Nearly Quadratic Broadcast Without Trusted Setup Under Dishonest Majority

Georgios Tsimos
tsimos@umd.edu
University of Maryland

Julian Loss
jloss@umiacs.umd.edu
University of Maryland

Charalampos Papamanthou
cpap@umd.edu
University of Maryland

July 15, 2020

Abstract

Broadcast (BC) is a crucial ingredient for many protocols in distributed computing and cryptography. In this paper we study its communication complexity against an adversary that controls a majority of the parties. In this setting, all known protocols either exhibit a communication complexity of more than $O(n^3)$ bits (where n is the number of parties) or crucially rely on a trusted party to generate cryptographic keys before the execution of the protocol. We give the first protocol for BC that achieves $\tilde{O}(n^2 \cdot \kappa)$ bits of communication (where κ is the security parameter) under a dishonest majority and minimal cryptographic setup assumptions, i.e., where no trusted setup is required and parties just need to generate their own cryptographic keys. Our protocol is randomized and combines the classic Dolev-Strong protocol with network gossiping techniques to minimize communication. Our analysis of the main random process employs Chernoff bounds for negatively-associated variables and might be of independent interest.

1 Introduction

Consensus is the fundamental distributed computing problem of agreeing on a common output v among n mutually distrustful parties, some t of which may be corrupted. In the sender-centric consensus version of *broadcast* (BC), a designated sender s distributes a value v such that (1) all parties output the same value v' (consistency) (2) all parties output v in case s is honest (validity). BC lies at the core of many cryptographic protocols such as secret sharing and multi-party computation (MPC). More recently, it has also proven useful in the context of blockchain protocols (e.g., [15]).

Broadcast without trusted setup. As expected, feasibility and quality of BC protocols greatly depends on the underlying assumptions. For example, in absence of cryptography (such as digital signatures), no BC protocol can tolerate more than $t = n/3$ malicious parties [13]. However, when digital signatures are used, it is widely known that BC can be solved for arbitrary $t < n$ malicious parties with $O(n^3 \cdot \kappa)$ bits of communication [5], where κ is the security parameter. Digital signatures are considered a minimal cryptographic assumption in that they require *no trusted setup*: All parties register their public keys to a public bulletin board before the start of the protocol and no assumption is made on how parties generate their keys. This model is known as the *bulletin board PKI* model.

Better communication complexity in the trusted PKI model. To obtain BC with better (subcubic) communication complexity, one can strengthen the underlying (cryptographic) assumptions. In particular all existing protocols with less than $O(n^3)$ bits of communication (e.g., [1, 4]) are in the *trusted PKI model* which stands in stark contrast to the bulletin board PKI model: In the trusted PKI model there is a *trusted setup* phase where a trusted party, Alice, generates all keys

Table 1: Comparison of RandAuthBroadcast, in terms of communication complexity, to existing BC protocols. We denote with n the number of parties and ϵ is a fixed constant in $(0, 1)$.

protocol	model	communication	adversary	malicious parties
Abraham et al. [1]	trusted PKI	$\tilde{O}(n \cdot \kappa)$	adaptive	$< n/2$
Chan et al. [4]	trusted PKI	$\tilde{O}(n^2 \cdot \kappa)$	adaptive	$< (1-\epsilon) \cdot n$
Dolev and Strong [5]	bulletin board PKI	$O(n^3 \cdot \kappa)$	adaptive	$< n$
RandAuthBroadcast	bulletin board PKI	$\tilde{O}(n^2 \cdot \kappa)$	static	$< (1-\epsilon) \cdot n$

honestly and distributes them to the protocol participants while at the same time she must remain uncompromised for the rest of the protocol. Other drawbacks of the trusted PKI model include lack for support in dynamic settings, e.g., in blockchain systems, where new parties frequently join (and leave) as well as bad usability due to their requirement of a very specific (and often not very efficient) type of signature scheme such as VRFs or threshold signatures [2, 14] (Note that protocols for bulletin board PKIs are far more convenient, as they work with *any digital signature scheme*.) Unsurprisingly, using a trusted PKI, BC can easily be solved within $O(n^2 \cdot \kappa)$ bits of communication for any $t < (1-\epsilon) \cdot n$, where $0 < \epsilon < 1$. For a complete comparison of different BC protocols using different models and assumptions see Table 1.

Given the importance of removing any centralized trust in distributed systems (which is the goal of distributed systems to begin with) while still having an efficient protocol, we ask the following fundamental question:

Is there a protocol that solves BC for $t < n$ corruptions with $o(n^3)$ bits of communication in the bulletin board PKI model, i.e., with no trusted setup?

Our result. We resolve this question by introducing RandAuthBroadcast (see Figure 2), the first protocol for BC that achieves communication complexity of $\tilde{O}(n^2 \cdot \kappa)$ in the bulletin board PKI model (In particular, the exact communication complexity in bits is $O(n^2 \cdot f(n) \cdot \kappa)$ for $f(n) = \omega(1) \cdot \log n$, which guarantees a negligible probability of failure, as required in a cryptographic setting.) Our protocol is randomized and works for $t < (1-\epsilon) \cdot n$, where $0 < \epsilon < 1$ is a constant. On the downside, we assume a static model of corruption where the adversary must decide which t parties to corrupt before the execution—but the corrupted parties are byzantine (In Section 5 we discuss challenges in achieving adaptive security.) At a technical level, our protocol combines techniques used in gossip networks with the structure of the Dolev-Strong protocol to facilitate efficient communication. Although our approach is rather intuitive, its analysis is subtle, e.g., requiring a proof for negatively-associated random variables. For this reason, our main propagation protocol AddRandomEdges (see Section 3) could be of independent interest.

