# Multi-Valued Byzantine Broadcast: the $t < n$ Case

Martin Hirt, Pavel Raykov

ETH Zurich, Switzerland
{hirt,raykovp}@inf.ethz.ch

**Abstract.** Byzantine broadcast is a distributed primitive that allows a specific party to consistently distribute a message among $n$ parties in the presence of potential misbehavior of up to $t$ of the parties. All known protocols implementing broadcast of an $\ell$-bit message from point-to-point channels tolerating any $t < n$ Byzantine corruptions have communication complexity at least $\Omega(\ell n^2)$. In this paper we give cryptographically secure and information-theoretically secure protocols for $t < n$ that communicate $\mathcal{O}(\ell n)$ bits when $\ell$ is sufficiently large. This matches the optimal communication complexity bound for any protocol allowing to broadcast $\ell$-bit messages. While broadcast protocols with the optimal communication complexity exist for $t < n/2$, this paper is the first to present such protocols for $t < n$.

## 1 Introduction

### 1.1 Byzantine Broadcast

The Byzantine broadcast problem (aka Byzantine generals) is stated as follows [PSL80]: A specific party (the sender) wants to distribute a message among $n$ parties in such a way that all correct parties obtain the same message, even when some of the parties are malicious. The malicious misbehavior is modeled by a central adversary who corrupts up to $t$ parties and takes full control of their actions. Corrupted parties are called *Byzantine* and the remaining parties are called *correct*. Broadcast requires that all correct parties agree on the same value $v$, and if the sender is correct, then $v$ is the value proposed by the sender. Broadcast is one of the most fundamental primitives in distributed computing. It is used to implement various protocols like voting, bidding, collective contract signing, etc. Basically, this list can be continued with all protocols for secure multi-party computation as defined by Yao [Yao82, GMW87].

There exist various implementations of Byzantine broadcast from synchronous point-to-point communication channels with different security guarantees. In the model without trusted setup, perfectly-secure Byzantine broadcast is achievable when $t < n/3$ [PSL80, BGP92, CW92]. In the model with trusted setup, cryptographically or information-theoretically secure Byzantine broadcast is achievable for any $t < n$ [DS83, PW96].

Closely related to the broadcast problem is the consensus problem. In consensus each party holds a value as input, and then parties agree on a common value as output of consensus. In this paper we consider the case where any number of parties may be Byzantine. In this case the consensus problem is not well-defined, and hence we do not treat it here.

### 1.2 Efficiency of Byzantine Broadcast

In this paper we focus on the efficiency of broadcast protocols. In particular, we are interested in optimizing their *communication complexity*. The communication complexity of a protocol is defined by Yao [Yao79] to be the number of bits sent/received by correct parties during the protocol run.[1]

Historically, the broadcast problem was introduced for binary values [PSL80]. However, in various applications *long* values are broadcast rather than bits. Examples of such applications are general purpose multiparty computation protocols and specific tasks like voting. Such a broadcast of long values is called *multi-valued* broadcast. In this paper we study the communication complexity of multi-valued broadcast protocols.

Many known protocols for multi-valued broadcast [TC84,FH06,LV11, Pat11] are actually *constructions* from a broadcast of short messages and point-to-point channels. Communication complexity of such constructions is computed in terms of the point-to-point channels and the broadcast for short messages usage. The security of the protocol is based on the security of the construction and the security of the broadcast for short messages.

Let us denote the communication complexity of a short $s$-bit message broadcast with $\mathcal{B}(s)$. The most trivial construction is to broadcast the message bit by bit, which is perfectly secure for $t < n$ and has communication complexity $\ell\mathcal{B}(1)$. The construction by Turpin and Coan [TC84] is perfectly secure and tolerates $t < n/3$ while communicating $\mathcal{O}(\ell n^2 + n\mathcal{B}(1))$ bits. The construction by Fitzi and Hirt [FH06] is information-theoretically secure and tolerates $t < n/2$ while communicating $\mathcal{O}(\ell n + n^3\kappa + n\mathcal{B}(n+\kappa))$ bits, where $\kappa$ denotes a security parameter. The construction by Liang and Vaidya [LV11] is perfectly secure and tolerates $t < n/3$ while communicating $\mathcal{O}(\ell n + \sqrt{\ell}n^2\mathcal{B}(1) + n^4\mathcal{B}(1))$ bits.

[1] When counting the number of bits received by correct players, we take into account only messages which were *actively* received by them, i.e., messages which should be received according to the protocol specification.

This construction can even be extended to tolerate more than $n/3$ corruptions [LV11]. However, the extended protocol inherently requires $t < n/2$ (see Appendix A for the details). The construction by Patra [Pat11] is perfectly secure and tolerates $t < n/3$ while communicating $\mathcal{O}(\ell n + n^2\mathcal{B}(1))$ bits.

In this paper we consider the case where $t < n$. In this model existing protocols [DS83, PW96] were designed to broadcast bits, but they can be easily adopted to broadcast long messages. A simple modification of the protocol by Dolev and Strong [DS83] is cryptographically secure and has communication complexity $\Omega(\ell n^2 + n^3\kappa)$. Analogously, the protocol by Pfitzmann and Waidner [PW96] is information-theoretically secure and has communication complexity $\Omega(\ell n^2 + n^6\kappa)$ [Fit03]. Also the protocols of [HMR14] can be seen as multi-valued constructions for $t < n$. However, their resulting communication complexity is $\Omega(\ell n^3)$.

Another measure of protocol efficiency often considered is round complexity. There are two principal classes of protocols with respect to this measure: constant-round and non-constant round. In the model without trusted setup, constant-round binary Byzantine broadcast is achievable when $t < n/3$ [FM88]. In the model where public-key infrastructure (PKI) has been set up via a trusted party, constant-round binary Byzantine broadcast is achievable for $t < n/2$ [KK06], but is not achievable for $t < n$ [GKKO07].

