On the Complexity of Broadcast Setup*

Martin Hirt, Pavel Raykov
ETH Zurich, Switzerland
{hirt,raykovp}@inf.ethz.ch

July 5, 2013

Abstract

Byzantine broadcast is a distributed primitive that allows a specific party (called "sender") to consistently distribute a value v among n parties in the presence of potential misbehavior of up to t of the parties. Broadcast requires that correct parties always agree on the same value and if the sender is correct, then the agreed value is v. Broadcast without a setup (i.e., from scratch) is achievable from point-to-point channels if and only if $t < n/3$. In case $t \geq n/3$ a trusted setup is required. The setup may be assumed to be given initially or generated by the parties in a setup phase.

It is known that generating setup for protocols with cryptographic security is relatively simple and only consists of setting up a public-key infrastructure. However, generating setup for information-theoretically secure protocols is much more involved. In this paper we study the complexity of setup generation for information-theoretic protocols using point-to-point channels and temporarily available broadcast channels. We optimize the number of rounds in which the temporary broadcast channels are used while minimizing the number of bits broadcast with them. We give the first information-theoretically secure broadcast protocol tolerating $t < n/2$ that uses the temporary broadcast channels during only 1 round in the setup phase. Furthermore, only $\mathcal{O}(n^3)$ bits need to be broadcast with the temporary broadcast channels during that round, independently of the security parameter employed. The broadcast protocol presented in this paper allows to construct the first information-theoretically secure MPC protocol which uses a broadcast channel during only one round. Additionally, the presented broadcast protocol supports refreshing, which allows to broadcast an a priori unknown number of times given a fixed-size setup.

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

*This is the full version of the paper presented in ICALP 2013 [HR13]

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 (MPC) [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 an input, and then parties agree on a common value as an output of consensus. Consensus and broadcast are reducible to each other with the help of point-to-point channels in case $t < n/2$.

1.2 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 $s \in \{1, \ldots, n\}$. We assume that each pair of parties is connected with a secure synchronous channel, where synchronous means that the parties share a common clock and that the message delay is bounded by a constant.

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

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 $v_i = v_s$.
CONSISTENCY: All correct parties in $\mathcal{P}$ decide on the same value.
TERMINATION: Every correct party in $\mathcal{P}$ terminates.

Adversary. The faultiness of parties is modeled in terms of a central adversary corrupting some of the parties. The adversary can corrupt up to $t < n/2$ parties, making corrupted parties deviate from the protocol in any desired manner.
We consider *information-theoretic* security which captures the fact that the protocol may fail only with some negligible probability even in the settings where the adversary has unbounded computing power.

1.3 Broadcast with a Trusted Setup

Broadcast is achievable from point-to-point channels if and only if $t < n/3$. In order to tolerate $t \geq n/3$ a broadcast protocol must additionally use a trusted setup which is provided to the parties beforehand. Such a trusted setup may be assumed to be a part of the model or to be distributed by a temporarily available trusted party. In this paper we consider the alternative case where parties themselves generate the setup using point-to-point communication and temporarily available during a setup phase broadcast channels.
Formally, a *broadcast scheme* is a pair of protocols (Setup, Broadcast), where Setup generates the parties' secret states with which they start the execution of Broadcast. The Setup protocol makes uses of *temporary* broadcast and point-to-point communication channels, while Broadcast employs point-to-point channels only. It is required that the combination of Setup and Broadcast achieves broadcast defined above and Setup is independent of the sender's input v_s provided in Broadcast.

In this paper we study the efficiency of broadcast schemes, in particular the *temporary broadcast-efficiency* of the setup protocol. We employ two measures of efficiency for the temporary broadcast used in the setup protocol: *round complexity* and *bit complexity*. Round complexity denotes the number of rounds in which temporary broadcast is used and bit complexity denotes the number of bits broadcast by it.

1.4 Contributions

If computational security suffices, then in the setup phase it is enough to consistently generate a *Public-Key Infrastructure* (PKI) and then employ [DS83] to broadcast. Generating PKI requires only 1 round of temporary broadcast, in which each party broadcasts his public key. In case of information-theoretic security known solutions require $\Omega(n^2)$ rounds of temporary broadcast while broadcasting $\Omega(n^8\kappa)$ bits [PW96], where κ is a security parameter. Motivated by the gap between computational and information-theoretically secure protocols, Garay et al. [GGO12] initiated the study of information-theoretically secure broadcast schemes for $t < n/2$.[1] They focused on optimizing the temporary broadcast round complexity in the setup phase and gave a broadcast scheme which requires only 3 rounds of the temporary broadcast. The number of bits broadcast by their protocol in the setup phase is $\Omega(n^6\kappa)$. Another construction by Hirt et al. [BHR07] yields a broadcast scheme for $t < n/2$ which needs $\Omega(n)$ rounds of temporary broadcast with $\Omega(n\kappa)$ bits broadcast in the setup phase.[2]
This paper gives the first information-theoretically secure broadcast scheme that requires only 1 round of the temporary broadcast in the setup phase for $t < n/2$, which is trivially optimal. This result not only improves the broadcast round complexity of all existing broadcast schemes, but allows to construct an MPC protocol which uses only one round of broadcast (existence of such schemes was unresolved before [KK07, GGO12]).[3] Furthermore, our protocol needs to broadcast only $\mathcal{O}(n^3)$ bits in the setup phase *regardless* of how long the security parameter κ is. To our knowledge, this is the first protocol with such a property. Additionally, we give an efficient refresh protocol which allows to broadcast many values given a fixed setup. The table below summarizes the existing broadcast schemes:

Security	**Threshold**	**BC rounds**	**BC bits**	**Ref.**
comp.	$t < n$	1	$\Omega(n\kappa)$	[DS83]
inf.-theor.	$t < n$	$\Omega(n^2)$	$\Omega(n^8\kappa)$	[PW96]
	$t < n/2$	$\Omega(n)$	$\Omega(n\kappa)$	[BHR07]
		3	$\Omega(n^6\kappa)$	[GGO12]
		1	$\mathcal{O}(n^3)$	This paper

1.5 Organization of the Paper

First, we give a broadcast scheme for three players ($n = 3$) which tolerates any $t \leq 3$ number of corruptions (Section 2). Then, in Section 3 we show how to use the scheme for three

[1] Their original motivation was to optimize the number of broadcast rounds needed for information-theoretically secure MPC. The broadcast scheme they give is used as a core component of an MPC protocol.

[2] One can view this construction as a broadcast scheme by interpreting the refresh protocol presented in this paper as the setup protocol.

[3] Additionally, this shows that the lower bound of at least 2 broadcast rounds for MPC protocols given in [GGO12] is wrong.

players to obtain a broadcast scheme for arbitrary number of players n tolerating $t < n/2$ corruptions.

2 The Broadcast Scheme for $n = 3$ and $t \leq 3$

In this section we consider a setting consisting of only *three* players and show how one can prepare a setup that allows a designated player to broadcast one bit. The broadcast scheme is based on a series of reductions among modifications of well known cryptographic primitives. On the highest level we show that: (1) temporary broadcast allows to construct information checking, (2) information checking allows to construct verifiable secret sharing, and (3) verifiable secret sharing allows to construct a setup protocol.
Additionally, we present an optimization which allows players to efficiently generate many setups in parallel. This reduces the number of bits broadcast in the setup phase while still requiring only one round of a temporary broadcast.

2.1 Detectable Information Checking

Information checking (IC) is an information-theoretically secure method for authenticating data among three players D, I, R. The dealer D holds a secret value s from some finite field $\mathbb{F}$ which he sends to an intermediary I together with an authentication information. The intended recipient R gets a verification information. Later I sends s' and some authentication information to R, who uses the verification information to check whether $s = s'$. We propose a non-robust modification of IC which we call *Detectable Information Checking* (DIC), where the authentication phase may either *succeed* or *abort*. If it aborts then the parties output a *dispute* Δ, which is a pair of players (i.e., $\Delta \subseteq \{D, I, R\}$), at least one of them being Byzantine.
Formally, a DIC scheme is a pair[4] of protocols $(\mathsf{ICSetup}, \mathsf{ICReveal})$, where in $\mathsf{ICSetup}$ D inputs s and then parties either abort with a dispute or succeed by saving a local state. If $\mathsf{ICSetup}$ succeeds then parties invoke $\mathsf{ICReveal}$ such that R outputs s' or $\perp$. A DIC scheme must satisfy the following security properties:

Completeness: If D, I and R are correct, then $\mathsf{ICSetup}$ succeeds and R will output $s' = s$ in $\mathsf{ICReveal}$.

Non-Forgery: If D and R are correct and $\mathsf{ICSetup}$ succeeds, then R will output $s' = s$ or $s' = \perp$ in $\mathsf{ICReveal}$.

Commitment: If I and R are correct and $\mathsf{ICSetup}$ succeeds, then at the end of $\mathsf{ICSetup}$ I knows a value s' such that R will output s' in $\mathsf{ICReveal}$.

Privacy: If D and I are correct, then R obtains no information on s during $\mathsf{ICSetup}$.

Detection: In case $\mathsf{ICSetup}$ aborts every correct party outputs the same dispute Δ.

Termination: Every correct party terminates $\mathsf{ICSetup}$, respectively $\mathsf{ICReveal}$.

We say that a DIC scheme is information-theoretically secure if the properties above are guaranteed with overwhelming probability.

The protocol. In this section we present a protocol for DIC based on [CDD+99]. The secret and the verification information will lie on a line (a polynomial of degree 1), where the secret will be the value in 0. Intermediary I gets to know the line, while the recipient R learns a value on this line in some secret point α unknown to I. In the reveal phase, R accepts a line from I only if the evaluation of this line in α is correct. In order to ensure commitment property I verifies whether D distributed consistent information to him and to R. This is done during one

[4]Sometimes (as in [CDD+99, GGO12]) IC is presented as a triple of protocols Distr, $\mathsf{AuthVal}$, $\mathsf{RevealVal}$ where Distr and $\mathsf{AuthVal}$ is a more fine-grained representation of $\mathsf{ICSetup}$.

broadcast round in ICSetup. As an outcome of the broadcast parties may succeed in ICSetup or abort with a dispute.