1.1 Related Work

The problem of BC was originally introduced in the celebrated work of Lamport, Shostak, and Pease [13]. Their work also gave the first (setup-free) protocol for $t < n/3$ and showed optimality of their parameters. However, their solution required an exponential amount of communication and was soon improved upon by protocols requiring only polynomial amounts of communication [7, 10]. More recently, a line of work initiated by King et al. [12, 11, 3] gave setup-free protocols for the case of $t < n/3$ that require $\tilde{O}(n^{3/2})$ communication. For the setting of $t < n$ corruptions, Dolev and Strong [5] gave the first protocol with polynomial efficiency. Their protocol uses a bulletin board PKI, requires $O(n^3 \cdot \kappa)$ bits of communication, and solves BC for any $t < n$. Much more recently work of Chan et al. gives a protocol that requires $\tilde{O}(n^2 \cdot \kappa)$ bits of communication and

requires trusted setup. In the range of $t < n/3$ and $t < n/2$, the works of Micali [15], Micali and Vaikuntnathan [16], and Abraham et al. [1] present solution with subquadratic communication complexity using trusted setup. We give an overview over communication efficient protocols in Table 1. Somewhat surprisingly, in the setting with setup (for $t < n$), any efficiency improvement to the early work of Dolev and Strong has been aimed exclusively at improving the *round complexity* rather than the communication complexity. This has been the subject of several works [9, 8, 4, 17].

1.2 Organization

We introduce basic notation, concentration inequalities, the corruption and network model, and definitions of broadcast protocol in Section 2. In Section 3, we analyze the properties of the random process AddRandomEdges that occurs in our protocol. In Section 4, we show how to use these properties to build our protocol RandAuthBroadcast. We conclude in Section 5 with some discussion concerning challenges to make the protocol adaptively secure as well as to further reduce the communication complexity.

2 Preliminaries and Notation

We now continue with some preliminary notations and definitions that we will be using throughout the paper. We will make use of the notation $f(x) \nearrow$ in x to denote that function $f(x)$ is *increasing in* x. Also for $m \in \mathbb{N}$ and a set S, such that $m \leq |S|$, we denote with $m \sim S$ the set S' that contains m elements of S, chosen uniformly at random with replacement.

2.1 Bulletin board PKI

We assume that parties share a *public key infrastructure* (PKI). That is, each party i has a secret key sk_i and a public key pk_i, where pk_i is known to all parties. The secret key sk_i and the public key pk_i are not assumed to be computed in a trusted manner. Each party i can compute a *signature* σ on a message m via $\sigma \leftarrow sig(sk_i, m)$. Later, anybody can verify σ via calling $ver(pk_i, \sigma, m)$. As is standard for this line of work, we assume that signatures are *idealized*, that is, we treat signatures as *perfectly unforgeable* in the sense that it is impossible, without sk_i, to create a signature σ on a message m such that $ver(pk_i, \sigma, m) = 1$. We also assume *perfect correctness*, meaning that for any m, $ver(pk_i, sig(sk_i, m), m) = 1$. To simplify notation, we write $sig_i(m)$ to indicate $sig(sk_i, m)$ and $ver_i(\sigma, m)$ to indicate $ver(pk_i, \sigma, m)$.

2.2 Network Model

We consider the standard synchronous model of communication. In this model, parties are assumed to share a global clock that progresses at the same rate for all parties. Furthermore, they are connected via pairwise, authenticated channels. Any message that is sent by an honest party at time T is guaranteed to arrive at every honest party at time $T + \Delta$, where Δ is the maximum network delay. In particular, this means that messages of honest parties can not be dropped from the network and are always delivered. As such we consider protocols that execute in a round based fashion, where every round in the protocol is of length Δ. It is assumed that all parties start executing the r-th round of a protocol at time $(r-1) \cdot \Delta$. Let $\mathcal{M}$ be a set of messages and $\mathcal{P}$ be a set of parties. When a party i calls Send$(\mathcal{M}, \mathcal{P})$ at round r, then the set of messages $\mathcal{M}$ is delivered to parties in $\mathcal{P}$ at round $r + 1$. Finally, when a party i calls Receive() at round r, then all messages that were sent to i at round $r - 1$ via Send commands are stored in i's local storage.

2.3 Adversary Model

Let $0 < \epsilon < 1$ be a positive constant. We consider a polynomial-time adversary that can corrupt up to $t \leq (1-\epsilon) \cdot n$ parties in a malicious fashion, i.e., can make them deviate from a protocol description arbitrarily. In particular, the adversary learns a corrupted party's entire state, including secret keys and its entire view of the protocol execution up to that point. In addition, we consider a *rushing adversary* that can observe the honest parties' messages in any synchronous round r of a protocol, and delay them until the end of that round. In this way, it can choose its own messages for that round before delivering any of the honest messages. We consider the *static model of corruption*. In this model, the adversary chooses what parties to corrupt *before* an execution of the protocol.

2.4 Definitions

We begin with the formal definition of a t-secure broadcast protocol.

Definition 1 (t-secure broadcast)**.** *A protocol Π executed by n parties, where a designated party s (the sender) holds an input v is a* t-secure broadcast protocol *if the following properties are satisfied with overwhelming probability in n whenever at most t parties are corrupted:*

> ***Validity:*** *if the sender is honest, all honest parties output v.*
>
> ***Consistency:*** *all honest parties output the same value v'.*

Note that we require our result to hold with overwhelming probability in n (which is $1 - negl(n)$ where $negl(n)$ is a function that is $o(1/n^c)$ for all constants c), which is much stronger than whp. This is due to the considered adversary being polynomial-time since we are in the cryptographic setting. Our protocol is randomized and for its analysis we will be using the notion of negatively associated random variables.

