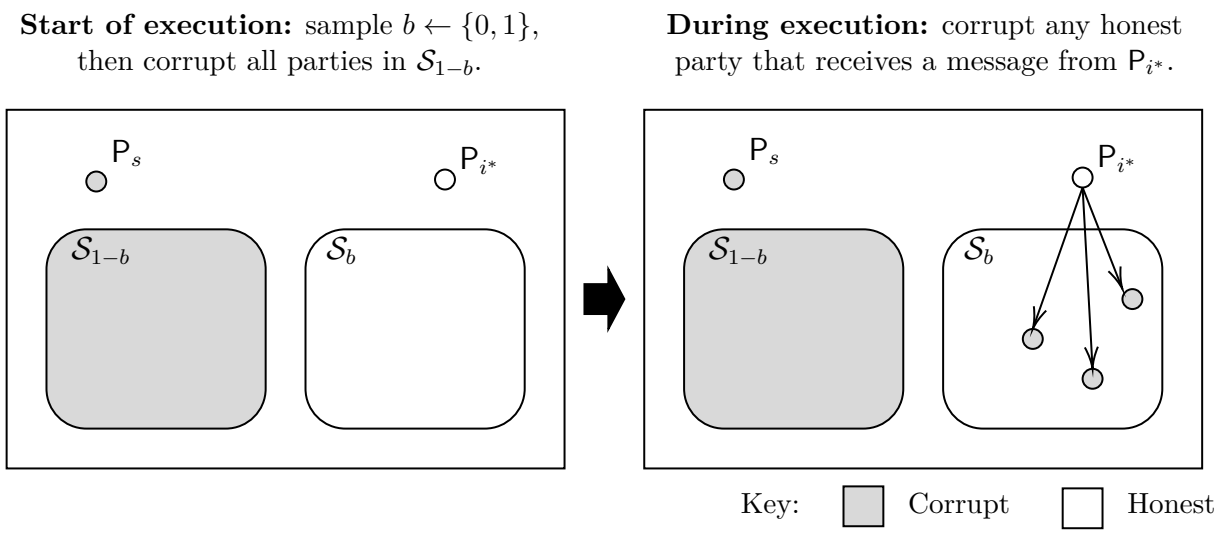

Figure 2: The partition of the parties used in the proof of Theorem 1.3. At the start of the execution, the adversary partitions all parties except for the sender P_s and targeted party P_{i^*} into equal-size sets $\mathcal{S}_0$ and $\mathcal{S}_1$, and corrupts the parties in $\mathcal{S}_{1-b}$ (for a random bit b). During the execution, whenever an honest party $\mathsf{P}_j \in \mathcal{S}_b$ receives a message from the targeted party P_{i^*}, the adversary corrupts P_j.

1.3 Further Related Work

Since the classical results from the '80s, a significant line of work has been devoted to understanding the complexity of broadcast protocols.[4]

[4]In this work we consider broadcast protocols that achieve the usual properties of *termination*, *agreement*, and *validity*. We note that stronger notions of broadcast have been considered in the literature, e.g., in the adaptive setting, the works of [HZ10, GKKZ11, CGZ23] study *corruption fairness* ensuring that once any receiver learns the sender's input, the adversary cannot corrupt the sender and change its message. As our main technical contributions are lower bounds, focusing on weaker requirements yields stronger results.

Communication complexity. In the honest-majority regime, we know of several protocols, deterministic [BGP92, CW92, MR21] or randomized [Mic17, FLL21], that match the known lower bounds [DR85, ACD+19] for strongly adaptive adversaries. When considering static, or weakly adaptive security, a fruitful line of works achieved sub-quadratic communication, with information-theoretic security [KSSV06, KS09, KS11, BGH13] or with computational security [CM19, ACD+19, CKS20, BKLL20, BCG21] .

In the dishonest-majority regime, the most communication-efficient broadcast constructions are based on the protocol of Dolev and Strong [DS83]. This protocol is secure facing any number of strongly adaptive corruptions and the communication complexity is $O(n^3)$. When considering weakly adaptive corruptions, Chan et al. [CPS20] used cryptography and trusted setup to dynamically elect a small, polylog-size committee in each round and improved the communication to $\tilde{O}(n^2)$. In the static-corruption setting, Tsimos et al. [TLP22] achieved $\tilde{O}(n^2)$ communication by running the protocol of [DS83] over a "gossiping network" [DGH+87, KSSV00]. This work further achieved *amortized* sub-quadratic communication facing weakly adaptive corruptions when all parties broadcast in parallel.

A line of works focused on achieving *balanced* protocols, where all parties incur the same work in terms of communication complexity [KS11, BCG21, ADD+22]. The work of [BCG21] also showed lower bounds on the necessary setup and cryptographic assumptions to achieve balanced protocols when extending almost-everywhere agreement to full agreement.[5] Message dissemination protocols [DXR21, LZMT22] have also been proven useful for constructing balanced protocols.

The work in [HKK08] showed that without trusted setup assumptions, at least one party must send $\Omega(n^{1/3})$ messages, in the *static filtering* model, where each party must decide which set of parties it will accept messages from in each round before the rounds begins. We remark that our lower bounds hold also given trusted setup, and in the dynamic-filtering model (in which sub-quadratic upper bounds have been achieved).

Connectivity. Obtaining communication-efficient protocols inherently relies on using a strict subgraph of the communication network. Early works [Dol82, FLM86] showed that *deterministic* broadcast is possible in an incomplete graph only if the graph is $(t+1)$-connected. The influential work of King et al. [KSSV06] laid a path not only for randomized Byzantine agreement with sub-quadratic communication, but also for protocols that run over a partial graph [KS09, KS11, BGH13, BCG21]. The graphs induced by those protocols yield expander graphs, and the work of [BCDH18] showed that in the strongly adaptive setting and facing a linear number of corruptions, no protocol for all-to-all broadcast in the plain model (without PKI setup) can maintain a non-expanding communication graph against all adversarial strategies. Further, feasibility of broadcast with a non-expander communication graph, admitting a sub-linear cut, was demonstrated in weaker settings [BCDH18].

Round complexity. In terms of round complexity, when considering deterministic protocols, $t+1$ rounds are known to be sufficient [PSL80, DS83, GM93] and necessary [FL82, DS83]. Ben-Or [Ben83] and Rabin [Rab83], showed that this lower bound can be overcome using randomiza-

[5] *Almost-everywhere agreement* [DPPU88] is a relaxed problem in which all but an $o(1)$ fraction of the parties must reach agreement. For this relaxation, King et al. [KSSV06] showed an efficient protocol, with poly-logarithmic locality, communication, and rounds. This protocol serves as a stepping stone to several sub-quadratic Byzantine agreement protocols, by extending almost-everywhere agreement to *full agreement* [KSSV06, KS09, KS11, BGH13, BCG21].

tion. In the case of fixed-round protocols, the works of [FM97, Mic17, FLL21] showed protocols achieving 2^{-r} error within $O(r)$ rounds. On the other hand, Karlin and Yao [KY86] and Chor, Merritt and Shmoys [CMS89] showed that any r-round protocol incurs an error probability of r^{-r} when the number of corruptions is linear, a bound that has recently been matched by Ghinea, Goyal and Liu-Zhang [GGL22]. For protocols with probabilistic termination, randomized broadcast with expected-constant number of rounds was achieved in the honest-majority setting [FM97, FG03, KK06], even under composition [CCGZ19, CCGZ21]. It was further shown that two rounds are unlikely to suffice for reaching agreement, even with weak guarantees, as long as $t > n/4$ [CHM+22] (as opposed to *three* rounds [Mic17]). In the dishonest-majority setting, there are sublinear-round broadcast protocols [GKKO07, FN09, CPS20, WXDS20], and even expected-constant-round protocols [WXSD20, SLM+23]. These results match the lower bound of $\Omega(n/(n-t))$ rounds (allowing up to constant failure probability) [GKKO07].

Outline of Paper

The paper is organized as follows. In Section 2, preliminary content including notations, security, and network model is introduced. In Section 3, we present the message-complexity lower bound for static corruptions. In Section 4, we present the locality lower bound for weakly adaptive corruptions. Finally, in Section 5, we describe a statically secure broadcast protocol with sub-quadratic communication and poly-logarithmic locality.

2 Preliminaries

In this section, we present the security model and preliminary definitions.

Notations. We use calligraphic letters to denote sets or distributions (e.g., $\mathcal{S}$), uppercase for random variables (e.g., R), lowercase for values (e.g., r), and sans-serif (e.g., A) for algorithms (i.e., Turing machines). For $n \in \mathbb{N}$, let $[n] = \{1, \ldots, n\}$. Let poly denote the set all positive polynomials and let PPT denote a probabilistic (interactive) Turing machines that runs in *strictly* polynomial time. We denote by κ the security parameter. A function $\nu \colon \mathbb{N} \mapsto [0,1]$ is *negligible*, denoted $\nu(\kappa) = \mathsf{negl}(\kappa)$, if $\nu(\kappa) < 1/p(\kappa)$ for every $p \in \mathsf{poly}$ and sufficiently large κ. Moreover, we say that $\nu \colon \mathbb{N} \mapsto [0,1]$ is *noticeable* if $\nu(\kappa) \geq 1/p(\kappa)$ for some $p \in \mathsf{poly}$ and sufficiently large κ. When using the $\tilde{O}(n)$ notation, polynomial factors in $\log(n)$ and the security parameter κ are omitted.

Protocols. All protocols considered in this paper are PPT (probabilistic polynomial time): the running time of every party is polynomial in the (common) security parameter κ, given as a unary string. For simplicity, we consider Boolean-input Boolean-output protocols, where apart from the common security parameter, a designated sender P_s has a single input bit, and each of the honest parties outputs a single bit. We note that our protocols can be used for broadcasting longer strings, with an additional dependency of the communication complexity on the input-string length.