Let $s, y, z, \alpha \in \mathbb{F}$. We say that the triple (s, y, z) is 1_α-consistent provided that three points $(0, s)$, $(1, y)$ and (α, z) lie on a line in $\mathbb{F}$ (that is, for some $L(x) = bx + c$ over $\mathbb{F}$, we have $L(0) = s$, $L(1) = y$ and $L(\alpha) = z$).

Protocol ICSetup(s):

1. Dealer D chooses a random value $\alpha \in \mathbb{F} \setminus \{0, 1\}$ and additional random $y, z \in \mathbb{F}$ such that (s, y, z) is 1_α-consistent. In addition it chooses a random 1_α-consistent vector (s', y', z'). D sends s, s', y, y' to I and α, z, z' to R.
2. Intermediary I chooses a random $d \in \mathbb{F}$. I sends d to D and $d, s' + ds, y' + dy$ to R. Let d_1 denote the value received by D and d_2, s'', y'' the values received by R.
3. Every player **broadcasts** the following (in parallel):
   - 3.1 Dealer broadcasts triple $T_D = (d_1, s' + d_1 s, y' + d_1 y)$.
   - 3.2 Intermediary broadcasts triple $T_I = (d, s' + ds, y' + dy)$.
   - 3.3 Recipient broadcasts triple $T_R = (d_2, s'', y'')$ and a bit b, where b is 1 if $(s'', y'', z' + d_2 z)$ is 1_α-consistent and 0 otherwise.
4. Every player checks for abortion:
   If $T_D \neq T_I$ then abort with $\Delta = \{D, I\}$.
   If $T_R \neq T_I$ then abort with $\Delta = \{R, I\}$.
   If $T_D = T_R$ and $b = 0$ then abort with $\Delta = \{D, R\}$.
   Otherwise, the protocol succeeds and D stores nothing, I stores (s, y) and R stores (α, z).

Protocol ICReveal(s):

1. Intermediary I sends s, y to R. If (s, y, z) is 1_α-consistent then R decides on s, otherwise on $\perp$.

Lemma 1. The pair of protocols (ICSetup, ICReveal) achieves DIC (except with probability $\mathcal{O}(1/|\mathbb{F}|)$). Furthermore, ICSetup uses the underlying broadcast channel during one predetermined round (where each player broadcasts $\mathcal{O}(\log |\mathbb{F}|)$ bits) and ICReveal does not use broadcast at all.

PROOF First, we show that each DIC property is satisfied:

COMPLETENESS: It is clear that if all parties are honest, then ICSetup succeeds and R outputs s in ICReveal.

NON-FORGERY: If D and R are correct and ICSetup succeeds, then prior to ICReveal I has no information on α except that it is different from 0 and 1. For I making R output in ICReveal any value different from s is equivalent to guessing α, which he can do with probability at most $1/(|\mathbb{F}| - 2)$.

COMMITMENT: If I and R are correct and ICSetup succeeds, then $(s' + ds, y' + dy, z' + dz)$ is 1_α-consistent with a randomly chosen d. R accepts s if (s, y, z) is 1_α-consistent. The probability that (s, y, z) is not 1_α-consistent, given that $(s' + ds, y' + dy, z' + dz)$ is 1_α-consistent, is at most $1/|\mathbb{F}|$.

PRIVACY: Here D and I are correct. In ICSetup R observes only $\alpha, z, z', d, s' + ds, y' + dy$. R knows that $(s' + ds, y' + dy, z' + dz)$ is 1_α-consistent, hence his knowledge is equivalent to $\alpha, z, z', d, s' + ds$ which is independent of s.

DETECTION: Since parties compute Δ based only on the information that was broadcast they decide on the same Δ. We are left to show that Δ forms a dispute. If D and I are correct then it must be that $T_D = T_I$, analogously if I and R are correct then $T_I = T_R$. If $T_D = T_R$ and D is correct then R must hold a 1_α-consistent triple $(s'', y'', z' + d_2 z)$.

TERMINATION: Due to their specifications, protocols ICSetup and ICReveal always terminate.

Finally, the protocol ICSetup uses broadcast during only one round at Step 3 where each party

broadcasts $\mathcal{O}(\log |\mathbb{F}|)$ bits, while the protocol ICReveal does not broadcast. ∎

2.2 Detectable Verifiable Secret Sharing

Verifiable secret sharing (VSS) is a classical cryptographic primitive for secure sharing of a secret. It lies in a core of many protocols for multi-party computation and is used in various applications. In this section we consider a very restricted setting for VSS consisting of only three players D, P_1, P_2. The dealer D holds a secret value s from some finite field $\mathbb{F}$ and shares it among two recipients P_1, P_2, such that individually they have no information about s. Later in the reconstruction phase all correct recipients reconstruct the same s' which equals s if the dealer is correct. We consider a non-robust version of VSS which we call *Detectable Verifiable Secret Sharing* (or short DVSS), where the sharing phase can *abort* in the presence of malicious behavior.