Definition 2 (Negatively associated random variables [6])**.** *Let $\mathbf{X} = \{X_1, \ldots, X_n\}$ be a set of random variables. We say that the random variables $\mathbf{X}$ are negatively associated (or that $\mathbf{X}$ is NA) if for every two disjoint sets $I, J \subseteq [n]$, $\mathbb{E}[f(X_i, i \in I) g(X_j, j \in J)] \leq \mathbb{E}[f(X_i, i \in I)]\,\mathbb{E}[g(X_j, j \in J)]$ for all functions $f : R^{|I|} \to R$ and $g : R^{|J|} \to R$ that are both non-decreasing or both non-increasing.*

Some key properties that we will be using to prove negative association are listed in the Appendix. See Properties 1, 2, 3 and 4. It is easy to show that a standard Chernoff bound holds for negatively associated variables. For completeness, we present the proof in the Appendix.

Lemma 1 (Lower tail for negatively associated variables)**.** *Let $X_1, \ldots, X_n$ be negatively associated Boolean random variables and X be their sum. Let $\mu = \mathbb{E}[X]$. Then, for any $0 < \delta < 1$, the standard Chernoff bound holds*

$$\Pr[X < (1-\delta)\mu] \leq \left(\frac{e^{-\delta}}{(1-\delta)^{(1-\delta)}} \right)^{\mu}.$$

3 The Procedure AddRandomEdges

In this section, we present a random procedure, that we call AddRandomEdges. Procedure AddRandomEdges simulates the propagation of messages from honest nodes to the rest of the network in our protocol, between two consecutive rounds. Separately analyzing it allows us to argue about the consistency and validity of RandAuthBroadcast in a more structured manner.

1: **procedure** $\mathbf{Z} \leftarrow$ AddRandomEdges(S_1, S_2, S_3, S, m)
Input: Partition of n nodes into disjoint sets S_1, S_2, S_3; $S \subseteq S_1$; Integer $m \leq n$.
Output: Boolean variables $\mathbf{Z}$.

2: **for** every node $v \in S$ **do**
3: **for** $i = 1, \ldots, m$ **do**
4: Pick a node u uniformly at random from the set of nodes $S_1 \cup S_2 \cup S_3$;
5: Add an edge (v, u);
6: Let $Z_u \in \{0, 1\}$ such that $Z_u = 1$ iff $u \in S_2$ has nonzero degree;
7: **return** $\{Z_u\}_{u \in S_2}$;

Figure 1: The AddRandomEdges procedure.

AddRandomEdges works over a graph whose vertices are partitioned into three disjoint sets: In particular, given a partition of n nodes into three disjoint sets S_1, S_2, S_3, and a set $S \subseteq S_1$, AddRandomEdges picks, for every node in S, m random neighbors from $S_1 \cup S_2 \cup S_3$ and outputs a Boolean variable for every node u in S_2 indicating whether u has obtained a neighbor as a result of this procedure (The value m will be taken to be roughly $\omega(1) \cdot \log n$ by our protocol to achieve negligible probability of failure.) Looking ahead, S is the set of nodes that send a message q at a specific round, S_2 is the set of nodes that have not heard q before, and we are trying to figure out how many in S_2 will get q. The procedure is formally described in Figure 1.

We now prove some useful properties of AddRandomEdges. In Lemma 2 we show that the output random Boolean variables are negatively associated. Then in Lemma 3 we show that the number of nodes that acquire an edge in S_2 is roughly at least two times the number of nodes in S. Intuitively this will allow us to show that enough nodes will be seeing the propagated messages in our final protocol.

Lemma 2. *Random variables $\{Z_u\}_{u \in S_2}$ output by* AddRandomEdges(S_1, S_2, S_3, S, m) *are negatively associated (as defined in Definition 2).*

Proof. Let us define indicator random variables

$$X_{vuk} = \begin{cases} 1, & \text{if node } u \in S_2 \text{ is chosen as the } k\text{-th neighbor of } v \in S; \\ 0, & \text{else} \end{cases},$$

where $k \leq m$. For each pair (v, k), it holds that $\sum_{u \in S_2} X_{vuk} = 1$, since exactly one node $u \in S_2$ is chosen as the k-th neighbor of v. Thus, by Property 3 (also Lemma 8 in [6]) it holds that each set $\mathcal{X}_{(v,k)} = \{X_{vuk}\}_{u \in S_2}$ is negatively associated. Furthermore, for each pair $(v, k) \neq (v', k')$ the sets $\mathcal{X}_{(v,k)}, \mathcal{X}_{(v',k')}$ contain mutually independent random variables, since choosing the neighbors of v and v' are independent procedures. Thus, by Property 1 in the Appendix (also Proposition 7.1 in [6]) it holds that the set $\mathcal{X} = \{X_{vuk}\}$ for $v \in S$, $u \in S_2$ and $k = 1, \ldots, m$ is negatively associated. Now, for every $u \in S_2$, let us define the sets $U_u = \{X_{vuk}\}_{v \in S, k \in [m]}$, which are pairwise disjoint. Also, for every $u \in S_2$, let us also define functions $h_u(X_{vuk}, U_u) = \sum_{X_{vuk} \in U_u} X_{vuk}$. Note that function h_u counts the incoming edges towards u from all $v \in S$. Each such function is a non-decreasing function and thus by Property 2 (also Proposition 7.2 in [6]) the set of random variables $\{h_u\}_{u \in S_2}$ is negatively associated. Observe now that the variables Z_u for $u \in S_2$ output by AddRandomEdges can be defined as

$$Z_u = f(Y_u) := \begin{cases} 0, & \text{if } Y_u = 0; \\ 1, & \text{if } Y_u > 0. \end{cases}$$