### 1.3 Contributions

Consider any protocol for multi-valued broadcast. Since every correct player must learn the value proposed by the sender, the communication costs of the broadcast protocol must be at least $\mathcal{O}(\ell n)$. In this paper we give two generic constructions for a multi-valued broadcast which allow to achieve optimal communication complexity of $\mathcal{O}(\ell n)$ bits for $t < n$. The first construction is cryptographically secure and communicates $\mathcal{O}(\ell n + n(\mathcal{B}(\kappa) + n\mathcal{B}(1)))$ bits. The second construction is information-theoretically secure and communicates $\mathcal{O}(\ell n + n^3(\mathcal{B}(\kappa) + n\mathcal{B}(1)))$ bits. The constructions take $\mathcal{O}(n^2)$ and $\mathcal{O}(n^3)$ rounds, respectively. Table 1 summarizes the complexity costs of the existing constructions for multi-valued broadcast.[2]

In order to obtain a concrete protocol for multi-valued broadcast one takes the above constructions and composes them with the existing proto-

[2] In order to facilitate comparison we substitute $\mathcal{B}(s)$ with $s\mathcal{B}(1)$ in the communication complexity of the constructions, which is trivially possible since $\mathcal{B}(s) \leq s\mathcal{B}(1)$ for all $s$ and such arguments appear as summands inside the big $\mathcal{O}$.

| Threshold | Security | Bits Communicated | Literature |
|---|---|---|---|
| $t < n/3$ | perfect | $\mathcal{O}(\ell n^2 + n\mathcal{B}(1))$ | [TC84] |
| | | $\mathcal{O}(\ell n + (\sqrt{\ell} n^2 + n^4)\mathcal{B}(1))$ | [LV11] |
| | | $\mathcal{O}(\ell n + n^2\mathcal{B}(1))$ | [Pat11] |
| $t < n/2$ | inf.-theor. | $\mathcal{O}(\ell n + n^3\kappa + (n^2 + n\kappa)\mathcal{B}(1))$ | [FH06] |
| $t < n$ | perfect | $\ell\mathcal{B}(1)$ | Trivial |
| | inf.-theor. | $\mathcal{O}(\ell n + (n^4 + n^3\kappa)\mathcal{B}(1))$ | This paper |
| | cryptographical | $\mathcal{O}(\ell n + (n^2 + n\kappa)\mathcal{B}(1))$ | This paper |

**Table 1.** The overview of multi-valued broadcast constructions

cols for a bit broadcast (e.g., [BGP92, DS83, PW96]). The security of the composed protocol is then the "minimal" security provided by the construction and the bit broadcast protocol employed. For example, when composing information-theoretical construction for $t < n/2$ [FH06] with cryptographically secure protocol for $t < n$ [DS83] we obtain multi-valued broadcast protocol with cryptographic security tolerating $t < n/2$ and communication complexity $\mathcal{O}(\ell n + n^4(n + \kappa))$. Further instantiations are described in Table 2.

| Threshold | Security | Bits Communicated | Literature |
|---|---|---|---|
| $t < n/3$ | perfect | $\mathcal{O}(\ell n^2)$ | Trivial with [BGP92] |
| | | $\mathcal{O}(\ell n + \sqrt{\ell} n^4 + n^6)$ | [LV11] with [BGP92] |
| | | $\mathcal{O}(\ell n + n^4)$ | [Pat11] with [BGP92] |
| $t < n/2$ | inf.-theor. | $\mathcal{O}(\ell n + n^7\kappa)$ | [FH06] with [PW96] |
| | cryptogr. | $\mathcal{O}(\ell n + n^4(n + \kappa))$ | [FH06] with [DS83] |
| $t < n$ | inf.-theor. | $\Omega(\ell n^2 + n^6\kappa)$ | [PW96] |
| | | $\mathcal{O}(\ell n + n^{10}\kappa)$ | This with [PW96] |
| | cryptogr. | $\Omega(\ell n^2 + n^3\kappa)$ | [DS83] |
| | | $\mathcal{O}(\ell n + n^5\kappa)$ | This with [DS83] |

**Table 2.** Instantiations of multi-valued broadcast constructions

We note that all multi-valued constructions are only *asymptotically* optimal in $\ell$, i.e., they only outperform the trivial construction when relatively long messages are broadcast. Such long messages appear, for example, in voting protocols [CGS97] (where the set of authorities agree on the set of ballots), or in multi-party computation protocols [GMW87] (when all gates on a particular level of the circuit are evaluated in parallel). In particular, multi-party computation protocols for $t < n$ (e.g., [AJLA$^+$12, GGHR14]) achieve better communication complexity when combined with the broadcast constructions presented in this paper.

Furthermore, we investigate the round complexity of constructions for multi-valued broadcast. While for the case of $t < n/2$ constant-round constructions exist (e.g., [FH06]), we prove that in the settings with $t < n$ constant-round constructions do not exist.[3] This is a generalization of the impossibility result given in [GKKO07], because the underlying broadcast procedure for small messages can be used to distribute PKI (by letting the parties broadcast their public keys) and hence PKI cannot be sufficient to implement broadcast in a constant number of rounds.

## 2 Model and Definitions

**Parties.** We consider a setting consisting of $n$ parties (players) $\mathcal{P} = \{P_1, \ldots, P_n\}$ with some designated party called the sender, which we denote with $P_s$ for some $s \in \{1, \ldots, n\}$. For a set of parties $A \subseteq \mathcal{P}$ let $\overline{A}$ denote $\mathcal{P} \setminus A$. We assume that the parties are connected with a synchronous authentic point-to-point network. Synchronous means that all parties share a common clock and that the message delay in the network is bounded by a constant.

**Broadcast definition.** A broadcast protocol allows the sender $P_s$ to distribute a value $v_s$ among parties $\mathcal{P}$ such that:

Termination: Every correct party $P_i \in \mathcal{P}$ terminates.

Consistency: All correct parties in $\mathcal{P}$ decide on the same value.

Validity: If the sender $P_s$ is correct, then every correct party $P_i \in \mathcal{P}$ decides on the value proposed by the sender $v_i = v_s$.

**Adversary.** The faultiness of parties is modeled in terms of a central adversary corrupting up to $t < n$ parties, making them deviate from the protocol in any desired manner. We distinguish two types of security in this paper: *cryptographic* and *information-theoretic*. Cryptographic security guarantees that the protocol is secure based on some computational assumptions (e.g., signatures and/or collision-resistant hash functions),

[3] In the notation of [HMR14] this means that no non-trivial constant-round broadcast-amplification protocols tolerating $t < n$ exist.

while information-theoretical (also called statistical) security captures the fact that even a computationally unbounded adversary cannot violate the security of the protocol with a non-negligible probability.

## 3 Protocols Overview

We present cryptographically and information-theoretically secure constructions for multi-valued broadcast. Both constructions are built over point-to-point channels and an oracle for broadcasting short messages. When describing protocols we often say that players broadcast messages, while meaning that they actually use the given broadcast oracle.

On the highest level both constructions broadcast the long message block by block, where each block is broadcast using a special protocol for block broadcast. This block broadcast protocol achieves optimal communication complexity only in *good* executions, while in *bad* executions more bits need to be communicated. We select the number of blocks in such a way that good executions outnumber bad ones and the total communication complexity is optimal. Whether an execution is good or bad is determined using the *Dispute Control Framework* [BH06]. Dispute control is a technique which keeps track of disputes (also called conflicts) between players and ensures that occurred disputes cannot show up again. Intuitively, an execution is good if it is dispute-free, and bad otherwise.