Formally, a DVSS scheme is a pair of protocols (VSS-Share, VSS-Rec), where in VSS-Share D inputs s and then parties either abort with a dispute Δ or succeed by saving a local state. If VSS-Share succeeds then the parties invoke VSS-Rec such that the recipients reconstruct the shared secret. A DVSS scheme must satisfy the following security properties:

Correctness: If VSS-Share succeeds, then there exists a fixed value $s' \in \mathbb{F}$ which will be reconstructed as a result of VSS-Rec by every correct recipient. If D is correct then $s' = s$.

Privacy: If D is correct, then corrupted parties obtain no information on s in VSS-Share.

Detection: If D, P_1 and P_2 are correct then VSS-Share always succeeds. In case VSS-Share aborts every correct party outputs the same dispute Δ.

Termination: Every correct party terminates VSS-Share, respectively VSS-Rec.

We say that a DVSS scheme is information-theoretically secure if the properties above are guaranteed with overwhelming probability. Furthermore, by t-DVSS we denote a DVSS scheme tolerating $\leq t$ corruptions, and by t-DVSS$^+$ we denote a scheme which is t-DVSS but Detection and Termination hold for arbitrary number of corruptions.

The protocol. In this section we present an implementation of 1-DVSS$^+$ based on DIC. In order to share a secret s the dealer generates two random values s_1, s_2 such that $s_1 + s_2 = s$. Then the dealer authenticates s_1 by invoking ICSetup(s_1) where P_1 acts as intermediary and P_2 as recipient. In parallel, the dealer authenticates s_2 by invoking ICSetup(s_2) where P_2 acts as intermediary and P_1 as recipient. If one of the ICSetup invocations aborts then VSS-Share aborts as well. The reconstruction consists of running ICReveal among P_1, P_2 and the dealer sending the secret s to both recipients. If a correct recipient obtains non-$\perp$ value in ICReveal, then it reconstructs $s' = s_1 + s_2$, otherwise it outputs $s' = s$ received from the dealer.

Protocol VSS-Share(s):

1. The dealer D chooses random $s_1, s_2 \in \mathbb{F}$ such that $s = s_1 + s_2$.
2. Then D, P_1, P_2 execute in parallel ICSetup(s_1) where P_1 is intermediary and P_2 is recipient and ICSetup(s_2) where P_2 is intermediary and P_1 is recipient.
3. Every player checks for abortion:
   If any of ICSetup aborts, then VSS-Share aborts as well with a dispute Δ output by one of ICSetup (in case both aborts, then output the dispute from the first).
   Otherwise, the protocol succeeds and each player saves the states obtained from both ICSetup invocations.

Protocol VSS-Rec(s):

1. The dealer D sends s to both recipients.
2. Players P_1, P_2 invoke ICReveal(s_1) and ICReveal(s_2). If P_i obtains a share $s_j \neq \perp$ from the other recipient P_j then it decides on $s_i + s_j$, otherwise it decides on a value s received from the dealer.

Lemma 2. The pair of protocols (VSS-Share, VSS-Rec) achieves 1-DVSS$^+$ (except with probability $\mathcal{O}(1/|\mathbb{F}|)$). Furthermore, VSS-Share uses the underlying broadcast channel in only one predetermined round (where each player broadcasts $\mathcal{O}(\log|\mathbb{F}|)$ bits) and VSS-Rec does not use broadcast at all.

PROOF First, we show that each 1-DVSS$^+$ property is satisfied:

CORRECTNESS: We prove that Correctness holds when the number of corrupted parties $t \leq 1$. If all parties are correct ($t = 0$) then, due to the Completeness property of DIC, correct recipients will output $s_1 + s_2 = s$ in VSS-Rec. Assume now only one party is corrupted ($t = 1$). Consider two cases: (1) *the dealer is corrupted* and (2) *one of the recipients is corrupted*. In case (1), due to the Commitment property of DIC, player P_1 knows s_1 such that P_2 outputs s_1 in ICReveal, and P_2 knows s_2 such that P_1 outputs s_1 in ICReveal (except with probability $\mathcal{O}(1/|\mathbb{F}|)$). Hence, since both P_1 and P_2 are correct they will both output $s' = s_1 + s_2$ in VSS-Rec. In case (2), wlog assume that a correct recipient is P_1. Due to the Non-Forgery property of DIC, player P_1 may obtain in ICReveal only s_2 or $\perp$ (except with probability $\mathcal{O}(1/|\mathbb{F}|)$). In both cases P_1 outputs $s' = s_1 + s_2$ or $s' = s$, which is the same for a correct D.