Since function f is non-decreasing, by Property 2 (also Proposition 7.2 in [6]) we have that $\{Z_u\}_{u\in S_2}$ are negatively associated. □

Lemma 3. *Let (S_1, S_2, S_3) be a partition of n nodes into disjoint sets with $\tau \le |S_1| \le \epsilon\cdot n/3$, $|S_2| = \epsilon\cdot n - |S_1|$, $|S_3| = n - \epsilon\cdot n$, where $\epsilon \in (0,1)$ is a constant and $\tau \ge 1$. Let also $S \subseteq S_1$ with $|S| \ge 2\cdot\tau/3$ and let $\{Z_u\}_{u\in S_2}$ be the random variables output by* AddRamdomEdges(S_1, S_2, S_3, S, m). *Then for $m \ge 15/\epsilon$,*

$$\Pr\left[\sum_{u\in S_2} Z_u \ge 2\cdot\tau\right] \ge 1 - p\,,$$

where $p = \frac{\epsilon\cdot n}{5}\cdot e^{-\epsilon\cdot m/9}$.

Proof. The proof will consider two cases, one for $|S_1| > \epsilon\cdot n/6$ and one for $|S_1| \le \epsilon\cdot n/6$.

Case $|S_1| > \epsilon\cdot n/6$: For this case we have that $|S| > \epsilon\cdot n/9$ and therefore the probability that $Z_u = 0$ for a fixed $u \in S_2$ is

$$(1-1/n)^{m\cdot|S|} > (1-1/n)^{m\cdot\epsilon\cdot n/9} \le e^{-\epsilon\cdot m/9}\,.$$

Note now that since $|S_2| = \epsilon\cdot n - |S_1| \ge 2\cdot\epsilon\cdot n/3 \ge 2\cdot\tau$ it is

$$\begin{aligned}\Pr\left[\sum_{u\in S_2} Z_u \ge 2\cdot\tau\right] &\ge \Pr\left[\sum_{u\in S_2} Z_u = |S_2|\right] = \Pr\left[\bigcap_{u\in S_2} Z_u = 1\right] = 1 - \Pr\left[\bigcup_{u\in S_2} Z_u = 0\right]\\ &\ge 1 - (\epsilon\cdot n - |S_1|)\cdot e^{-\epsilon\cdot m/9} > 1 - \frac{\epsilon\cdot n}{5}\cdot e^{-\epsilon\cdot m/9}\,.\end{aligned}$$

Case $|S_1| \le \epsilon\cdot n/6$: By Lemma 2, random variables Z_u ($u \in S_2$) are negatively associated. By Lemma 1, we can use the lower tail Chernoff bound for $Z = \sum_{u\in S_2} Z_u$, i.e.,

$$\Pr[Z < (1-\delta)\mu] < \left(\frac{e^{-\delta}}{(1-\delta)^{1-\delta}}\right)^{\mu}\,.$$

Note that $\mu = \mathbb{E}[Z] = \sum_{u\in S_2}\Pr[Z_u = 1] = (\epsilon\cdot n - |S_1|)\left(1 - (1-1/n)^{m\cdot|S|}\right)$. Therefore

$$\begin{aligned}\mu &\ge (5/6)\cdot\epsilon\cdot n\cdot\left(1-(1-1/n)^{m\cdot|S|}\right) \text{ (since } |S_1| \le \epsilon\cdot n/6)\\ &\ge (5/6)\cdot\epsilon\cdot n\cdot(1-(1-1/n)^{m}) \text{ (since } |S| \ge 1)\\ &\ge (5/6)\cdot\epsilon\cdot m\cdot(1-(1-1/m)^{m}) \text{ (since } n \ge m \text{ and } x\cdot(1-(1-1/x)^m) \nearrow \text{ in } x)\\ &\ge (5/6)\cdot\epsilon\cdot m\cdot(1-1/e)\\ &> 0.5\cdot\epsilon\cdot m\,.\end{aligned}$$

For $\delta = 1/2$ this yields

$$\Pr[Z < \mu/2] < \left(\frac{e^{-0.5}}{(0.5)^{0.5}}\right)^{0.5\cdot\epsilon\cdot m} = \left(\frac{2}{e}\right)^{0.25\cdot\epsilon\cdot m}\,.$$

Recall however that we must bound the probability $\Pr[Z < 2\cdot\tau]$. Therefore it is enough to also show that $\mu \ge 4\cdot\tau$. Indeed, starting with the expression $\mu = (\epsilon\cdot n - |S_1|)\left(1-(1-1/n)^{m\cdot|S|}\right)$

we have that

$$\begin{aligned}
\mu &\geq 5 \cdot |S_1| \cdot \left(1 - \left(1 - \frac{\epsilon}{6|S_1|}\right)^{m \cdot |S|}\right) \text{ (since } n \geq \frac{6|S_1|}{\epsilon} \text{ and } \mu \nearrow \text{ in } n) \\
&\geq 5 \cdot \tau \cdot \left(1 - \left(1 - \frac{\epsilon}{6\tau}\right)^{m \cdot |S|}\right) \text{ (since } |S_1| \geq \tau \text{ and } 5x\left(1 - (1 - \epsilon/6x)^{m \cdot |S|}\right) \nearrow \text{ in } x) \\
&\geq 5 \cdot \tau \cdot \left(1 - \left(1 - \frac{\epsilon}{6\tau}\right)^{2 \cdot m \cdot \tau/3}\right) \text{ (since } |S| \geq 2 \cdot \tau/3) \\
&\geq 5 \cdot \tau \cdot \left(1 - e^{-m \cdot \epsilon/9}\right) \qquad (1) \\
&> 4 \cdot \tau \text{ (since } m \geq 15/\epsilon)\,,
\end{aligned}$$

where to derive Inequality 1, we used the properties that for $0 < x \leq 1$ it is $(1-x)^{1/x} \leq e^{-1}$, for $a, b, c > 0$ it is $a \leq b \Leftrightarrow a^c \leq b^c$, and the fact that $\tau \geq 1 \Rightarrow \tau \geq \epsilon/6 \Rightarrow \frac{\epsilon}{6\tau} \in (0, 1]$.

