The Broadcast Message Complexity of Secure Multiparty Computation

Sanjam Garg[1], Aarushi Goel[2], and Abhishek Jain[2]

[1] University of California, Berkeley
sanjamg@berkeley.edu
[2] Johns Hopkins University
{aarushig,abhishek}@cs.jhu.edu

Abstract. We study the *broadcast message complexity* of secure multiparty computation (MPC), namely, the total number of messages that are required for securely computing any functionality in the broadcast model of communication.

MPC protocols are traditionally designed in the simultaneous broadcast model, where each round consists of *every* party broadcasting a message to the other parties. We show that this method of communication is sub-optimal; specifically, by eliminating simultaneity, it is, in fact, possible to reduce the broadcast message complexity of MPC.

More specifically, we establish tight lower and upper bounds on the broadcast message complexity of n-party MPC for every $t < n$ corruption threshold, both in the plain model as well as common setup models. For example, our results show that the optimal broadcast message complexity of semi-honest MPC can be much lower than $2n$, but necessarily requires at least three rounds of communication. We also extend our results to the malicious setting in setup models.

1 Introduction

The ability to securely compute on private datasets of individuals has wide applications of tremendous benefits to society. Secure multiparty computation (MPC) [25, 19, 2, 8] provides a solution to the problem of computing on private data by allowing a group of mutually distrusting parties to jointly evaluate any function over their private inputs in a manner that reveals nothing beyond the output of the function.

Broadcast Message Complexity. Traditionally, the most popular communication model for the design of MPC protocols is the *broadcast* model, where parties communicate with each other by sending messages over an authenticated broadcast channel. Indeed, starting from [19], most computationally secure protocols in the literature are designed in the broadcast model.

Viewing a broadcast channel as a resource, in this work, we initiate the study of the *broadcast message complexity* of MPC, namely, the number of messages that are required for securely computing any functionality in the broadcast model. Specifically, we ask the following basic question:

What is the broadcast message complexity of n-party MPC w.r.t. $t < n$ corruptions (for every t)?

At first, it may seem that the above question can be easily resolved by appealing to the known bounds on the round complexity of MPC. For example, in the case of semi-honest corruptions (for any $t \geqslant 1$), *two* rounds are known to be necessary [21]. Then, it may seem that the broadcast message complexity in this case must be at least $2n$ since each of the n parties must presumably send a message in each of the two rounds. In this work, we show that the above intuition is *incorrect*.

Simultaneity is wasteful. MPC protocols are traditionally designed in the simultaneous broadcast model, where in each round, *every* party sends a message. We show that this model is wasteful, and that by eliminating simultaneity, it is possible to reduce the number of required messages.

Specifically, we consider the general setting where in each round, any subset of parties may send a message. In this setting, we show that the broadcast message complexity of MPC can be much lower than in the simultaneous broadcast model.

1.1 Our Results

We study the broadcast message complexity of MPC in the plain model, as well as common setup models, including the public-key infrastructure (PKI) model and the common reference string (CRS) model. We provide a tight characterization of broadcast message complexity as well as the number of rounds necessary for achieving optimal number of broadcasts. In particular, our results show that two rounds are insufficient for achieving optimal broadcasts; instead, at least three rounds are necessary. We elaborate on our results below.

I. Broadcast Message Complexity. We first investigate the broadcast message complexity of semi-honest MPC in the plain model. We provide a tight characterization that varies with the number of corrupted parties t and the number of output parties $|\mathcal{O}|$, where $\mathcal{O}$ denotes the set of parties who can learn the output.

Theorem 1 (Informal). *For any $t < n-1$ semi-honest, static corruptions and $|\mathcal{O}| > 1$ (resp., $|\mathcal{O} = 1|$) output parties, $n + t + 1$ (resp., $n + t$) broadcasts are necessary and sufficient for MPC in the plain model. For $t = n - 1$ corruptions, the broadcast message complexity is $n + t$ (resp., $n + t - 1$) when $|\mathcal{O}| > 1$ (resp., $|\mathcal{O} = 1|$).*

A few remarks about the above theorem are in order: (1) Our lower bound also holds in the CRS model. (2) Our positive result is based on any two-round semi-honest oblivious transfer (OT), which is the *optimal* assumption for $t \geqslant n/2$. In the CRS model, it can be extended to achieve UC security against malicious corruptions based on two-round malicious-secure OT.

We next show that the broadcast message complexity of MPC is lower in the PKI model, where the parties first post their respective public keys on a bulletin board.

Theorem 2 (Informal). *For any $t \leqslant n-1$ semi-honest, static corruptions and $|\mathcal{O}| > n-t$ (resp., $|\mathcal{O}| \leqslant n-t$) output parties, $n+t$ (resp., $n+t-1$) broadcasts are necessary and sufficient for MPC in the bare PKI model.*

We, in fact, provide two different positive results in this model:

- Our first construction works in the honest majority setting (i.e., $t < n/2$) and achieves security even against *malicious* adversaries. It does not require any additional assumptions beyond public-key encryption, and is therefore *optimal* in that sense. Interestingly, we achieve this result by drawing a connection to the security notion of guaranteed output delivery.
- Our second construction works for any $t < n$ corruptions and achieves security against semi-honest adversaries based on any two-round semi-honest OT. In the CRS model, this construction can be extended using standard techniques to achieve UC security against malicious adversaries.

II. Round Complexity. While two rounds are known to be necessary and sufficient for semi-honest MPC, we show that they are insufficient for achieving optimal broadcast message complexity. In particular, we show that *three* rounds are necessary and sufficient. This result holds both in the plain model as well as the PKI and CRS models.

Theorem 3. *Three rounds are necessary and sufficient for semi-honest MPC with optimal broadcast message complexity.*

We, in fact, prove two strengthenings of the above theorem:

- We show that three rounds are also sufficient for achieving security against malicious adversaries, either in the PKI model for $t < n/2$ corruptions or in the CRS model for $t < n$ corruptions.
- We show that in the plain model, any three round protocol that achieves optimal broadcast message complexity must necessarily utilize a *unique communication pattern*, where a communication pattern specifies which parties speak in which round(s). We also prove an analogous result in the PKI model.

The table below provides a summary of our lower bounds for broadcast message complexity and round complexity.

Model	Corruptions	Rounds	Output Parties	Broadcasts
Plain	$1 \leqslant t < n-1$	3	$\|\mathcal{O}\| > 1$	$n+t+1$
			$\|\mathcal{O}\| = 1$	$n+t$
Plain	$t = n-1$	3	$\|\mathcal{O}\| > 1$	$2n-1$
			$\|\mathcal{O}\| = 1$	$2n-2$
PKI	$1 \leqslant t \leqslant n-1$	3	$\|\mathcal{O}\| > n-t$	$n+t$
			$\|\mathcal{O}\| \leqslant n-t$	$n+t-1$

III. Application to P2P Model. While we focus on broadcast message complexity in this work, our positive results can also be used to obtain MPC

protocols in the point-to-point (P2P) communication model with *optimal* P2P message complexity for *any* corruption threshold.

The P2P message complexity of (computationally secure) MPC was recently studied by Ishai et al. [22] who established lower and upper bounds for $t = n-1$. Subsequently, [23] extended their lower bound to any $t < n$, but left open the problem of obtaining a matching upper bound for general t. We show that our construction from Theorem 2 for general $t < n$ can be used to resolve this open problem.

Theorem 4 (Informal). *Assuming the existence of two-round semi-honest OT, for any $t < n$ semi-honest, static corruptions and $|\mathcal{O}|$ output parties, $(n + t + |\mathcal{O}| - 2)$ messages are sufficient for MPC in the P2P communication model.*

1.2 Technical Overview

Starting Ideas. Recall that two rounds are known to be necessary for MPC, even for achieving security against semi-honest adversaries [21]. Further, in all known two-round MPC protocols in the broadcast model [16, 24, 14, 5, 17, 6, 18, 3], each party broadcasts a message in each round, resulting in a total of $2n$ messages.

At first, it may seem that this is inherent. Indeed, consider the scenario where one of the n parties, say P_i, does not send any message in the first round, and instead only sends a message in the second round. Can we construct a secure protocol in the plain model with this communication pattern? The answer is no, and to see this, recall that the security guarantee of semi-honest MPC stipulates that an adversary can only learn a *single* function output corresponding to a fixed set of inputs. To achieve this guarantee, a protocol transcript must somehow "fix" an input of each party. In the above scenario, since P_i only sends a message in the second round, its input cannot be fixed.[3] Therefore, an adversarial P_i can launch the following *residual function attack*: it first completes an execution of the protocol to obtain a transcript, and then replaces its message in the transcript with a freshly computed message w.r.t. a different input. It now computes its output function to learn a new output, w.r.t. the same set of inputs of the other parties, thereby violating MPC security.

The above still leaves open the possibility of designing a protocol where P_i only sends a message in the first round. In this case, P_i's input can indeed be fixed by the honest party messages sent in the second round. Unfortunately, it turns out that this is still not sufficient, even against the minimal corruption threshold of $t = 1$. The reason is that an adversary can simply "spoof" all the other parties P_j (where $j \neq i$). That is, after obtaining a protocol transcript, the adversary can simply replace the messages of all the other parties P_j with freshly computed messages w.r.t. different inputs, while keeping P_i's message

[3] Recall that the messages sent by honest parties in any round are independent of each other.

intact. Now, the adversary is able to learn multiple outputs w.r.t. a fixed input of P_i, thereby violating MPC security.

Towards a Template. Our main insight is that *by increasing the round complexity, we can decrease the broadcast message complexity.* To explain the basic idea, let us consider a toy scenario with $n = 5$ and $t = 2$. Our goal here is to obtain a template that requires $n+t+1 = 8$ messages, as per Theorem 1. Clearly, to achieve this broadcast message complexity, two parties must send only one message each. Now, consider the following communication pattern:

R1 In the first round, P_4 and P_5 send a message.
R2 In the second round, P_1, P_2 and P_3 send a message.
R3 In the third round, P_1, P_2 and P_3 send a message.

Since $t = 2$, at least one of P_1, P_2, P_3 must be honest. Let P_i be that party. Then, P_i's message in the third round must fix the inputs of *all* the other parties. Therefore, it would seem that this template should already work. Unfortunately, this is not the case as the spoofing attack we discussed earlier is also applicable here. Indeed, an adversary can simply spoof P_1, P_2, P_3 and launch a residual function attack on the inputs of P_4 or P_5 in a similar manner as above.

Upon closer inspection, we find that the reason for the spoofing attack in the above template is that while P_i can fix the input of all the parties, P_i's *input itself is not fixed by any other party.* Indeed, this is why the spoofing attack includes spoofing of P_i as well.

To address this issue, we modify the above template by exchanging rounds one and two. That is, we consider the following template:

R1 In the first round, P_1, P_2 and P_3 send a message.
R2 In the second round, P_4 and P_5 send a message.
R3 In the third round, P_1, P_2 and P_3 send a message.