As our main results are lower bounds, we consider protocols in the *correlated randomness* model; that is, prior to the beginning of the protocol π a trusted dealer samples values $(r_1, \ldots, r_n) \leftarrow \mathcal{D}_\pi$ from an efficiently sampleable known distribution $\mathcal{D}_\pi$ and gives the value r_i to party P_i. This model captures, for example, a trusted PKI setup for digital signatures and verifiable random functions (VRFs), where the dealer samples a public/private keys for each party and hands to each P_i its

secret key and a vector of all public keys; this is the setup needed for our upper bound result in Section 5. The model further captures more involved distributions, such as setup for *threshold signatures*, information-theoretic PKI [PW92], pairwise correlations for *oblivious transfer* [Bea95], and more.

We define the *view* of a party P_i as its setup information r_i, its random coins, possibly its input (in case P_i is the sender), and its set of all messages received during the protocol.

Communication model. The communication model that we consider is *synchronous*, meaning that protocols proceed in rounds. In each round every party can send a message to every other party over an authenticated channel, where the adversary can see the content of all transmitted messages, but cannot drop/inject messages. We emphasize that our lower bounds hold also in the private-channel setting which can be established over authenticated channels using public-key encryption and a PKI setup; our protocol construction only requires authenticated channels. It is guaranteed that every message sent in a round will arrive at its destination by the end of that round. The adversary is *rushing* in the sense that it can use the messages received by corrupted parties from honest parties in a given round to determine the corrupted parties' messages for that round.

Adversary model. The adversary runs in probabilistic polynomial time and may corrupt a subset of the parties and instruct them to behave in an arbitrary (malicious) manner. Some of our results (the lower bound in Section 3 and the protocol in Section 5) consider a static adversary that chooses which parties to corrupt *before* the beginning of the protocol, i.e., *before* the setup information is revealed to the parties. Note that this strengthens the lower bound, but provides a weaker feasibility result. Our second lower bound (Section 4) considers an adaptive adversary that can choose which parties to corrupt during the course of the protocol, based on information it dynamically learns. We consider the *atomic-multisend model* (also referred to as a weakly adaptive adversary), meaning that once a party P_i starts sending messages in a given round, it cannot be corrupted until it completes sending all messages for that round, and every message sent by P_i is delivered to its destination. This is weaker than the standard model for adaptive corruptions [Fel88, CFGN96, Can01] (also referred to as a strongly rushing adversary), which enables the adversary to corrupt a party at any point during the protocol and drop/change messages that were not delivered yet. Again, we note that the weaker model we consider yields a stronger lower bound. Further, in the stronger model, a result by Abraham et al. [ACD+19] rules out sub-quadratic protocols with linear resiliency, even in the honest-majority setting.

Broadcast. We consider the standard, property-based definition of broadcast.

Definition 2.1 (Broadcast protocol)**.** *An n-party protocol π, where a distinguished sender P_s holds an initial input message $x \in \{0,1\}$, is a* broadcast protocol *secure against t corruptions, if the following conditions are satisfied for any PPT adversary that corrupts up to t parties:*

- ***Termination:*** *There exists an a-priori-known round R such that the protocol is guaranteed to complete within R rounds (i.e., every so-far honest party produces an output value).*
- ***Agreement:*** *For every pair of parties P_i and P_j that are honest at the end of the protocol, if party P_i outputs y_i and party P_j outputs y_j, then $y_i = y_j$ with all but negligible probability in κ.*

- ***Validity:*** *If the sender is honest at the end of the protocol, then for every party* P_i *that is honest at the end of the protocol, if* P_i *outputs* y_i *then* $y_i = x$ *with all but negligible probability in* κ*.*

The communication locality [BGT13, BCDH18] of a protocol corresponds to the maximal degree of any honest party in the communication graph induced by the protocol execution. While defining the incoming communication edges to a party can be subtle (as adversarial parties may "spam" honest parties; see e.g., a discussion in [BCDH18]), out-edges of honest parties are clearly identifiable from the protocol execution. In this paper, we will focus on this simpler notion of output-locality, and use the terminology *locality* of the protocol to simply refer to this value. Our results provide a lower bound on output locality of given protocols, which in turn directly lower bounds standard locality as in [BGT13, BCDH18].

Definition 2.2 (Output locality)**.** *An* n*-party* t*-secure broadcast protocol* π *with setup distribution* $\mathcal{D}_\pi$ *has locality* ℓ*, if for every PPT adversary* Adv *corrupting up to* t *parties and every sender input* x *it holds that*

$$\Pr\left[\mathsf{OutEdges}(\pi, \mathsf{Adv}, \mathcal{D}_\pi, \kappa, x) > \ell\right] \leq \mathsf{negl}(\kappa),$$

where $\mathsf{OutEdges}(\pi, \mathsf{Adv}, \mathcal{D}_\pi, \kappa, x)$ *is the random variable of the maximum number of parties any honest party sends messages to, defined by running the protocol* π *with the adversary* Adv *and setup distribution* $\mathcal{D}_\pi$*, security parameter* κ *and sender input* x*. The probability is taken over the random coins of the honest parties, the random coins of* Adv*, and the sampling coins from the setup distribution* $\mathcal{D}_\pi$*.*

3 Message-Complexity Lower Bound for Static Corruptions

We begin with the proof of Theorem 1.2. The high-level idea of the lower bound is that if a protocol has $o(n^2)$ messages, then, with noticeable probability, a randomly chosen pair of parties do not communicate even under certain attacks.

Theorem 3.1 (Theorem 1.2, restated)**.** *Let* $\epsilon(n) \in o(1)$*. If there exists a broadcast protocol that is secure against* $(1 - \epsilon(n)) \cdot n$ *static corruptions, then the message complexity of the protocol is* $\Omega(n \cdot \frac{1}{\epsilon(n)})$*.*

Proof. Let $\psi(n) = \frac{1}{12\epsilon(n)}$ and let π be a broadcast protocol with message complexity $\mathsf{MC} = n \cdot \psi(n)$ that is secure against $(1 - \epsilon(n)) \cdot n$ static corruptions. (In fact, we will prove a stronger statement than claimed, where the message complexity of the protocol must be greater than $n \cdot \frac{1}{12\epsilon(n)}$.) Without loss of generality, we assume that the setup information sampled before the beginning of the protocol $(r_1, \ldots, r_n) \leftarrow \mathcal{D}_\pi$ includes the random string used by each party. That is, every party P_i generates its messages in each round as a function of r_i, possibly its input (if P_i is the sender), and its incoming messages in prior rounds. Again, without loss of generality, let P_1 denote be the sender, and split the remaining parties to two equal-size subsets $\mathcal{A}$ and $\mathcal{B}$ (for simplicity, assume that n is odd).

Consider the adversary Adv_1 that proceeds as follows:

1. Choose randomly a set $\mathcal{S} \subseteq \mathcal{A}$ of size $\epsilon(n) \cdot n - 1$ and a party $\mathsf{P}_{i^*} \in \mathcal{B}$.
2. Corrupt all parties except for $\mathcal{S} \cup \{\mathsf{P}_{i^*}\}$.

3. Receive the setup information of the corrupted parties $\{r_i \mid \mathsf{P}_i \notin \mathcal{S} \cup \{\mathsf{P}_{i^*}\}\}$.

4. Maintain two independent executions, denoted Exec_0 and Exec_1, as follows.

   - In the execution Exec_0, the adversary runs in its head the parties in $\mathcal{A} \setminus \mathcal{S}$ honestly on their setup information $\{r_i \mid \mathsf{P}_i \in \mathcal{A} \setminus \mathcal{S}\}$ and a copy of the sender, denoted P_1^0, running on input 0 and setup information r_1.

     The adversary communicates on behalf of the virtual parties in $(\mathcal{A} \setminus \mathcal{S}) \cup \{\mathsf{P}_1^0\}$ with the honest parties in $\mathcal{S}$ according to this execution. Every corrupted party in $\mathcal{B} \setminus \{\mathsf{P}_{i^*}\}$ crashes in this execution, and the adversary drops every message sent by the virtual parties in $(\mathcal{A} \setminus \mathcal{S}) \cup \{\mathsf{P}_1^0\}$ to P_{i^*} and does not deliver any message from P_{i^*} to these parties.
   - In the execution Exec_1, the adversary runs in its head the parties in $\mathcal{B} \setminus \{\mathsf{P}_{i^*}\}$ honestly on their setup information $\{r_i \mid \mathsf{P}_i \in \mathcal{B} \setminus \{\mathsf{P}_{i^*}\}\}$ and a copy of the sender, denoted P_1^1, running on input 1 and setup information r_1.

     The adversary communicates on behalf of the virtual parties in $(\mathcal{B} \setminus \{\mathsf{P}_{i^*}\}) \cup \{\mathsf{P}_1^1\}$ with the honest P_{i^*} according to this execution. Every corrupted party in $\mathcal{A} \setminus \mathcal{S}$ crashes in this execution, and the adversary drops every message sent by the virtual parties in $(\mathcal{B} \setminus \{\mathsf{P}_{i^*}\}) \cup \{\mathsf{P}_1^1\}$ to honest parties in $\mathcal{S}$ and does not deliver any message from $\mathcal{S}$ to these parties.

We start by defining a few notations. Consider the following random variables

$$\textsc{SetupAndCoins} = (R_1, \ldots, R_n, S, I^*)\,,$$

where $R_1, \ldots, R_n$ are distributed according to $\mathcal{D}_\pi$, and S takes a value uniformly at random in the subsets of $\mathcal{A}$ of size $\epsilon(n) \cdot n - 1$, and I^* takes a value uniformly at random in $\mathcal{B}$. During the proof, R_i represents the setup information (including private randomness) of party P_i, whereas the pair (S, I^*) corresponds to the random coins of the adversary Adv_1 (used for choosing $\mathcal{S}$ and P_{i^*}). Unless stated otherwise, all probabilities are taken over these random variables.