PRIVACY: Wlog assume that a corrupted recipient is P_2 and P_1 is correct. Due to the Privacy property of DIC, player P_2 who acts as a recipient in ICSetup(s_1) has no information about s_1 in the end of ICSetup. Hence, after VSS-Share P_2's view contains only s_2 which is independent of s.

DETECTION: If all parties are correct then, due to the Completeness property of DIC, both ICReveal(s_1) and ICReveal(s_2) succeed and hence VSS-Share succeeds. Parties abort if and only if one of ICSetup aborts. Due to the Detection property of ICSetup parties output the same dispute Δ.

TERMINATION: Due to their specifications, protocols VSS-Share and VSS-Rec always terminate.

Finally, the protocol VSS-Share uses the underlying broadcast channel only at Step 2 where ICSetup uses it, while the protocol VSS-Rec does not broadcast. Since two instances of ICSetup are invoked in parallel, according to Lemma 1 there is only one predetermined round where the underlying broadcast channel is invoked and each player broadcasts $\mathcal{O}(\log|\mathbb{F}|)$ bits. ∎

2.3 Broadcast Scheme

We consider three players $\{D, P_1, P_2\}$, where D is the sender (also called the dealer) and P_1, P_2 are the recipients. In the setup phase parties execute protocol Setup_3 which makes use of a temporary broadcast channel. The setup generation may abort by outputting a dispute Δ. Later the dealer D can broadcast a bit value with the protocol $\mathsf{Broadcast}_3$ using the setup created. This is done differently depending on whether the preceding Setup_3 aborted or succeeded.

The protocol. The protocol for generating a setup uses 1-DVSS$^+$ together with *Message Authentication Codes* (MACs). We employ a MAC scheme which uses a preshared key. In the setup phase, the dealer shares a random key from some finite field $\mathbb{F}$ using VSS-Share. In order to broadcast a message later, the dealer sends the message together with its authentication information to recipients. Then the recipients exchange the messages they received from the dealer. It is guaranteed that if both recipients are correct then they hold the same set of at most two messages. Then the parties invoke VSS-Rec and learn the key for the MAC scheme used.

Recipients decide on the message with a valid MAC, respectively on $\perp$ if no/both messages match.

In order to construct an information-theoretically secure MAC scheme, we employ a trivially unforgeable authentication scheme for bits. To authenticate a message consisting of one bit b the dealer sends to the recipients P_1, P_2 either the left part of the preshared key ($b = 0$) or its right part ($b = 1$).

For the simplicity of the presentation, let $\overline{P_1}$ denote P_2 and $\overline{P_2}$ denote P_1. Additionally, let $\mathbb{F}$ be a finite field of $2^{2\kappa}$ elements interpreted as bit strings of length 2κ.

Protocol Setup_3:

1. The dealer generates random $k_0||k_1 \in \{0,1\}^{2\kappa}$.
   Parties execute $\mathsf{VSS\text{-}Share}(k_0||k_1)$. If $\mathsf{VSS\text{-}Share}$ aborts, then abort as well and output the dispute Δ.

Protocol $\mathsf{Broadcast}_3(b)$:

1. If Setup_3 succeeded
   - 1.1 The dealer sends k_b to both P_1, P_2. Denote the values received by the players P_1 and P_2 with a_1 and a_2, respectively.
   - 1.2 The recipients exchange a_1 and a_2 and form a set of authenticators $A = \{a_1, a_2\}$.
   - 1.3 The players execute $\mathsf{VSS\text{-}Rec}$ and get $k_0||k_1$ as an output.
   - 1.4 The recipients decide on 0 if $k_0 \in A$ and on 1 otherwise. The dealer decides on b.
2. else Setup_3 aborted with Δ then
   - 2.1 If $\Delta = \{D, P_i\}$ then the dealer sends b to $\overline{P_i}$ who forwards it to P_i. All parties decide on the values received. The dealer decides on b.
   - 2.2 If $\Delta = \{P_1, P_2\}$ then the dealer sends b to P_1 and P_2. All parties decide on the values received. The dealer decides on b.

Lemma 3. The protocol $\mathsf{Broadcast}_3$ achieves broadcast (of b) given that Setup_3 has been executed before (except with probability $\mathcal{O}(2^{-\kappa})$). Furthermore, Setup_3 is independent of b and uses the temporary available broadcast channel in only one predetermined round (where each party broadcasts $\mathcal{O}(\kappa)$ bits).

PROOF First, we prove that broadcast properties are satisfied. If Setup_3 outputted Δ, then by inspection of the cases it is clear that $\mathsf{Broadcast}_3$ achieves broadcast. Assume now that Setup_3 succeeded. We consider four possible cases of player corruption:

(All players are correct) Due to the Correctness and Detection properties of DVSS$^+$, $\mathsf{VSS\text{-}Share}$ succeeds and all players reconstruct dealer's $k_0||k_1$ in $\mathsf{VSS\text{-}Rec}$. Hence, all players decide on dealer's b in $\mathsf{Broadcast}_3$.