We employ the dispute control framework as follows. We consider a set of unordered pairs of parties $\Delta$, where $\{P_i, P_j\} \in \Delta$ represents the fact that parties $P_i$ and $P_j$ accuse each other of being Byzantine. Parties start a protocol by setting $\Delta$ to be the empty set. Then during the protocol run they add new disputes to $\Delta$ when they learn about new accusations. We ensure that $\Delta$ always remains *valid*, meaning that if $\{P_i, P_j\} \in \Delta$ then at least one of the players $P_i, P_j$ is Byzantine.

## 4 Cryptographically Secure Construction

First, we present a protocol `CryptoBlockBC` for broadcasting blocks. The protocol `CryptoBlockBC` makes use of an external procedure for broadcasting short values and a set of disputes $\Delta$. Then we plug `CryptoBlockBC` in the protocol `CryptoBC`, which broadcasts an $\ell$-bit message block by block $q$ times. In each invocation of `CryptoBlockBC` we will use the same global variable $\Delta$ with the disputes among the players. This means that if parties $P_i$ and $P_j$ conflict during some block broadcast, then they conflict in all later invocations of `CryptoBlockBC`. Then, we count the communication complexity of the resulting construction and select $q$ which makes its optimal.

### 4.1 Block Broadcast Protocol `CryptoBlockBC`

The protocol `CryptoBlockBC` employs a collision-resistant hash function $\mathsf{CRHash}$, i.e., no efficient algorithm can find two different inputs $v, v'$ with $\mathsf{CRHash}(v) = \mathsf{CRHash}(v')$.[4] In the beginning of the protocol the sender broadcasts a hash $h = \mathsf{CRHash}(v_s)$ of the value it holds. The goal of the protocol is to ensure that all correct players learn $v_s$. All parties during the protocol run are divided into two sets: $H$ and $\overline{H}$. The set $H$ consists of happy players who have already learned $v_s$, and $\overline{H}$ who have not. At each iteration of `CryptoBlockBC` we try to move a player from $\overline{H}$ to $H$. We select a pair of players $P_x, P_y$ such that $P_x \in H$ and $P_y \in \overline{H}$. Then $P_x$ sends the value it holds to $P_y$. This procedure is meaningless if parties $P_x, P_y$ are in the dispute, so the pair is chosen such that $\{P_x, P_y\} \notin \Delta$. Once $P_y$ receives a value from $P_x$ it verifies that its hash is $h$; in the positive case $P_y$ is included in $H$ and in the negative case a conflict between $P_x$ and $P_y$ is found. Hence at each iteration we either include one player into $H$ or we discover a new conflict between a pair of players.

```
Protocol CryptoBlockBC($v_s$):
 1. Parties initialize happy set $H$ to be $\{P_s\}$.
 2. Sender $P_s$: Broadcast $h := \mathsf{CRHash}(v_s)$.
 3. While $\exists\, P_x, P_y \in \mathcal{P}$ s.t. $P_x \in H$ and $P_y \in \overline{H}$ and $\{P_x, P_y\} \notin \Delta$ do
    $r$.1 $P_x$: Send $v_x$ to player $P_y$. Denote received value by $v_y$.
    $r$.2 $P_y$: If $h = \mathsf{CRHash}(v_y)$ broadcast 1, else broadcast 0.
    $r$.3 If $P_y$ broadcasted 1 then parties add $P_y$ to $H$, otherwise they
          add $\{P_x, P_y\}$ to $\Delta$.
 4. $\forall P_i \in \mathcal{P}$: If $P_i \in H$ decide on $v_i$, otherwise decide on $\bot$.
```

**Lemma 1.** *Given that the initial dispute set $\Delta_s$ is valid and* $\mathsf{CRHash}$ *is a collision-resistant hash function, protocol* `CryptoBlockBC` *achieves broadcast (of $v_s$) and terminates with a valid dispute set $\Delta_e$. Furthermore, the protocol terminates in $\mathcal{O}(n+d)$ rounds communicating at most $\mathcal{B}(|h|) + (n+d)(|v_s| + \mathcal{B}(1))$ bits, where $d = |\Delta_e| - |\Delta_s|$, $|h|$ is the output length of* $\mathsf{CRHash}$*, and $|v_s|$ is the block length.*

[4] This is rather informal definition of collision resistance for unkeyed hash functions, for a more formal treatment see [Rog06].

*Proof.* First, we prove that at each iteration of the while loop all correct players in $H$ always hold the same value $v$ such that $\mathsf{CRHash}(v) = h$. A player is included into $H$ under condition that it broadcasts 1 at Step $r.2$, which he does only if it holds a value $v$ with $\mathsf{CRHash}(v) = h$. Hence for any two correct players $P_i, P_j \in H$ it must hold that $\mathsf{CRHash}(v_i) = h$ and $\mathsf{CRHash}(v_j) = h$. Since $\mathsf{CRHash}$ is collision-resistant it implies that $v_i = v_j$.[5]

**(Validity of $\Delta_e$)** We show that whenever $P_x$ and $P_y$ are correct then $\{P_x, P_y\}$ is not added to $\Delta$ at Step $r.3$. A correct $P_x \in H$ holds $v_x$ with $\mathsf{CRHash}(v_x) = h$ and sends $v_x = v_y$ to $P_y$ at Step $r.1$, who successfully verifies that $\mathsf{CRHash}(v_y) = h$ and broadcasts 1 at Step $r.2$, hence $\{P_x, P_y\}$ is not added to $\Delta$ at Step $r.3$.

**(Termination)** At each iteration of the while loop either the happy set $H$ or the dispute set $\Delta$ grows. $|H|$ is limited by $n$ and $|\Delta|$ is limited by $n^2$, hence the number of iterations is limited.

**(Consistency)** We prove that in the end of the protocol all correct players belong either to $H$ (and decide on the same value $v$) or to $\overline{H}$ (and decide on $\perp$). As shown above $\Delta$ remains valid in all iterations, hence for correct players $P_x$ and $P_y$ the pair $\{P_x, P_y\} \notin \Delta$. Hence, if $P_x \in H$ and $P_y \in \overline{H}$ then the while loop does not terminate.

**(Validity)** The sender $P_s$ is always in $H$. If $P_s$ is correct then it decides on $v_s$ and due to the consistency criterion all other correct players decide on $v_s$ as well.

**(Complexity analysis)** At each iteration of the while loop either $H$ or $\Delta$ grows. Hence, the total number of iterations of the while loop is upper bounded by $n + d$ where $d$ is $|\Delta_e| - |\Delta_s|$. This implies that the number of rounds the construction employs is $\mathcal{O}(n + d)$. Furthermore, the total communication costs of the protocol are upper bounded by $\mathcal{B}(|h|) + (n + d)(|v_s| + \mathcal{B}(1))$. □

## 4.2 Constructing Broadcast for Long Messages

Now we plug in `CryptoBlockBC` in the protocol `CryptoBC` which broadcasts a message block by block.

[5] More formally, when an adversary can provoke two correct players to hold colliding values for $\mathsf{CRHash}$ with non-negligible probability, then this adversary can be used to construct an efficient collision-finding algorithm for $\mathsf{CRHash}$.