In the technical sections, we show that the above template indeed works. The key point is that now, not only can P_i fix the inputs of all the other adversarial and honest parties, but crucially, the other honest parties, *who send messages in round two*, can also fix P_i's input (which must be used to compute its first round message). This raises several questions:

- Does the above idea generalize to any n and t?
- How can we prove a lower bound on the broadcast message complexity?
- Does the lower bound stay intact in the public-key model, or does it change?
- Are three rounds really necessary?
- Is the above communication pattern necessary, or can we also achieve security with other communication patterns?
- How can we construct protocols with optimal broadcast message complexity?

In the technical sections, we show that the above idea indeed generalizes to any n and t; the specific details vary depending upon the number of output parties and the number of corruptions, as well as whether we are working in the plain model or the public-key model.

We now proceed to address the remaining questions.

Lower Bounds on Broadcast Message Complexity. Our proof for lower bound on broadcast message complexity in Section 3 is simple and relies on establishing the minimum number of parties who must send *two* messages in the protocol.

We start by proving that in the plain model, for $t < n - 1$ and $|\mathcal{O}| > 1$ output parties, the number of parties that broadcast more than one message in the protocol must be strictly greater than the number of corrupted parties. Let us consider a toy example with $n = 5$ and $t = 2$, where all parties receive the output. Let us assume for contradiction that the number of parties that broadcast more than one message is equal to the number of corrupted parties. For example, suppose that P_1, P_2, P_3 broadcast a single message, while P_4 and P_5 broadcast more than one messages. Now, consider an adversary that corrupts both P_4 and P_5. Let P_1 be the first party to broadcast its message amongst P_1, P_2, P_3. Note that in this case, the message of P_1 does not depend on the messages (or inputs) of P_2 and P_3. The adversary can simply spoof P_2 and P_3 and launch a residual function attack on the inputs of P_1 as discussed before. Therefore, in order to prevent such attacks at least three parties must broadcast at least two messages each.

As we show in the technical section, the above idea generalizes to any n and $t < n - 1$ and any $|\mathcal{O}| > 1$ output parties. Now, in an n-party protocol, each party must broadcast at least one message for its input to be included in the computation. Thus, the total broadcast message complexity in this setting must be at least $2(t + 1) + 1(n - (t + 1)) = n + t + 1$. We use this result to establish lower bounds for $|\mathcal{O}| = 1$ output party when $t < n - 1$ parties are corrupted.

We prove our lower bound for $t = n - 1$ in the plain model and for any $t < n$ in the PKI model using a similar approach as above. However the optimal lower bound in this case is lower than in the plain model; intuitively, this is because in this case, spoofing attacks are not possible. Specifically, we establish that the number of parties that broadcast more than one messages in this case must be greater than or equal to the number of corrupted parties. This means that the total broadcast message complexity in this case must be at least $2(t)+1(n-t) = n + t$.

Lower Bounds on Round Complexity. In Section 4, we prove that at least three rounds are necessary for achieving optimal broadcast message complexity. Our proof generalizes the intuition discussed earlier for the toy example:

- We first show that any party that broadcasts a single message must not do so in the last round of the protocol. Roughly, this is because in this case, there is no opportunity for the input of this party to be "fixed" by messages of the other parties. Indeed, an adversary can otherwise simply corrupt this party and launch a residual function attack on the inputs of all the other honest parties in the manner as discussed earlier in the toy example.
- Given the observation, we show that two round MPC with optimal message complexity is impossible. Roughly, this is because, in any such two round

protocol, all the parties that send a single message must necessarily do so in the first round. However, in this case, the adversary can launch a residual function attack on the inputs of any one of the honest parties that broadcast their only message in the first round, as discussed earlier in the toy example.

Interestingly, this lower bound also holds in the public-key model, regardless of the number of corruptions.

In Section 4.3, we also prove that in the plain model, there is a *unique* communication pattern that must be used to achieve optimal broadcast message complexity. This communication pattern is a generalization of the one discussed above; roughly, we prove that the parties who send one message must speak in the second round, while the parties who send two messages must speak in the first and the third rounds! In the public-key model, we do not establish uniqueness of communication pattern; instead, we show that there is a specific class of communication patterns that must be used to achieve optimal broadcast message complexity.

Upper Bounds. To establish positive results on optimal broadcast message complexity, we provide multiple transformations.

Let us first focus on $t < n/2$ corruptions. In this setting, we establish our positive result in the PKI model by drawing a connection with the notion of *guaranteed output delivery*, which, roughly speaking, concerns with ensuring that the honest parties are able to compute the output even if the corrupted parties abort the computation prematurely. More Specifically,

- We show a general compiler from any two round MPC protocol with "*strong*" guaranteed output delivery (namely, where in the second round, for any $t < n/2$, only $t + 1$ honest parties are required to send a message in order for all the honest parties to compute output[4]) against fail-stop adversaries into a three round *semi-honest* protocol with optimal broadcast message complexity.
- Further, we show that if the underlying two round protocol additionally achieves security with abort against malicious adversaries, then our resulting three round protocol also achieves security against *malicious* adversaries security.

Instantiating our compiler with the recent protocol of Ananth et al. [1] (which satisfies both of the aforementioned properties) yields a malicious-secure MPC protocol with optimal broadcast message complexity based on public-key encryption. Finally, we remark that this transformation inherently fails in the plain model since two round MPC over broadcast channels with guaranteed output delivery is impossible in the plain model, even in the semi-honest setting [20].

Let us now consider the case where $t < n$. We show a general transformation from any two round MPC that achieves security against *dishonest majority*

[4] When $n = 2t + 1$, this is the same as guaranteed output delivery. However, for $n > 2t + 1$, this is a strengthening of guaranteed output delivery.

into a three round MPC with optimal broadcast message complexity. The transformation is simple and works by having a subset of parties (say S_1) send out encrypted secret shares of their private states to the other parties (say S_2), who can compute upon these states to generate messages on behalf of the parties in S_1. In order for this approach to work, it is crucial that at least one party in S_2 is honest, and indeed, this is why we require that the underlying two-round protocol achieves security against dishonest majority. This transformation works both in the plain model as well as the public-key model. Instantiating it with the two-round protocols of [18, 3], we obtain a protocol with optimal broadcast message complexity based on OT.

Finally, we also show that our positive results in the PKI model can be used to obtain tight upper bounds on P2P message complexity (for any corruption threshold), resolving an open question from [22, 23]. We note that if we use a naive transformation from a broadcast model protocol to a P2P model protocol (as discussed above), the resulting protocol would also require the public-key infrastructure used by the underlying protocol. To overcome this, we show alternative, direct transformations from our specific broadcast model protocols to obtain P2P model protocols with optimal P2P message complexity. We refer the reader to Section 7.2 for more details.

1.3 Related Work

To the best of our knowledge, no prior work has studied the broadcast message complexity of MPC.

P2P Message Complexity. The most closely related work to ours is the recent work of Ishai et al [22] who study the message complexity of computationally-secure MPC in the P2P model (i.e., where the parties only rely on P2P channels). For $t = n - 1$ corruptions and $|\mathcal{O}|$ output parties, they show that $2n + |\mathcal{O}| - 3$ P2P messages are necessary and sufficient for semi-honest MPC, even in the correlated randomness setup model. Mittal [23] extended their lower bound to any $t < n$ corruptions and showed that at least $(n + t + |\mathcal{O}| - 2)$ P2P messages are necessary for $|\mathcal{O}|$ output parties. However, they left open the problem of obtaining a matching upper bound, which we resolve in this work.

We note that the lower bounds on P2P message complexity can be used to derive some lower bounds on broadcast message complexity. The basic idea is simple and works by transforming any broadcast model protocol Π into a P2P model protocol Π' executed over a "chain" communication pattern, where the parties are arranged as per their speaking order in Π.[5] At each step, each party computes its new message and sends it together with the aggregated transcript it received from the previous party to the next party on the chain. The last party on the chain computes the output. We give a formal description of this transformation in the full version our paper.

However, this approach has two important shortcomings. First, it does *not* give optimal lower bounds in the plain model. Second, it only works in extremal

[5] if multiple parties speak in the same round in Π, they can be arranged in any order.

cases where the number of output parties $|\mathcal{O}|$ in Π are either $|\mathcal{O}| = 1$ or $|\mathcal{O}| = n$.[6] In particular, it is unclear how to obtain P2P message complexity $k - 1$ if the number of output parties is $1 < |\mathcal{O}| < n$. This is because, in this case, the underlying protocol Π in the broadcast model may not contain an output party in the last round.

To obtain a full picture with optimal lower bounds, our work instead provides a *direct* approach to proving lower bounds on broadcast message complexity. Previously, the P2P message complexity of MPC in the information-theoretic setting with a bounded fraction of corrupted parties was studied in [9, 12, 13], in different models.

Other Works. We also mention the works of [11, 7, 4, 15] who consider MPC protocols that achieve sublinear communication complexity by assigning the computation to a small random subset of parties in the honest majority setting. They do not give any specific bounds on the P2P or broadcast message complexity of MPC.

2 Preliminaries

2.1 Secure Multi-Party Computation

A secure multi-party computation protocol is a protocol executed by n parties $P_1, \cdots, P_n$ for a n-party functionality $\mathcal{F}$.

Communication Model. We consider two kinds of protocols: (1) ones that only rely upon an authenticated broadcast channel, (2) and ones that only rely upon private point-to-point channels. We discuss each case separately:

- *Broadcast model:* In almost all prior work in the broadcast model of communication, in each round of the protocol, *all* parties broadcast a message. We consider a generalization of this setting, where in any round, any subset of parties may broadcast a message. We define the broadcast message complexity as follows:

  Definition 1 (Broadcast Message Complexity). *The broadcast message complexity of a protocol is the total number of broadcast messages sent by all the parties in the protocol. The broadcast message complexity of MPC for a functionality f is the minimum number of broadcast messages required for securely computing f.*

[6] In the case of $|\mathcal{O}| = 1$, we simply add the output party as the last node on the chain. With this approach, we obtain a protocol with P2P message complexity k, starting from a protocol with broadcast message complexity k. For the case of $|\mathcal{O}| = n$, we can actually do better, and simply delete the last message (since the last node on the chain can compute it on its own), resulting in a protocol with P2P message complexity $k - 1$.

