On Composability of Reliable Unicast and Broadcast

Abstract. In the recent past composability has emerged as a key requirement for various distributed protocols. The robustness of a composable protocol *depends* on the environment in which the protocol runs. In other words it is not enough for a protocol to be robust when it runs in isolation or in a "stand-alone" setting but it should be robust even in an environment where several copies of the same protocol or other protocols are simultaneously running.

We investigate the composability for protocols that tolerate a bounded adversary modelled as a probabilistic polynomial time Turing machine. We examine composability of protocols for two fundamental problems in distributed computing - reliable unicast and reliable broadcast. We show that any composable protocol for reliable unicast tolerating an adversary, that corrupts up to any t nodes, requires $2t+1$ connectivity and any composable protocol for reliable broadcast tolerating an adversary, that corrupts up to any t nodes, requires $n > 3t$ and $2t+1$ connectivity.

Key words: Composability, Reliable Message Transmission, Authenticated Byzantine Agreement.

1 Introduction

The problem of reliable communication is one of the fundamental problems in distributed computing. The communication can be from one node to another (unicast), one to many (multicast), or from one node to all other nodes (broadcast). To state informally, problem of unicast is a sender S wishes to communicate a message m to a receiver R through an arbitrary synchronous network, some of whose nodes are under the control of an adversary. It was first studied in detail by Dolev *et al.* [9] under the setting of threshold Byzantine adversary that can actively corrupt up to any t nodes in the network. For a computationally unbounded adversary, it is proved that reliable communication between any pair of nodes is possible if and only if the network is $(2t+1)$-connected [9, 16]. In line with the observations of Dolev *et al.*[9], we completely abstract the synchronous network as a collection of κ wires connecting the sender and the receiver of which up to t wires may be Byzantine corrupt. In [9], it is proved that such an abstraction is actually *without loss of generality* in the study of the *possibility* of reliable communication. Intuitively, these κ wires represent κ vertex disjoint paths from the sender to the receiver in the network. Later, the problem of reliable communication was also studied in several other settings. For instance, [25] consider asynchronous network, [2, 28] work with *mobile* adversaries; multicast lines and hypergraphs are considered in [16, 17, 30] while non-threshold adversaries are considered in [21]; the case of directed graphs is dealt with in [7, 29].

The problem of broadcast (Byzantine agreement) was introduced by Pease *et al.* [23] in 1980. The challenge is to maintain a coherent view of the world among all the honest players in spite of faulty players trying to disrupt the same. Specifically, in a Byzantine agreement protocol over a *synchronous* network of n players, each player starts with an input from a fixed set $V = \{0,1\}$. At the end of the protocol (which may involve finitely many rounds of interaction), even if up to any t of the n players are faulty, all non-faulty players output the same value $u \in V$ and if all non-faulty players start with the same input $v \in V$, then $u = v$. There exists a very rich literature on the problem of Byzantine agreement (BA) as briefly described below. Pease *et al.* [23] showed that a protocol for BA tolerating a t-adversary over a completely connected synchronous network exists if and only if $n > 3t$. Later, studies were initiated in this problem under various settings like asynchronous networks [13], partially synchronous networks [12], incomplete networks [11], hypernetworks [15], non-threshold adversaries [14], mixed-adversaries [1], mobile adversaries [18], and probabilistic correctness [24] to name a few. An important variant of BA is the authenticated model proposed by Pease *et al.* [23]. In this model, which we hereafter refer to as *authenticated Byzantine agreement* (ABA), the players are supplemented with "magical" powers (say a Public Key Infrastructure(PKI) and digital signatures) using which the players can authenticate themselves and their messages. It is proved that in such a model, the tolerability against a t-adversary can be amazingly increased to as high as $t < n$. Dolev [10] presented efficient protocols for ABA thereby confirming the usefulness of authentication in both possibility as well as feasibility of distributed protocols. Subsequent papers on this subject include [3, 5, 27, 4, 20, 19, 26].

There has been a growing concern on the composability of protocols in the recent past. Traditionally, the robustness of a protocol was analyzed with a tacit assumption that the protocol is executed in the stand-alone setting. A protocol is said to be robust in the stand-alone sense, if the protocol is robust under the assumption that it is the only protocol that is to be executed. However in reality, several protocols are executed simultaneously with each execution oblivious of the existence of other executions. Thus the stand-alone notion of robustness is grossly inadequate. Especially in the case of reliable communication which is a fundamental primitive used in design of almost all fault-tolerant distributed protocols, a stand-alone notion of reliability is highly inappropriate. “Protocol composition” refers to an environment where participating parties are involved in many protocol executions. Further, each of the executions are oblivious of the other executions. Protocol composition although does not produce all kinds of environments, it does produce a rich variety of environments. There are several different protocol compositions that have been studied in literature. We present the definitions of a few of them below:

1. **Self Composition :** A protocol is said to be self composable if several executions of the same protocol run in a network, the protocol still remains robust.
2. **General Composition :** The protocol needs to be robust even when it is run along with several executions of other protocols.
3. **Sequential Composition :** In sequential composition, there is only one execution of a particular protocol at one point of time.
4. **Parallel Composition :** In parallel composition, there can be several executions running simultaneously.

In [6] the notion of universal composability was introduced. This model is used to prove the security of protocols under general composition. In general, one would expect that when the adversary is computationally bounded, we can achieve greater fault tolerance than against the adversary who has unlimited computational powers. To motivate our study we now give an example where the above intuition does not hold good.

2 Our Model

We consider a set of n players, denoted by $\mathbb{P}$, communicating over a synchronous network. That is, the protocol is executed in a sequence of *rounds* where in each round, a player can perform some local computation, send new messages to all the players, receive the messages sent to him in the same round by the players (and if necessary perform some more local computation), in that order. We further assume that the communication channels between any two processors is perfectly reliable. During the execution, the adversary may take full control of up to any t players in that execution. We denote by t-adversary an adversary that controls up to t of the n players. For the purpose of unicast we assume two special nodes **S** and **R** apart from $\mathbb{P}$. We further assume that adversary $\mathcal{A}$ cannot corrupt **S** and **R**. We also assume existence of a (signature/authentication) scheme where the sender signs the message to be sent. No player can forge any other player’s signature and the receiver can uniquely identify the sender of the message using the signature.