**Protocol** `CryptoBC`$(v_s, q)$:

1. Parties initialize dispute set $\varDelta$ with the empty set.
2. Sender $P_s$: Cut $v_s$ in $q$ pieces $v^1, \dots, v^q$ (add padding if required).
3. For $r = 1, \dots, q$ invoke `CryptoBlockBC`$(v^r)$, denote the output of party $P_i$ by $v_i^r$.
4. $\forall P_i \in \mathcal{P}$: If one of $v_i^r = \bot$ then output $\bot$, otherwise output $v_i^1 || \cdots || v_i^q$.

Since block broadcast is invoked $q$ times, due to Lemma 1 the total communication complexity is at most

$$\sum_{i=1}^{q} \Big[ \mathcal{B}(|h|) + (n + d_i)(\ell/q + \mathcal{B}(1)) \Big] = q\mathcal{B}(|h|) + (qn + \sum_{i=1}^{q} d_i)(\ell/q + \mathcal{B}(1))$$

bits. We know that the sum of $d_i$ is upper bounded by the total number of possible disputes $n^2$. Hence we have that communication complexity is upper bounded by $q\mathcal{B}(|h|) + (qn + n^2)(\ell/q + \mathcal{B}(1))$. By setting $q = n$ we get that the total communication is at most $2\ell n + 2n^2\mathcal{B}(1) + n\mathcal{B}(|h|)$ which is $\mathcal{O}(\ell n + n(\mathcal{B}(\kappa) + n\mathcal{B}(1)))$.

The number of rounds the construction employs is $\sum_{i=1}^{q} r_i$, where each $r_i \in \mathcal{O}(n + d_i)$. Hence, for $q = n$ we have that the total number of rounds is $\mathcal{O}(n^2)$.

The following theorem summarizes the cryptographically secure construction presented in this section:

**Theorem 1.** *In the setting with $t < n$, the construction* `CryptoBC` *with $q = n$ achieves cryptographically secure broadcast of $\ell$-bit messages in $\mathcal{O}(n^2)$ rounds by communicating $\mathcal{O}(\ell n + n(\mathcal{B}(\kappa) + n\mathcal{B}(1)))$ bits (where $\kappa$ is a security parameter and $\mathcal{B}(s)$ is the complexity of the underlying broadcast for short $s$-bit messages).*

In order to obtain a concrete multi-valued broadcast protocol we instantiate `CryptoBC` with the protocol [DS83]:

**Theorem 2.** *Instantiating the construction* `CryptoBC` *with $q = n$ and [DS83] as underlying broadcast for short messages results in a cryptographically secure multi-valued broadcast protocol for $t < n$ with communication complexity $\mathcal{O}(\ell n + n^5\kappa)$ (where $\kappa$ is a security parameter).*

## 5 Information-Theoretically Secure Construction

This section is organized similar to the cryptographic case. First, we present a protocol `ITBlockBC` for broadcasting blocks which is analogous to `CryptoBlockBC`, with the difference that it relies on a universal hash function instead of a collision-resistant one. As in the cryptographic case we then plug ITBlockBC in the ITBC protocol, which broadcasts a message block by block $q$ times. Then, we count the communication complexity of the resulting protocol ITBC, and select the number of blocks $q$ which makes it optimal.

### 5.1 Universal Hash Functions

Consider a family of functions $\mathcal{U} = \{U_k\}_{k \in \mathcal{K}}$ indexed with a key set $\mathcal{K}$, where each function $U_k$ maps elements of some set $\mathcal{X}$ to a fixed set of bins $\mathcal{Y}$. The family $\mathcal{U}$ is called $\varepsilon$-universal if for any two distinct messages $v_1$ and $v_2$,

$$\frac{|\{k \in \mathcal{K} \mid U_k(v_1) = U_k(v_2)\}|}{|\mathcal{K}|} \leq \varepsilon.^{6}$$

A $\varepsilon$-universal hash function can for example be constructed as follows: Let $\mathcal{X} = \{0,1\}^\ell$, $\mathcal{K} = \mathcal{Y} = \mathrm{GF}(2^\nu)$, and any value $v \in \{0,1\}^\ell$ be interpreted as a polynomial $f_v$ over $\mathrm{GF}(2^\nu)$ of degree $\lceil \ell/\nu \rceil - 1$. The hash function is defined as $U_k(v) = f_v(k)$. We know that two distinct polynomials of degree $\lceil \ell/\nu \rceil - 1$ can match in at most $\lceil \ell/\nu \rceil - 1$ points. Hence, for any two distinct $v_1, v_2 \in \{0,1\}^\ell$,

$$\frac{|\{k \in \{0,1\}^\nu \mid U_k(v_1) = U_k(v_2)\}|}{2^\nu} \leq \frac{\lceil \ell/\nu \rceil - 1}{2^\nu} \leq 2^{-\nu}\ell.$$

So, $\{U_k\}_{k \in \{0,1\}^\nu}$ is a family of $(2^{-\nu}\ell)$-universal hash functions.

We will denote a $\varepsilon$-universal hash function with ITHash.

### 5.2 Block Broadcast Protocol ITBlockBC

Similarly to the cryptographic case all parties during the run of the protocol ITBlockBC are divided into two sets: $H$ and $\overline{H}$. The set $H$ consists of happy players who have already learned $v_s$, and $\overline{H}$ who have not. The difference to the cryptographic case is that the set $H$ is not monotonically growing—it may happen that the same player may be added/removed from $H$ several times. At each iteration of ITBlockBC we try to move a player from $\overline{H}$ to $H$. We select a pair of players $P_x, P_y$ such that $P_x \in H$, $P_y \in \overline{H}$ and $\{P_x, P_y\} \notin \Delta$. Then $P_x$ sends the value it holds to $P_y$. Now player $P_y$ needs to verify that the value received from $P_x$ is the value that correct parties in $H$ hold. In order to do so, $P_y$ broadcasts a key $k$

[6] This is a combinatorial definition of a universal hash function, usually the last condition is written probabilistically as $\Pr[k \xleftarrow{\$} \mathcal{K} : U_k(v_1) = U_k(v_2)] \leq \varepsilon$.