– *Point-to-point model:* While our focus is on the broadcast message complexity, we also consider protocols in the point-to-point (P2P) communication model (i.e., where the parties only use private point-to-point channels, and no broadcasts). We define the P2P message complexity as follows:

Definition 2 (P2P Message Complexity). *The P2P message complexity of a protocol is the total number of private messages sent by all the parties in the protocol. The P2P message complexity of MPC is the minimum number of P2P messages required for securely computing any functionality.*

Security. One of the primary goals in secure computation is to protect the honest parties against dishonest behavior from the corrupted parties. This is usually modeled using a central adversarial entity, that controls the set of corrupted parties and instructs them on how to operate. That is, the adversary obtains the views of the corrupted parties, consisting of their inputs, random tapes and incoming messages, and provides them with the messages that they are to send in the execution of the protocol. In our protocols we only consider static adversaries, meaning that the adversary selects the set of parties that it wants to corrupt at the start of the protocol. We discuss the following security models in this work:

– **Security against Semi-Honest Adversaries:** A semi-honest adversary always follows the instructions of the protocol. This is an "honest but curious" adversarial model, where the adversary might try to learn extra information by analyzing the transcript of the protocol later.

Definition 3. *Let f be an n-party functionality. We say that a protocol Π t-securely computes $\mathcal{F}$ in the presence of a semi-honest, non-uniform PPT adversary $\mathcal{A}$ that corrupts a subset A of t parties, if there exists a PPT simulator algorithm $\mathcal{S}$ such that for every security parameter λ, and all input vectors $\mathbf{x} \in \{0,1\}^{\lambda \times n} = \{x_1, \ldots, x_n\}$, it holds that:*

$$\{\mathcal{S}(1^\lambda, \{x_i\}_{i \in \mathsf{A}}, f(\mathbf{x})), f(\mathbf{x})\} \approx_c \{\mathsf{view}^{\Pi}_{\mathcal{A}}(1^\lambda, \mathbf{x}), \mathsf{out}^{\Pi}_{\mathcal{A}}(1^\lambda, \mathbf{x})\}$$

– **Security against Fail-Stop Adversaries:** A fail-stop adversary instructs the corrupted parties to follow the protocol as a semi-honest adversary, but it may also instruct a corrupted party to halt early (only sending some of its messages in a round). The decision to abort or not may depend on its view. Fail-stop adversaries may be rushing or non-rushing. We consider the following security notions against fail-stop adversaries:

Guaranteed Output Delivery: Secure computation against fail-stop adversaries with guaranteed output delivery ensures that the honest parties always learn the function output (computed over the inputs of "active" parties) even if some parties prematurely abort the protocol. It is well known that guaranteed output delivery is impossible to realize for general functions in the dishonest majority setting [10].

Strong Guaranteed Output Delivery: Note that guaranteed output delivery ensures that the honest parties can always reconstruct the output as long as *all* the honest parties send messages in the last round. We require a stronger variant of this notion, where it suffices for any $t+1$ honest parties to broadcast messages in the last round. Observe that if $n = 2t + 1$ (i.e., if there are exactly $t + 1$ honest parties in the system), then this notion is equivalent to the standard notion of guaranteed output delivery. However for $n > 2t + 1$, this notion is strictly stronger than standard guaranteed output delivery.

2.2 MPC with Strong Guaranteed Output Delivery

For our semi-honest construction in the honest majority setting (i.e., for $t < n/2$) with optimal message complexity, we make use of a two round MPC protocol that achieves *strong guaranteed output* delivery against fail-stop adversaries. As defined above, in a protocol that achieves strong guaranteed output delivery, it suffices for any $t+1$ honest parties to broadcast a message in the last round. Thus, the message complexity of such a two round protocol is $n+t+1$. Although note that with $n + t + 1$ messages, this protocol only achieves security with abort, since the adversary can always corrupt the parties that send messages in the last round. However, this weakened security is sufficient for us. We show how to reduce the message complexity of this protocol to $n + t$ by adding an additional round.

Interestingly, we observe that if such a two round protocol also achieves security with abort against malicious adversaries, then our resulting protocols with optimal message complexity in the PKI model for $t < n/2$ can also be proved maliciously secure, without requiring any additional assumptions. We observe that the two round MPC protocol from Corollary 7 of Ananth et. al [1] satisfies both these properties that we require. We give a sketch of the proof for the following theorem in in the full version our paper.

Theorem 5 (Implicit from [1]). *Assuming the existence of public-key encryption, there exists a two-round secure MPC protocol that achieves strong guaranteed output against $t < n/2$ fail-stop corruptions and achieves security with abort against $t < n/2$ malicious corruptions in the PKI model.*

2.3 Functionalities of Interest

Our lower bounds rely on residual function attacks (as described earlier), where only one honest party's input is fixed, and the adversary can evaluate the function on multiple different inputs (by changing the inputs of other parties). For most functionalities, this is a *non-trivial* information that cannot be simulated. In fact, it usually leads to a complete break in the privacy of the honest party whose input is fixed. For concreteness, we present our lower-bounds using one such functionality called the *multi-party OT functionality* defined in [22]. This functionality is a variant of oblivious transfer where each party has three-input

bits. At the end, based on the first input bits of all parties, the output parties only learn one of the input bits of all parties.

Definition 4 (MOT Functionality). *For $n > 2$ and nonempty $\mathcal{O} \subseteq [n]$, let $MOT : X^n \to Y^n$ be the $n-$party functionality defined as follows:*

- *The input domain of each party is $X = \{0,1\}^3$ and the output domain is $Y = \{0,1\}^{n+1}$.*
- *Given input (c_i, x_i^0, x_i^1) from each party P_i, the functionality lets $c = c_1 \oplus \ldots \oplus c_n$ and outputs $(c, x_1^c, \ldots, x_n^c)$ to all parties $P_j, j \in \mathcal{O}$ (the output of party P_j for $j \notin \mathcal{O}$ is the fixed string 0^{n+1}).*

For simplicity, the proofs of all the lower bounds in this work are described with respect to the MOT functionality. However, it should be easy to see that all our proofs extend to any such non-trivial functionality.

3 Lower Bounds on Broadcast Message Complexity

In this section, we provide lower bounds on the broadcast message complexity of MPC in the plain model and the bare public-key model. We show that the broadcast message complexity is different in the two models, and further depends on the number of corruptions t and the number of output parties $|\mathcal{O}|$, where $\mathcal{O}$ is the set of parties who learn the output.

3.1 Plain model

We first investigate the lower bounds on broadcast message complexity of semi-honest MPC protocols in the plain model. We start by proving our lower bound for $t < n-1$ corruptions and $|\mathcal{O}| > 1$ output parties. In this case, we first show that for a secure MPC protocol, at least $t+1$ parties must broadcast more one messages in Lemma 1. Using this result, it is easy to see that even if these $t+1$ parties send two messages each, the total number of messages required are $2(t+1)+1(n-(t+1)) = n+t+1$, since each party in an n-party protocol must broadcast at least one message. Finally we show how this result can be used to get a lower bound for $|\mathcal{O}| = 1$ output party.

We now formally state the following two theorems, for $1 \leqslant t < (n-1)$, and $t = n-1$, respectively. In this subsection, we only give a proof for Theorem 6. The proof of Theorem 7, which follows similarly to Theorem 6, except that it does not rely on spoofing attacks, is deferred to the full version our paper.

Theorem 6. *In the plain model, the broadcast message complexity of n-party MPC for non-trivial functionalities secure against $1 \leqslant t < (n-1)$ semi-honest, static corruptions is $n+t+1$ if the number of output parties is $|\mathcal{O}| > 1$, and $n+t$ if $|\mathcal{O}| = 1$.*

Theorem 7. *In the plain model, the broadcast message complexity of n-party MPC for non-trivial functionalities secure against $t = n-1$ semi-honest, static corruptions is $2n-1$ if the number of output parties is $|\mathcal{O}| > 1$, and $2n-2$ if $|\mathcal{O}| = 1$.*

Proof of Theorem 6.. We divide the proof into two cases, depending upon the number of output parties.

Case1 : $|\mathcal{O}|>1$ Let Π be a secure n-party broadcast channel MPC protocol for the MOT functionality in the plain model with $|\mathcal{O}| > 1$, that is secure against a semi-honest adversary that corrupts $1 \leqslant t < (n-1)$ parties. Let $\mathcal{S}_1$ be the set of parties that broadcast a single message in Π and $\mathcal{S}_{>1}$ be the set of parties that broadcast more than one messages. We start by proving the following lemma.

Lemma 1. *There must be at least $t+1$ parties in Π that broadcast more that 1 message, i.e., $|\mathcal{S}_{>1}| \geqslant t+1$*

Proof. Let us assume for the sake of contradiction that $|\mathcal{S}_{>1}| \leqslant t$. Let $\mathcal{A}$ be an adversary who corrupts all the parties in $\mathcal{S}_{>1}$ and no other party, i.e., all parties in $\mathcal{S}_1$ are honest. Since $|\mathcal{S}_{>1}| \leqslant t$, this is a valid adversary.

Let $P^* \in \mathcal{S}_1$ be the first party to broadcast a message amongst all parties in $\mathcal{S}_1$. We note that there might be more than one such party in $\mathcal{S}_1$ that broadcast their messages simultaneously in a round. In that case, w.l.o.g. we let P^* be the lexicographically first party amongst those. Let $\mathcal{S}_1^* = \mathcal{S}_1 \setminus \{P^*\}$ be the set of all other parties in $\mathcal{S}_1$. Let $\mathcal{P}$ denote the set of all parties in Π and let (x_i, r_i) denote the input and randomness of party $P_i \in \mathcal{P}$. We now describe the strategy of $\mathcal{A}$.

- $\mathcal{A}$ runs an honest execution of Π where it sets $x_j = 000$ for every corrupted party $P_j \in \mathcal{S}_{>1}$. Let trans_Π be the transcript of this execution.
- Let P_k be any party in $\mathcal{S}_1^*$. $\mathcal{A}$ then runs two mental experiments where in the first experiment it sets $x_j = 000$ for every $P_j \in \mathcal{S}_1^*$ and in the second experiment it sets $x_k = 100$ and $x_j = 000$ for every $P_j \in \mathcal{S}_1^* \setminus \{P_k\}$. Let P^* broadcast its message in round ℓ and let $\mathsf{trans}_\Pi^{<\ell}$ be the transcript of honest execution up to round ℓ. Given $\mathsf{trans}_\Pi^{<\ell}$, the message sent by P^* in the honest execution and the new set of inputs, $\mathcal{A}$ uses the next message function of the remaining parties $P_j \in \mathcal{P} \setminus \{P^*\}$ to compute their messages from round ℓ onwards. Let $\mathsf{trans}_{\Pi 1}$ and $\mathsf{trans}_{\Pi 2}$ denote the transcripts of the two mental experiments. Note that since $|\mathcal{O}| > 1$, there is at least one output party in the set $\mathcal{P} \setminus \{P^*\}$. It uses the output function of any output party $P_i \in \mathcal{O} \cap (\mathcal{S}_{>1} \cup \mathcal{S}_1^*)$ to compute outputs $y_1 = \mathsf{Out}_\Pi(i, x_i, r_i, \mathsf{trans}_{\Pi 1})$ and $y_2 = \mathsf{Out}_\Pi(i, x_i, r_i, \mathsf{trans}_{\Pi 2})$.