Since now $1 - \frac{\epsilon \cdot n}{5} \cdot e^{-\epsilon \cdot m/9} < 1 - (2/e)^{0.25 \cdot \epsilon \cdot m}$, the lemma follows. □

4 The Protocol RandAuthBroadcast

We now describe our protocol RandAuthBroadcast. For simplicity, we will describe our protocol for the case where the values that are agreed upon are from the binary domain, but it is trivial to generalize to an arbitrary domain of values.

4.1 Intuition: From Send-To-All to Gossiping

Our protocol is inspired by the classic protocol of Dolev-Strong [5] that achieves $O(n^3 \cdot \kappa)$ communication complexity in the bulletin PKI model. Dolev-Strong uses a Send-To-All approach repeatedly—we briefly recall the protocol here: Each party $i \in [n]$ maintains a set $\mathsf{Extracted}_i$ that initialized as empty. The protocol proceeds in t rounds as follows. In the first round, the designated sender s signs her input bit and sends the signature to all $n-1$ parties. In rounds $2 \leq r \leq t$, for each bit $b \in \{0, 1\}$, if an honest party i has seen at least r signatures on b (including a signature from the designated sender) and b is not in her extracted set, then party i adds b to her local extracted set, signs b and sends the $r+1$ signatures to all $n-1$ parties. What makes the above protocol work is the fact that when party i sends the $r+1$ signatures, *all* honest parties see these signatures in the next round, since these signatures are sent to *all* $n-1$ parties. However this comes at the cost of increased communication complexity: For example, all honest parties could send an $O(t \cdot \kappa)$-size message to $n-1$ parties, which results in $O(n^2 \cdot t \cdot \kappa)$ communication which is $O(n^3 \cdot \kappa)$ for linear t.

Our protocol does away with the Send-To-All, and introduces a form of gossiping: it does not require an honest party to send the $r+1$ signatures to all $n-1$ parties. Instead, an honest party sends the $r+1$ signatures to a small set of $m = \log n \cdot \omega(1)$ parties chosen uniformly at random from the n parties (recall from Section 2 that we denote this set with $m \sim [n]$), with the hope that after a certain number of rounds, enough honest parties will see these messages. As expected, the total number of rounds must now increase. Fortunately, our protocol requires just an additional $R = O(\log n)$ rounds, yielding a communication complexity of $O(n^2 \cdot m \cdot \kappa) = \tilde{O}(n^2 \cdot \kappa)$ and a round complexity of $O(n)$.

```
1: procedure b' ← RandAuthBroadcast_p(b)
       Input: Initial bit b.
       Output: Decision bit b'.

2:     Extracted_p = Local_p = ∅;
3:     for round r = 1 to t + R + 1 do
4:         if r = 1 and p is the designated sender s then
5:             Send(sig_s(b), m ∼ [n]);
6:         if 1 < r ≤ t + R then
7:             Local_p ← Local_p ∪ Receive();
8:             for bit x ∈ {0, 1} do
9:                 if |S ← DistinctSigs(x, Local_p, s)| ≥ min{r, t + 1} & x ∉ Extracted_p then
10:                    Extracted_p = Extracted_p ∪ x;
11:                    Send(sig_p(x) ∪ S, m ∼ [n]);
12:        if r = t + R + 1 then
13:            return b' ∈ Extracted_p if |Extracted_p| = 1 otherwise return the canonical bit 0;
```

Figure 2: Our RandAuthBroadcast$_p$ protocol for honest party p: n is the number of parties, $t = (1 - \epsilon) \cdot n$ is the number of malicious parties, m is the number of uniform parties selected at every round, s is the designated sender, $R = O(\log n)$ is the number of additional rounds, $\mathsf{Extracted}_p$ and Local_p are local structures initialized as empty. DistinctSigs(x, Local_p, s) returns the set of distinct valid signatures on x contained in Local_p if this set includes a signature from s, otherwise it returns $\emptyset$. Note that only the designated sender's input matters for the protocol but we give an input bit to all parties to simplify the description of the protocol.

4.2 Formal description and proof of RandAuthBroadcast

Figure 2 contains the pseudocode of our protocol from the view of an honest party p (i.e., this is the algorithm that runs at each distributed (honest) node p). It takes as input the identity p of the party (given as subscript of the procedure name) and the initial bit b and returns the final bit b' (Note that the initial bit b is only relevant for the designated sender s.) Note three major differences from the Dolev-Strong protocol [5]: (1) we increase the number of rounds from t to $t + R + 1$ (Line 3); (2) instead of sending to all parties we send to m randomly selected parties (Lines 5 and 11); (3) for all rounds $r \geq t + 1$ we do not require $r + 1$ signatures to add to the extracted set but just $t + 1$—that is why we use the expression $\min\{r, t + 1\}$ in Line 9. We now continue with the proof of consistency and validity of RandAuthBroadcast. We first define, using notation consistent with AddRandomEdges, the following sets of parties (with respect to a specific bit b and a specific round r):

1. $S(b, r)$ contains honest parties x that added b to their $\mathsf{Extracted}_x$ set *at* round r;
2. $S_1(b, r)$ contains honest parties x that added b to their $\mathsf{Extracted}_x$ set *by* round r;
3. $S_2(b, r)$ contains the set of honest parties x that have *not* added b to their $\mathsf{Extracted}_x$ set by round r.