We also assume that players can run more than one executions of the same protocol in parallel . For our purpose we deal only with *stateless* composition of protocols, i.e. use of session key or any other kind of execution identifier is not permitted. The adversary $\mathcal{A}$ can use messages from one execution of the protocol in other executions of the same protocol.

3 Motivation

We now show that the stand-alone notion of robustness is not a satisfactory model and we need to examine the protocol for its composability properties. For the problem of reliable unicast in the stand alone setting even the “sign-flood-verify” protocol (given in Figure 1) is easily seen to tolerate a bounded adversary who can corrupt up to $t = \kappa - 1$ wires. However, when the adversary has unlimited computational powers, it is well known that perfect reliable communication protocol exists if and only if $\kappa > 2t$ [9].

We show that when majority of nodes (of any vertex cut-set) in a communication network is faulty, it is impossible to design a composable protocol for reliable unicast. In the case where majority of the nodes are non-faulty, we also design a single round bit optimal composable protocol that, with an arbitrarily high probability, is reliable.

For the case of reliable broadcast, the study on effects of composability was initiated by Lindel *et al.* [22]. They proved that for Byzantine agreement under composition, the additional power of authentication is rendered useless. Specifically, they showed that protocols for authenticated Byzantine agreement over complete graphs can be composed if and only if $n > 3t$ as compared to the bound of $n > t$ in stand alone setting. These results, in conjunction with the extant literature, imply that bounding the powers of adversary does not improve either fault tolerance or the communication complexity for the problem of reliable communication.

Our first major and interesting result is the impossibility of parallel self composable protocols for reliable unicast when the number of faults $t \geq \lceil \frac{\kappa}{2} \rceil$ even under the assumption that the adversary is computationally bounded. Since this result matches the bound established in [9], we arrive at an interesting conclusion that, in the composability setting the weaker

Sign-and-Flood Protocol

The protocol assumes a PKI (Public key infrastructure) for authentication of messages. The sender wishes to send a message $\mathbf{m}$ to the receiver.

1. The sender digitally *signs* the message $\mathbf{m}$ with his private key using a well-known digital signature algorithm (say DSS); let $SIGN_S(\mathbf{m})$ denote the resultant data.
2. The sender sends $\mathbf{m}, SIGN_S(\mathbf{m})$ to the receiver along *all* the κ wires.
3. The receiver receives, say, m_i, c_i along the i^{th} wire.
4. The receiver *verifies* the validity of the received c_i's, $1 \leq i \leq \kappa$, using the (corresponding) verification algorithm with the sender's public key. Let i^* be the minimum index such that the verification of c_{i^*} succeeded.
5. The receiver outputs the message m_{i^*} corresponding to signature c_{i^*}.

Fig. 1. A Naïve Protocol for Reliable Unicast Tolerating Computationally Bounded Adversaries.

adversary does not lead to higher fault tolerance. Observe that composability needs to be examined only for the case where the adversary is computationally bounded. Our next major result is concerned with a single round δ-reliable protocol (A protocol is said to be δ-reliable if it succeeds to reliably transmit a message with a probability greater than $1-\delta$). Note that the reliability of protocols tolerating bounded adversaries are based on certain assumptions on hardness of problems which are basically probabilistic in nature. In other words every such protocol may be viewed as a δ-reliable protocol for some appropriate δ corresponding to the problem. However, we have observed that these protocols are not composable, Hence a natural question is can we design an efficient composable protocol that is δ-reliable? We provide an affirmative answer by designing a δ-reliable protocol tolerating an unbounded adversary. It turns out that our protocol is trivially composable and hence environmentally robust. Further we reliably send ℓ field elements by communicating $O(\ell)$ field elements which is optimal up to a constant factor. Thus, our results imply that contrary to the intuition that we may incur higher communication cost in a composable protocol, we have obtained a composable protocol with an optimal cost. For the case reliable broadcast, we give complete characterization of the graphs over which ABA is possible. We show that ABA over a synchronous graph G is possible if and only if $n > 3t$ and G is $2t+1$ connected. Note that these conditions are same as those for the case of Byzantine agreement over graph G without use of authenticators. This shows under composition, additional power of authentication breaks down.

4 Composability of Reliable Unicast

4.1 Impossibility of Parallel Composability

In this section, we prove the impossibility of universally composable reliable communication when the number of faulty wires $t \geq \lceil \frac{\kappa}{2} \rceil$. Infact we prove a much stronger result that there exist no protocol for reliable communication that is parallel composable even twice.

The proof relies heavily on the fact that there are no unique session IDs that the sender S and the receiver R have agreed upon. Intuitively it is impossible in a stateless model for the sender and receiver to agree on a session ID, because agreeing on a unique session ID would amount to communication which itself is the objective of the protocol.

Theorem 1. *There exists no reliable protocol under parallel self composition (for even just two executions) that can tolerate* $\lceil \frac{\kappa}{2} \rceil$ *or more faults.*

Proof: On the contrary let us assume that there exists a protocol Π for reliable unicast that remains reliable under parallel self composition and tolerates $\lceil \frac{\kappa}{2} \rceil$ faults. Let the sender S and the receiver R be connected by a set $W = \{w_1, w_2 \ldots w_\kappa\}$ of κ wires out of which the adversary $\mathcal{A}$ can corrupt any set of $\lceil \frac{\kappa}{2} \rceil$ wires. Let Γ_1, Γ_2 be two concurrent executions of the protocol Π between S and R. Let m_1, m_2 be the inputs to the executions Γ_1, Γ_2. Without loss of generality we can assume that the protocol Π terminates within N rounds.

Let W_1 be the set of wires $\{w_1, w_2, \ldots, w_{\lceil \frac{\kappa}{2} \rceil}\}$. Similarly let W_2 be the set of wires $\{w_{\lceil \frac{\kappa}{2} \rceil+1}, w_{\lceil \frac{\kappa}{2} \rceil+2} \ldots w_\kappa\}$. Then consider the two scenarios depicted in the figure 2. The nodes S_1 and S_2 denote the sender side of the two executions Γ_1 and Γ_2, while the nodes R_1 and R_2 denote the receiver side.