for $\varepsilon$-universal hash function $\mathsf{ITHash}$, and then $P_s$ broadcasts a hash $h$ for this key. As long as $P_y$ honestly chooses $k$ uniformly at random, with overwhelming probability correct players will obtain different hashes if they hold different values. If a party in $H \cup \{P_y\} \setminus \{P_s\}$ holds a value with a hash $h$, then he broadcasts 1, and 0 otherwise (the sender $P_s$ does not broadcast because if he is correct he can broadcast only 1). If every party broadcasts 1, then the iteration was successful and $P_y$ is added to $H$. Otherwise, some of the parties in $H \cup \{P_y\}$ do not hold the right value and we search for new disputes.
An important difference from the cryptographic case is that disputes may occur not only between $P_x$ and $P_y$, but between any two parties in $H$. In order to find such disputes, one must be able to reason about the history of how $H$ was formed. We will keep a history set $T$ which will contain pairs of players $(P_x, P_y)$ such that $P_y$ learned the value it holds from $P_x$.

```
Protocol ITBlockBC($v_s$):
 1. Parties initialize happy set $H$ to be $\{P_s\}$ and history set $T$ to be $\emptyset$.
 2. While $\exists\, P_x, P_y \in \mathcal{P}$ s.t. $P_x \in H$ and $P_y \in \overline{H}$ and $\{P_x, P_y\} \notin \Delta$ do
    r.1 $P_x$: Send $v_x$ to player $P_y$. Denote received value by $v_y$. Add
        $(P_x, P_y)$ to $T$.
    r.2 $P_y$: Generate random $k \in \mathcal{K}$ and broadcast it.
        Sender $P_s$: Broadcast $h := \mathsf{ITHash}_k(v_s)$.
    r.3 $\forall P_i \in H \cup \{P_y\} \setminus \{P_s\}$: If $h = \mathsf{ITHash}_k(v_i)$ then broadcast 1,
        otherwise 0.
    r.4 If all parties broadcasted 1
          - Add $P_y$ to $H$.
        else
          - For all $(P_i, P_j) \in T$ s.t. $P_i$ broadcasted 1 (resp. $P_i = P_s$) and
            $P_j$ broadcasted 0, add $\{P_i, P_j\}$ to $\Delta$.
          - Set $H$ to $\{P_s\}$, $T$ to $\emptyset$.
 3. $\forall P_i \in \mathcal{P}$: If $P_i \in H$ decide on $v_i$, otherwise decide on $\bot$.
```

**Lemma 2.** *Given that the initial dispute set $\Delta_s$ is valid and* $\mathsf{ITHash}$ *is a universal hash function, protocol* `ITBlockBC` *achieves broadcast (of $v_s$) and terminates with a valid dispute set $\Delta_e$ (except with negligible probability). Furthermore, the protocol terminates in $\mathcal{O}(n + nd)$ rounds communicating at most $(n + nd)(|v_s| + \mathcal{B}(|h|) + \mathcal{B}(|k|) + n\mathcal{B}(1))$ bits, where $d = |\Delta_e| - |\Delta_s|$, $|h|$ is the output length of* $\mathsf{ITHash}$*, $|k|$ is the key length of* $\mathsf{ITHash}$*, and $|v_s|$ is the block length.*

*Proof.* First, we prove that at each iteration of the while loop all correct players in $H$ always hold the same value $v$. More precisely, we need to show that if a correct player $P_y$ is added to $H$, then, given that all correct players in $H$ hold the same value $v$, it holds that $v_y = v$. We have that all parties in $H \cup \{P_y\} \setminus \{P_s\}$ broadcast 1 at Step $r$.3. This implies that $P_y$ successfully verifies that $\mathsf{ITHash}_k(v_y) = h$, and all correct parties in $H$ verify that $\mathsf{ITHash}_k(v) = h$. Due to the fact that $P_y$ is correct, the key $k$ is chosen uniformly at random, so given that $\mathsf{ITHash}_k(v_y) = \mathsf{ITHash}_k(v)$, it must hold with overwhelming probability $1 - \varepsilon$ that $v_y = v$.

Second, we show that if the condition at Step $r$.4 is false then at least one new conflict is found. We have that not all players in $H \cup \{P_y\} \setminus \{P_s\}$ broadcasted 1. Consider two possible cases:

*(Exists $P_z \in H \setminus \{P_s\}$ which broadcasts 0 at step r.3)* Since $P_z$ is in $H$ there must exist a sequence of players $P_{i_1}, P_{i_2}, \ldots, P_{i_k}$ in $H$ such that $P_{i_1} = P_s, P_{i_k} = P_z$ and $(P_{i_j}, P_{i_{j+1}}) \in T$ for all $j = 1, \ldots, k-1$ (see illustration in Figure 1). In the $r^{th}$ iteration some of the players in $H$ stayed happy ($P_s$ and those who broadcasted 1) and some become unhappy (broadcasted 0). We know that $P_s$ stayed happy and $P_z$ became unhappy. Hence in a row $P_{i_1}, P_{i_2}, \ldots, P_{i_k}$ there are players of both types. Then we have that exist two players $P_{i_u}, P_{i_{u+1}}$ such that $P_{i_u}$ stays happy and $P_{i_{u+1}}$ becomes unhappy. By construction of $T$, $(P_{i_u}, P_{i_{u+1}}) \in T$ implies that $\{P_{i_u}, P_{i_{u+1}}\}$ is not yet in $\Delta$. Consequently, the pair $\{P_{i_u}, P_{i_{u+1}}\}$ will be identified as having a conflict and will be added to $\Delta$.

*(Each $P_i \in H \setminus \{P_s\}$ broadcasts 1 at step r.3)* It means that $P_x$ broadcasts 1 (or $P_x = P_s$) and $P_y$ broadcasts 0. Hence the new dispute $\{P_x, P_y\}$ will be added to $\Delta$.

Now we proceed with the proof of the current lemma.

**(Validity of $\Delta_e$)** We show that whenever $P_i$ and $P_j$ are correct then $\{P_i, P_j\}$ is never added to $\Delta$. The pair $\{P_i, P_j\}$ is added to $\Delta$ only when $P_i$ sent some $v$ to $P_j$ (i.e., $(P_i, P_j) \in T$), and they disagree for some key $k$ whether $\mathsf{ITHash}_k(v)$ equals $h$. Hence, $P_i$ or $P_j$ is corrupted.

**(Termination)** There can be at most $n$ successive iterations where the set $H$ grows (condition at Step $r$.4 is true). As shown above whenever condition at Step $r$.4 is false a new conflict is found. The number of conflicts is limited and so must be the number of the while loop iterations.