Let $\textsc{AttackMain}$ be the random variable defined by running the protocol π with the adversary Adv_1 over $\textsc{SetupAndCoins}$. That is, $\textsc{AttackMain}$ consists of a vector of $n+1$ views: of the honest parties in $S \cup \{\mathsf{P}_{I^*}\}$ and of the corrupted parties in $\mathcal{A} \setminus S$ and $\mathcal{B} \setminus \{\mathsf{P}_{I^*}\}$, where the i^{th} view is denoted by $\textsc{view}_i^{\mathsf{main}}$, and of two copies of the sender P_1^0 and P_1^1, denoted $\textsc{view}_{1\text{-}0}^{\mathsf{main}}$ and $\textsc{view}_{1\text{-}1}^{\mathsf{main}}$, respectively. Each view consists of the setup information R_i, possibly the input, and the set of received messages in each round. Specifically,

$$\textsc{AttackMain} = \left(\textsc{view}_{1\text{-}0}^{\mathsf{main}}, \textsc{view}_{1\text{-}1}^{\mathsf{main}}, \textsc{view}_2^{\mathsf{main}}, \ldots, \textsc{view}_n^{\mathsf{main}}\right).$$

Denote by $\mathcal{E}_{\mathsf{disconnect}}^{\mathsf{main}}$ the event that P_{I^*} and S do not communicate in $\textsc{AttackMain}$; that is, P_{I^*} does not send any message to parties in S (according to $\textsc{view}_{I^*}^{\mathsf{main}}$) and every party P_J with $J \in S$ does not send any message to P_{I^*} (according to $\textsc{view}_J^{\mathsf{main}}$). We proceed to prove that the event $\mathcal{E}_{\mathsf{disconnect}}^{\mathsf{main}}$ occurs with noticeable probability.

Lemma 3.2. $\Pr\left[\mathcal{E}_{\mathsf{disconnect}}^{\mathsf{main}}\right] \geq \frac{1}{3}$.

Proof. Denote by $\mathcal{E}_{\mathsf{S}\to\mathsf{P}}^{\mathsf{main}}$ the event that a party in S sends a message to P_{I^*} in $\textsc{AttackMain}$, and P_{I^*} did not send any message to any party in S in any prior round. We begin by upper bounding the probability of $\mathcal{E}_{\mathsf{S}\to\mathsf{P}}^{\mathsf{main}}$.

Claim 3.3. $\Pr\left[\mathcal{E}^{\mathsf{main}}_{\mathsf{S}\to\mathsf{P}}\right] \leq \frac{1}{3}$.

Proof. Consider a different adversary for π, denoted Adv_B, that statically corrupts all parties in $\mathcal{B}$ and crashes them (all other parties including the sender are honest). Let AttackCrashB denote the random variable defined by running the protocol π with the adversary Adv_B over SetupAndCoins, in which the honest sender's input is 1. That is, AttackCrashB consists of a vector of $n/2+1$ views: of the honest parties in $\mathcal{A}$, where the i^{th} view is denoted by $\text{VIEW}_i^{\mathsf{crash\text{-}B}}$, and the sender P_1 denoted by $\text{VIEW}_1^{\mathsf{crash\text{-}B}}$. Each view consists of the setup information R_i, the input 1 for P_1, and the set of received messages in each round. Specifically,

$$\text{AttackCrashB} = \left(\text{VIEW}_i^{\mathsf{crash\text{-}B}}\right)_{i\in\mathcal{A}\cup\{1\}}.$$

Denote by $\mathcal{E}^{\mathsf{crash\text{-}B}}_{\mathsf{S}\to\mathsf{P}}$ the event that a party in S sends a message to P_{I^*} in AttackCrashB such that P_{I^*} did not send any message to any party in S in any prior round. Note that as long as parties in S do not receive a message from P_{I^*} until some round ρ in AttackMain, their joint view is identically distributed as their joint view in AttackCrashB up until round ρ. Therefore,

$$\Pr\left[\mathcal{E}^{\mathsf{main}}_{\mathsf{S}\to\mathsf{P}}\right] = \Pr\left[\mathcal{E}^{\mathsf{crash\text{-}B}}_{\mathsf{S}\to\mathsf{P}}\right].$$

Note that, by the definition of Adv_B, the distribution of AttackCrashB, and therefore $\Pr\left[\mathcal{E}^{\mathsf{crash\text{-}B}}_{\mathsf{S}\to\mathsf{P}}\right]$, is *independent* of the random variables S and I^*. Hence, one can consider the mental experiment where $R_1,\ldots,R_n$ are first sampled for setting AttackCrashB, and later, S and I^* are independently sampled at random. This does not affect the event $\mathcal{E}^{\mathsf{crash\text{-}B}}_{\mathsf{S}\to\mathsf{P}}$.

Recall that the message complexity of π is $\mathsf{MC} = n\cdot\psi(n)$ for $\psi(n) = \frac{1}{12\epsilon(n)}$. Further, S is of size $|S| = \epsilon(n)\cdot n - 1$ and $|\mathcal{A}| = |\mathcal{B}| = n/2$. Observe that the message complexity upper-bounds the number of communication edges between $\mathcal{A}$ and $\mathcal{B}$. Further, the probability that a party in S talks first to P_{I^*} is upper-bounded by the probability that there exists a communication edge between S and P_{I^*}. Since S and I^* are uniformly distributed in $\mathcal{A}$ and $\mathcal{B}$, respectively, we obtain that this probability is bounded by

$$\begin{aligned}
\Pr\left[\mathcal{E}^{\mathsf{crash\text{-}B}}_{\mathsf{S}\to\mathsf{P}}\right] &\leq \mathsf{MC}\cdot\frac{1}{|\mathcal{B}|}\cdot\frac{|S|}{|\mathcal{A}|} \\
&= n\cdot\psi(n)\cdot\frac{1}{n/2}\cdot\frac{\epsilon(n)\cdot n-1}{n/2} \\
&\leq n\cdot\psi(n)\cdot\frac{1}{n/2}\cdot\frac{\epsilon(n)\cdot n}{n/2} \\
&= 4\cdot\psi(n)\cdot\epsilon(n) \\
&= \frac{4\cdot\epsilon(n)}{12\cdot\epsilon(n)} = \frac{1}{3}.
\end{aligned}$$

□

Similarly, denote by $\mathcal{E}^{\mathsf{main}}_{\mathsf{P}\to\mathsf{S}}$ the event that P_{I^*} sends a message to a party in S in AttackMain, such that no party in S sent a message to P_{I^*} in any prior round; i.e., changing the order from $\mathcal{E}^{\mathsf{main}}_{\mathsf{S}\to\mathsf{P}}$. We upper bound the probability of $\mathcal{E}^{\mathsf{main}}_{\mathsf{P}\to\mathsf{S}}$ in an analogous manner.

Claim 3.4. $\Pr\left[\mathcal{E}^{\mathsf{main}}_{\mathsf{P}\to\mathsf{S}}\right] \leq \frac{1}{3}$.

Proof. Consider a different adversary for π, denoted Adv_A, that statically corrupts all parties in $\mathcal{A}$ and crashes them. Let AttackCrashA be a random variable defined by running the protocol π with the adversary Adv_A over SetupAndCoins, in which the honest sender's input is 0. That is, AttackCrashA consists of a vector of $n/2+1$ views: of the honest parties in $\mathcal{B}$, where the i^{th} view is denoted by $\text{VIEW}_i^{\mathsf{crash\text{-}A}}$, and the sender P_1 denoted by $\text{VIEW}_1^{\mathsf{crash\text{-}A}}$. Each view consists of the setup information R_i, the input 0 for P_1, and the set of received messages in each round. Specifically,

$$\text{AttackCrashA} = \left(\text{VIEW}_i^{\mathsf{crash\text{-}A}}\right)_{i\in\mathcal{B}\cup\{1\}}.$$

Denote by $\mathcal{E}_{\mathsf{P}\to\mathsf{S}}^{\mathsf{crash\text{-}A}}$ the event that P_{I^*} sends a message to a party in S in AttackCrashA, and no party in S sent a message to P_{I^*} in any prior round. As long as P_{I^*} does not receive a message from parties in S until some round ρ in AttackMain, its view is identically distributed as its view in AttackCrashA up until round ρ. Therefore,

$$\Pr\left[\mathcal{E}_{\mathsf{P}\to\mathsf{S}}^{\mathsf{main}}\right] = \Pr\left[\mathcal{E}_{\mathsf{P}\to\mathsf{S}}^{\mathsf{crash\text{-}A}}\right].$$

An analogue analysis to the previous case shows that $\Pr\left[\mathcal{E}_{\mathsf{P}\to\mathsf{S}}^{\mathsf{crash\text{-}A}}\right] \leq 1/3$, as desired. □

Combined together, we get that

$$\Pr\left[\neg\mathcal{E}_{\mathsf{disconnect}}^{\mathsf{main}}\right] = \Pr\left[\mathcal{E}_{\mathsf{S}\to\mathsf{P}}^{\mathsf{main}} \cup \mathcal{E}_{\mathsf{P}\to\mathsf{S}}^{\mathsf{main}}\right] \leq \Pr\left[\mathcal{E}_{\mathsf{S}\to\mathsf{P}}^{\mathsf{main}}\right] + \Pr\left[\mathcal{E}_{\mathsf{P}\to\mathsf{S}}^{\mathsf{main}}\right] \leq \frac{2}{3}.$$

Therefore, $\Pr\left[\mathcal{E}_{\mathsf{disconnect}}^{\mathsf{main}}\right] \geq 1/3$. This concludes the proof of Lemma 3.2. □

We proceed to show that conditioned on $\mathcal{E}_{\mathsf{disconnect}}^{\mathsf{main}}$, *agreement* of the protocol π is broken. Denote by Y_i^{main} the random variable denoting the output of P_i according to AttackMain. Further, denote by J^* the random variable corresponding to the minimal value in S.

Lemma 3.5. $\Pr\left[Y_{I^*}^{\mathsf{main}} \neq Y_{J^*}^{\mathsf{main}} \mid \mathcal{E}_{\mathsf{disconnect}}^{\mathsf{main}}\right] \geq 1 - \mathsf{negl}(\kappa)$.

Proof. We begin by showing that conditioned on $\mathcal{E}_{\mathsf{disconnect}}^{\mathsf{main}}$, party P_{I^*} outputs 0 with overwhelming probability.

Claim 3.6. $\Pr\left[Y_{I^*}^{\mathsf{main}} = 0 \mid \mathcal{E}_{\mathsf{disconnect}}^{\mathsf{main}}\right] \geq 1 - \mathsf{negl}(\kappa)$.