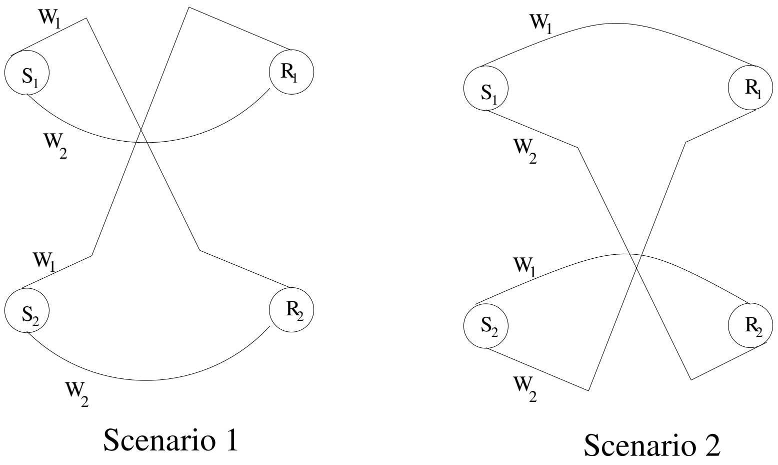

Fig. 2. The two indistinguishable Scenarios $\mathcal{S}_1$ and $\mathcal{S}_2$.

Scenario $\mathcal{S}_1$

The inputs to protocol executions Γ_1 and Γ_2 are m_1, m_2 respectively. The adversary corrupts the wires W_1 and does the following.

- Swap the messages of Γ_1 and Γ_2 along the wires W_1. More formally if $msg_i(A,B,w_k)$ denotes the message sent from A to B in round i along w_k then
  - The adversary sends $msg_i(S_1,R_1,w_k)$ along wire w_k to R_2 and the message $msg_i(S_2,R_2,w_k)$ to R_1.
  - The adversary sends $msg_i(R_1,S_1,w_k)$ along wire w_k to S_2 and the message $msg_i(R_2,S_2,w_k)$ to S_1.

Scenario $\mathcal{S}_2$

The inputs to protocol executions Γ_1 and Γ_2 are m_2, m_1 respectively. The adversary corrupts the wires W_2 and does the following.

- Swap the messages of Γ_1 and Γ_2 along the wires W_2. More formally let $msg_i(A,B,w_k)$ denotes the message sent from A to B in round i along w_k. Then for each $w_k \in W_2$,
  - The adversary sends $msg_i(S_1,R_1,w_k)$ along wire w_k to R_2 and the message $msg_i(S_2,R_2,w_k)$ to R_1.
  - The adversary sends $msg_i(R_1,S_1,w_k)$ along wire w_k to S_2 and the message $msg_i(R_2,S_2,w_k)$ to S_1.

We prove that the messages received by S and R are the same for both the scenarios. In order to prove this, we proceed by induction on the number of rounds. At the beginning of the protocol, trivially the view of the receiver is the same in both the scenarios. Let us say the view of the receiver is the same in both the scenarios is the same for the first $r-1$ rounds. In the r^{th} round

- In scenario $\mathcal{S}_1$, the node S_1 sends the message m_{11}, m_{12} to receiver R_1, and node S_2 sends the message m_{21}, m_{22} to node R_2. Because of adversarial strategy the receiver receives the messages m_{21}, m_{12} at node R_1 and messages m_{11}, m_{22} at node R_2.
- In scenario $\mathcal{S}_2$, the node S_1 sends the message m_{21}, m_{22} to receiver R_1, and node S_2 sends the message m_{11}, m_{12} to node R_2. The receiver receives the messages m_{21}, m_{12} at node R_1 and messages m_{11}, m_{22} at node R_2.

Thus the receiver R receives the same messages in both the scenarios. Similarly it can be seen that the messages received by S in round r in both the scenarios are the same. Hence by induction, the messages received by R is the same in both the scenarios. Thus it is impossible for the receiver to distinguish between the two scenarios. This implies that it is impossible for R to decide whether the output message for Γ_1 is m_1 or m_2. Hence no protocol for reliable unicast is parallel composable even two times when the number of faults $t \geq \lceil \frac{\kappa}{2} \rceil$. □

The above result imposes a serious limitation on the possibility of fault tolerant distributed protocols over an incomplete network. All protocols for fault tolerant distributed protocols on an incomplete network heavily rely on reliable message transmission protocol for communication between players that are not connected by a direct link. Therefore parallel composable fault tolerant distributed protocols over an incomplete network is impossible whenever the number of faults $t \geq \lceil \frac{\kappa}{2} \rceil$.

4.2 Sequential Composability

Theorem 2. *No r-round protocol for Reliable Message Transmission that tolerates $t \geq \lceil \frac{\kappa}{2} \rceil$ faults is sequentially composable more than $2r+1$ times.*

Proof: The impossibility result is derived by showing that for any deterministic protocol Π r-rounds of execution of the two scenarios shown in figure 2 above can be simulated by $2r+1$ sequential executions of Π.

Assume by contradiction that there exists a deterministic r-round protocol Π for Reliable message transmission that tolerates $t \geq \lceil \frac{\kappa}{2} \rceil$ and is sequentially composable $2r+1$ times. The adversary can simulate one round of execution in scenario $\mathcal{S}_1$ by two sequential executions of Π. For this purpose consider $2r$ sequential executions $\{\Pi_1, \Pi_2, \ldots \Pi_{2r}\}$of Π. For each $i, 1 \leq i \leq r$ the inputs to Π_{2i-1} ,Π_{2i} are m_1, m_2 respectively. Hence the *odd* executions Π_{2i-1} are used to simulate S_1, R_1 while the *even* executions Π_{2i} simulate S_2, R_2. We denote by $msg_j(A,B,w_i)$ the message sent by A to B along wire w_i in the j^{th} round of a protocol execution. The adversary follows the following strategy to simulate Scenario $\mathcal{S}_1$.