Claim. $y_1 = (c^*, x_1^{c^*}, \ldots, x_n^{c^*})$, where $x_i = 000$ for each $P_i \in \mathcal{P} \setminus \{P^*\}$ and c^* is the first input bit of party P^*.

Proof. Since P^* is the first party to broadcast a message amongst all parties in $\mathcal{S}_1$, its message does not depend on messages from any other party in $\mathcal{S}_1$. Thus $\mathsf{trans}_{\Pi 1}$ represents honestly computed transcript of Π where the inputs of all parties in $\mathcal{S}_1^*$ have been replaced with 000. Therefore, from the correctness of Π, we get $y_1 = (c^*, x_1^{c^*}, \ldots, x_n^{c^*})$. □

Claim. $y_2 = ((1-c^*), x_1^{(1-c^*)}, \ldots, x_n^{(1-c^*)})$ where $x_k = 100$ for any one party $P_k \in \mathcal{S}_1^*$ and $x_i = 000$ for each $P_i \in \mathcal{P} \setminus \{P^*, P_k\}$.

The proof of this claim is similar to the proof of the previous claim.
From the above claims, we can see that the two outputs y_1 and y_2 reveal both the input bits of the honest party P^*. Such a protocol is clearly not secure. Therefore our assumption is wrong and there must be at least $t+1$ parties in Π that broadcast more that 1 messages. This concludes the proof of Lemma 1. We now use this lemma to prove the first part of Theorem 6. □

Let us assume for the sake of contradiction that there exists a semi-honest secure MPC protocol Π in the plain model, that has a broadcast message complexity of $n+t$ for $|\mathcal{O}| > 1$ output parties. From Lemma 1, we know that at least $t+1$ parties must broadcast at least 2 messages. This means that there is at least one party out of the remaining $n-(t+1)$ parties, that doesn't send a message, i.e., the output of the protocol is independent of this party's input. But we know that for correctness of output of the MOT functionality, every party's input is necessary for computation. Therefore the output computed by this protocol is incorrect. Thus, our assumption is wrong and such a protocol with broadcast complexity $n+t$ cannot exist.

Case2 : $|\mathcal{O}|$=1 We now prove the second part of Theorem 6. Let us assume for the sake of contradiction that there exists an n-party broadcast channel MPC protocol for any non-trivial functionality in the plain model with $|\mathcal{O}| = 1$, that is secure against a semi-honest adversary that corrupts $1 \leqslant t < (n-1)$ parties and has a broadcast message complexity of $n+t-1$. This protocol can be easily transformed into another protocol Π' with the same corruption threshold, where $|\mathcal{O}| = n$. This can be obtained by adding a round at the end of Π where the output party broadcasts the output. Clearly, every party learns the output in Π' and it only has a broadcast message complexity of $n+t$. But from the first part of Theorem 6, we know that such a protocol with a broadcast message complexity of $n+t$ cannot exist. Therefore, our assumption is wrong and the minimum broadcast message complexity for such a protocol with one output party is $n+t$. This completes the proof of Theorem 6. □

3.2 PKI Model

We now investigate the lower bounds on broadcast message complexity of semi-honest MPC protocols against $1 \leqslant t \leqslant (n-1)$ corruptions in the PKI model. We observe that the lower bound for $|\mathcal{O}| \leqslant n-t$ output parties follows similar to the proof of Theorem 7 using the lower bounds of Mittal in [23] on the P2P message complexity.

For $|\mathcal{O}| > n-t$ parties, we start by showing that in a secure MPC protocol in the PKI model, at least t parties must broadcast more than one message each in Lemma 2. The proof of this lemma is very similar to the proof of Lemma 1, except that now it is not possible for the adversary to spoof other parties in the PKI model. Using this result, as before it is easy to see that even if these t parties send two messages each, the total number of messages required are $2(t)+1(n-t) = n+t$, since each party must broadcast a message in an n-party protocol. We now formally prove the following theorem.

Theorem 8. *In the PKI model, the broadcast message complexity of n-party MPC for non-trivial functionalities secure against* $1 \leqslant t \leqslant (n-1)$ *semi-honest, static corruptions is* $n+t$ *if the number of output parties is* $|\mathcal{O}| > n-t$*, and* $n+t-1$ *if* $|\mathcal{O}| \leqslant n-t$*.*

Proof. The lower bound for $|\mathcal{O}| \leqslant n-t$ and $|\mathcal{O}| = n$ output parties can be derived from the lower bound given in [23]. Intuitively, it can be shown that, if there exists a broadcast protocol in this setting with $n+t-2$ broadcast message complexity, and $|\mathcal{O}| \leqslant n-t$ output parties, then the lower bound given in [23] on the P2P message complexity will be violated. Similarly the bound of $n+t$ messages for $|\mathcal{O}| = n$ output parties also holds. We defer the full proof for this setting to the full version of our paper.

But this still leaves open the question about broadcast message complexity of MPC with $1 \leqslant t \leqslant n-1$ semi-honest corruptions and $n-t < |\mathcal{O}| < n$ output parties in th PKI model. We know prove the lower bound of $n+t$ messages for $n-t < |\mathcal{O}| < n$ output parties. The following proof also works for $|\mathcal{O}| = n$ output parties. We prove this bound for the MOT functionality, but this proof can be easily extended for any non-trivial functionality. Let Π be an secure n-party broadcast channel MPC protocol for the MOT functionality in the PKI model with $|\mathcal{O}| > t+1$, that is secure against a semi-honest adversary that corrupts $1 \leqslant t \leqslant (n-1)$ parties. Let $\mathcal{S}_1$ be the set of parties that broadcast a single message in Π and $\mathcal{S}_{>1}$ be the set of parties that broadcast more than one messages. We start by proving the following lemma.

Lemma 2. *There must be at least* t *parties in* Π *that broadcast more that* 1 *message, i.e.,* $|\mathcal{S}_{>1}| \geqslant t$

Proof. Let us assume for the sake of contradiction that $|\mathcal{S}_{>1}| = (t-1)$. Let $P_{\mathsf{last}} \in \mathcal{S}_1$ be the last party to broadcast a message amongst all parties in $\mathcal{S}_1$. Note that there might be more than one such parties in $\mathcal{S}_1$ that broadcast their messages simultaneously in a round. In that case we let P_{last} be the lexicographically last party amongst those. Let $\mathcal{A}$ be an adversary who corrupts P_{last} and all the parties in $\mathcal{S}_{>1}$. Since $|\mathcal{S}_{>1}| = (t-1)$, this is a valid adversary. Let P^* be any honest party. From the above definition of $\mathcal{A}$, we know that $P^* \in \mathcal{S}_1$. Let $\mathcal{P}$ denote the set of all parties in Π and let (x_i, r_i) denote the input and randomness of party $P_i \in \mathcal{P}$. We now describe the strategy of $\mathcal{A}$.

- $\mathcal{A}$ runs an honest execution of Π where it sets $x_j = 000$ for every corrupted party $P_j \in \mathcal{S}_{>1} \cup \{P_{\mathsf{last}}\}$. Let trans_Π be the transcript of this execution. It uses the output function of any corrupted output party $P_o \in \mathcal{O} \cap (\mathcal{S}_{>1} \cup \{P_{\mathsf{last}}\})$ to compute the output $(c, x_1^c, \ldots, x_n^c) = \mathsf{Out}_\Pi(o, x_o, r_o, \mathsf{trans}_\Pi)$. Since $|\mathcal{O}| > n-t$, there is at least one corrupt output party.
- $\mathcal{A}$ then runs a mental experiment where it sets $x_{\mathsf{last}} = 100$. Let P_{last} broadcast its message in round ℓ and let $\mathsf{trans}_\Pi^{<\ell}$ be the transcript of honest execution up to round ℓ. Given $\mathsf{trans}_\Pi^{<\ell}$ and the new sets of inputs, $\mathcal{A}$ uses the next message function of all the corrupted parties to compute their messages from round ℓ onwards. Let trans'_Π denote the transcript of the mental experiment.

It uses the output function of the output party P_o to compute the output $y' = \mathsf{Out}_{\Pi}(o, x_o, r_o, \mathsf{trans}'_{\Pi})$.

Claim. $y' = ((1-c), x_1^{1-c}, \ldots, x_n^{1-c})$, where $x_{\mathsf{last}} = 100$.

Proof. Since P_{last} is the last party to broadcast a message amongst all parties in $\mathcal{S}_1$, messages of other parties (honest parties) in $\mathcal{S}_1$ do not depend on P_{last}'s message. Thus trans'_{Π} represents an honestly computed transcript of Π where only the input of P_{last} has been replaced with 100. Therefore, from the correctness of Π, $y' = ((1-c), x_1^{1-c}, \ldots, x_n^{1-c})$. □

From the above claim, we see that the output of the honest execution and y' reveal both the input bits of the honest party P^*. Such a protocol is clearly not secure. Therefore our assumption is wrong and there must be at least t parties in Π that broadcast more that 1 messages. □

We now use this lemma to prove Theorem 8. Let us assume for the sake of contradiction that there exists a semi-honest secure MPC protocol Π in the PKI model for $1 \leqslant t \leqslant n-1$, that has a broadcast message complexity of $n+t-1$. From Lemma 2, we know that at least t parties must broadcast at least 2 messages. Since $2 \times (t) = 2t$ and $(n+t-1)-(2t) = n-t-1$, there is at least one party that doesn't send a message, i.e., the output of the protocol is independent of this party's input. But we know that for correctness of output of the MOT functionality, every party's input is necessary for computation. Therefore the output computed by this protocol is incorrect. Thus, our assumption is wrong and such a protocol with broadcast complexity $n + t - 1$ cannot exist. This completes the proof of Theorem 8. □

4 Lower Bounds on Round Complexity

In this section, we investigate the minimal round complexity of semi-honest MPC with optimal broadcast message complexity. The following theorem summarizes our results.

Theorem 9. *Three-rounds are necessary for semi-honest MPC with optimal broadcast message complexity. This result holds regardless of the model (plain or bare public key), the number of corruptions or the number of output parties.*

We divide this proof into two parts. In section 4.1, we consider MPC protocols in the plain model with $1 \leqslant t < (n-1)$ corruptions. Later in section 4.2, we consider MPC protocols in the plain model with $t = n-1$ corruptions and in the PKI model with $1 \leqslant t < n$ corruptions.

4.1 Plain model : $1 \leqslant t < (n-1)$

We start by proving in Lemma 3, that the last round in a secure MPC protocol in the plain model with $t < n-1$ semi-honest corruptions cannot consist of messages from parties that broadcast a single message in the protocol.