Proof. Consider again the adversary Adv_A that statically corrupts all parties in $\mathcal{A}$ and crashes them, with the corresponding random variable AttackCrashA. Denote by $\mathcal{E}_{\mathsf{disconnect}}^{\mathsf{crash\text{-}A}}$ the event that P_{I^*} does not send any message to parties in S (according to $\text{VIEW}_{I^*}^{\mathsf{crash\text{-}A}}$). It holds that

$$\Pr\left[\mathcal{E}_{\mathsf{disconnect}}^{\mathsf{crash\text{-}A}}\right] = \Pr\left[\neg\mathcal{E}_{\mathsf{P}\to\mathsf{S}}^{\mathsf{crash\text{-}A}}\right] = 1 - \Pr\left[\mathcal{E}_{\mathsf{P}\to\mathsf{S}}^{\mathsf{crash\text{-}A}}\right] \geq 2/3.$$

First, since the sender is honest and has input 0, by *validity* all honest parties in $\mathcal{B}$ output 0 in such execution, except for negligible probability. This holds even conditioned on $\mathcal{E}_{\mathsf{disconnect}}^{\mathsf{crash\text{-}A}}$ (since $\mathcal{E}_{\mathsf{disconnect}}^{\mathsf{crash\text{-}A}}$ occurs with noticeable probability). Denote by $Y_i^{\mathsf{crash\text{-}A}}$ the random variable denoting the output of P_i according to AttackCrashA. Then,

$$\Pr\left[Y_{I^*}^{\mathsf{crash\text{-}A}} = 0 \;\middle|\; \mathcal{E}_{\mathsf{disconnect}}^{\mathsf{crash\text{-}A}}\right] \geq 1 - \mathsf{negl}(\kappa). \tag{1}$$

Second, note that conditioned on $\mathcal{E}^{\mathsf{crash\text{-}A}}_{\mathsf{disconnect}}$ (by an analogous analysis of Lemma 3.2, this probability is non-zero), the view of P_{I^*} is identically distributed in AttackCrashA as its view in AttackMain conditioned on $\mathcal{E}^{\mathsf{main}}_{\mathsf{disconnect}}$. Indeed, conditioned on $\mathcal{E}^{\mathsf{main}}_{\mathsf{disconnect}}$, party P_{I^*} receives messages only from corrupt parties in AttackMain, which are consistently simulating precisely this execution where $\mathcal{A}$ has crashed and the sender has input 0. Therefore,

$$\Pr\left[Y^{\mathsf{crash\text{-}A}}_{I^*} = 0 \;\middle|\; \mathcal{E}^{\mathsf{crash\text{-}A}}_{\mathsf{disconnect}}\right] = \Pr\left[Y^{\mathsf{main}}_{I^*} = 0 \;\middle|\; \mathcal{E}^{\mathsf{main}}_{\mathsf{disconnect}}\right]. \tag{2}$$

The proof follows from Equations 1 and 2. This concludes the proof of Claim 3.6. □

We proceed to show that, conditioned on $\mathcal{E}^{\mathsf{main}}_{\mathsf{disconnect}}$, parties in S output 1 with overwhelming probability under the attack of Adv_1. Recall that J^* denotes the random variable corresponding to the minimal value in S.

Claim 3.7. $\Pr\left[Y^{\mathsf{main}}_{J^*} = 1 \mid \mathcal{E}^{\mathsf{main}}_{\mathsf{disconnect}}\right] \geq 1 - \mathsf{negl}(\kappa)$.

Proof. The proof follows in nearly an identical manner. Namely, consider the adversary Adv_B that statically corrupts all parties in $\mathcal{B}$ and crashes them, and the random variable AttackCrashB. Denote by $\mathcal{E}^{\mathsf{crash\text{-}B}}_{\mathsf{disconnect}}$ the event that for every $J \in S$, party P_J does not send any message to P_{I^*} (according to $\text{VIEW}^{\mathsf{crash\text{-}B}}_J$). It holds that

$$\Pr\left[\mathcal{E}^{\mathsf{crash\text{-}B}}_{\mathsf{disconnect}}\right] = \Pr\left[\neg\mathcal{E}^{\mathsf{crash\text{-}B}}_{\mathsf{S}\to\mathsf{P}}\right] = 1 - \Pr\left[\mathcal{E}^{\mathsf{crash\text{-}B}}_{\mathsf{S}\to\mathsf{P}}\right] \geq 2/3.$$

Since the sender is honest and has input 1, by *validity* all honest parties in $\mathcal{A}$ output 1 except for negligible probability. This holds even conditioned on $\mathcal{E}^{\mathsf{crash\text{-}B}}_{\mathsf{disconnect}}$ (since $\mathcal{E}^{\mathsf{crash\text{-}A}}_{\mathsf{disconnect}}$ occurs with noticeable probability). Denote by $Y^{\mathsf{crash\text{-}B}}_i$ the random variable denoting the output of P_i according to AttackCrashB, and recall that J^* corresponds to the minimal value in S. Then,

$$\Pr\left[Y^{\mathsf{crash\text{-}B}}_{J^*} = 1 \;\middle|\; \mathcal{E}^{\mathsf{crash\text{-}B}}_{\mathsf{disconnect}}\right] \geq 1 - \mathsf{negl}(\kappa). \tag{3}$$

Conditioned on $\mathcal{E}^{\mathsf{crash\text{-}B}}_{\mathsf{disconnect}}$, the view of P_{J^*} is identically distributed in AttackCrashB as its view in AttackMain conditioned in $\mathcal{E}^{\mathsf{main}}_{\mathsf{disconnect}}$. Therefore,

$$\Pr\left[Y^{\mathsf{crash\text{-}B}}_{J^*} = 1 \;\middle|\; \mathcal{E}^{\mathsf{crash\text{-}B}}_{\mathsf{disconnect}}\right] = \Pr\left[Y^{\mathsf{main}}_{J^*} = 1 \;\middle|\; \mathcal{E}^{\mathsf{main}}_{\mathsf{disconnect}}\right]. \tag{4}$$

The proof follows from Equations 3 and 4. This concludes the proof of Claim 3.7. □

Since P_{I^*} and P_{J^*} are honest, the proof of Lemma 3.5 follows from Claim 3.6 and Claim 3.7. □

Collectively, we have demonstrated an adversarial strategy Adv_1 that violates the agreement property of protocol π with noticeable probability:

$$\begin{aligned}
\Pr\left[Y^{\mathsf{main}}_{I^*} \neq Y^{\mathsf{main}}_{J^*}\right] &= \Pr\left[Y^{\mathsf{main}}_{I^*} \neq Y^{\mathsf{main}}_{J^*} \mid \mathcal{E}^{\mathsf{main}}_{\mathsf{disconnect}}\right] \cdot \Pr\left[\mathcal{E}^{\mathsf{main}}_{\mathsf{disconnect}}\right] \\
&\quad + \Pr\left[Y^{\mathsf{main}}_{I^*} \neq Y^{\mathsf{main}}_{J^*} \mid \neg\mathcal{E}^{\mathsf{main}}_{\mathsf{disconnect}}\right] \cdot \Pr\left[\neg\mathcal{E}^{\mathsf{main}}_{\mathsf{disconnect}}\right] \\
&\geq \Pr\left[Y^{\mathsf{main}}_{I^*} \neq Y^{\mathsf{main}}_{J^*} \mid \mathcal{E}^{\mathsf{main}}_{\mathsf{disconnect}}\right] \cdot \Pr\left[\mathcal{E}^{\mathsf{main}}_{\mathsf{disconnect}}\right] \\
&\geq (1 - \mathsf{negl}(\kappa)) \cdot \frac{1}{3}.
\end{aligned}$$

Note that the attack succeeds for any choice of distribution for setup information, and that the adversarial strategy runs in polynomial time, thus applying even in the presence of computational hardness assumptions. This concludes the proof of Theorem 3.1. □

4 Locality Lower Bound for Adaptive Corruptions

We proceed with the proof of Theorem 1.3. Here we show how a weakly adaptive adversary that can corrupt $n/2+k$ parties can target any party of its choice and force a that party to communicate with k neighbors. We refer to Section 1.2 for a high-level overview of the attack.

Theorem 4.1 (Theorem 1.3, restated)**.** *Let $0 < k < (n-1)/2$ and let π be an n-party broadcast protocol secure against $t = n/2+k$ adaptive corruptions. Then, for any non-sender party P_{i^*} there exists a PPT adversary that can force the locality of P_{i^*} to be larger than k, except for negligible probability.*

Proof. Let π be a broadcast protocol that is secure against $t = n/2 + k$ adaptive corruptions. Without loss of generality, we assume that the setup information sampled before the beginning of the protocol $(r_1, \ldots, r_n) \leftarrow \mathcal{D}_\pi$ includes the random string used by each party. That is, every party P_i generates its messages in each round as a function of r_i, possibly its input (if P_i is the sender), and its incoming messages in prior rounds. Again, without loss of generality, let P_1 denote be the sender. Further, fix the party P_{i^*}, and split the remaining parties (without P_1 and P_{i^*}) to two equal-size subsets $\mathcal{S}_0$ and $\mathcal{S}_1$ (for simplicity, assume that n is even).

Consider the following adversary Adv that proceeds as follows:

1. Wait for the setup phase to complete. Later on, whenever corrupting a party P_i, the adversary receive its setup information r_i.
2. Corrupt the sender P_1.
3. Toss a random bit $b \leftarrow \{0,1\}$ and corrupt all parties in $\mathcal{S}_{1-b}$.
4. Maintain two independent executions, denoted Exec_0 and Exec_1, as follows.
   - In the execution Exec_b, the adversary runs in its head a copy of the sender, denoted P_1^b, honestly running on input b and setup r_1. The adversary communicates on behalf of the virtual party P_1^b, and eventually corrupted parties in $\mathcal{S}_b$, with all honest parties $\mathcal{S}_b \cup \{\mathsf{P}_{i^*}\}$ according to this execution. The virtual parties in $\mathcal{S}_{1-b}$ are emulated as crashed in this execution.