- *Execution* $2k-1$*:* The adversary $\mathcal{A}$ corrupts the set of wires W_1. For each $j < k$, the adversary $\mathcal{A}$ works as follows in the j^{th} round of Π_{2k-1}
  - On each wire $w_i \in W_1$, $\mathcal{A}$ communicates $msg_j(S_2, R_2, w_i)$ to R_1.
  - On each wire $w_i \in W_1$, $\mathcal{A}$ communicates $msg_j(R_2, S_2, w_i)$ to S_1

  Note that the for all $j < k$ the messages $msg_j(S_2, R_2, w_i), w_i \in W_1$ were obtained in the previous execution. In round k the adversary records the messages $msg_k(S_1, R_1, w_i)$ and $msg_k(R_1, S_1, w_i)$ for all $w_i \in W_1$.
- *Execution* $2k$*:* The adversary $\mathcal{A}$ corrupts the set of wires W_1. For each $j < k$, the adversary $\mathcal{A}$ works as follows in the j^{th} round of Π_{2k}
  - On each wire $w_i \in W_1$, $\mathcal{A}$ communicates $msg_j(S_1, R_1, w_i)$ to R_2.
  - On each wire $w_i \in W_1$, $\mathcal{A}$ communicates $msg_j(R_1, S_1, w_i)$ to S_2.

  Note that the for all $j < k$ the messages $msg_j(S_1, R_1, w_i), w_i \in W_1$ were obtained in the previous execution. In round k the adversary records the messages $msg_k(S_2, R_2, w_i)$ and $msg_k(R_2, S_2, w_i)$ for all $w_i \in W_1$.

At the end of the round 1 of the second execution Π_2 the views of S and R is the same as the view of S_2 and R_2 at the end of round 1 in scenario $\mathcal{S}_1$. Then in the fourth sequential execution Π_4 the messages received by S and R are in the first round is the same as those received by S_2 and R_2 in round 1 of Scenario $\mathcal{S}_1$. Since Π is a deterministic protocol, the messages sent by S and R in round two of Π_4 are same as the messages sent by S_2 and R_2 in round two of Scenario $\mathcal{S}_1$. Thus even after round 2 the messages received by S and R are consistent with their views in Scenario $\mathcal{S}_1$. Using the same argument, we have that for every i, the views of S and R in Π_{2i} for the first i-rounds are identical to the views of S_2 and R_2 for the first i-rounds of Scenario $\mathcal{S}_1$. Similarly the views of S and R in Π_{2i-1} for the first i-rounds are identical to the views of S_1 and R_1.

By symmetry, it is possible for the adversary to simulate Scenario $\mathcal{S}_2$ in $2r$ sequential executions of Π. The protocol Π terminates within r rounds. Therefore after the r-rounds in Π_{2r}, S and R must terminate with output message. But it is impossible since the view of S and R at the r^{th} round of Π_{2r} is the same as that of S_2 and R_2 after r rounds in Scenario $\mathcal{S}_1$. Hence there does not exist an r-round protocol for reliable message transmission that tolerates $t \geq \lceil \frac{\kappa}{2} \rceil$ faults and is sequentially composable more than $2r-1$ times. □

4.3 Randomized Composable Reliable Unicast for Free

We now present a single phase universally composable protocol that with a high probability reliably transmits ℓ field elements with a overall communication complexity of $O(\ell)$ field elements. We represent the block of field elements $\mathbf{M}$ that S wishes to send to R as $\mathbf{M} = [m_0\ m_1\ \ldots\ m_{\kappa(\kappa-t)}]$. In other words, a finite field $\mathbb{F}$ is so chosen that the message can be represented as a concatenation of $\kappa(\kappa-t)$ elements from $\mathbb{F}$. The protocol is given in the Figure 3 where κ is the number of wires (or more generally the number of vertex disjoint paths from the sender to the receiver) denoted as $\mathcal{W} = \{w_1, w_2, \ldots, w_\kappa\}$. Let ε be a bound on the probability that the protocol does not work correctly. We require that the size of the field $\mathbb{F}$ be $\Omega(\frac{Q(\kappa)}{\varepsilon})$, for some polynomial $Q(\kappa)$, but this is of course acceptable since the complexity of the protocol increases logarithmically with field size. We now discuss the correctness of the protocol. As usual, we focus only on the case wherein both the sender S and the receiver R are honest throughout the protocol. It is sufficient to prove that (a) If R does not output "FAILURE" then R always recovers the correct message $\mathbf{M}$, and (b) There exists a suitable choice of $\mathbb{F}$ such that protocol terminates with a non-"FAILURE" output with probability $(1-\varepsilon)$, for any $\varepsilon > 0$. We do exactly this in the following two lemmas.

The Single Phase δ-Reliable Universally Composable Unicast Protocol

1. S selects $(\kappa - t)$ polynomials $p_i, 0 < i \leq (\kappa - t)$ over $\mathbb{F}$, each of degree $\kappa - 1$ with the coefficients assigned as follows: the coefficient of x^j in polynomial p_i is assigned to be $m_{i\kappa+j}$. From these $(\kappa - t)$ polynomials, S "extrapolates" another t polynomials each of degree $\kappa - 1$ by having the coefficients of x^j in all these κ polynomials (p_i's) lie on a $\kappa - t - 1$ degree polynomial.
2. S sends the polynomial p_i through w_i.
3. S chooses another κ^2 field elements at random, say r_{ij}, $0 < i, j \leq \kappa$.
4. S sends along wire w_i, the n ordered pairs $(r_{ij}, p_j(r_{ij}))$, for all j. Let $v_{ij} = p_j(r_{ij})$.
5. Let p_i' and (r_{ij}', v_{ij}') be the values received by R. Among all the wires in $\mathcal{W}$, we say that wire w_i *contradicts* wire w_j if: $v_{ij}' \neq p_j'(r_{ij}')$.
6. Among all the wires in $\mathcal{W}$, R checks if there is a wire contradicted by at least $t+1$ wires. All such wires are removed.
7. If there is at least one contradiction among the remaining wires, R outputs "FAILURE" and halts.
8. If there is no contradiction left, R checks whether the coefficients of x^j for the remaining polynomials (that is, after excluding those that are eliminated in the step 6) lie on a $(\kappa - t - 1)$ degree polynomial. If not, R outputs "FAILURE" and halts. If yes, R corrects the polynomials $p_i(x)$ of each corrupted wire w_i (i.e, he "corrects" those wires) using the polynomials received along the uncorrupted wires. R now knows all the polynomials $p_i(x)$.
9. R recovers **M** from the coefficients of the first $(\kappa - t)$ polynomials.