(Two players are correct) Consider two possibilities: (1) *both recipients are correct*, and (2) *the dealer and one of the recipients are correct.* In both cases Termination clearly holds.
In case (1) Validity clearly holds. Due to the Correctness property of DVSS$^+$ for $t = 1$, both recipients reconstruct the same $k_0||k_1$ (except with probability $\mathcal{O}(1/|\mathbb{F}|)$). Since they hold the same set A and use the same key $k_0||k_1$, the recipients decide on the same value.
Consider case (2). Wlog assume that P_1 is correct and P_2 is Byzantine. Due to the Correctness property of DVSS$^+$ for $t = 1$, player P_1 obtains the key $k_0||k_1$ shared by the dealer (except with probability $\mathcal{O}(1/|\mathbb{F}|)$). Due to the Privacy property of DVSS$^+$ for $t = 1$, player P_2 does not learn the authentication information for the bit which was not sent by the dealer. The probability that P_2 forges a MAC for an unknown bit is the same as guessing the corresponding part of the key which is at most $2^{-\kappa}$. Hence, a correct recipient decides on the value send by the correct sender (except with probability $\mathcal{O}(2^{-\kappa})$). Since the dealer always decides on the value b, both Validity and Consistency hold.

(One player is correct) If a recipient is correct, then broadcast properties are clearly satisfied. If the dealer is correct then Termination and Consistency are clearly satisfied, while Validity

follows from the fact that the dealer decides on the value he proposes.
(No player is correct) No properties are needed to be proven here.

The protocol Setup_3 uses temporary broadcast only during the VSS-Share invocation at Step 1. Due to Lemma 2 VSS-Share uses broadcast only one predetermined round (where each player broadcasts $\mathcal{O}(\kappa)$ bits) and so does Setup_3. ∎

2.4 Efficient Parallel Setup

In the previous section we have shown how to generate a setup for one bit broadcast. An obvious approach for generating a setup for an ℓ bit message is to prepare ℓ such setups. In this section we show how to efficiently parallelize ℓ setup invocations such that the temporary broadcast is used only small number of times. As a key ingredient of such a parallelization, we present the protocol $\mathsf{BCFromTenBits}_3$ which given an opportunity to broadcast 10 bits allows to broadcast a message of arbitrary fixed length. The protocol is perfectly secure, i.e., its security depends only on the security of the underlying broadcast.
The protocol $\mathsf{BCFromTenBits}_3$ works recursively. At each iteration the parties reduce the broadcast of a long message to broadcasting a shorter message. The recursion stops once the dealer is supposed to broadcast 10 bits only. Each recursive iteration has the same structure: first the dealer distributes the long message using point-to-point channels, then the recipients exchange the messages received from the dealer, and afterwards each recipient forwards to the dealer the message received from the other recipient. Finally, the dealer broadcasts a hint that allows each correct recipient to decide on one of the messages he has received. The hint consists of a key k for a special *c-identifying predicate* (see later). The task of broadcasting a long message is now reduced to broadcasting the key k which has a smaller bit-length than the message.

Identifying predicates. An identifying predicate allows to identify a specific element v from some small subset $S \subseteq \mathcal{D}$ where $\mathcal{D}$ is a potentially large domain. More formally, a c-identifying predicate is a function $Q : \mathcal{D} \times \mathcal{K} \to \{0,1\}$ such that for any $S \subseteq \mathcal{D}$ with $|S| \leq c$ and any value $v \in S$ there exists a key $k \in \mathcal{K}$ with $Q(v,k) = 1$ and $Q(v',k) = 0$ for all $v' \in S \setminus \{v\}$. The goal of constructing such a function Q is to have $|\mathcal{K}|$ as small as possible given c and $|\mathcal{D}|$.
Assume now $\mathcal{D} = \{0,1\}^\ell$, then Q can be constructed as following: for a set S and a value v, we find $c-1$ bit positions $p_1, \ldots, p_{c-1}$ such that v differs from any other value $v' \in S$ in some position p_i. Then the key k for a set S and a value v is defined by positions $p_1, \ldots, p_{c-1}$ and bits $b_1, \ldots, b_{c-1}$ which v has at these positions. Hence, for this Q the key space $\mathcal{K} = \{0,1\}^{(c-1)(\lceil \log \ell \rceil + 1)}$.

Protocol $\mathsf{BCFromTenBits}_3(v \in \{0,1\}^\ell)$:

1. If $\ell \leq 10$ then the parties use the underlying broadcast given in order to broadcast v.
2. Otherwise:
   - 2.1 The dealer sends v to P_1 and P_2. Denote the values received by v_1 and v_2.
   - 2.2 P_1 sends v_1 to P_2, denoted by v_{12}, and P_2 sends v_2 to P_1, denoted by v_{21}. P_1 forms the set $V_1 = \{v_1, v_{21}\}$ and P_2 forms the set $V_2 = \{v_2, v_{12}\}$.
   - 2.3 The recipients send v_{21} and v_{12} to the dealer. Denote received values by v_{210} and v_{120}. Let $S = \{v, v_{120}, v_{120}\}$.
   - 2.4 The dealer chooses a key k for the 3-identifying predicate Q with the domain $\{0,1\}^\ell$, the set of values S and the value v. The parties invoke $\mathsf{BCFromTenBits}_3(k)$ recursively. Let k' denote the result of the broadcast.
   - 2.5 Each recipient P_i decides on a unique $v' \in V_i$ such that $Q(v',k') = 1$ (if $k' = \perp$ or none/both values in V_i have Q equal to 1, then decide on $\perp$). The dealer decides on v.