     Whenever P_{i^*} sends a message to a party $\mathsf{P}_i \in \mathcal{S}_b$ this party gets corrupted and ignores this message (i.e., the adversary does not deliver messages from P_{i^*} to P_i).
   - In the execution Exec_{1-b}, the adversary runs in its head the parties in $\mathcal{S}_{1-b}$ honestly on their setup information $\{r_i \mid \mathsf{P}_i \in \mathcal{S}_{1-b}\}$ and a copy of the sender, denoted P_1^{1-b}, running on input $1-b$ and setup r_1. The adversary communicates on behalf of the virtual parties in $(\mathcal{S}_{1-b}) \cup \{\mathsf{P}_1^{1-b}\}$ with P_{i^*} according to this execution. The honest parties in $\mathcal{S}_b$ are emulated as crashed in this execution; that is, the adversary drops every message sent by the virtual parties in $(\mathcal{S}_{1-b}) \cup \{\mathsf{P}_1^{1-b}\}$ to $\mathcal{S}_b$ and does not deliver any message from $\mathcal{S}_b$ to these parties.

     Whenever P_{i^*} sends a message to a party $\mathsf{P}_i \in \mathcal{S}_{1-b}$ this party ignores this message (i.e., the adversary does not deliver the message to P_i).

We start by defining a few notations. Consider the following random variables

$$\textsc{SetupAndCoins} = (R_1, \ldots, R_n, B)\,,$$

where $R_1, \ldots, R_n$ are distributed according to $\mathcal{D}_\pi$, and B takes a value uniformly at random in $\{0,1\}$. During the proof, R_i represents the setup information (including private randomness) of party P_i, whereas B corresponds to the adversarial choice of which set to corrupt. Unless stated otherwise, all probabilities are taken over these random variables.

Let AttackMain be the random variable defined by running the protocol π with the adversary Adv over SetupAndCoins. That is, AttackMain consists of a vector of $n+1$ views: of the parties in $S_b \cup \{\mathsf{P}_{I^*}\}$, of the corrupted parties in $\mathcal{S}_{1-b}$, where the i^{th} view is denoted by $\text{VIEW}_i^{\mathsf{main}}$, and of two copies of the sender P_1^0 and P_1^1, denoted $\text{VIEW}_{1\text{-}0}^{\mathsf{main}}$ and $\text{VIEW}_{1\text{-}1}^{\mathsf{main}}$, respectively. Each view consists of the setup information R_i, possibly the input (for the sender), and the set of received messages in each round. Specifically,

$$\text{AttackMain} = \left(\text{VIEW}_{1\text{-}0}^{\mathsf{main}}, \text{VIEW}_{1\text{-}1}^{\mathsf{main}}, \text{VIEW}_2^{\mathsf{main}}, \ldots, \text{VIEW}_n^{\mathsf{main}}\right).$$

Denote by $\mathcal{E}_{\mathsf{low\text{-}locality}}^{\mathsf{main}}$ the event that the output-locality of P_{I^*} is at most k in AttackMain; that is, P_{I^*} sends messages to at most k parties (according to $\text{VIEW}_{I^*}^{\mathsf{main}}$). If $\Pr[\mathcal{E}_{\mathsf{low\text{-}locality}}^{\mathsf{main}}] = \mathsf{negl}(\kappa)$, then the proof is completed. Otherwise, it holds that $\Pr[\mathcal{E}_{\mathsf{low\text{-}locality}}^{\mathsf{main}}]$ is non-negligible (in particular, $\Pr[\mathcal{E}_{\mathsf{low\text{-}locality}}^{\mathsf{main}}] > 0$). We will show that conditioned on $\mathcal{E}_{\mathsf{low\text{-}locality}}^{\mathsf{main}}$, *agreement* is broken. Denote by Y_i^{main} the random variable denoting the output of P_i according to AttackMain.

First, note that conditioned on $\mathcal{E}_{\mathsf{low\text{-}locality}}^{\mathsf{main}}$, the view of P_{i^*} is identically distributed no matter which set $\mathcal{S}_b$ is corrupted.

Claim 4.2. *For every $\beta \in \{0,1\}$ it holds that*

$$\Pr\left[Y_{i^*}^{\mathsf{main}} = \beta \mid \mathcal{E}_{\mathsf{low\text{-}locality}}^{\mathsf{main}} \cap (B=0)\right] = \Pr\left[Y_{i^*}^{\mathsf{main}} = \beta \mid \mathcal{E}_{\mathsf{low\text{-}locality}}^{\mathsf{main}} \cap (B=1)\right].$$

Proof. By the construction of Adv, for each $\beta \in \{0,1\}$ party P_{i^*} receives from the parties in $\mathcal{S}_\beta$ and from P_1 messages that correspond to an execution by honest parties on sender input β as if the parties in $\mathcal{S}_{1-\beta}$ all crashed, and where every party in $\mathcal{S}_\beta$ that P_{i^*} talks to ignores its message (since P_{i^*} talks to at most k parties conditioned on $\mathcal{E}_{\mathsf{low\text{-}locality}}^{\mathsf{main}}$, the adversary can corrupt all of them).

Further, P_{i^*} receives from the parties in $\mathcal{S}_{1-\beta}$ and from P_1 messages that correspond to a simulated execution by honest parties on sender input $1-\beta$ as if the parties in $\mathcal{S}_\beta$ all crashed, and where every party in $\mathcal{S}_{1-\beta}$ that P_{i^*} talks to ignores its message.

Clearly, the view of P_{i^*} is identically distributed in both cases; hence, its output bit is identically distributed as well. □

We proceed to show that conditioned on $\mathcal{E}_{\mathsf{low\text{-}locality}}^{\mathsf{main}}$, party P_{i^*} outputs 0 for $B=0$ and outputs 1 for $B=1$.

Claim 4.3. *For every $\beta \in \{0,1\}$ it holds that*

$$\Pr\left[Y_{i^*}^{\mathsf{main}} = \beta \mid \mathcal{E}_{\mathsf{low\text{-}locality}}^{\mathsf{main}} \cap (B=\beta)\right] = 1 - \mathsf{negl}(\kappa).$$

Proof. Consider a different adversary for π, denoted Adv_β, that proceeds as follows:

1. Wait for the setup phase to complete.
2. Corrupt all parties in $\mathcal{S}_{1-\beta}$ and crash them.

3. Whenever P_{i^*} sends a message to a party $\mathsf{P}_i \in \mathcal{S}_\beta$ this party gets corrupted and ignores this message (i.e., the adversary does not deliver messages from P_{i^*} to P_i).

Let $\text{AttackCrashS}_\beta$ be the random variable defined by running the protocol π with the adversary Adv_β over SetupAndCoins, in which the honest sender's input is β. That is, $\text{AttackCrashS}_\beta$ consists of a vector of $n/2$ views: of the parties in $\mathcal{S}_\beta \cup \{\mathsf{P}_{i^*}\}$ (both honest and corrupted), where the i^{th} view is denoted by $\text{VIEW}_i^{\mathsf{crash\text{-}S}_\beta}$, and the sender P_1 denoted by $\text{VIEW}_1^{\mathsf{crash\text{-}S}_\beta}$. Each view consists of the setup information R_i, the input β for P_1, and the set of received messages in each round. Specifically,

$$\text{AttackCrashS}_\beta = \left(\text{VIEW}_i^{\mathsf{crash\text{-}S}_\beta}\right)_{i \in \mathcal{S}_\beta \cup \{1, i^*\}}.$$

Let us denote by $\mathcal{E}^{\mathsf{crashS}_\beta}_{\mathsf{low\text{-}locality}}$ the event that the output-locality of P_{I^*} is at most k in $\text{AttackCrashS}_\beta$; that is, P_{I^*} sends messages to at most k parties (according to $\text{VIEW}_{i^*}^{\mathsf{crash\text{-}S}_\beta}$). If $\Pr[\mathcal{E}^{\mathsf{crashS}_\beta}_{\mathsf{low\text{-}locality}}] = \mathsf{negl}(\kappa)$, then Adv_β can force the locality of P_{I^*} to be high in $\text{AttackCrashS}_\beta$, and the proof is completed. Otherwise, it holds that $\Pr[\mathcal{E}^{\mathsf{crashS}_\beta}_{\mathsf{low\text{-}locality}}]$ is non-negligible.

Note that since $|\mathcal{S}_\beta| = (n-1)/2$ and $k < (n-1)/2$, then conditioned on $\mathcal{E}^{\mathsf{crashS}_\beta}_{\mathsf{low\text{-}locality}}$ there exists at least one remaining honest party in $\mathcal{S}_\beta$ at the end of the execution with Adv_β. By *validity*, each such honest party must output β with overwhelming probability. Denote by $Y_i^{\mathsf{crash\text{-}S}_\beta}$ the random variable denoting the output of P_i according to $\text{AttackCrashS}_\beta$. Denote by J^* the random variable corresponding to the minimal index of an honest party in $\mathcal{S}_\beta$ at the end of the execution with Adv_β. Then

$$\Pr\left[Y_{J^*}^{\mathsf{crash\text{-}S}_\beta} = \beta \mid \mathcal{E}^{\mathsf{crashS}_\beta}_{\mathsf{low\text{-}locality}}\right] = 1 - \mathsf{negl}(\kappa). \tag{5}$$

Further, note that the set of all honest parties in $\mathcal{S}_\beta$ and their joint view in an execution with Adv_β conditioned on $\mathcal{E}^{\mathsf{crashS}_\beta}_{\mathsf{low\text{-}locality}}$ is identically distributed as in an execution with Adv conditioned on $\mathcal{E}^{\mathsf{main}}_{\mathsf{low\text{-}locality}} \cap (B = \beta)$. Therefore,

$$\Pr\left[Y_{J^*}^{\mathsf{crash\text{-}S}_\beta} = \beta \mid \mathcal{E}^{\mathsf{crashS}_\beta}_{\mathsf{low\text{-}locality}}\right] = \Pr\left[Y_{J^*}^{\mathsf{main}} = \beta \mid \mathcal{E}^{\mathsf{main}}_{\mathsf{low\text{-}locality}} \cap (B = \beta)\right]. \tag{6}$$

Finally, by *agreement*, since both P_{J^*} and P_{i^*} are honest at the end of the execution with Adv, conditioned on $\mathcal{E}^{\mathsf{main}}_{\mathsf{low\text{-}locality}} \cap (B = \beta)$, it holds that

$$\Pr\left[Y_{i^*}^{\mathsf{main}} = \beta \mid \mathcal{E}^{\mathsf{main}}_{\mathsf{low\text{-}locality}} \cap (B = \beta)\right] = \Pr\left[Y_{J^*}^{\mathsf{main}} = \beta \mid \mathcal{E}^{\mathsf{main}}_{\mathsf{low\text{-}locality}} \cap (B = \beta)\right] - \mathsf{negl}(\kappa). \tag{7}$$

The claim follows from Equations 5, 6, and 7. □

By Claim 4.2 and Claim 4.3 it follows that $\Pr[\mathcal{E}^{\mathsf{main}}_{\mathsf{low\text{-}locality}}] = \mathsf{negl}(\kappa)$. This concludes the proof of Theorem 4.1. □

5 Statically Secure Sub-Quadratic Broadcast

In this section we prove Proposition 1.1 by presenting a broadcast protocol secure against a constant fraction of static corruptions that requires $\tilde{O}(n)$ bits of total communication, given a trusted-PKI setup for VRFs. The protocol is balanced, and each party communicates $\mathsf{polylog}(n) \cdot \mathsf{poly}(\kappa)$ bits.