Fig. 3. An Optimal Single Phase δ-Reliable Universally Composable Unicast Protocol.

Lemma 1. *If R does not output "FAILURE" then R always recovers the correct message* **M**.

Proof. Consider the runtime environment E, modeled as a Turing machine that freely interacts with the adversary, and initiates several protocols (with possibly different inputs), including many concurrent copies of the present protocol. We now show that irrespective of what E does, it cannot stop R from recovering the message **M** provided R does not output failure.

Note that any corruption among the wires involves changing the polynomial corresponding to that wire. Consequently, no corrupted wire can escape undetected since their coefficients lie on a $(\kappa - t - 1)$ degree polynomial. This is because if no more than t wires accused a wire among the κ wires, then at least $\kappa - t$ of them are honest wires, each of them carrying a point on the polynomial which implies that the polynomial is unchanged. Therefore, at the start of "if yes" part of step 8, all the wires which are used in calculation of the output could not have corrupted their polynomials. This guarantees that R's output in step 9 is correct, irrespective of how that environment E has helped the adversary modify the data sent along the corrupted wires. However, we stress that this holds only as long as *fresh* randomness is used in each and every initiation of the protocol[1]. Since S is assumed to be honest, this is indeed the case. □

Lemma 2. *The protocol terminates with an output that is not a "FAILURE" with high probability.*

Proof. No matter what the runtime environment is, we know that no uncorrupted wire changes the value sent on that wire. Therefore, it follows that no honest wire can contradict another honest wire. Thus, if wire i contradicts wire j, then either wire i or wire j is faulty. From this it is easy to see that an honest wire can be contradicted by at most t other wires, and therefore any wire that is contradicted $t+1$ or more wires has to be faulty. Hence R can be sure that all the wires removed by him are indeed faulty.

We need to show that if a wire is corrupted, then it will be contradicted by all the honest players with high probability. Let π_{ij} be the probability that the corrupted wire j will not be contradicted by an honest wire i. This means that the adversary can ensure that $p_j(r_{ij}) = p_j'(r_{ij})$ with a probability of π_{ij}. Since there are only $\kappa - 1$ points at which these two polynomials intersect, this allows the adversary to guess the value of r_{ij} with a probability of at least $\frac{\pi_{ij}}{\kappa-1}$. But since r_{ij} was selected uniformly in $\mathbb{F}$, and the random coins are independent of those used by the other protocols initiated by the runtime environment, the probability of guessing it is at most $\frac{1}{|\mathbb{F}|}$. Therefore we have $\pi_{ij} \leq \frac{\kappa-1}{|\mathbb{F}|}$ for each i, j. Thus the total probability that the adversary can find i, j such that corrupted wire j will not be contradicted by i is at most $\sum_{i,j} \pi_{ij} \leq \frac{\kappa^2(\kappa-1)}{|\mathbb{F}|}$.

Since $\mathbb{F}$ is chosen such that $|\mathbb{F}| \geq \frac{Q(\kappa)}{\varepsilon}$, it follows that the protocol outputs a non-"FAILURE" value with probability $\geq 1 - \varepsilon$ if we set $Q(\kappa) = \kappa^3$. □

[1] It can be easily shown that if randomness is re-used among several invocations of the protocol, then the environment may help the adversary to adroitly modify the messages along corrupted wires such that R is duped into accepting an erroneous **M** with a non-negligible probability.

A Discussion on Our Composable Protocol: The single phase δ-reliable universally composable communication protocol of Figure 3 is a *special* kind of a δ-reliable communication protocol in that the receiver R actually *knows* if the protocol outputs the correct message or not. Note that according to our definition of δ-reliable communication, it is acceptable even if R is unable to "detect" every occurrence of an error. Thus, fortunately, our protocol has a strictly stronger property than what is needed — nevertheless, since the protocol is optimal, one need not search for a more efficient definition-adhering protocol. Of what use is such a special property? We state the three main advantages of such a "detection" property below:

1. Even the most ardent opponent of randomized (i.e. non-zero error-prone) algorithms will not hesitate to deploy an algorithm that detects *every* erratic execution.
2. Due to the error-detection capabilities of our protocol, it is easy to convert our protocol's error-probability into its termination-probability. Specifically, whenever R detects that the execution is erroneous, he may inform S about it and re-run the protocol. As a result, we end up having designed a protocol for *perfectly* reliable communication (that is, $\delta = 0$) with the property that it terminates with a very high probability in a *single* phase![2]
3. Lastly, the special property concerned is for *free* — recall that ℓ field elements cannot be sent by communicating less than $O(\ell)$ field elements.[3]

5 Composability of Reliable Broadcast

We first formally define *ABA* where the expected consistency is that all honest players (who are outside the adversary's control) must agree. Note that the problem of reliable broadcast can be reduced to the problem of ABA (and vice-versa) in the case of honest majority.

Definition 1 (*ABA*). *Each player starts with an input from a fixed set $V = \{0,1\}$. The goal is for the players to eventually output decisions from the set V upholding the following conditions, even in the presence of a t-adversary:*

- Agreement: *All non-faulty players decide on the same value $u \in V$.*
- Validity: *If all non-faulty players start with the same initial value $v \in V$, then $u = v$.*
- Termination: *All non-faulty players eventually decide.*

5.1 Impossibility of Parallel Composition

In this section we show the impossibility of composition of protocol for *ABA* over a general graph. First, we prove that given $n > 3t$, there does not exists any *ABA* protocol over general graph that composes in parallel and remains secure when connectivity is at most $2t$[4]. This result when combined with the result of Lindell *et al.* [22], shows that the advantage gained by use of authentication for Byzantine Agreement over any graph is lost when the protocols for *ABA* is composed in parallel.