**(Consistency)** We prove that in the end of the protocol all correct players belong either to $H$ (and decide on the same value $v$) or to $\overline{H}$ (and decide on $\perp$). As shown above $\Delta$ remains valid in all iterations, hence for any two correct players $P_x, P_y$, the pair $\{P_x, P_y\} \notin \Delta$. Hence, if $P_x \in H$

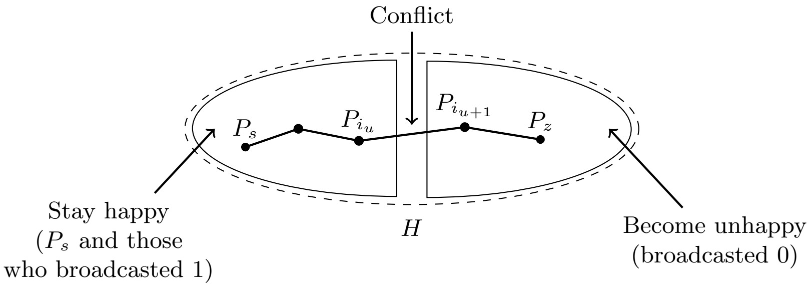

**Fig. 1.** Conflict finding in `ITBlockBC`

and $P_y \in \overline{H}$ then the while loop does not terminate.
**(Validity)** The correct sender $P_s$ is always in $H$. The sender $P_s$ decides on $v_s$ and due to the consistency criterion all other correct players decide on $v_s$ as well.
**(Complexity analysis)** There can be at most $n$ consecutive iterations, where no conflict is found, hence the total number of iterations is at most $n+nd$, where $d = |\varDelta_e| - |\varDelta_s|$. This implies that the number of rounds the construction employs is $\mathcal{O}(n+nd)$. Furthermore, since the communication costs of each iteration are at most $|v_s| + \mathcal{B}(|h|) + \mathcal{B}(|k|) + n\mathcal{B}(1)$, we have that the total communication costs of the protocol are upper bounded by $(n + nd)(|v_s| + \mathcal{B}(|h|) + \mathcal{B}(|k|) + n\mathcal{B}(1))$. □

### 5.3 Constructing Broadcast for Long Messages

Similarly to the cryptographic case, we plug `ITBlockBC` in the protocol `ITBC` which simply broadcasts a message block by block. The protocol `ITBC` is a copy of the protocol `CryptoBC` with the only difference that `CryptoBlockBC` is substituted with `ITBlockBC`.

Due to Lemma 2 the total communication complexity of `ITBC` is at most

$$\sum_{i=1}^{q} \left[ (n + d_i n)(\ell/q + \mathcal{B}(|h|) + \mathcal{B}(|k|) + n\mathcal{B}(1)) \right] =$$

$$n(q + \sum_{i=1}^{q} d_i)(\ell/q + \mathcal{B}(|h|) + \mathcal{B}(|k|) + n\mathcal{B}(1)).$$

This expression is bound by $n(q + n^2)(\ell/q + \mathcal{B}(|h|) + \mathcal{B}(|k|) + n\mathcal{B}(1))$. By setting $q = n^2$ we have that communication costs are at most $2\ell n + 2n^3(\mathcal{B}(|h|) + \mathcal{B}(|k|) + n\mathcal{B}(1)))$ which is $\mathcal{O}(\ell n + n^3(\mathcal{B}(\kappa) + n\mathcal{B}(1)))$.

The number of rounds the construction employs is $\sum_{i=1}^{q} r_i$, where each $r_i \in \mathcal{O}(n + nd_i)$. Hence, for $q = n^2$ we have that the total number of rounds is $\mathcal{O}(n^3)$.

The following theorem summarizes the information-theoretically secure construction presented in this section:

**Theorem 3.** *In the setting with $t < n$, the construction* `ITBC` *with $q = n^2$ achieves information-theoretically secure broadcast of $\ell$-bit messages in $\mathcal{O}(n^3)$ rounds by communicating $\mathcal{O}(\ell n + n^3(\mathcal{B}(\kappa) + n\mathcal{B}(1)))$ bits (where $\kappa$ is a security parameter and $\mathcal{B}(s)$ is the complexity of the underlying broadcast for short $s$-bit messages).*

In order to obtain a concrete multi-valued broadcast protocol we instantiate `ITBC` with the protocol [PW96]:

**Theorem 4.** *Instantiating the construction* `ITBC` *with $q = n^2$ and [PW96] as underlying broadcast for short messages results in an information-theoretically secure multi-valued broadcast protocol for $t < n$ with communication complexity $\mathcal{O}(\ell n + n^{10}\kappa)$ (where $\kappa$ is a security parameter).*

## 6 On The Round Complexity of Multi-Valued Constructions

While the primary goal of this paper is to build communication efficient protocols, one often optimizes the protocols with respect to another measure of the protocols' efficiency, number of rounds employed by a protocol. According to this measure there are two principal classes of the protocols: constant-round and non-constant round. In the following we investigate whether it is possible to obtain protocols optimal in both measures, that is, constant-round multi-valued broadcast protocols with optimal communication complexity for $t < n$.

The goal of this paper is to build protocols for efficient multi-valued constructions. We stress that by construction we understand a protocol for $n$ players which realizes multi-valued broadcast on top of bilateral channels and a special procedure for broadcasting bits. We explicitly distinguish such constructions and *plain* multi-valued broadcast protocols (e.g., [DS83, PW96]) that directly implement broadcast from bilateral channels.

When $t < n/2$ both communication and round optimal multi-valued broadcast protocols can be built by combining constant-round construction [FH06] with a constant-round binary broadcast protocol (e.g., [KK06, GKKO07]). For the case of arbitrary $t < n$ it has been shown that no plain protocol can achieve broadcast in a constant number of rounds [GKKO07]. In the context of this paper this shows that no concrete instantiation of a multi-valued construction and a procedure for broadcasting bits can be constant-round. However, it is still interesting to understand whether a non-trivial constant-round construction for multi-valued broadcast exists separately. Next we show that this is not possible, i.e., there is a separation between $t < n/2$ and $t < n$ cases not only for broadcast protocols but between constructions for multi-valued broadcast as well.

**A construction's failure probability (based on [GY89]).** Consider any multi-valued construction protocol $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_n)$. A scenario is a triple $(v, B, \mathcal{A})$ where $v \in \{0,1\}^\ell$ is a value that the sender broadcasts, $B \subseteq \mathcal{P}$ is a set of malicious players controlled with an adversarial strategy $\mathcal{A}$. We call an execution of the protocol $\boldsymbol{\pi}$ in a scenario *successful* if the outputs of honest parties $\mathcal{P} \setminus B$ satisfy broadcast properties (validity and consistency). We define the error $\varepsilon_{\boldsymbol{\pi},v,B,\mathcal{A}}$ to be the probability of an unsuccessful execution over the randomness used by honest parties and the adversary in the corresponding scenario.[7] Then the failure probability of $\boldsymbol{\pi}$ is defined as $\max\limits_{v,B,\mathcal{A}} \varepsilon_{\boldsymbol{\pi},v,B,\mathcal{A}}$, i.e., as the maximum failure among all scenarios.