Proposition 5.1 (Proposition 1.1, restated)**.** *Let $0 < \epsilon < 1$ be any constant. Assuming a trusted-PKI for VRFs and signatures, it is possible to compute broadcast with $\tilde{O}(n)$ total communication ($\tilde{O}(1)$ per party) facing a static adversary corrupting $(1-\epsilon) \cdot n$ parties.*

Our protocol is a simple variant of Chan, Pass, and Shi [CPS20], where every step that requires all-to-all communication is substituted by a more communication-efficient message-propagation mechanism. We first describe the message-propagation mechanism, which follows the spirit of Tsimos, Loss and Papamanthou [TLP22], and afterwards the modified broadcast protocol of [CPS20].

A message-propagation mechanism. Consider the problem where each party has (possibly) an input message it would like to disseminate to all the other parties. This can trivially be solved by letting each party send its input message to all other parties, which would incur a communication complexity of $n^2 \cdot \ell$, where ℓ is the length of the input message.

In some cases, however, we are interested in disseminating only messages of a certain type. More precisely, we want that if any honest party has a message of type T, then all honest parties obtain at least one message of type T, but we do not need that all parties obtain all input messages of type T.

Definition 5.2 (Message-propagation protocol)**.** *Let T be a predicate. An n-party protocol π, where each party P_i has an initial input x_i (or no input), is a* message-propagation *protocol for type-T messages secure against t static corruptions, if for any PPT adversary that statically corrupts up to t parties, the following holds except for negligible probability in κ: If any honest party holds an input of type-T, then all honest parties output a value of type-T.*

The trivial approach described above (where every party sends its input message to every other party) still requires quadratic communication, since a linear number of parties may distribute a type-T message towards all parties. A more efficient solution employs instead a flooding mechanism over a communication graph that forms an expander (i.e., a sparse graph with strong connectivity properties). More concretely, it is possible to form a communication graph where each party is connected only to $O(\log(n)+\kappa)$ other parties, and the honest parties form a connected component except with negligible probability in κ.

Each party P_i can then send its input to its neighbors (if the input is of type T), and if P_i has no input or an input that is not of type T, party P_i can simply forward to its neighbors the first type-T message that it received. It is easy to see that, since the honest parties form a connected component, if any honest party has a message of type T, then all honest parties obtain at least one message of type T. Moreover, the communication complexity is $O(n \cdot (\log(n)+\kappa) \cdot \ell)$, where ℓ is the length of a type-T message. We describe the protocol in Figure 3 and obtain the following lemma.

Lemma 5.3. *Let κ be a security parameter, let n be the number of parties, and let $0 < \epsilon < 1$ be a constant. Protocol $\mathsf{Flood}(\mathsf{T}, n, \epsilon, \kappa)$ is a message-propagation protocol for type-T messages, secure against static $(1-\epsilon) \cdot n$ corruptions. The communication complexity is $O(n \cdot (\log(n)+\kappa) \cdot \ell)$ bits, where ℓ is the size of a type-T message.*

Proof. From [LZMM+22, Lem. 15], we know that the communication graph induced by the honest parties during an execution of Flood is connected and has diameter at most $\rho = 7 \ln\left(\frac{n}{2 \cdot (\ln(n)+\kappa)}\right) + 2$ with overwhelming probability in κ. Therefore, if any honest party has a type-T input message, all honest parties receive a type-T message within ρ rounds.

Since each honest party only sends a type-T message at most once, and the neighborhood of a party is of size $O(\log(n)+\kappa)$, the claimed communication complexity follows. □

$\mathsf{Flood}(\mathsf{T}, n, \epsilon, \kappa)$

Parameters: T is the type of messages to propagate, n is the number of parties, ϵ is the fraction of honest parties, and κ is the security parameter.

Variables: P_i sets local variables $\mathcal{N}_i = \emptyset$ and $\mathsf{Relayed}_i = 0$.

Message propagation:
Each party P_i has an input x_i (no input is interpreted as $\perp$).
Each party P_i locally adds P_j to the set $\mathcal{N}_i$ with uniform probability $p_{\mathsf{flood}} := \frac{\log(n)+\kappa}{\epsilon n}$.[a]
Let $\rho = 7 \ln\left(\frac{n}{2\cdot(\ln(n)+\kappa)}\right) + 2$. Each party P_i does the following.
Round 1: If the input message x_i is of type T, send x_i to each $\mathsf{P}_j \in \mathcal{N}_i$, set $\mathsf{Relayed}_i = 1$, and set $y_i = x_i$.
for *each round* $r \in \{2, \ldots, \rho\}$ **do**
  Let $\mathcal{S}$ be the messages received in round $r-1$. If there is a message $m \in \mathcal{S}$ of type-T and $\mathsf{Relayed}_i = 0$, then send m to each $\mathsf{P}_j \in \mathcal{N}_i$, set $\mathsf{Relayed}_i = 1$, and set $y_i = m$.
Output y_i.

[a] We note that with static corruptions, each party P_i can maintain the same neighborhood set $\mathcal{N}_i$ across several instances of Flood.

Figure 3: Message-propagation mechanism for n parties for messages of type T.

Chan et al.'s modified protocol. We describe a modified version of Chan et al.'s broadcast protocol, where every all-to-all communication step is simply substituted by the message-propagation mechanism described above.

Following [ACD+19, CPS20], we describe the protocol in a hybrid world assuming an ideal functionality $\mathcal{F}_{\mathsf{mine}}$ parameterized by probability $p := \min\{1, \frac{\kappa+1}{\epsilon n}\}$, which can be realized assuming a trusted-PKI for VRFs as setup. $\mathcal{F}_{\mathsf{mine}}$ serves as a committee-election oracle, and has the following interface:

- Mining: when a party P_i calls $\mathcal{F}_{\mathsf{mine}}.\mathtt{mine}(b)$ on a bit $b \in \{0,1\}$ for the first time, $\mathcal{F}_{\mathsf{mine}}$ flips a p-weighted coin and returns the result $b' \in \{0,1\}$. (1 indicates success, 0 indicates failure.) Calling $\mathcal{F}_{\mathsf{mine}}.\mathtt{mine}(b)$ again in the future returns the same result b'.
- Verifying: any node can call $\mathcal{F}_{\mathsf{mine}}.\mathtt{verify}(b, i)$ on a bit b and index i. If P_i has already called $\mathcal{F}_{\mathsf{mine}}.\mathtt{mine}(b)$ and received result $b' = 1$, then $\mathcal{F}_{\mathsf{mine}}.\mathtt{verify}(b, i)$ returns 1; otherwise, $\mathcal{F}_{\mathsf{mine}}.\mathtt{verify}(b, i)$ returns 0.

At the start of the protocol, parties invoke the ideal functionality $\mathcal{F}_{\mathsf{mine}}$ to choose a committee $\mathcal{C}$ of size $|\mathcal{C}| = R = O(\kappa)$ parties, which will contain at least one honest party (with high probability). The protocol is based on the Dolev-Strong protocol [DS83], where only the committee members $\mathcal{C}$ (and the sender) contribute signatures on the input message and every protocol stage r is split into two mini-stages. We define a message of type-$\mathsf{T}_{b,r}$ as a message that contains the bit b along with at least r correct signatures, including the signatures from the sender and at least $r-1$ distinct committee-parties.

- Each party P_i keeps a set Ext_i (initially empty). In stage 0, the sender signs its input bit and sends b and the signature to all parties (this is a $\mathsf{T}_{b,1}$ message).
- For each stage $r = 1$ to $R+1$, each party P_i does the following.

1. First mini-stage: for every $b \notin \mathsf{Ext}_i$ such that P_i has received a type-$\mathsf{T}_{b,r}$ message, add b to Ext_i and run the message-propagation protocol $\mathsf{Flood}(\mathsf{T}_{b,r}, n, \epsilon, \kappa)$ to distribute a type-$\mathsf{T}_{b,r}$ to all parties (and wait until the protocol ends).
2. Second mini-stage: for every $b \notin \mathsf{Ext}_i$ such that a committee member $\mathsf{P}_i \in \mathcal{C}$ has received a type-$\mathsf{T}_{b,r}$ message, pick any such message, add b to Ext_i, create a type-$\mathsf{T}_{b,r+1}$ message by adding its own signature, and run the message-propagation protocol $\mathsf{Flood}(\mathsf{T}_{b,r+1}, n, \epsilon, \kappa)$ to distribute a type-$\mathsf{T}_{b,r+1}$ message to all parties (and wait until the protocol ends).

- Stage $R+2$: Each party P_i outputs the bit contained in Ext_i if $|\mathsf{Ext}_i| = 1$; otherwise, output 0.

See Figure 4 for the formal description of the protocol.