Lemma 4. The protocol $\mathsf{BCFromTenBits}_3$ perfectly secure achieves broadcast (of v from a predetermined domain $\{0,1\}^\ell$) given that the dealer can broadcast 10 bits. Furthermore, the dealer needs to broadcast 10 bits only in one predetermined round which index depends only on ℓ.

PROOF **($\mathsf{BCFromTenBits}_3$ implements broadcast)** We prove that $\mathsf{BCFromTenBits}_3$ implements broadcast under the assumption that the recursive invocation of $\mathsf{BCFromTenBits}_3(k)$ already guarantees broadcast properties. In case all players are correct or at least two are Byzantine the broadcast properties are clearly satisfied. The only two non-trivial cases left are as follows:

(The sender D is corrupted, both recipients are correct) Then we need to prove only Consistency property. We have that both recipients have the same set of values $V_1 = V_2$. Since they both reconstruct the same key k' as an outcome of the recursive call to $\mathsf{BCFromTenBits}_3$ (its Consistency property) they must decide on the same value as well.

(One of the recipients is corrupted, the sender is correct) Then we need to prove only Validity property. Wlog assume P_2 is corrupted and P_1 is correct. Since sender's $v \in V_1$ and $V_1 \subseteq S$, the dealer chooses k such that $Q(v,k) = 1$ and for all other values in V_1 the predicate Q is 0. Since the recursive call to $\mathsf{BCFromTenBits}_3$ satisfies the Validity property, the player P_1 gets a correct key $k' = k$ and, after applying Q, decides on v.

(Complexity analysis) The protocol $\mathsf{BCFromTenBits}_3$ works recursively. We denote the broadcast input length at each recursive invocation r to be T_r. We have that $T_0 = \ell$ and T_{i+1} is defined recursively to be $2\lceil \log T_i \rceil + 2$. Hence, this implies that $T_{i+1} < T_i$ for any $T_i > 10$. Consequently, in the end we need to broadcast at most 10 bits. In total, there are $\mathcal{O}(\log^* \ell)$ recursive iterations, where $\log^*(\cdot)$ is the iterated logarithm function. ∎

Parallelizing Setup_3. We run many instances of Setup_3 in parallel such that each player consolidates the values it needs to broadcast in one string which is broadcast with $\mathsf{BCFromTenBits}_3$. The protocol $\mathsf{BCFromTenBits}_3$ uses temporary broadcast as its underlying primitive for broadcasting 10 bits. The following lemma summarizes the properties achieved by this construction:

Lemma 5. The broadcast scheme described above generates ℓ setups. Furthermore, the temporary broadcast is used in one predetermined round only where each player broadcasts at most 10 bits.

3 Broadcast Scheme for any n and $t < n/2$

Broadcast is achievable from point-to-point channels without a trusted setup if and only if $t < n/3$. Fitzi and Maurer [FM00] proposed a construction of a broadcast protocol for $t < n/2$ from the broadcast channels among every triple of players. In this paper we use following theorem from [FM00]:

Theorem 1. In the model where broadcast is available among every triple of players there is a protocol that implements broadcast among n players tolerating $t < n/2$ corruptions. This protocol invokes underlying triple broadcast channels at most n times for each triple (P_i, P_j, P_k), where P_i is the sender and P_j, P_k are the recipients.

The protocol we present generates n setups for each triple of players where P_i is the sender and P_j, P_k are the recipients. It is done by invoking in parallel procedure Setup_3.

Protocol Setup_n:

1. For each possible sender P_i and recipients P_j, P_k (i, j, k are all different): Parties P_i, P_j, P_k invoke Setup_3 n times in parallel.

Protocol $\mathsf{Broadcast}_n(b)$:

1. Use protocol by [FM00], whenever P_i needs to broadcast 1 bit among P_j, P_k use the protocol $\mathsf{Broadcast}_3$ with the prepared setup.

Theorem 2. The protocol $\mathsf{Broadcast}_n$ achieves broadcast (of b) for $t < n/2$ given that Setup_n has been executed before (except with probability $\mathcal{O}(n^4 2^{-\kappa})$). Furthermore, Setup_n is independent of b and uses temporary broadcast procedure in only one predetermined round (where players needs to broadcast $\mathcal{O}(n^3)$ bits).