**Impossibility framework.** We employ a standard indistinguishability argument that is used to prove that certain security goals cannot be achieved by any protocol in the Byzantine environment [PSL80]. Such a proof goes by contradiction, i.e., by assuming that the security goals can be satisfied by means of some protocol $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_n)$. Then the programs $\pi_i$ are used to build a *configuration* with contradictory behavior. The configuration consists of (possibly) multiple copies of $\pi_i$ connected with bilateral channels and given admissible inputs. Once the configuration is built, one simultaneously starts all the programs in the configuration and analyzes the outputs produced by the programs locally. By arguing that the view of some programs $\pi_i$ and $\pi_j$ in the configuration is indistinguishable from their view when run by the corresponding players $P_i$ and $P_j$ (while the adversary corrupts the remaining players in $\mathcal{P} \setminus \{P_i, P_j\}$) we can deduce consistency conditions on the outputs by $\pi_i$ and $\pi_j$ that lead to a contradiction. The main novelty in the following proof is that we consider an extended communication model where in addition to bilateral channels players are given access to a special procedure for broadcasting short messages. While following the path described

[7] In all executions we assume that the procedure to broadcast bits is perfectly secure, i.e., the values broadcast with it are consistently delivered to the parties.

above, we need to additionally describe how the calls to this procedure are handled.

**Theorem 5.** *Every non-trivial*[8]*multi-valued broadcast construction for* $t < n$ *which takes less than* $n - 1$ *rounds fails with probability at least* $1/(2n)$.

*Proof.* Take any non-trivial construction $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_n)$ which requires $q < n - 1$ rounds and has error probability $\varepsilon$. Without loss of generality, assume as well that the sender is $P_1$, i.e., the sender's program is $\pi_1$. On the highest level our proof consists of three steps. (i) we define a configuration (inspired by [GKKO07]). (ii) we show that all programs in the configuration must output the same value $v$ with probability $1 - n\varepsilon$. (iii) we use an information flow argument to prove that there is a program in the configuration that outputs $v$ with probability at most $1/2$. Finally, we combine the probability inequalities given by (ii) and (iii) to conclude that $\varepsilon \geq 1/(2n)$.
**(i)** Consider a chain of $n$ programs $\pi_1, \pi_2, \pi_3, \ldots, \pi_n$ connected with bilateral channels as shown in Figure 2. In this configuration only programs that are connected communicate, i.e., $\pi_1$ communicates only with $\pi_2$ and receives no messages from parties in $\mathcal{P} \setminus \{P_1, P_2\}$. Let $\pi_1$ be given as input a uniform random variable $V$ chosen from the input domain $\{0, 1\}^\ell$. Now we execute the programs. Whenever any program broadcasts any value using the broadcast procedure this value is delivered to all programs in the configuration.
**(ii)** First, we prove that any pair of connected programs $(\pi_i, \pi_{i+1})$ in the chain outputs the same value. One can view the configuration as the player $P_i$ running the program $\pi_i$ and $P_{i+1}$ running $\pi_{i+1}$ while the adversary corrupting $\mathcal{P} \setminus \{P_i, P_{i+1}\}$ is simulating the programs $\pi_1, \ldots, \pi_{i-1}$ and $\pi_{i+2}, \ldots, \pi_n$. Due to the consistency property, $\pi_i$ and $\pi_{i+1}$ must output the same value with probability at least $1 - \varepsilon$. Since every connected pair of programs in the chain outputs the same value with probability at least $1 - \varepsilon$, then all the programs in the configuration output the same value with probability at least $1 - (n - 1)\varepsilon$. Moreover, the configuration can be viewed as $P_1$ executing $\pi_1$ while the adversary corrupts $\mathcal{P} \setminus \{P_1\}$ and simulates the remaining programs. Due to the validity property, $\pi_1$ must output $V$ with probability at least $1 - \varepsilon$. Finally, all the programs in the chain output $V$ with probability $1 - n\varepsilon$.
**(iii)** Let $S_i^r$ be a random variable denoting the state of the program $\pi_i$

[8] By non-trivial we mean every construction which broadcasts strictly less bits with the broadcast procedure than the length of the message broadcast $\ell$.

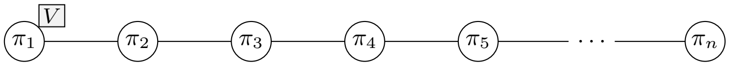

**Fig. 2.** The configuration to show the impossibility of non-trivial construction

in the chain after $r$ rounds of the protocol execution. By state we understand the input that the program has, the set of all messages that the program received up to the $r^{th}$ round over point-to-point channels and via the underlying broadcast procedure together with the random coins it has used. Let $B^r$ be a random variable denoting the list of the values that have been broadcast with the broadcast procedure up to the $r^{th}$ round. After $r$ rounds only programs $\pi_1, \pi_2, \ldots, \pi_{r+1}$ can receive full information about $V$. The remaining programs in the chain $\pi_{r+2}, \pi_{r+3}, \ldots, \pi_n$ can receive only the information that was distributed with the broadcast procedure, i.e., the information contained in $B^r$. That is, one can verify by induction that for any $r$ and for all $i \geq r+2$ holds $I(V; S_i^r | B^r) = 0$. Hence, for the last program in the chain $\pi_n$ after $q$ rounds of computation it holds that $I(V; S_n^q | B^q) = 0$ and hence $I(V; S_n^q) \leq H(B^q)$. Because we assumed that the construction is non-trivial, at most $\ell - 1$ bits can be broadcast with the broadcast procedure. Hence, we have that $H(B^q) \leq \ell - 1$. Combining these facts we get that $I(V; S_n^q) \leq \ell - 1$. Hence, the last program $\pi_n$ outputs $V$ with probability at most $1/2$. However, we have shown above that all programs (including $\pi_n$) output $V$ with probability at least $1 - n\varepsilon$. Hence, we have that $1/2 \geq 1 - n\varepsilon$ which implies that $\varepsilon \geq 1/(2n)$. □

## 7 Conclusions

Existing multi-valued broadcast protocols achieve optimal communication complexity only for $t < n/3$ [LV11] or $t < n/2$ [FH06]. In this paper we proposed the first multi-valued broadcast protocols that tolerate any $t < n$ Byzantine corruptions and achieve optimal communication complexity $\mathcal{O}(\ell n)$ for sufficiently long messages of $\ell$ bits. One of the proposed protocols is cryptographically secure and the other one is information-theoretically secure. The cryptographically secure protocol is based on the security of the signature scheme and a collision-resistance of the hash function employed. It communicates $\mathcal{O}(\ell n + n^5 \kappa)$ bits. The information-theoretically secure protocol may fail with a negligible probability and needs to communicate $\mathcal{O}(\ell n + n^{10} \kappa)$ bits.

The presented constructions `CryptoBC` and `ITBC` require $\mathcal{O}(n^2)$ and $\mathcal{O}(n^3)$

rounds, respectively. While constant-round constructions are unachievable, it is still unresolved whether more round-efficient constructions exist. We leave round-complexity optimizations and proving stronger lower bounds as open questions.

## References

[AJLA$^+$12] G. Asharov, A. Jain, A. López-Alt, E. Tromer, V. Vaikuntanathan, and D. Wichs. Multiparty computation with low communication, computation and interaction via threshold fhe. In *Proceedings of the 31st Annual International Conference on Theory and Applications of Cryptographic Techniques*, EUROCRYPT'12, pages 483–501, Berlin, Heidelberg, 2012. Springer-Verlag.