Intuition: We first provide some intuition as to why added power of authentication in Byzantine Agreement is rendered useless during composition. In the stand alone model for *ABA*, players use authentication and send messages to each other. Since the authentication scheme is assumed to be unforgeable, adversary $\mathcal{A}$ cannot alter the message sent by an honest player but can atmost act in a fail-stop fashion. For *ABA* over complete graph, PSL [23] showed that the fault tolerance can be as high as $n > t$. However during parallel composition, $\mathcal{A}$ can use messages from one execution in another execution, thus $\mathcal{A}$ can exhibit byzantine behavior even w.r.t value of honest players. For such a case Lindell et al. [22] showed that parallel composition protocols of *ABA* over complete graphs is possible if and only if $t < \frac{n}{3}$. It is not difficult to see that *ABA* over general graph G in a stand-alone model is possible if and only if $n > t$ and G is $t+1$ connected. The reason behind $t+1$ connectivity is evident from the fact that given adversary can corrupt upto t nodes, two honest players using authentication can communicate reliably if they have $t+1$ node disjoint paths between them. But this still will leave atleast one path via which honest players can communicate. However when composed in parallel, situation changes. In parallel composition adversary can use values from different executions, thus can send different messages on behalf of an honest player. In such a case $\mathcal{A}$ can ensure that an honest player gets different messages for another honest player, yet both the messages are properly authenticated. This is similar to setting of Byzantine agreement over incomplete graphs. Dolev [8] proved that Byzantine agreement over incomplete graph G

[2] Recall that for the kind of fault-tolerance being discussed, no single phase perfectly reliable unicast (that is, $\delta = 0$) protocol can exist [9]! Thus, the existence of an expected-single phase perfectly reliable unicast protocol is quite a pleasant surprise! Moreover, such a protocol also turning out to be bit-optimal — is wonderful!

[3] We have assumed throughout the paper that the message space is *incompressible*.

[4] We are considering only stateless compositions.

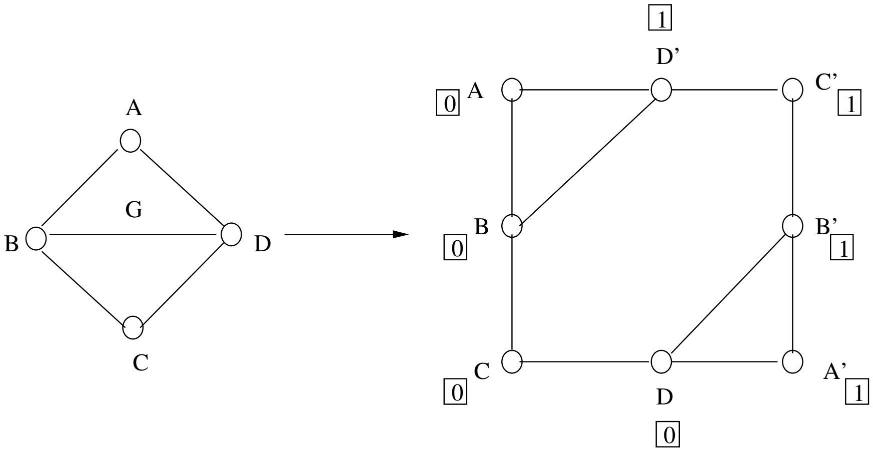

Fig. 4. System G and S.

in a stand-alone model is possible iff $n > 3t$ and G is $2t+1$ connected. We now show that same holds when protocol for ABA over incomplete graph G is composed parallely.

Basic Outline: We assume there exists a protocol π that solves ABA for four players A,B,C,D over two connected graph where atmost one player may be byzantine faulty. We further assume that π remains secure even when composed in parallel twice. Let the original system of four players be G. We construct a new system S as shown in figure 4 . Each player in S runs π. We now formally define S and further show that there exists a contradiction in π.

Construction of S: Take two copies of each player in G and construct an octagonal system S that interwines two independent copies of π as shown in Figure 4. Player A is connected to B,D'; B is connected to A,C,D'; C is connected to B,D and so on. A node a behaving in a byzantine fashion with a pair of honest nodes, is captured by connecting one of the honest nodes to a and other to a'. a and a' are independent copies of a with same authentication keys. Also note that each player in S knows only its immediate neighbors and not the complete graph. Also, in reality a player may be connected to either of a or a', but it cannot differentiate between the two. It knows its neighbor only by its local name which may be a. Here we neither know what system S is supposed to do nor what π solves and therefore the definitions of ABA does not tell us anything directly about what the players' output should be. All we know is that S is a synchronous system and π has a well defined behavior.

Let α_0 be an execution of π in G where A,B and C are honest players and start with input 0. Adversary $\mathcal{A}$ corrupts D in byzantine fashion. Let α_1 is an execution of π in G in which A,B and C are honest players and start with input 1. $\mathcal{A}$ corrupts D in byzantine fashion. Let α_2 is an execution of π in G in which A,C and D are honest players and start with input values 1,0 and 0 respectively. $\mathcal{A}$ corrupts B in byzantine fashion. Let α is an execution of π in S in which each player starts with input value as shown in Figure 4. Notice that all the players in S are honest and follow the prescribed protocol correctly.
We now describe the adversary strategy. Let $msg_i(a,b)$ denote the message send by player a to b in ith round of execution of protocol π.

1. *Send outgoing messages of round i*: $\mathcal{A}$ obtains the messages $msg_i(D,A)$,$msg_i(D,B)$ and $msg_i(D,C)$ from D in α_0. (these are round i messages that D would have sent to players A,B and C if D would have been honest).Similarly, $\mathcal{A}$ obtains the messages $msg_i(D',A')$, $msg_i(D',B')$ and $msg_i(D',C')$ from D' in α_1 (these are round i messages that D' would have sent to players A',B' and C' if D' would have been honest).
   - In α_0, $\mathcal{A}$ sends the messages $msg_i(D',A')$, $msg_i(D',B')$ and $msg_i(D,C)$ to A,B and C respectively. Thus, the directed edges (D,A) and (D,B) are replaced by directed edges (D',A) and (D',B) respectively.
   - In α_1, $\mathcal{A}$ sends $msg_i(D,A)$,$msg_i(D,B)$ and $msg_i(D',C')$ to A',B' and C' respectively. Thus, the directed edges (D',A') and (D',B') are replaced by directed edges (D,A') and (D,B') respectively.
2. *Receive incoming messages from round i*: In α_0, $\mathcal{A}$ obtains the messages $msg_i(A,D)$,$msg_i(B,D)$ and $msg_i(C,D)$ via player D. Similarly, in α_1, $\mathcal{A}$ gets $msg_i(A',D')$, $msg_i(B',D')$ and $msg_i(C',D')$ via player D'.
   - $\mathcal{A}$ passes the messages $msg_i(A',D')$,$msg_i(B',D')$ and $msg_i(C,D)$ to D in α_0. Thus, the directed edges (A,D) and (B,D) are replaced by directed edges (A',D) and (B',D) respectively.