$\mathsf{Flood\text{-}BC}^{n,\epsilon,\kappa}_{\mathcal{F}_{\mathsf{mine}}}$

Parameters: Let n be the number of parties, let ϵ be the fraction of honest nodes, and let κ the security parameter. Let $(\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Ver})$ be a digital signature scheme.

Setup: Let $\mathcal{F}_{\mathsf{mine}}$ be the committee-election oracle. The parties have access to a PKI setup where each party P_i samples $(\mathtt{pk}_i, \mathtt{sk}_i) \leftarrow \mathsf{KeyGen}(1^\kappa)$ and all parties obtain $(\mathtt{pk}_1, \ldots, \mathtt{pk}_n)$.

Input: Let b_s be the input of the sender P_s.

Stage 0 **(Initialization):**

- The sender P_s computes $\sigma = \mathsf{Sign}_{\mathtt{sk}_s}(b_s)$ and sends the message (b_s, σ) to all parties.
- Each party P_i sets its extracted set $\mathsf{Ext}_i \leftarrow \emptyset$ and also queries $\mathcal{F}_{\mathsf{mine}}$, parameterized with $p := \min\{1, \frac{\kappa+1}{\epsilon n}\}$. Let $\mathcal{C}$ denote the elected committee.

Let the number of stages be $R := \frac{3}{\epsilon}(\kappa + 1)$. In the following, we denote by type-$\mathsf{T}_{b,r}$ as a message that contains the bit b along with at least r correct signatures (i.e., such that Ver outputs 1) from public keys of distinct committee-parties and also including the sender.

Stage $r = 1, \ldots, R+1$**:**

1. (Stage r.1) Each party P_i does the following:
   - For each bit b such that P_i has received a type-$\mathsf{T}_{b,r}$ message m and $b \notin \mathsf{Ext}_i$: add b to Ext_i and invoke protocol $\mathsf{Flood}(\mathsf{T}_{b,r}, n, \epsilon, \kappa)$, to propagate one type-$\mathsf{T}_{b,r}$ message m.
2. (Stage r.2) Each party $\mathsf{P}_i \in \mathcal{C}$ does the following:
   - For each bit b such that P_i has received a type-$\mathsf{T}_{b,r}$ message m and $b \notin \mathsf{Ext}_i$: add b to Ext_i and if P_i can create a type $\mathsf{T}_{b,r+1}$ message m by adding a signature $\mathsf{Sign}_{\mathtt{sk}_i}(b)$ to the received $\mathsf{T}_{b,r}$ message, execute protocol $\mathsf{Flood}(\mathsf{T}_{b,r+1}, n, \epsilon, \kappa)$, to propagate one type-$\mathsf{T}_{b,r+1}$ message m.

Stage $R+2$ **(Termination):** Each party P_i does the following:

- If $|\mathsf{Ext}_i| = 1$, then P_i outputs the unique bit $b_i \in \mathsf{Ext}_i$; else, P_i outputs the default bit 0.

Figure 4: Chan et al.'s modified Broadcast Protocol in the $\mathcal{F}_{\mathsf{mine}}$-hybrid world

We proceed to prove Proposition 5.1.

Proof of Proposition 5.1. Termination is trivial, since all parties output a value at stage $R+2$.

To prove validity, let the sender be an honest party with input b. At the end of stage 0, all honest parties receive a type-$\mathsf{T}_{b,1}$ message. Therefore, each party P_i adds b to the set Ext_i in the first mini-stage of stage 1. Moreover, no other value is added to Ext_i (except for negligible probability), since the sender does not sign any other value. At stage $R+2$, all honest parties output b except for negligible probability.

To prove agreement, we show that the sets Ext_i of each honest party P_i contain the same set of values the end of the protocol at stage $R+2$.

- First, consider the case where the first honest party P_i that adds a bit b to its set Ext_i, does so in the first mini-stage of stage r. This means that P_i received a type-$\mathsf{T}_{b,r}$ message and a type-$\mathsf{T}_{b,r}$ message was subsequently propagated through Flood. Note that since a type-$\mathsf{T}_{b,r}$ message contains r signatures, and $\mathcal{C}$ contains at least one honest party, then it holds that $r < R+1$ (otherwise an honest committee-member added b to its extracted set previously, contradicting the assumption that P_i is the first honest party that adds b to its set).

  The message-propagation mechanism then ensures that all honest parties receive a type-$\mathsf{T}_{b,r}$ message at the end of this first mini-stage. Therefore, in the second mini-stage of stage r, there is an honest committee-member that can form a type-$\mathsf{T}_{b,r+1}$ message, which is distributed through Flood. Therefore, in the next stage $r+1$ all parties add b to their extracted set.

- Second, consider the case where the first honest party P_i that adds a bit b to its set Ext_i, does so in the second mini-stage of stage r. Since the second mini-stage is only executed by committee members, $\mathsf{P}_i \in \mathcal{C}$. Moreover, P_i received a type-$\mathsf{T}_{b,r}$ message for the first time and can form a type-$\mathsf{T}_{b,r+1}$ message, which is distributed through Flood. Note that $r < R+1$ since the type-$\mathsf{T}_{b,r}$ message does not contain P_i's signature. Therefore, in the next stage $r+1$ (first mini-stage), all parties add b to their extracted set.

In each stage r, there are at most 4 invocations to Flood (one per bit, per mini-stage), and the message contains up to $R+1$ signatures and a bit value. Assuming each signature is of size $O(\kappa)$, the size of the message is bounded by $\ell = O((R+1)\kappa + 1) = O(\kappa^2)$. Since the number of stages invoking Flood is $R+1 = O(\kappa)$ and the cost of each instance of Flood is $O(n \cdot (\log(n) + \kappa) \cdot \ell)$, the total incurred communication complexity is $O(\kappa \cdot n \cdot (\log(n) + \kappa) \cdot \kappa^2) = O(\kappa^3 n \log(n) + \kappa^4 n)) = \tilde{O}(n)$ and the per-party communication is $\tilde{O}(1)$. □

Acknowledgments

Part of E. Blum's work was done while the author was an intern at NTT Research. E. Boyle's research is supported in part by AFOSR Award FA9550-21-1-0046 and ERC Project HSS (852952). R. Cohen's research is supported in part by NSF grant no. 2055568 and 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 authors and do not necessarily reflect the views of Algorand Foundation. Part of C.D. Liu-Zhang's work was done while the author was at NTT Research.

Bibliography

[ACD+19] Ittai Abraham, T.-H. Hubert Chan, Danny Dolev, Kartik Nayak, Rafael Pass, Ling Ren, and Elaine Shi. Communication complexity of byzantine agreement, revisited. In *Proceedings of the*

38th Annual ACM Symposium on Principles of Distributed Computing (PODC), pages 317–326, 2019.

[ADD+22] Nicolas Alhaddad, Sourav Das, Sisi Duan, Ling Ren, Mayank Varia, Zhuolun Xiang, and Haibin Zhang. Balanced byzantine reliable broadcast with near-optimal communication and improved computation. In *Proceedings of the 41st Annual ACM Symposium on Principles of Distributed Computing (PODC)*, pages 399–417, 2022.

[BCDH18] Elette Boyle, Ran Cohen, Deepesh Data, and Pavel Hubáček. Must the communication graph of MPC protocols be an expander? In *38th Annual International Cryptology Conference (CRYPTO), part III*, pages 243–272, 2018.

[BCG21] Elette Boyle, Ran Cohen, and Aarushi Goel. Breaking the $O(\sqrt{n})$-bit barrier: Byzantine agreement with polylog bits per party. In *Proceedings of the 40th Annual ACM Symposium on Principles of Distributed Computing (PODC)*, pages 319–330, 2021.

[Bea95] Donald Beaver. Precomputing oblivious transfer. In *14th Annual International Cryptology Conference (CRYPTO)*, pages 97–109, 1995.

[Ben83] Michael Ben-Or. Another advantage of free choice: Completely asynchronous agreement protocols (extended abstract). In *Proceedings of the 2nd Annual ACM Symposium on Principles of Distributed Computing (PODC)*, pages 27–30, 1983.

[BGH13] Nicolas Braud-Santoni, Rachid Guerraoui, and Florian Huc. Fast Byzantine agreement. In *Proceedings of the 32th Annual ACM Symposium on Principles of Distributed Computing (PODC)*, pages 57–64, 2013.

[BGP92] Piotr Berman, Juan A Garay, and Kenneth J Perry. Bit optimal distributed consensus. *Computer Science Research*, pages 313–322, 1992.

[BGT13] Elette Boyle, Shafi Goldwasser, and Stefano Tessaro. Communication locality in secure multi-party computation - how to run sublinear algorithms in a distributed setting. In *Proceedings of the 10th Theory of Cryptography Conference (TCC)*, pages 356–376, 2013.

[BKLL20] Erica Blum, Jonathan Katz, Chen-Da Liu-Zhang, and Julian Loss. Asynchronous Byzantine agreement with subquadratic communication. In *Proceedings of the 18th Theory of Cryptography Conference (TCC), part I*, pages 353–380, 2020.

[Can01] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *Proceedings of the 42nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 136–145, 2001.

[CCGZ19] Ran Cohen, Sandro Coretti, Juan A. Garay, and Vassilis Zikas. Probabilistic termination and composability of cryptographic protocols. *Journal of Cryptology*, 32(3):690–741, 2019.

[CCGZ21] Ran Cohen, Sandro Coretti, Juan A. Garay, and Vassilis Zikas. Round-preserving parallel composition of probabilistic-termination cryptographic protocols. *Journal of Cryptology*, 34(2):12, 2021.

[CFGN96] Ran Canetti, Uriel Feige, Oded Goldreich, and Moni Naor. Adaptively secure multi-party computation. In *Proceedings of the 28th Annual ACM Symposium on Theory of Computing (STOC)*, pages 639–648, 1996.

[CGZ23] Ran Cohen, Juan A. Garay, and Vassilis Zikas. Completeness theorems for adaptively secure broadcast. In *43rd Annual International Cryptology Conference (CRYPTO), part I*, pages 3–38, 2023.