Next we use this result to show that at least three rounds are necessary for optimal message complexity in the plain model with $t < n - 1$ semi-honest corruptions. Intuitively, assuming for contradiction that there exists such a two-round protocol with optimal message complexity, from Lemma 3 we know that all the parties that send a single message in the protocol must broadcast their message in the first round, and all the other parties broadcast their messages in both the first and second rounds. The adversary can now launch a residual function attack on the inputs of any one party that sends a single message by spoofing all other honest parties. This is possible because the message of this honest party that sends its only message in the first round, does not depend on the messages (or the inputs) of any other party. The adversary can keep recomputing the remaining transcript using different inputs of other parties to compute different outputs of the function.

We now give a formal proof for the MOT functionality, but it is easy to see that this proof can be extended to any non-trivial functionality. Let Π be a secure n-party broadcast channel MPC protocol for the MOT functionality in the plain model with minimum broadcast message complexity, that is secure against a semi-honest adversary who may corrupt up to $1 \leqslant t < (n-1)$ parties. Let $\mathcal{S}_1$ be the set of parties that broadcast a single message in Π and $\mathcal{S}_2$ be the set of parties that broadcast two messages. We start by proving the following lemma.

Lemma 3. *The last round in Π does not consist of messages from parties that broadcast a single message in the protocol.*

Proof. Let us assume for the sake of contradiction that there is a party $P_{\mathsf{last}} \in \mathcal{S}_1$ that broadcasts its message in the last round ℓ. Let $\mathcal{A}$ be an adversary who corrupts any output party $P_o \in \mathcal{O}$. Let $\mathcal{P}$ denote the set of all parties in Π and let (x_i, r_i) denote the input and randomness of party $P_i \in \mathcal{P}$. We now describe the strategy of $\mathcal{A}$.

- $\mathcal{A}$ runs an honest execution of Π where it sets $x_o = 000$ for the corrupt output party P_o. Let trans_{Π} be the transcript of this execution.
- $\mathcal{A}$ then runs two mental experiments where in the first experiment it sets the input of party P_{last}, $x_{\mathsf{last}} = 000$ and in the second experiment it sets $x_{\mathsf{last}} = 100$. Let $\mathsf{trans}_{\Pi}^{<\ell}$ denote the transcript of the honest execution of Π, up to round ℓ. Given $\mathsf{trans}_{\Pi}^{<\ell}$ and the new inputs, $\mathcal{A}$ computes P_{last}'s new messages in the two mental experiments. Let $\mathsf{trans}_{\Pi 1}$ and $\mathsf{trans}_{\Pi 2}$ denote the final transcripts of the two mental experiments. It uses the output function of the output party P_o to compute outputs $y_1 = \mathsf{Out}_{\Pi}(o, x_o, r_o, \mathsf{trans}_{\Pi 1})$ and $y_2 = \mathsf{Out}_{\Pi}(o, x_o, r_o, \mathsf{trans}_{\Pi 2})$.

Claim. $y_1 = (c, x_1^c, \ldots, x_n^c)$, where $x_{\mathsf{last}} = 000$ and c is the xor of the first input bits of all other parties in $\mathcal{P}$.

Proof. Since P_{last} broadcasts its message only in the last round, the messages of all other parties in the protocol are independent of its message. Thus

$\mathsf{trans}_{\Pi 1}$ represents an honestly computed transcript of Π where the input of P_{last} is replaced with 000. Therefore, from the correctness of Π, we get that $y_1 = (c, x_1^c, \ldots, x_n^c)$. □

Claim. $y_2 = ((1-c), x_1^{(1-c)}, \ldots, x_n^{(1-c)})$ where $x_{\mathsf{last}} = 100$ and $1-c$ is the xor of the first input bits of all other parties in $\mathcal{P}$.

The proof of this claim is similar to the proof of the previous claim.
From the above claims, we can see that the two outputs y_1 and y_2 reveal both the input bits of all the honest parties $P_i \in \mathcal{P} \setminus \{P_{\mathsf{last}}, P_o\}$. Such a protocol is clearly not secure. Therefore our assumption is wrong and there must be at least $t+1$ parties in Π that broadcast more that 1 messages. □

We now use this lemma to prove the first part of theorem 9.
We prove this theorem separately for the following cases:

Case 1: $|\mathcal{O}| = 1$**.** From Theorem 6, we know that the broadcast message complexity of Π is $n+t$, i.e., $|\mathcal{S}_1| = n-t$ and $|\mathcal{S}_2| = t$. Let us assume for the sake of contradiction that there are only 2 rounds in Π. From Lemma 3, we know that in a 2 round protocol, all the parties in $\mathcal{S}_1$ must broadcast their messages in the first round. Let P_o be the output party.

Let $\mathcal{A}$ be an adversary who corrupts all parties in $\mathcal{S}_2$. Since $|\mathcal{S}_2| = t$, this is a valid adversary. Let $\mathcal{P}$ denote the set of all parties in Π and let (x_i, r_i) denote the input and randomness of party $P_i \in \mathcal{P}$. We now describe the strategy of $\mathcal{A}$. $\mathcal{A}$ runs an honest execution of Π where it sets $x_i = 000$ for every corrupted party $P_i \in \mathcal{S}_2$. Let trans_{Π} be the transcript of this execution. We now have the following cases:

$\mathbf{P_o} \in \mathcal{S}_\mathbf{2}$: The adversary computes the output of the honest execution $y = (c, x_1^c, \ldots, x_n^c) = \mathsf{Out}_{\Pi}(o, x_o, r_o, \mathsf{trans}_{\Pi})$.

It then runs a mental experiment where it sets the input of P_o, $x_o = 100$. Given the first round messages of all other parties from the honest execution, it computes the new first round message of P_o and the second round messages of all parties in $\mathcal{S}_2$. Let trans'_{Π} denote the transcript of the mental experiment. It uses the output function of the output party P_o to compute the new output $y' = \mathsf{Out}_{\Pi}(o, x_o, r_o, \mathsf{trans}'_{\Pi})$.

$\mathbf{P_o} \in \mathcal{S}_\mathbf{1}$: $\mathcal{A}$ runs two mental experiments, where it sets $x_o = 000$ and $x_o = 100$ respectively. It computes P_o's messages in the two mental experiments using these new inputs. It also computes the second round messages of all corrupted parties given P_o's new message and the remaining first round messages from honest execution. Let $\mathsf{trans}_{\Pi 1}$ and $\mathsf{trans}_{\Pi 2}$ denote the transcripts in the two mental experiments. It then uses the output function of P_o to compute outputs $y_1 = \mathsf{Out}_{\Pi}(o, x_o, r_o, \mathsf{trans}_{\Pi 1})$ and $y_2 = \mathsf{Out}_{\Pi}(o, x_o, r_o, \mathsf{trans}_{\Pi 2})$.
We now analyze these two cases separately:

Analysis for $\mathbf{P_o} \in \mathcal{S}_\mathbf{2}$.

Claim. $y' = ((1-c), x_1^{(1-c)}, \ldots, x_n^{(1-c)})$

Proof. Since all honest parties only broadcast a message in the first round, their messages are independent of the messages from any other party. Thus trans'_{Π} represents an honestly computed transcript of Π where the input of x_o has been replaced with 100. Therefore, from the correctness of Π, $y' = ((1-c), x_1^{(1-c)}, \ldots, x_n^{(1-c)})$ □

From the above claim, we can see that y and y' reveal both the input bits of all the honest parties. Such a protocol is clearly insecure. Therefore our assumption is wrong there must be at least 3 rounds in Π in this case.

Analysis for $\mathbf{P_o} \in \mathcal{S}_1$.

Claim. $y_1 = (c, x_1^c, \ldots, x_n^c)$, where $x_o = 000$ and c is the xor of the first input bits of all other parties in $\mathcal{P}$.

Proof. Since all parties in $\mathcal{S}_1$ broadcast their message in the first round, their messages are independent of the messages from any other party. Thus $\mathsf{trans}_{\Pi 1}$ represents an honestly computed transcript of Π where the input of P_o is replaced with 000. Therefore, from the correctness of Π, we get that $y_1 = (c, x_1^c, \ldots, x_n^c)$. □

Claim. $y_2 = ((1-c), x_1^{(1-c)}, \ldots, x_n^{(1-c)})$ where $x_o = 100$ and c is the xor of the first input bits of all other parties in $\mathcal{P}$.

The proof of this claim is similar to the proof of the previous claim.
From the above claims, we can see that y_1 and y_2 reveal both the input bits of all the honest parties. Such a protocol is clearly insecure. Therefore our assumption is wrong there must be at least 3 rounds in Π in this case.

Case 2: $|\mathcal{O}| > 1$. From theorem 6, we know that the broadcast message complexity of Π is $n + t + 1$, i.e., $|\mathcal{S}_1| = n - t - 1$ and $|\mathcal{S}_2| = t + 1$. Let us assume for the sake of contradiction that there are only 2 rounds in Π. From Lemma 3, we know that in a 2 round protocol, all the parties in $\mathcal{S}_1$ must broadcast their messages in the first round.

Let $\mathcal{A}$ be an adversary who corrupts all but one party in $\mathcal{S}_2$. Let P_{remain} be the remaining honest party in $\mathcal{S}_2$. Since $|\mathcal{S}_2| = t + 1$, this is a valid adversary. Let $\mathcal{P}$ denote the set of all parties in Π and let (x_i, r_i) denote the input and randomness of party $P_i \in \mathcal{P}$. We now describe the strategy of $\mathcal{A}$. $\mathcal{A}$ runs an honest execution of Π where it sets $x_i = 000$ for every corrupted party $P_i \in \mathcal{S}_2 \setminus \{P_{\mathsf{remain}}\}$. Let trans_{Π} be the transcript of this execution. We can have the following cases:

One of the parties in $\mathcal{S}_2$ is an output party: It then runs two mental experiments where in the first experiment, it sets the input $x_{\mathsf{remain}} = 000$ and in the second experiment it sets $x_{\mathsf{remain}} = 100$. Given the new input and first round messages of all other parties from the honest execution, it computes the new first and second round messages of P_{remain} and the second round messages of all other parties in $\mathcal{S}_2$. Let $\mathsf{trans}_{\Pi 1}$ and $\mathsf{trans}_{\Pi 2}$ denote the transcripts of the two mental experiments. It uses the output function of the output party $P_o \in \mathcal{S}_2$ to compute outputs $y_1 = \mathsf{Out}_{\Pi}(o, x_o, r_o, \mathsf{trans}_{\Pi 1})$ and $y_2 = \mathsf{Out}_{\Pi}(o, x_o, r_o, \mathsf{trans}_{\Pi 2})$.