Also, define S_3 contain the set of malicious parties ($|S_3| = n - \epsilon \cdot n$). We now prove our main technical lemma showing (roughly) that the number of parties that receive a message at round r' that was sent at round $r < r'$ increases exponentially with $r' - r$ with overwhelming probability.

Lemma 4 (Gossiping bounds)**.** *For a specific bit b, let r be the first round of* RandAuthBroadcast *where an honest party p adds b to* $\mathsf{Extracted}_p$*. Then, for all rounds r' such that $r \leq r' \leq t+R$ we have that*

$$|S(b,r')| \geq (2/3)3^{r'-r} \text{ and } |S_1(b,r')| \geq 3^{r'-r}$$

with probability at least $(1-p)^{r'-r}$, where $p = \frac{\epsilon \cdot n}{5} \cdot e^{-\epsilon \cdot m/9}$.

Proof. For the base case where $r' = r$, we have that by definition of $S(b,r)$ and $S_1(b,r)$, it is $i \in S(b,r)$ and $i \in S_1(b,r)$. Therefore $|S(b,r)| \geq \tau$ and $|S_1(b,r)| \geq \tau$ for some $\tau \geq 1$, with probability 1, and the base case holds.

For the inductive step, assume the claim holds for some round $\rho < t+R$, i.e., with probability at least $(1-p)^{\rho-r}$ it is

$$|S(b,\rho)| \geq (2/3)3^{\rho-r} \text{ and } |S_1(b,\rho)| \geq 3^{\rho-r}\,. \tag{2}$$

Recall now that the protocol proceeds from round ρ to round $\rho+1$ by having parties in $S(b,\rho)$ send a valid message on b to m random parties. We define the events

$$A : |S_2(b,\rho)| \geq 2 \cdot 3^{\rho-r}$$

$$B : |S(b,r')| \geq (2/3)3^{r'-r} \text{ and } |S_1(b,r')| \geq 3^{r'-r}.$$

To figure out a bound on how many new honest parties will receive this message, it is enough to call

$$\textsc{AddRamdomEdges}(S_1(b,\rho), S_2(b,\rho), S_3, S(b,\rho), m)$$

from Figure 1, which, by Lemma 3, tells us that, assuming $m \geq 15/\epsilon$, it is $\Pr(A|B) \geq 1-p$, and thus $\Pr(|S(b,\rho+1)| \geq 2 \cdot 3^{\rho-r}|B) \geq 1-p$. By the inductive hypothesis in Equation 2 we know that $\Pr(B) \geq (1-p)^{\rho-r}$, and therefore by the identity $\Pr(A) \geq \Pr(B) \cdot \Pr(A|B)$ we have that

$$|S(b,\rho+1)| \geq 2 \cdot 3^{\rho-r}\,, \tag{3}$$

with probability at least $(1-p)^{\rho-r} \cdot (1-p) = (1-p)^{\rho+1-r}$. Note also that by definition of $S_1(.,.)$ and $S(.,.)$ we have

$$|S_1(b,\rho+1)| = |S(b,\rho+1)| + |S_1(b,\rho)| \geq 2 \cdot 3^{\rho-r} + 3^{\rho-r} = 3 \cdot 3^{\rho-r} = 3^{(\rho+1)-r}\,.$$

Now by Equation 3 we have $|S(b,\rho+1)| \geq 2 \cdot 3^{\rho-r} = (2/3) \cdot 3^{(\rho+1)-r}$, as required. □

We now present our main results, i.e., proofs for consistency/validity as defined in Definition 1.

Theorem 1 (RandAuthBroadcast consistency)**.** *Let $R = \lceil \log_3(\epsilon \cdot n) \rceil$ and $m = \omega(1) \cdot \log n$. At the end of* RandAuthBroadcast*, all honest parties agree on the same output bit, with probability at least $1 - negl(n)$.*

Proof. Suppose an honest party i adds bit b to $\mathsf{Extracted}_i$ at some round r. We prove by the end of the protocol all honest parties j will add b to their $\mathsf{Extracted}_j$ sets with probability at least $1 - negl(n)$—this will mean that all honest parties will have identical $\mathsf{Extracted}$ sets by the end of the protocol, which is equivalent to consistency. We distinguish the two cases:

> Case $r < t+1$: For this case we make use of Lemma 4. Based on Lemma 4, if an honest party i adds b to $\mathsf{Extracted}_i$ at round $r < t+1$, at least $3^R \geq 3^{\log_3(\epsilon n)} = \epsilon \cdot n$ honest parties will add b to their $\mathsf{Extracted}$ set by round $r+R$ (and thus by the end of the protocol), with probability at least $(1-p)^R \geq 1 - R \cdot p$, by Bernoulli's inequality and since $-p \geq -1$. Note however that for $m = \omega(1) \cdot \log n$, $p = \frac{\epsilon \cdot n}{5} \cdot e^{-\epsilon \cdot m/9} = negl(n)$ and therefore the probability is at least $1 - negl(n)$, as required.

Case $r \geq t+1$: Suppose an honest party i adds bit b to $\mathsf{Extracted}_i$ at some round $r \geq t+1$. This means that i has received valid signatures on b from $t+1$ distinct parties. That means that an honest party j added bit b to $\mathsf{Extracted}_j$ at some round $r'' < t+1$. Therefore, the case $r'' < t+1$ from above applies for honest party j and therefore all honest parties will add b to their $\mathsf{Extracted}$ sets by the end of the protocol, with probability at least $1 - negl(n)$.