- $\mathcal{A}$ passes the messages $msg_i(A,D)$,$msg_i(B,D)$ and $msg_i(C',D')$ to D' in α_0.Thus, the directed edges (A',D') and (B',D') are replaced by directed edges (A,D') and (B,D') respectively.

We now show that no protocol can achieve *ABA* for graph *G* under composition of protocol. Evidently, the above statement completes the contradiction to the assumption that consensus is possible under composition. This, therefore, completes the proof. We show the following two lemmas as a prelude to proving above statement.

Lemma 3. *Players A, B and C output* 0 *in the system S.*

Proof. First we show that whatever messages D and D' send to A, B and C in α, $\mathcal{A}$ can send the same to A, B and C in α_0. In the execution α all the players follow π except from that the player D is connected to A' and B' in place of A and B and D' is connected to A and B in place of A' and B'. However, from the adversary strategy it is evident that this change in connectivity does not make any difference to the views of A,B and C i.e. A,B and C receive same messages in α and α_0. Thus to A,B and C, α and α_0 are indistinguishable. $\alpha \overset{A}{\sim} \alpha_0$, $\alpha \overset{B}{\sim} \alpha_0$, and $\alpha \overset{C}{\sim} \alpha_0$. From the validity condition of *ABA* we can say that A,B and C will eventually output 0 in α_0. Since $\alpha \overset{A}{\sim} \alpha_0$, $\alpha \overset{B}{\sim} \alpha_0$, and $\alpha \overset{C}{\sim} \alpha_0$, A,B and C in α will output 0.[5] □

Lemma 4. *Players A', B' and C' output* 1 *in the system S.*

Proof. The faulty party here is D' and the proof works in the same way as above. Players A',B' and C' started with initial values 1 and hence they output 1 in the system S. □

Theorem 3. *ABA when composed in parallel over graph G cannot tolerate even one byzantine fault.*

Proof. Suppose there exists a protocol π which achieves *ABA* under composition when the executions α_0, α_1 and α_2 are run in parallel. Let α_2 be an execution with players C, D and A' being honest and B as corrupt. On similar lines as proof of Lemma 3, we can show that C and A' will output the same value in system S. But, C and A' have already decided on 0 and 1 respectively(Lemma 3 and Lemma 4). This contradicts the agreement condition for *ABA*. Thus, there does not exists such a π. □

Theorem 4. *ABA tolerating t-adversary cannot be composed in parallel over a general network $\mathcal{N}$ if $n \leq 3t$ or $\mathcal{N}$ is not $2t+1$ connected*

Proof. The necessity and sufficient condition of $n \leq 3t$ is shown by Lindell *et al.* [22]. For the connectivity part, suppose there exists a protocol π' that solves *ABA* in network $\mathcal{N}$ tolerating upto t faulty players, with $n \geq 3t$ and $\mathcal{N}$ is $2t$ connected. Let π solves *ABA* in graph G(see Figure 4) for nodes A,B,C and D. Partition players of π' into four nonempty sets I_A, I_B, I_C and I_D such that sets I_B and I_D contains at most t players each . Each player A,B ,C and D in G keeps track of all the players in sets I_A, I_B, I_C, I_D respectively. Each player i in G assigns its own initial value to every member of I_i and simulates the steps of all the players in I_i as well as the messages between pairs of players in I_i . Messages from players in I_i to players in another set I_j are sent from player i to player j in G. If any player in I_i decides on a value v, then i decides on the value v. If there are more of such values, then i can choose any one from those values. Note that if we remove all the nodes in I_B and I_D, then the nodes in I_A and I_C get disconnected. The edges in graph G can now be considered as the bundle of edges between the groups I_A, I_B, I_C and I_D in $\mathcal{N}$. Put all the faulty processes in $\mathcal{N}$ to either I_B or I_D. Since B or D in G simulate all the faulty players in $\mathcal{N}$, there are at most t faulty players in $\mathcal{N}$.

Fix any particular execution β' of π' with at most one faulty process and let β be the simulated execution of π Since π' is assumed to solve *ABA* for n players with at most t faults under composition of protocols, the usual agreement, validity and termination conditions for *ABA* hold in β'. We argue that these conditions carry over to β.
For termination, let i be a nonfaulty player in G. Then i simulates at least one process, j, of $\mathcal{N}$, and j must be nonfaulty since i is. The termination condition for β' implies that j must eventually decide and hence i decides. For validity, if all non faulty players of π begin with a value v, then all non faulty players in π' also begin with v. Validity for β' implies that v is the only decision for a nonfaulty player in β'. Then v is the only decision value for a nonfaulty player in β. For agreement, suppose i and j are nonfaulty players of π. Then they simulate only nonfaulty players of π'. Agreement for β' implies that all of these simulated players agree, so i and j also agree.
We conclude that π solves *ABA* for four players under composition tolerating one fault. But this contradicts Theorem 3. Thus, there does not exists any such protocol π'. □

[5] We are able to make claims regarding player's outputs in α as views of players are same as those in α_0. Thus by analyzing outputs in α_0, we can determine outputs in *S*.

6 Conclusion

Our first result shows that any composable protocol for reliable unicast tolerating bounded adversaries requires that $\kappa > 2t$. As the bounds for tolerating an unbounded adversary is also the same, we conclude that a weaker adversary does not result in improved fault tolerance for the problem. This is in line with the impossibility of Authenticated Byzantine agreement when $n < 3t+1$ shown in [22] which makes digital signatures useless in the case of Byzantine agreement. Our second result establishes something in the other direction. We have designed a constant overhead δ-reliable unicast protocol for a sufficiently large field size (we require $|\mathbb{F}| = \Omega(\frac{\kappa^3}{\delta})$). The optimality of the communication complexity encourages us to design protocols tolerating unbounded adversaries instead of weaker bounded adversaries. Summarizing, bounding the powers of adversary does not improve either fault tolerance or the communication complexity for the problem of reliable communication. Since reliable communication is a primitive used by almost all multiparty protocols these results are relevant for general distributed computation as well.