None of the parties in $\mathcal{S}_2$ is an output party: Let $P_o \in \mathcal{S}_1$ be an output party. $\mathcal{A}$ runs two mental experiments where in the first experiment, it sets $x_o = 000$ and $x_{\mathsf{remain}} = 000$ and in the second experiment it sets $x_o = 100$ and $x_{\mathsf{remain}} = 000$. Given the new sets of inputs and the first round messages of all other parties from the honest execution, it computes the new first round messages of P_o and P_{remain}. It also computes the new second round messages of all parties in $\mathcal{S}_2$. Let $\mathsf{trans}_{\Pi 1}$ and $\mathsf{trans}_{\Pi 2}$ denote the transcripts in the two mental experiments. It then uses the output function of P_o to compute outputs $y_1 = \mathsf{Out}_{\Pi}(o, x_o, r_o, \mathsf{trans}_{\Pi 1})$ and $y_2 = \mathsf{Out}_{\Pi}(o, x_o, r_o, \mathsf{trans}_{\Pi 2})$.

Remark. Note that if $t = n - 2$, since $|\mathcal{O}| > 1$, at least one of the parties in $\mathcal{S}_2$ will always be an output party. The second case can only occur if there are at least 3 honest parties. Therefore, if the adversary spoofs two honest parties P_o and P_{remain}, it can still compromise the privacy of at least one honest party.
We analyze the two cases separately:

Analysis for the case when one of the parties in $\mathcal{S}_2$ is an output party:

Claim. $y_1 = (c, x_1^c, \ldots, x_n^c)$, where $x_{\mathsf{remain}} = 000$ and c is the xor of first input bits of all parties other than P_{remain} in the honest execution.

Proof. Since all parties in $\mathcal{S}_1$ broadcast their messages in the first round, their messages are independent of the messages from any other party. Thus $\mathsf{trans}_{\Pi 1}$ represents an honestly computed transcript of Π where the input of P_{remain} has been changed is replaced with 000. Therefore, from the correctness of Π, we get that $y_1 = (c, x_1^c, \ldots, x_n^c)$. □

Claim. $y_2 = ((1-c), x_1^{(1-c)}, \ldots, x_n^{(1-c)})$, where c is the xor of first input bits of all parties other than P_{remain} in the honest execution.

The proof of this claim is similar to the proof of the previous claim.
From the above claims, we can see that y_1 and y_2 reveal both the input bits of all the honest parties. Such a protocol is clearly insecure. Therefore our assumption is wrong there must be at least 3 rounds in Π in this case.

Analysis for the case when none of the parties in $\mathcal{S}_2$ is an output party:

Claim. $y_1 = (c, x_1^c, \ldots, x_n^c)$, where $x_o = 000$, $x_{\mathsf{remain}} = 000$ and c is the xor of first input bits of all parties other than P_{remain} and P_o in the honest execution.

Claim. $y_2 = ((1-c), x_1^{(1-c)}, \ldots, x_n^{(1-c)})$, where $x_o = 100$, $x_{\mathsf{remain}} = 000$ and c is the xor of first input bits of all parties other than P_{remain} and P_o in the honest execution.

The proofs of these claims are similar to the proofs of the two claims in the previous case. From the above claims, we can see that y_1 and y_2 reveal both the input bits of all the other honest parties. Such a protocol is clearly insecure. Therefore our assumption is wrong there must be at least 3 rounds in Π in this case. This completes the proof for the first part of Theorem 9

4.2 Plain Model : $t = n - 1$ and PKI Model : $1 \leqslant t \leqslant n - 1$

Similar to the previous subsection, we start by proving that even in this setting, the last round round of a secure MPC protocol cannot consist of messages from parties that broadcast a single message in the protocol. The rest of the proof also works very similar to the one in the previous subsection. We now give a formal proof for the lower bound on number of rounds in the plain model with $t = n - 1$ corruptions. The proof for PKI model with $1 \leqslant t \leqslant n - 1$ corruptions follows similarly.

Let Π be any n-party broadcast channel MPC protocol for any non-trivial functionality in the plain model with minimum broadcast message complexity, that is secure against a semi-honest adversary who may corrupt up to $t = (n-1)$ parties. Let $\mathcal{S}_1$ be the set of parties that broadcast a single message in Π and $\mathcal{S}_2$ be the set of parties that broadcast two messages. Let $\mathcal{O}$ be the set of output parties. We start with the following lemma.

Lemma 4. *The last round in Π does not consist of messages from parties that broadcast a single message in the protocol.*

The proof of this lemma is similar to the proof of lemma 3. Now we prove Theorem 9. Let us assume for the sake of contradiction that there are only 2 rounds in Π. From Lemma 4, we know that in a 2 round protocol, all the parties in $\mathcal{S}_1$ must broadcast their messages in the first round. We have the following cases:

Case 1: $|\mathcal{O}| = 1$**.** From Theorem 7, we know that the broadcast message complexity of Π is $2n - 2$, i.e., $|\mathcal{S}_1| = 2$ and $|\mathcal{S}_2| = n - 2$. Let $P_o \in \mathcal{O}$ be the only output party.

Claim. In this case $P_o \in \mathcal{S}_1$.

Proof. If we have protocol with broadcast message complexity $2n-2$, where the output party sends a message in the last round, we can always get a protocol with broadcast message complexity $2n-3$ where the output party does not send a message in the last round. Instead it computes the last round message offline and learns the output. But this clearly violates the lower bound of $2n-2$ on the broadcast message complexity of such protocols. Therefore P_o does not send a message in the last round and hence $P_o \in \mathcal{S}_1$. □

Now let $\mathcal{A}$ be an adversary who corrupts all parties in $\mathcal{S}_2$ and the output party P_o. Note that since $\mathcal{A}$ only corrupts t parties, this is a valid adversary. Clearly in this case, the message of the honest party does not depend on the messages of any of the corrupted parties. After running an honest execution of Π, the adversary can simply change the inputs of the corrupted parties while keeping the message of the honest party same and learn multiple different outputs. Thus, Π is clearly insecure and it must have at least three-rounds.

Case 2: $|\mathcal{O}| > 1$**.** From Theorem 7, we know that the broadcast message complexity of Π is $2n-1$, i.e., $|\mathcal{S}_1| = 1$ and $|\mathcal{S}_2| = n-1$. Since there are at least

2 output parties, one of them is definitely in $\mathcal{S}_2$. Let $\mathcal{A}$ be an adversary that chooses to corrupt all the parties in $\mathcal{S}_2$. Clearly in this case, the honest party sends its message in the first round and therefore its message is not dependent on the messages from any of the corrupted parties. After running an honest execution of Π, the adversary can simply change the inputs of the corrupted parties while keeping the message of the honest party same and learn multiple different outputs. Thus, Π is clearly insecure and there must be at least three-rounds in Π.

4.3 Communication Pattern

Since our broadcast model of communication allows for only a subset of parties to send a message in each broadcast round, there are a number of possible communication patterns in which parties may broadcast their messages. However, not all these combinations are viable for obtaining a secure MPC protocol. In the previous section, we already established that all protocols with minimum broadcast message complexity must comprise of at least three-rounds. We inspect the exact communication patterns for secure MPC protocols with optimal broadcast message complexity.

Plain Model : $1 \leqslant t \leqslant n-1$ We show that any MPC protocol with optimal broadcast message complexity in the plain model must follow a unique communication pattern. We state the formal result and give a proof in the full version our paper.

PKI Model : $1 \leqslant t \leqslant n-1$ We show that any MPC protocol with optimal broadcast message complexity in the PKI model must use a communication pattern from a specific class of communication patterns (which is a strict subset of all possible communication patterns). We call it a class of communication patterns because there are more than one communication patterns that fall into the same category of communication patterns that can be use to obtain a secure MPC protocol. We state the formal result and give a proof in the full version our paper.

5 Positive Result in the PKI Model : $t < \frac{n}{2}$

In this section we describe a general compiler to get a three-round semi-honest MPC protocol secure against $t < \frac{n}{2}$ corruptions with optimal broadcast message complexity in the PKI model from any two round MPC protocol with strong guaranteed output delivery against $t < \frac{n}{2}$ fail-stop corruptions in the PKI model. Using the two-round protocol from Theorem 5, that also achieves security with abort against $t < \frac{n}{2}$ malicious adversaries, our resulting three round protocol with optimal broadcast message complexity is also secure against malicious corruptions.

5.1 Overview

To enable the honest output parties to learn the output in an MPC protocol that satisfies strong guaranteed output delivery against $t < \frac{n}{2}$ fail-stop corruptions and security with abort against $t < \frac{n}{2}$ malicious corruptions, only $t + 1$ honest parties are required to participate in the last round. It is easy to observe that such a protocol would provide security with abort against $t < \frac{n}{2}$ malicious corruptions if any $t+1$ parties participate in the last round. This already gives us a maliciously secure MPC protocol with broadcast message complexity of $n+t+1$. To further reduce the broadcast message complexity, we add an extra round in the middle where one of the parties sends it first and second message at the same time. This gives us a three-round maliciously secure MPC protocol against $t < \frac{n}{2}$ corruptions and minimal broadcast message complexity for $|\mathcal{O}| > n - t$ output parties. This protocol can also be transformed into a protocol for $|\mathcal{O}| \leqslant n - t$ output parties. Her we describe a compiler for $|\mathcal{O}| > n - t$ output parties. In the full version of our paper we discuss how this can be extended to $|\mathcal{O}| \leqslant n - t$ output parties in this setting. Let Φ be a two-round protocol that achieves strong guaranteed output delivery against $t < \frac{n}{2}$ fail-stop corruptions and security with abort against $t < \frac{n}{2}$ malicious corruptions, then the transformed three-round protocol has the following template:

R1: Parties $P_1, \ldots, P_{n-1}$ send their first round messages of Φ in the first round.
R2: Party P_n sends its first and second round messages of Φ in the second round.
R3: Parties $P_1, \ldots, P_t$ send their second round messages of Φ in the third round.

This gives us the following theorem statement.

Theorem 10. *Let Φ be a two-round MPC protocol with strong guaranteed output delivery against $t < \frac{n}{2}$ fail-stop corruptions and security with abort against $t < \frac{n}{2}$ malicious corruption with $|\mathcal{O}| = n$ output parties in the PKI model. Then there exists a general compiler that transforms Φ into a three-round maliciously secure MPC protocol with minimum broadcast message complexity in the PKI model that tolerates $t < \frac{n}{2}$ corruptions.*

Applying Theorem 10 to the protocol from Theorem 5, we get the following.

Corollary 1. *Assuming public-key encryption, there exists a three-round maliciously secure MPC protocol with minimum broadcast complexity in the PKI model that tolerates up to $t < \frac{n}{2}$ corruptions.*

5.2 Our Compiler for $|\mathcal{O}| > n - t$

Let $\mathcal{P} = \{P_1, \ldots, P_n\}$ be the set of parties in the protocol and let $\{x_1, \ldots, x_n\}$, $\{r_1, \ldots, r_n\}$, $\{\mathsf{pk}_1, \ldots, \mathsf{pk}_n\}$ and $\{\mathsf{sk}_1, \ldots, \mathsf{sk}_n\}$ be their respective inputs, randomness, public keys and secret keys. Let λ be the security parameter.