□

Theorem 2 (RandAuthBroadcast validity)**.** *Let $R = \lceil \log_3(\epsilon \cdot n) \rceil$ and $m = \omega(1) \cdot \log n$. If the sender is honest, then at the end of* RandAuthBroadcast *all honest parties agree on the same output bit, which is the input bit of the sender, with probability at least $1 - negl(n)$.*

Proof. Follows from the proof of consistency in Theorem 1. After $R = \lceil \log_3(\epsilon \cdot n) \rceil$ rounds, all honest parties will have received the bit of the honest sender, with probability at least $1 - negl(n)$. □

Theorem 3 (RandAuthBroadcast communication complexity)**.** *Let $R = \lceil \log_3(\epsilon \cdot n) \rceil$ and $m = \omega(1) \cdot \log n$. The total number of bits exchanged by all parties in* RandAuthBroadcast *is $\tilde{O}(n^2 \cdot \kappa)$.*

Proof. Every honest party sends at most one time to $m = \omega(1) \cdot \log n$ parties a message of at most t signatures. Since there are $O(n)$ honest parties, $t = O(n)$ and the size of each signature is κ, the total number of bits exchanged is $O(n^2 \cdot m \cdot \kappa) = \tilde{O}(n^2 \cdot \kappa)$. □

5 Conclusions and Discussion

In this paper, we have studied the communication complexity of broadcast with dishonest majority in the bulletin board PKI model. We have shown a protocol that achieves close to $O(n)$ improvement in communication complexity over the protocol of Dolev and Strong [5] against a static adversary. We believe that there is much room for future work in this direction.

The most immediate question is whether subcubic broadcast is possible against an adaptive adversary that controls a majority of the parties. In the *strongly adaptive* adversarial model, the adversary can corrupt parties *at any given point of an execution of a protocol*. On top of this, the adversary can observe a party p's messages for round r, then adaptively corrupt p and *delete any of p's messages for round r*. In this manner, the adversary can replace p's messages with its own, or send conflicting messages to the messages that p sent prior to being corrupted. It is clear that our protocol is not secure in this setting. A simple attack would be to send the wrong bit b during some round $r < t - m + 1$ to a single honest party p and then observe which m parties p sends to during the next round. Then, the adversary corrupts these m parties and thus by the end of the protocol, p will be the only honest party having b in her $\mathsf{Extracted}$ set.

Another question is whether the communication complexity in the quadratic protocol by Chan et al. [4] can be further improved to subquadratic complexity (using trusted setup). A potential improvement could be achieved by using some cryptographic primitive such as *cryptographic accumulators* or *succinct non-interactive arguments* in a way that the size of the signature list that needs to be propagated is kept sublinear, while not comprimising the security of the protocol.

Finally, another interesting direction would be to restrict our sender model to send-to-all channels (instead of pairwise channels considered here), as commonly found in popular blockchain protocols.

References

[1] Ittai Abraham, T.-H. Hubert Chan, Danny Dolev, Kartik Nayak, Rafael Pass, Ling Ren, and Elaine Shi. Communication complexity of byzantine agreement, revisited. In Peter Robinson

and Faith Ellen, editors, *38th ACM Symposium Annual on Principles of Distributed Computing*, pages 317–326. Association for Computing Machinery, July / August 2019.

[2] Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the Weil pairing. In Colin Boyd, editor, *Advances in Cryptology – ASIACRYPT 2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 514–532. Springer, Heidelberg, December 2001.

[3] Elette Boyle, Ran Cohen, and Aarushi Goel. Succinctly reconstructed distributed signatures and balanced byzantine agreement. ePrint Cryptology Archive, 2020.

[4] T.-H. Hubert Chan, Rafael Pass, and Elaine Shi. Sublinear-round byzantine agreement under corrupt majority. In Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas, editors, *PKC 2020: 23rd International Conference on Theory and Practice of Public Key Cryptography, Part II*, volume 12111 of *Lecture Notes in Computer Science*, pages 246–265. Springer, Heidelberg, May 2020.

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

[6] Devdatt P. Dubhashi and Desh Ranjan. Balls and bins: A study in negative dependence. *Random Struct. Algorithms*, 13(2):99–124, 1998.

[7] Paul Feldman and Silvio Micali. Optimal algorithms for byzantine agreement. In *20th Annual ACM Symposium on Theory of Computing*, pages 148–161. ACM Press, May 1988.

[8] Matthias Fitzi and Jesper Buus Nielsen. On the number of synchronous rounds sufficient for authenticated byzantine agreement. In *DISC*, volume 5805 of *LNCS*, pages 449–463. Springer, 2009.

[9] Juan A. Garay, Jonathan Katz, Chiu-Yuen Koo, and Rafail Ostrovsky. Round complexity of authenticated broadcast with a dishonest majority. In *48th Annual Symposium on Foundations of Computer Science*, pages 658–668. IEEE Computer Society Press, October 2007.

[10] Juan A. Garay and Yoram Moses. Fully polynomial byzantine agreement in t+1 rounds. In *25th Annual ACM Symposium on Theory of Computing*, pages 31–41. ACM Press, May 1993.

[11] Valerie King and Jared Saia. Breaking the $O(n^2)$ bit barrier: scalable byzantine agreement with an adaptive adversary. In Andréa W. Richa and Rachid Guerraoui, editors, *29th ACM Symposium Annual on Principles of Distributed Computing*, pages 420–429. Association for Computing Machinery, July 2010.

[12] Valerie King, Jared Saia, Vishal Sanwalani, and Erik Vee. Scalable leader election. In *17th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 990–999. ACM-SIAM, January 2006.