PROOF Due to their specifications, protocols Setup_n and $\mathsf{Broadcast}_n$ always terminate. Since $\mathsf{Broadcast}_3$ has failure probability $\mathcal{O}(2^{-\kappa})$ then due to Theorem 1 the probability that $\mathsf{Broadcast}_n$ does not achieve broadcast is bounded by $\mathcal{O}(n^4 2^{-\kappa})$. The protocol Setup_n uses the temporary broadcast only during the parallel invocation of Setup_3 at Step 1. Due to Lemma 3 Setup_3 uses broadcast only in one predetermined round and so does Setup_n. Due to Lemma 5, each party in a triple needs to broadcasts 10 bits in order to generate n setups. There are $\binom{n}{3}$ triples in total, so players broadcast $\mathcal{O}(n^3)$ bits in one round of the setup phase. ∎

3.1 Refreshing the Setup

A broadcast scheme, as we defined it, works in the following way: first we run the Setup_n protocol for the setup generation and then execute $\mathsf{Broadcast}_n$ to broadcast a value. The invocation of the $\mathsf{Broadcast}_n$ protocol actually "consumes" the setup, i.e., one setup may be used for broadcasting one value only. A concept of a *refresh* protocol allows to extend a fixed setup for many setups [PW96].

A refresh protocol for the presented scheme with arbitrary number of players n runs refresh protocol for each triple of players. A refresh procedure for a triple is sketched below. Let us denote the three players with P_1, P_2, P_3. We assume that the initial setup allows each player P_1, P_2, P_3 to broadcast 10 bits. We construct a setup allowing each party to broadcast any number of ℓ bits. For each possible sender (P_1,P_2 or P_3) the players run in parallel ℓ instances of Setup_3 where the values each player P_i is supposed to broadcast in every instance are consolidated in one string s_i of $\mathcal{O}(\ell\kappa)$ bits. In order to broadcast s_i, each P_i uses the protocol $\mathsf{BCFromTenBits}_3$ where 10 initial setups are used.

3.2 Application to MPC

One of the most important applications of broadcast schemes is a secure MPC [GMW87]. In the settings with $t < n/2$ MPC is achievable from point-to-point communication only when a broadcast channel is available additionally. Such a broadcast channel is usually simulated with a broadcast scheme which tolerates $t < n/2$. The broadcast scheme presented in this paper shows that there is an information-theoretically secure MPC protocol which uses a broadcast channel during only one round. This is achieved by running Setup_n sufficiently many times in parallel (this uses one round of temporary broadcast) to prepare enough setups, and simulating all invocations to broadcast in the MPC protocol by $\mathsf{Broadcast}_n$. Furthermore, if it is not known beforehand how many times the broadcast channel will be used, the refresh protocol is used. It allows to generate arbitrary many setups from a given fixed-size setup.

4 Conclusions

In this paper we study the efficiency of information-theoretically secure broadcast schemes in terms of the temporary broadcast usage during the setup phase. All known schemes [PW96,

BHR07, GGO12] use temporary broadcast in strictly more than one round in the setup phase. We give a broadcast scheme for $t < n/2$ which requires only 1 round of the temporary broadcast. Furthermore, the presented scheme requires only $\mathcal{O}(n^3)$ bits to be broadcast in the setup phase.

References

[BGP92] P. Berman, J. A. Garay, and K. J. Perry. Bit optimal distributed consensus. In *Comp. Sc. Res.*, pages 313–322. Plenum Publ. Corp., NY, USA, 1992.

[BHR07] Z. Beerliova-Trubiniova, M. Hirt, and M. Riser. Efficient Byzantine agreement with faulty minority. In *ASIACRYPT 2007*, volume 4833 of *LNCS*, pages 393–409, 2007.

[CDD+99] R. Cramer, I. Damgård, S. Dziembowski, M. Hirt, and T. Rabin. Efficient multiparty computations secure against an adaptive adversary. In J. Stern, editor, *EUROCRYPT '99*, volume 1592 of *LNCS*, pages 311–326, May 1999.

[CW92] B. A. Coan and J. L. Welch. Modular construction of a byzantine agreement protocol with optimal message bit complexity. *Inf. and C.*, 97:61–85, 1992.

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

[FM00] M. Fitzi and U. Maurer. From partial consistency to global broadcast. STOC '00, pages 494–503, New York, NY, USA, 2000. ACM.

[GGO12] J. A. Garay, C. Givens, and R. Ostrovsky. Broadcast-efficient secure multiparty computation. *IACR Cryptology ePrint Archive*, 2012:130, 2012.

[GMW87] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game. STOC '87, pages 218–229, New York, NY, USA, 1987. ACM.

[HR13] M. Hirt and P. Raykov. On the complexity of broadcast setup. In F. Fomin, R. Freivalds, M. Kwiatkowska, and D. Peleg, editors, *Automata, Languages, and Programming*, volume 7965 of *Lecture Notes in Computer Science*, pages 552–563. Springer Berlin Heidelberg, 2013.

[KK07] J. Katz and C.-Y. Koo. Round-efficient secure computation in point-to-point networks. In *EUROCRYPT '07*, volume 4515 of *LNCS*, pages 311–328, 2007.

[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.