[BGP92] P. Berman, J. A. Garay, and K. J. Perry. Bit optimal distributed consensus. In *Computer Science Research*, pages 313–322. Plenum Publishing Corporation, New York, NY, USA, 1992. Preliminary version appeared in STOC '89.

[BH06] Z. Beerliova-Trubiniova and M. Hirt. Efficient multi-party computation with dispute control. In S. Halevi and T. Rabin, editors, *Theory of Cryptography Conference — TCC 2006*, volume 3876 of *Lecture Notes in Computer Science*, pages 305–328. Springer-Verlag, March 2006.

[CGS97] R. Cramer, R. Gennaro, and B. Schoenmakers. A secure and optimally efficient multi-authority election scheme. In *Advances in Cryptology - EUROCRYPT '97, International Conference on the Theory and Application of Cryptographic Techniques, Konstanz, Germany, May 11-15, 1997, Proceeding*, volume 1233 of *Lecture Notes in Computer Science*, pages 103–118. Springer, 1997.

[CW92] B. A. Coan and J. L. Welch. Modular construction of a byzantine agreement protocol with optimal message bit complexity. *Information and Computation*, 97:61–85, March 1992. Preliminary version appeared in PODC '89.

[DS83] D. Dolev and H. R. Strong. Authenticated algorithms for Byzantine agreement. *SIAM Journal on Computing*, 12(4):656–666, 1983. Preliminary version appeared in STOC '82.

[FH06] M. Fitzi and M. Hirt. Optimally efficient multi-valued Byzantine agreement. In *Proceedings of the 26th annual ACM symposium on Principles of distributed computing*, PODC '06, pages 163–168, New York, NY, USA, 2006. ACM.

[Fit03] M. Fitzi. *Generalized Communication and Security Models in Byzantine Agreement*. PhD thesis, ETH Zurich, March 2003. Reprint as vol. 4 of *ETH Series in Information Security and Cryptography*, ISBN 3-89649-853-3, Hartung-Gorre Verlag, Konstanz, 2003.

[FM88] P. Feldman and S. Micali. Optimal algorithms for byzantine agreement. In J. Simon, editor, *STOC*, pages 148–161. ACM, 1988.

[GGHR14] S. Garg, C. Gentry, S. Halevi, and M. Raykova. Two-round secure mpc from indistinguishability obfuscation. In Y. Lindell, editor, *TCC*, volume 8349 of *Lecture Notes in Computer Science*, pages 74–94. Springer, 2014.

[GKKO07] J. A. Garay, J. Katz, C.-Y. Koo, and R. Ostrovsky. Round complexity of authenticated broadcast with a dishonest majority. In *Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science*, FOCS '07, pages 658–668, Washington, DC, USA, 2007. IEEE Computer Society.

[GMW87] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game. In *Proceedings of the 19th annual ACM symposium on Theory of computing*, STOC '87, pages 218–229, New York, NY, USA, 1987. ACM.

[GY89] R. L. Graham and A. C. Yao. On the improbability of reaching byzantine agreements. In *Proceedings of the twenty-first annual ACM symposium on Theory of computing*, STOC '89, pages 467–478, New York, NY, USA, 1989. ACM.

[HMR14] M. Hirt, U. Maurer, and P. Raykov. Broadcast amplification. In Y. Lindell, editor, *TCC*, volume 8349 of *Lecture Notes in Computer Science*, pages 419–439. Springer, 2014.

[KK06] J. Katz and C.-Y. Koo. On expected constant-round protocols for byzantine agreement. In *In Advances in Cryptology—Crypto '06*, pages 445–462. Springer-Verlag, 2006.

[LV10a] G. Liang and N. Vaidya. Complexity of multi-value byzantine agreement. Technical report, University of Illinois at Urbana-Champaign, 2010. Available at `http://www.crhc.illinois.edu/wireless/papers/ba_sum_capacity_0729.pdf`.

[LV10b] G. Liang and N. Vaidya. Short note on complexity of multi-value byzantine agreement. *CoRR*, abs/1007.4857, 2010.

[LV11] G. Liang and N. Vaidya. Error-free multi-valued consensus with Byzantine failures. In *Proceedings of the 30th annual ACM symposium on Principles of distributed computing*, PODC '11, pages 11–20, New York, NY, USA, 2011. ACM. The arxiv version is available at http://arxiv.org/abs/1101.3520.

[LV14] G. Liang and N. Vaidya. Personal Communication, 2014.

[Pat11] A. Patra. Error-free multi-valued broadcast and Byzantine agreement with optimal communication complexity. In *Proceedings of the 15th international conference on Principles of Distributed Systems*, OPODIS '11, pages 34–49. Springer, 2011.

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

[PW96] B. Pfitzmann and M. Waidner. Information-theoretic pseudosignatures and Byzantine agreement for $t \geq n/3$. Technical report, IBM Research, 1996.

[Rog06] P. Rogaway. Formalizing human ignorance. In P. Q. Nguyen, editor, *VIETCRYPT*, volume 4341 of *Lecture Notes in Computer Science*, pages 211–228. Springer, 2006.

[TC84] R. Turpin and B. A. Coan. Extending binary Byzantine agreement to multivalued Byzantine agreement. *Information Processing Letters*, 18(2):73–76, 1984.

[Yao79] A. C. Yao. Some complexity questions related to distributive computing (preliminary report). In *Proceedings of the eleventh annual ACM symposium on Theory of computing*, STOC '79, pages 209–213, New York, NY, USA, 1979. ACM.

[Yao82] A. C. Yao. Protocols for secure computations. In *Proceedings of the 23rd Annual Symposium on Foundations of Computer Science*, SFCS '82, pages 160–164, Washington, DC, USA, 1982. IEEE Computer Society.

## A On the Constructions of Liang and Vaidya [LV11, LV10a, LV10b]

In [LV11] it is stated that the broadcast constructions presented there can be extended to tolerate $t \geq n/3$. We contacted the authors and they said that this statement is misleading and it should have been "$t < n/2$" instead of "$t \geq n/3$" to be more clear [LV14]. Below we detail why [LV11] inherently requires $t < n/2$ and cannot be extended beyond this bound (this reasoning applies to the related constructions [LV10a, LV10b]).

Essentially, the construction relies on a player set $S$ such that all players in $S$ have the same value $v$ and $S$ is guaranteed to contain at least one correct player. The value $v$ is the value that should be agreed on. This technique requires that such $S$ is unique. Uniqueness of $S$ can be guaranteed only when $t < n/2$. When $t \geq n/2$, even if all correct players do share the same value $v$, the Byzantine players can always pretend to have a different value $v'$ and create a larger player set $S'$ just among themselves to prevent protocol from reaching agreement.