[CHM+22] Ran Cohen, Iftach Haitner, Nikolaos Makriyannis, Matan Orland, and Alex Samorodnitsky. On the round complexity of randomized byzantine agreement. *Journal of Cryptology*, 35(2):10, 2022.

[CKS20] Shir Cohen, Idit Keidar, and Alexander Spiegelman. Not a COINcidence: Sub-quadratic asynchronous Byzantine agreement WHP. In *Proceedings of the 34th International Symposium on Distributed Computing (DISC)*, pages 25:1–25:17, 2020.

[CM19] Jing Chen and Silvio Micali. Algorand: A secure and efficient distributed ledger. *Theoretical Computer Science*, 777:155–183, 2019.

[CMS89] Benny Chor, Michael Merritt, and David B Shmoys. Simple constant-time consensus protocols in realistic failure models. *Journal of the ACM*, 36(3):591–614, 1989.

[CPS20] T.-H. Hubert Chan, Rafael Pass, and Elaine Shi. Sublinear-round byzantine agreement under corrupt majority. In *Proceedings of the 23rd International Conference on the Theory and Practice of Public-Key Cryptography (PKC), part II*, pages 246–265, 2020.

[CW92] Brian A Coan and Jennifer L Welch. Modular construction of a byzantine agreement protocol with optimal message bit complexity. *Information and Computation*, 97(1):61–85, 1992.

[DGH+87] Alan J. Demers, Daniel H. Greene, Carl Hauser, Wes Irish, John Larson, Scott Shenker, Howard E. Sturgis, Daniel C. Swinehart, and Douglas B. Terry. Epidemic algorithms for replicated database maintenance. In *Proceedings of the 6th Annual ACM Symposium on Principles of Distributed Computing (PODC)*, pages 1–12, 1987.

[DLS88] Cynthia Dwork, Nancy A. Lynch, and Larry J. Stockmeyer. Consensus in the presence of partial synchrony. *Journal of the ACM*, 35(2):288–323, 1988.

[Dol82] Danny Dolev. The byzantine generals strike again. *J. Algorithms*, 3(1):14–30, 1982.

[DPPU88] Cynthia Dwork, David Peleg, Nicholas Pippenger, and Eli Upfal. Fault tolerance in networks of bounded degree. *SIAM Journal on Computing*, 17(5):975–988, 1988.

[DR85] Danny Dolev and Rüdiger Reischuk. Bounds on information exchange for Byzantine agreement. *Journal of the ACM*, 32(1):191–204, 1985.

[DS83] Danny Dolev and H. Raymond Strong. Authenticated algorithms for Byzantine agreement. *SIAM Journal on Computing*, 12(4):656–666, 1983.

[DXR21] Sourav Das, Zhuolun Xiang, and Ling Ren. Asynchronous data dissemination and its applications. In *Proceedings of the 28th ACM Conference on Computer and Communications Security (CCS)*, pages 2705–2721, 2021.

[Fel88] Paul Feldman. *Optimal Algorithms for Byzantine Agreement*. PhD thesis, Stanford University, 1988. https://dspace.mit.edu/handle/1721.1/14368.

[FG03] Matthias Fitzi and Juan A. Garay. Efficient player-optimal protocols for strong and differential consensus. In *Proceedings of the 22th Annual ACM Symposium on Principles of Distributed Computing (PODC)*, pages 211–220, 2003.

[FL82] Michael J. Fischer and Nancy A. Lynch. A lower bound for the time to assure interactive consistency. *Information Processing Letters*, 14(4):183–186, 1982.

[FLL21] Matthias Fitzi, Chen-Da Liu-Zhang, and Julian Loss. A new way to achieve round-efficient byzantine agreement. In *Proceedings of the 40th Annual ACM Symposium on Principles of Distributed Computing (PODC)*, pages 355–362, 2021.

[FLM86] Michael J. Fischer, Nancy A. Lynch, and Michael Merritt. Easy impossibility proofs for distributed consensus problems. *Distributed Computing*, 1(1):26–39, 1986.

[FLP83] Michael J. Fischer, Nancy A. Lynch, and Mike Paterson. Impossibility of distributed consensus with one faulty process. In *Proceedings of the Second ACM SIGACT-SIGMOD Symposium on Principles of Database Systems*, pages 1–7, 1983.

[FM97] Pesech Feldman and Silvio Micali. An optimal probabilistic protocol for synchronous Byzantine agreement. *SIAM Journal on Computing*, 26(4):873–933, 1997.

[FN09] Matthias Fitzi and Jesper Buus Nielsen. On the number of synchronous rounds sufficient for authenticated byzantine agreement. In *Proceedings of the 23th International Symposium on Distributed Computing (DISC)*, pages 449–463, 2009.

[GGL22] Diana Ghinea, Vipul Goyal, and Chen-Da Liu-Zhang. Round-optimal byzantine agreement. In *41st Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT), part I*, pages 96–119, 2022.

[GKKO07] Juan A. Garay, Jonathan Katz, Chiu-Yuen Koo, and Rafail Ostrovsky. Round complexity of authenticated broadcast with a dishonest majority. In *Proceedings of the 48th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 658–668, 2007.

[GKKZ11] Juan A. Garay, Jonathan Katz, Ranjit Kumaresan, and Hong-Sheng Zhou. Adaptively secure broadcast, revisited. In *Proceedings of the 30th Annual ACM Symposium on Principles of Distributed Computing (PODC)*, pages 179–186, 2011.

[GM93] Juan A. Garay and Yoram Moses. Fully polynomial byzantine agreement in t+1 rounds. In *Proceedings of the 25th Annual ACM Symposium on Theory of Computing (STOC)*, pages 31–41, 1993.

[HKK08] Dan Holtby, Bruce M. Kapron, and Valerie King. Lower bound for scalable Byzantine agreement. *Distributed Computing*, 21(4):239–248, 2008.

[HZ10] Martin Hirt and Vassilis Zikas. Adaptively secure broadcast. In *29th Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT)*, pages 466–485, 2010.

[KK06] Jonathan Katz and Chiu-Yuen Koo. On expected constant-round protocols for Byzantine agreement. In *25th Annual International Cryptology Conference (CRYPTO)*, pages 445–462, 2006.

[KS09] Valerie King and Jared Saia. From almost everywhere to everywhere: Byzantine agreement with $\tilde{o}(n^{3/2})$ bits. In *Proceedings of the 23th International Symposium on Distributed Computing (DISC)*, pages 464–478, 2009.

[KS11] Valerie King and Jared Saia. Breaking the $O(n^2)$ bit barrier: Scalable Byzantine agreement with an adaptive adversary. *Journal of the ACM*, 58(4):18:1–18:24, 2011. A preliminary version appeared at PODC'10.

[KSSV00] Richard M. Karp, Christian Schindelhauer, Scott Shenker, and Berthold Vöcking. Randomized rumor spreading. In *Proceedings of the 41st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 565–574, 2000.

[KSSV06] Valerie King, Jared Saia, Vishal Sanwalani, and Erik Vee. Scalable leader election. In *Proceedings of the 17th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 990–999, 2006.

[KY86] Anna Karlin and Andrew Yao. Probabilistic lower bounds for byzantine agreement. *Unpublished document*, 1986.

[LSP82] Leslie Lamport, Robert E. Shostak, and Marshall C. Pease. The Byzantine generals problem. *ACM Transactions on Programming Languages and Systems*, 4(3):382–401, 1982.

[LZMM+22] Chen-Da Liu-Zhang, Christian Matt, Ueli Maurer, Guilherme Rito, and Søren Eller Thomsen. Practical provably secure flooding for blockchains. In *28th International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT), part I*, pages 774–805, 2022.

[LZMT22] Chen-Da Liu-Zhang, Christian Matt, and Søren Eller Thomsen. Asymptotically optimal message dissemination with applications to blockchains. Cryptology ePrint Archive, Paper 2022/1723, 2022. https://eprint.iacr.org/2022/1723.

[Mic17] Silvio Micali. Very simple and efficient Byzantine agreement. In *Proceedings of the 8th Annual Innovations in Theoretical Computer Science (ITCS) conference*, pages 6:1–6:1, 2017.

[MR21] Atsuki Momose and Ling Ren. Optimal communication complexity of authenticated byzantine agreement. In *Proceedings of the 35th International Symposium on Distributed Computing (DISC)*, pages 32:1–32:16, 2021.

[MRV99] Silvio Micali, Michael O. Rabin, and Salil P. Vadhan. Verifiable random functions. In *Proceedings of the 40th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 120–130, 1999.

[PSL80] Marshall C. Pease, Robert E. Shostak, and Leslie Lamport. Reaching agreement in the presence of faults. *Journal of the ACM*, 27(2):228–234, 1980.

[PW92] Birgit Pfitzmann and Michael Waidner. Unconditional Byzantine agreement for any number of faulty processors. In *Proceedings of the 9th Annual Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 339–350, 1992.

[Rab83] Michael O. Rabin. Randomized byzantine generals. In *Proceedings of the 24th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 403–409, 1983.

[SLM+23] Shravan Srinivasan, Julian Loss, Giulio Malavolta, Kartik Nayak, Charalampos Papamanthou, and Sri Aravinda Krishnan Thyagarajan. Transparent batchable time-lock puzzles and applications to byzantine consensus. In *Proceedings of the 26th International Conference on the Theory and Practice of Public-Key Cryptography (PKC), part I*, pages 554–584, 2023.

[TLP22] Georgios Tsimos, Julian Loss, and Charalampos Papamanthou. Gossiping for communication-efficient broadcast. In *42nd Annual International Cryptology Conference (CRYPTO), part III*, pages 439–469, 2022.

[WXDS20] Jun Wan, Hanshen Xiao, Srinivas Devadas, and Elaine Shi. Round-efficient byzantine broadcast under strongly adaptive and majority corruptions. In *Proceedings of the 18th Theory of Cryptography Conference (TCC), part I*, pages 412–456, 2020.

[WXSD20] Jun Wan, Hanshen Xiao, Elaine Shi, and Srinivas Devadas. Expected constant round byzantine broadcast under dishonest majority. In *Proceedings of the 18th Theory of Cryptography Conference (TCC), part I*, pages 381–411, 2020.