Round 1. Each party P_i for $i \in [n - 1]$ does the following:

It computes the first round message Φ_i^1 using its input x_i, randomness r_i, secret key sk_i and public keys of all parties: $\Phi_i^1 \leftarrow \mathsf{NextMsg}_\Phi^1(1^\lambda, i, x_i, \mathsf{sk}_i, \{\mathsf{pk}_1, \ldots, \mathsf{pk}_n\}, \bot; r_i)$ and broadcasts $M_i^1 := \Phi_i^1$ to all other parties.

Round 2. Party P_n does the following:

1. Computes the first round message Φ_n^1 using its input x_n and randomness r_n: $\Phi_n^1 \leftarrow \mathsf{NextMsg}_\Phi^1(1^\lambda, n, x_n, \mathsf{sk}_i, \{\mathsf{pk}_1, \ldots, \mathsf{pk}_n\}, \bot; r_n)$
2. For $i \in [n-1]$, it parses M_i^1 as Φ_i^1 and sets $\mathsf{trans}_\Phi^1 := \{\Phi_i^1\}_{i\in[n]}$.
3. Computes the second round message Φ_n^2 using its input x_n, randomness r_n and previous round transcript trans_Φ^1: $\Phi_n^2 \leftarrow \mathsf{NextMsg}_\Phi^2(1^\lambda, n, x_n, \mathsf{sk}_i, \{\mathsf{pk}_1, \ldots, \mathsf{pk}_n\}, \mathsf{trans}_\Phi^1; r_n)$
4. Broadcasts $M_n^2 := (\Phi_n^1, \Phi_n^2)$

At the end of Round 2. Each party P_i for $i \in [n-1]$ does the following: For j from 1 to $n-1$, parses M_j^1 as Φ_j^1. It parses M_n^2 as (Φ_n^1, Φ_n^2). Finally it sets $\mathsf{trans}_\Phi^1 := \{\Phi_j^1\}_{j\in[n]}$.

Round 3. Each party P_i for $i \in [t]$ does the following:
It computes the second round message $\Phi_i^2 \leftarrow \mathsf{NextMsg}_\Phi^2(1^\lambda, i, x_i, \mathsf{sk}_i, \{\mathsf{pk}_1, \ldots, \mathsf{pk}_n\}, \mathsf{trans}_\Phi^1; r_i)$ using its input x_i, randomness r_i and previous round transcript trans_Φ^1 and broadcasts $M_i^3 := (\Phi_i^2)$ to all other parties.

Output Phase. Each party P_i for $i \in [n]$ does the following:
For $j \in [t]$, it parses M_j^3 as (Φ_j^2). Then it sets $\mathsf{trans}_\Phi^2 := \{\{\Phi_j^2\}_{j\in[t]}, \Phi_n^2\}$. Finally it runs the output phase of Φ, $\mathsf{Out}_\Phi(i, x_i, r_i, \mathsf{sk}_i, \{\mathsf{pk}_1, \ldots, \mathsf{pk}_n\}, \mathsf{trans}_\Phi^1, \mathsf{trans}_\Phi^2)$ to learn the output.

This completes the description of our compiler. We provide a proof of security in the full version our paper.

6 Positive Result in the PKI Model : $t < n$

In this section we describe a general compiler to get a three-round semi-honest MPC protocol against $t < n$ corruptions with optimal broadcast message complexity in the PKI model from any two-round semi-honest MPC with dishonest majority in the plain model.

6.1 Overview

We start with any two round N-party semi-honest MPC protocol Φ, secure against $N-1$ corruptions, where $N = (n-t+2)\times(t+1)$. Let $\mathcal{P} = \{P_1, \ldots, P_n\}$ be the set of parties in our protocol Π and $\{x_1, \ldots, x_n\}$ be their respective inputs.

Let the n-party functionality that they compute on these inputs be $f(x_1, \ldots, x_n)$. We consider an N-party functionality F such that

$$F(x_1, \ldots, x_t, x_{(t+1)_1}, \ldots, x_{(t+1)_{(t+1)}}, \ldots, x_{(n-1)_1}, \ldots, x_{(n-1)_{(t+1)}}, x_n)$$
$$:= f(x_1, \ldots, x_t, x_{(t+1)_1} \oplus \ldots \oplus x_{(t+1)_{(t+1)}}, \ldots, x_{(n-1)_1} \oplus \ldots \oplus x_{(n-1)_{(t+1)}}, x_n)$$

The main idea behind this compiler is to first let $n - t - 1$ parties in Π split their inputs into $t+1$ additive shares each. Then the n parties together compute the N-input function F using Φ. Here we describe a compiler for $|\mathcal{O}| > n - t$ output parties. In the full version we show how this can be extended to the case where the number of output parties are $|\mathcal{O}| \leqslant n - t$. In the full version our paper, we also show how to extend these protocols to the malicious setting. The transformed three-round protocol for $|\mathcal{O}| > n-t$ output parties has the following template:

R1: Parties $P_1, \ldots, P_{n-1}$ participate in the first round.
R2: Only Party P_n participates in the second round.
R3: Parties $P_1, \ldots, P_t$ participate in the third round.

This gives us the following theorem statement.

Theorem 11. *Let Φ be a two-round semi-honest MPC protocol with dishonest majority and $|\mathcal{O}| = n$ output parties in the plain model. Then there exists a general compiler that transforms Φ into a three-round MPC protocol with minimum broadcast message complexity in the PKI model that tolerates up to $t < n$ semi-honest corruptions.*

Applying Theorem 11 to the protocol from Theorem 5.1 from [18], we get the following Corollary.

Corollary 2. *Assuming the existence of two-message semi-honest OT, there exists a three-round semi-honest MPC protocol with minimum broadcast complexity in the PKI model that tolerates up to $t < n$ corruptions.*

6.2 Our Compiler

Next, we describe the compiler for $|\mathcal{O}| > n - t$ output parties in detail.

Building Blocks. The main primitives required in this construction are: (1) A two-round semi-honest MPC protocol Φ for N parties in the plain/CRS model that only uses broadcast channels. (2) An additive secret sharing scheme. We denote this by $\mathsf{SS} := (\mathsf{Share}, \mathsf{Reconstruct})$. (3) A public-key encryption scheme $\mathcal{E} := (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$

Protocol. Let $\mathcal{P} = \{P_1, \ldots, P_n\}$ be the set of parties in the protocol and let $\{x_1, \ldots, x_n\}$, $\{r_1, \ldots, r_n\}$, $\{\mathsf{pk}_1, \ldots, \mathsf{pk}_n\}$ and $\{\mathsf{sk}_1, \ldots, \mathsf{sk}_n\}$ be their respective inputs, randomness, public keys and secret keys. Let λ be the security parameter.
Round 1. Each party P_i for $i \in [t]$ does the following:

It computes the first round message $\Phi_i^1 \leftarrow \mathsf{NextMsg}_{\Phi}^1(1^\lambda, i, x_i, \bot; r_i)$ using its input x_i and randomness r_i and broadcasts $M_i^1 := (\Phi_i^1)$ to all other parties.
Each party P_i for $i \in \{t+1, \ldots, n-1\}$ does the following:

1. Uses an additive secret sharing scheme SS to compute $t+1$ shares of its input x_i and randomness r_i using some random string s_i as follows: $\{x_{i_1}, \ldots, x_{i_t}, x_{i_n}\} \leftarrow \mathsf{Share}(1^\lambda, x_i; s_i)$ and $\{r_{i_1}, \ldots, r_{i_t}, x_{i_n}\} \leftarrow \mathsf{Share}(1^\lambda, r_i; s_i)$
2. Encrypts each input and randomness share x_{i_j} and r_{i_j} under public key pk_j for $j \in \{1, \ldots, t, n\}$: $c_{i_j} \leftarrow \mathsf{Enc}(\mathsf{pk}_j, (x_{i,j}, r_{i,j}); s_i)$
3. Computes the first round message $\Phi^1_{i_j}$ using each of its input and randomness share x_{i_j} and r_{i_j} for $j \in \{1, \ldots, t, n\}$: $\Phi^1_{i_j} \leftarrow \mathsf{NextMsg}^1_\Phi(1^\lambda, i_j, x_{i_j}, \bot; r_{i_j})$
4. Broadcasts $M^2_i := (\{c_{i_j}, \Phi^1_{i_j}\}_{j \in \{1,\ldots,t,n\}})$ to all other parties.

Round 2. Party P_n does the following:

1. Computes the first round message Φ^1_n using its input x_n and randomness r_n. $\Phi^1_n \leftarrow \mathsf{NextMsg}^1_\Phi(1^\lambda, n, x_n, \bot; r_n)$
2. **For** j from $t+1$ to $n-1$:
   (a) Parses M^2_j as $\{c_{j_k}, \Phi^1_{j_k}\}_{k \in [t+1]}$.
   (b) Decrypts c_{j_n} to obtain x_{j_n} and r_{j_n}: $(x_{j_n}, r_{j_n}) := \mathsf{Dec}(\mathsf{sk}_i, c_{j_n})$
3. Sets $\mathsf{trans}^1_\Phi := \{\{\Phi^1_j\}_{j \in \{1,\ldots,t,n\}}, \{\Phi^1_{j_k}\}_{j \in \{t+1,\ldots,n-1\}, k \in [t+1]}\}$
4. Computes the second round message Φ^2_n using its input x_n, randomness r_n and previous round transcript trans^1_Φ: $\Phi^2_n \leftarrow \mathsf{NextMsg}^2_\Phi(1^\lambda, n, x_n, \mathsf{trans}^1_\Phi; r_n)$
5. For each $j \in \{t+1, \ldots, n-1\}$, it computes the second round message $\Phi^2_{j_n}$ using input and randomness share x_{j_n} and r_{j_n} and previous round transcript trans^1_Φ: $\Phi^2_{j_n} \leftarrow \mathsf{NextMsg}^2_\Phi(1^\lambda, j_n, x_{j_n}, \mathsf{trans}^1_\Phi; r_{j_n})$
6. Broadcasts $M^2_n := (\Phi^1_n, \Phi^2_n, \{\Phi^2_{j_n}\}_{j \in \{t+1,\ldots,n-1\}})$

At the end of Round 2. Each party P_i for $i \in [t]$ does the following:

1. **For** j from $t+1$ to $n-1$, it parses M^2_j as $\{c_{j_k}, \Phi^1_{j_k}\}_{k \in \{1,\ldots,t,n\}}$ and decrypts c_{j_i} to obtain x_{j_i} and r_{j_i}: $(x_{j_i}, r_{j_i}) := \mathsf{Dec}(\mathsf{sk}_i, c_{j_i})$
2. **For** j from 1 to t, it parses M^1_j as (Φ^1_j).
3. Parses M^2_n as $(\Phi^1_n, \Phi^2_n, \{\Phi^2_{j_n}\}_{j \in \{t+1,\ldots,n-1\}})$
4. Sets $\mathsf{trans}^1_\Phi := \{\{\Phi^1_j\}_{j \in \{1,\ldots,t,n\}}, \{\Phi^1_{j_k}\}_{j \in \{t+1,\ldots,n-1\}, k \in \{1,\ldots,t,n\}}\}$