References

1. Bernd Altmann, Matthias Fitzi, and Ueli M. Maurer. Byzantine agreement secure against general adversaries in the dual failure model. In *Proceedings of the 13th International Symposium on Distributed Computing*, pages 123–137, London, UK, 1999. Springer-Verlag.
2. M. Backes, C. Cachin, and R. Strobl. Proactive Secure Message Transmission in Asynchronous Networks. In *Proceedings of the twenty-second ACM Annual Symposium on Principles of Distributed Computing (PODC)*, pages 223–232, New York, NY, USA, 2003. ACM Press.
3. Malte Borcherding. On the number of authenticated rounds in byzantine agreement. In *WDAG '95: Proceedings of the 9th International Workshop on Distributed Algorithms*, pages 230–241, London, UK, 1995. Springer-Verlag.
4. Malte Borcherding. Levels of authentication in distributed agreement. In *WDAG '96: Proceedings of the 10th International Workshop on Distributed Algorithms*, pages 40–55, London, UK, 1996. Springer-Verlag.
5. Malte Borcherding. Partially authenticated algorithms for byzantine agreement. In *ISCA: Proceedings of the 9th International Conference on Parallel and Distributed Computing Systems*, pages 8–11, 1996.
6. R. Canetti. Universally Composable Security: A New Paradigm for Cryptographic Protocols. In *Proceedings of the 42nd Symposium on Foundations of Computer Science (FOCS)*, pages 136–145. IEEE, 2001. Full version available at `http://eprint.iacr.org/2000/067`.
7. Y. Desmedt and Y. Wang. Perfectly Secure Message Transmission Revisited. In *Proceedings of Advances in Cryptology EUROCRYPT '02*, volume 2332 of *Lecture Notes in Computer Science (LNCS)*, pages 502–517. Springer-Verlag, 2002.
8. D. Dolev. The Byzantine Generals Strike Again. *Journal of Algorithms*, 3(1):14–30, March 1982.
9. D. Dolev, C. Dwork, O. Waarts, and M. Yung. Perfectly Secure Message Transmission. *Journal of the Association for Computing Machinery (JACM)*, 40(1):17–47, January 1993.
10. D. Dolev and H. R. Strong. Authenticated algorithms for byzantine agreement. *SIAM Journal on Computing*, 12(4):656–666, 1983.
11. Danny Dolev. The byzantine generals strike again. Technical report, Stanford, CA, USA, 1981.
12. Danny Dolev, Cynthia Dwork, and Larry Stockmeyer. On the minimal synchronism needed for distributed consensus. *J. ACM*, 34(1):77–97, 1987.
13. Michael J. Fischer, Nancy A. Lynch, and Michael S. Paterson. Impossibility of distributed consensus with one faulty process. *J. ACM*, 32(2):374–382, 1985.
14. Matthias Fitzi and Ueli M. Maurer. Efficient byzantine agreement secure against general adversaries. In *International Symposium on Distributed Computing*, pages 134–148, 1998.
15. Mattias Fitzi and Ueli Maurer. From partial consistency to global broadcast. In *STOC '00: Proceedings of the thirty-second annual ACM symposium on Theory of computing*, pages 494–503, New York, NY, USA, 2000. ACM.
16. M. Franklin and R.N. Wright. Secure Communication in Minimal Connectivity Models. *Journal of Cryptology*, 13(1):9–30, 2000.
17. M. Franklin and M. Yung. Secure Hypergraphs: Privacy from Partial Broadcast. In *Proceedings of 27th Symposium on Theory of Computing (STOC)*, pages 36–44. ACM Press, 1995.
18. J. A. Garay. Reaching (and Maintaining) Agreement in the Presence of Mobile Faults. In *Proceedings of the 8th International Workshop on Distributed Algorithms – WDAG '94*, volume 857 of *Lecture Notes in Computer Science (LNCS)*, pages 253–264, 1994.
19. L. Gong, P. Lincoln, and J. Rushby. Byzantine agreement with authentication: Observations and applications in tolerating hybrid and link faults, 1995.
20. Jonathan Katz and Chiu-Yuen Koo. On expected constant-round protocols for byzantine agreement. 2007.
21. M.V.N.A. Kumar, P. R. Goundan, K. Srinathan, and C. Pandu Rangan. On perfectly secure communication over arbitrary networks. In *Proceedings of the 21st Symposium on Principles of Distributed Computing (PODC)*, pages 193–202, Monterey, California, USA, July 2002. ACM Press.
22. Y. Lindell, A. Lysysanskaya, and T. Rabin. On the Composition of Authenticated Byzantine Agreement. In *Proceedings of the 34th Symposium on Theory of Computing (STOC)*, pages 514–523. ACM Press, 2002.

23. M. Pease, R. Shostak, and L. Lamport. Reaching agreement in the presence of faults. *J. ACM*, 27(2):228–234, 1980.
24. M. O. Rabin. Randomized byzantine generals. In *Proc. of the 24th Annu. IEEE Symp. on Foundations of Computer Science*, pages 403–409, 1983.
25. H. Sayeed and H. Abu-Amara. Perfectly Secure Message Transmission in Asynchronous Networks. In *Seventh IEEE Symposium on Parallel and Distributed Processing*, 1995.
26. Ulrich Schmid and Bettina Weiss. Synchronous byzantine agreement under hybrid process and link failures. Research Report 1/2004, Technische Universität Wien, Institut für Technische Informatik, Treitlstr. 1-3/182-1, 1040 Vienna, Austria, 2004.
27. T. K. Srikanth and S. Toueg. Simulating authenticated broadcasts to derive simple fault-tolerant algorithms. *Distributed Computing*, 2(2):80–94, 1987.
28. Kannan Srinathan, Prasad Raghavendra, and C. Pandu Rangan. On proactive perfectly secure message transmission. In *2th Australisian Conference on Information Security and Privacy (ACISP 07), Australia,*, July 2007.
29. Kannan Srinathan and C. Pandu Rangan. Possibility and complexity of probabilistic reliable communications in directed networks. In *Proceedings of 25th ACM Symposium on Principles of Distributed Computing (PODC'06)*, 2006.
30. Y. Wang and Y. Desmedt. Secure Communication in Multicast Channels: The Answer to Franklin and Wright's Question. *Journal of Cryptology*, 14(2):121–135, 2001.