[13] Leslie Lamport, Robert Shostak, and Marshall Pease. The byzantine generals problem. *ACM Transactions on Programming Languages and Systems*, 4(382–401), July 1982.

[14] Benoît Libert, Marc Joye, and Moti Yung. Born and raised distributively: fully distributed non-interactive adaptively-secure threshold signatures with short shares. In Magnús M. Halldórsson and Shlomi Dolev, editors, *33rd ACM Symposium Annual on Principles of Distributed Computing*, pages 303–312. Association for Computing Machinery, July 2014.

[15] Silvio Micali. Very simple and efficient byzantine agreement. In Christos H. Papadimitriou, editor, *ITCS 2017: 8th Innovations in Theoretical Computer Science Conference*, volume 4266, pages 6:1–6:1, 67, January 2017. LIPIcs.

[16] Silvio Micali and Vinod Vaikuntanathan. Optimal and player-replaceable consensus with an honest majority. Technical report, MIT, 2017.

[17] Jun Wan, Hanshen Xiao, Elaine Shi, and Srinivas Devadas. Expected constant round byzantine broadcast under dishonest majority. ePrint Cryptology Archive, 2020.

Appendix

Proof of Lemma 1

Proof. Define the variables $Y_i = 1 - X_i$. Since $X_i \leq 1$, Y_i are also NA (see proof of Proposition 5 [6]). For $i = 1, \ldots, n$, let $p_i = \Pr[X_i = 1]$. By linearity of expectation, we have that $\mathbb{E}[Y] = \mu = \sum_{i=1}^{n} p_i$. Since variables Y_i are negatively associated, from Property 4, we have that for $t > 0$,

$$\mathbb{E}\left[e^{tY}\right] = \mathbb{E}\left[e^{tY_1} \cdot e^{tY_2} \ldots e^{tY_n}\right] \leq \prod_{i=1}^{n} \mathbb{E}\left[e^{tY_i}\right] = \prod_{i=1}^{n} \mathbb{E}\left[e^{t(1-X_i)}\right] = \prod_{i=1}^{n} e^{t} \cdot \mathbb{E}\left[e^{-tX_i}\right] = e^{tn} \prod_{i=1}^{n} \mathbb{E}\left[e^{-tX_i}\right] .$$

For each $\mathbb{E}\left[e^{-tX_i}\right]$, it holds that $\mathbb{E}\left[e^{-tX_i}\right] = p_i e^{-t} + (1 - p_i) = 1 + p_i(e^{-t} - 1) \leq e^{p_i(e^{-t}-1)}$, where we used the fact that for any k, it holds that $1 + k \leq e^{k}$. Replacing above we get that

$$\mathbb{E}\left[e^{tY}\right] \leq e^{tn} \prod_{i=1}^{n} e^{p_i(e^{-t}-1)} = e^{tn + \sum_{i=1}^{n} p_i(e^{-t}-1)} = e^{tn + \mu(e^{-t}-1)} .$$

Finally, applying Markov's inequality, for any $t > 0$ we get that

$$\begin{aligned}
\Pr[X \leq (1-\delta)\cdot\mu] &= \Pr[n - Y \leq (1-\delta)\cdot\mu] \\
&= \Pr[Y \geq n - (1-\delta)\cdot\mu] \\
&= \Pr[e^{tY} \geq e^{tn - t(1-\delta)\cdot\mu}] \\
&\leq \frac{e^{tn+\mu(e^{-t}-1)}}{e^{tn - t(1-\delta)\mu}} \\
&= \frac{e^{\mu(e^{-t}-1)}}{e^{-t(1-\delta)\mu}} \\
&= \left(\frac{e^{e^{-t}-1}}{e^{-t(1-\delta)}}\right)^{\mu} ,
\end{aligned}$$

By now we set $t = -\ln(1-\delta) > 0$, since $\delta \in (0,1)$ and thus we get:

$$\Pr[X \leq (1-\delta)\cdot\mu] \leq \left(\frac{e^{-\delta}}{(1-\delta)^{(1-\delta)}}\right)^{\mu} .$$

□

Other properties of negatively associated variables

Property 1 (Proposition 7.1 [6]). *If* $\mathbf{X}$ *and* $\mathbf{Y}$ *are sets of NA random variables and* $\forall x \in \mathbf{X}, \forall y \in \mathbf{Y}, x, y$ *are mutually independent, then* $\mathbf{X} \cup \mathbf{Y}$ *is also a set of NA random variables.*

Property 2 (Proposition 7.2 [6]). *Let* $\mathbf{X} = \{X_1, \ldots, X_n\}$ *be a set of NA random variables. Let* $(I_1, \ldots, I_k) \subset [n]$ *be disjoint index sets, for some positive integer* k*. For* $j \in [k]$*, let* $h_j : R^{|I_j|} \to R$ *be functions that are all non-decreasing or all non-increasing, and define* $Y_j = h_j(X_i, i \in I_j)$*. Then* $\mathbf{Y} = \{Y_1, \ldots, Y_k\}$ *is also a set of NA random variables.*

Property 3 (Lemma 8 [6]). *Let* $\mathbf{X} = \{X_1, \ldots, X_n\}$ *be a set of Boolean random variables with* $\sum_i X_i = 1$*. Then* $\mathbf{X}$ *is a set of NA random variables.*

Property 4 (Lemma 2 [6]). *Let* $X_1, \ldots, X_n$ *be NA. Then for any non-decreasing functions* f_i *it is* $\mathbb{E}\left[\prod_i f_i(X_i)\right] \leq \prod_i \mathbb{E}\left[f_i(X_i)\right]$.