Round 3. Each party P_i for $i \in [t]$ does the following:

1. Computes the second round message Φ^2_i using its own input x_i, randomness r_i and previous round transcript trans^1_Φ: $\Phi^2_i \leftarrow \mathsf{NextMsg}^2_\Phi(1^\lambda, i, x_i, \mathsf{trans}^1_\Phi; r_i)$
2. For each $j \in \{t+1, \ldots, n-1\}$, it computes the second round message $\Phi^2_{j_i}$ using input and randomness share x_{j_i} and r_{j_i} and previous round transcript trans^1_Φ. For each $j \in \{t+1, \ldots, n\}$: $\Phi^2_{j_i} \leftarrow \mathsf{NextMsg}^2_\Phi(1^\lambda, j_i, x_{j_i}, \mathsf{trans}^1_\Phi; r_{j_i})$
3. Broadcasts $M^3_i := (\Phi^2_i, \{\Phi^2_{j_i}\}_{j \in \{t+1,\ldots,n\}})$

Output Phase. Each party P_i for $i \in [n]$ does the following: For j from 1 to t, it parses M^3_j as $(\Phi^2_j, \{\Phi^2_{k_j}\}_{k \in \{t+1,\ldots,n-1\}})$. Then it sets $\mathsf{trans}^2_\Phi := \{\{\Phi^2_j\}_{j \in \{1,\ldots,t,n\}}, \{\Phi^2_{j_k}\}_{j \in \{t+1,\ldots,n-1\}, k \in \{1,\ldots,t,n\}}\}$. Finally, it runs the output phase of Φ, $\mathsf{Out}_\Phi(i, x_i, r_i, \mathsf{trans}^1_\Phi, \mathsf{trans}^2_\Phi)$ to learn the output.

This completes the description of the compiler. We prove its security in the full version our paper.

7 Extensions

The protocols presented in the previous sections can be extended in various ways to obtain different protocols with additional properties.

7.1 Protocol in the Plain Model for $1 \leqslant t < n - 1$:

Such a three-round protocol can be obtained by slightly modifying the compiler from section 6.2. In the first round, parties $P_1, \ldots, P_t$ behave exactly as they do in the previous compiler, additionally they also send their public keys. P_n also sends its first message of the underlying protocol in the first round along with its public key. Parties $P_{t+1}, \ldots, P_{n-1}$ compute their messages exactly as do are doing in the previous compiler. The only difference is that now they send these messages in the second round. Then in the third round, parties $P_1, \ldots, P_t$ behave exactly as they do in the previous compiler. Additionally P_n also sends its remaining message in the third round. This compiler can also be instantiated using two-round protocols from [18, 3]. The resulting protocol has a broadcast message complexity of $n + t + 1$ messages which is optimal for $|\mathcal{O}| > 1$ output parties. The broadcast message complexity of this protocol can be reduced by one if there is a *single* output party. If one less party broadcasts a message in the third round and instead computes this message and output offline, we get a protocol with broadcast message complexity of $n + t$, which is also optimal for $|\mathcal{O}| = 1$ output party. Similar to the previous one, this result can also be extended to achieve malicious security in the CRS model while preserving the optimal broadcast message complexity. This can be done by instantiating the above compiler using the two-round maliciously secure protocol from the work of Garg et al. in [18] based on two-round OT in the CRS model with simulation-based security against malicious receivers and semi-honest senders along with an equivocation property. We give a similar to extension to obtain a protocol for $t = n - 1$ in the plain model with optimal message complexity in the full version our paper.

7.2 P2P Message Complexity

In [22, 23], Ishai et al. and Mittal give a lower bound of $n + t - 1$ messages on the P2P message complexity of MPC for $|\mathcal{O}| = 1$ output party with $t < n$ corruptions. While Ishai et. al. do give a construction for $t = n - 1$, the work of Mittal in [23] does not give a positive result for this lower bound for $t < n - 1$. In this section we give a protocol with optimal P2P message complexity. At first, it might seem that the positive results discussed in our work in broadcast setting would directly give a protocol with optimal P2P message complexity by applying a simple. But this is in fact not true. If we apply this transformation to our protocol in the plain model, we only get a protocol with P2P message complexity of $n + t$ for $|\mathcal{O}| = 1$, which is not optimal. If we instead apply this transformation to our protocols from the PKI model, we do get a protocol with optimal P2P message complexity, but the resulting protocol is also in the PKI model, which is not optimal in the assumptions. Below we describe an extension

to our protocols from the PKI model to obtain protocols that are optimal in the assumptions as well as the P2P message complexity. The protocol given in section 6.2 can transformed as follows:

- P_1 computes its first round message as described in that protocol and forwards it to party P_2 along with the public key.
- For $i \in \{2, \ldots, t\}$, Party P_i computes its first round message as described in that protocol and forwards it to party P_{i+1} along with its public key and all the messages received from P_{i-1}.
- Now that party P_{t+1} has access to the public keys of the first t parties, it computes its first round message as described in that protocol except that it does not encrypt the shares for party P_n, instead the shares for P_n are kept in the clear. It forwards its message along with all the messages received from P_t to P_{t+2}.
- For $i \in \{t+2, \ldots, n-1\}$, party P_i computes its message exactly as P_t does above and forwards it along along with all the messages received from P_{i-1} to P_{i+1}.
- Party P_n computes its message exactly as it does in the described protocol, except that it does not need to decrypt the shares, instead it receives all the shares in the clear from P_{n-1}. It forwards its message along with all the other messages (except the secret shares intended for P_n) received from P_{n-1} to P_1.
- For $i \in \{1, \ldots, t-1\}$, Party P_i computes its second message as described in that protocol and forwards it to party P_{i+1} along with all the messages received from P_{i-1}.
- At the end Party P_t can compute the output.

This gives us an $n+t-1$ message P2P protocol without any setup assumptions. This protocol can be trivially extended to obtain a protocol with $|\mathcal{O}|$ output parties that has $n+t+|\mathcal{O}|-2$ messages. This can be done by having P_t forward the output of the protocol to all the other output parties, using $|\mathcal{O}|-1$ additional messages.

Acknowledgments. The first author is supported in part from DARPA/ARL SAFEWARE Award W911NF15C0210, AFOSR Award FA9550-15-1-0274, AFOSR Award FA9550-19-1-0200, AFOSR YIP Award, NSF CNS Award 1936826, DARPA and SPAWAR under contract N66001-15-C-4065, a Hellman Award and research grants by the Okawa Foundation, Visa Inc., and Center for Long-Term Cybersecurity (CLTC, UC Berkeley). The second and third authors are supported in part by NSF SaTC grant 1814919 and Darpa Safeware grant W911NF-15-C-0213. The views expressed are those of the authors and do not reflect the official policy or position of the funding agencies.

References

1. Ananth, P., Choudhuri, A.R., Goel, A., Jain, A.: Round-optimal secure multiparty computation with honest majority. Advances in Cryptology - CRYPTO 2018 - 38th Annual Cryptology Conference (2018)

2. Ben-Or, M., Goldwasser, S., Wigderson, A.: Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In: 20th Annual ACM Symposium on Theory of Computing. pp. 1–10. ACM Press, Chicago, IL, USA (May 2–4, 1988)
3. Benhamouda, F., Lin, H.: k-round multiparty computation from k-round oblivious transfer via garbled interactive circuits. In: Nielsen, J.B., Rijmen, V. (eds.) Advances in Cryptology – EUROCRYPT 2018, Part II. Lecture Notes in Computer Science, vol. 10821, pp. 500–532. Springer, Heidelberg, Germany, Tel Aviv, Israel (Apr 29 – May 3, 2018)
4. Boyle, E., Chung, K.M., Pass, R.: Large-scale secure computation: Multi-party computation for (parallel) RAM programs. In: Gennaro, R., Robshaw, M.J.B. (eds.) Advances in Cryptology – CRYPTO 2015, Part II. Lecture Notes in Computer Science, vol. 9216, pp. 742–762. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 16–20, 2015)
5. Boyle, E., Gilboa, N., Ishai, Y.: Group-based secure computation: Optimizing rounds, communication, and computation. In: Coron, J., Nielsen, J.B. (eds.) Advances in Cryptology – EUROCRYPT 2017, Part II. Lecture Notes in Computer Science, vol. 10211, pp. 163–193. Springer, Heidelberg, Germany, Paris, France (Apr 30 – May 4, 2017)
6. Boyle, E., Gilboa, N., Ishai, Y., Lin, H., Tessaro, S.: Foundations of homomorphic secret sharing. In: Karlin, A.R. (ed.) ITCS 2018: 9th Innovations in Theoretical Computer Science Conference. vol. 94, pp. 21:1–21:21. LIPIcs, Cambridge, MA, USA (Jan 11–14, 2018)
7. Boyle, E., Goldwasser, S., Tessaro, S.: Communication locality in secure multi-party computation - how to run sublinear algorithms in a distributed setting. In: Sahai, A. (ed.) TCC 2013: 10th Theory of Cryptography Conference. Lecture Notes in Computer Science, vol. 7785, pp. 356–376. Springer, Heidelberg, Germany, Tokyo, Japan (Mar 3–6, 2013)
8. Chaum, D., Crépeau, C., Damgård, I.: Multiparty unconditionally secure protocols (extended abstract). In: 20th Annual ACM Symposium on Theory of Computing. pp. 11–19. ACM Press, Chicago, IL, USA (May 2–4, 1988)
9. Chor, B., Kushilevitz, E.: A communication-privacy tradeoff for modular addition. Inf. Process. Lett. 45(4), 205–210 (1993), https://doi.org/10.1016/0020-0190(93)90120-X
10. Cleve, R.: Limits on the security of coin flips when half the processors are faulty (extended abstract). In: 18th Annual ACM Symposium on Theory of Computing. pp. 364–369. ACM Press, Berkeley, CA, USA (May 28–30, 1986)
11. Cramer, R., Damgård, I., Nielsen, J.B.: Multiparty computation from threshold homomorphic encryption. In: Pfitzmann, B. (ed.) Advances in Cryptology – EUROCRYPT 2001. Lecture Notes in Computer Science, vol. 2045, pp. 280–299. Springer, Heidelberg, Germany, Innsbruck, Austria (May 6–10, 2001)
12. Damgård, I., Nielsen, J.B., Ostrovsky, R., Rosén, A.: Unconditionally secure computation with reduced interaction. In: Fischlin, M., Coron, J.S. (eds.) Advances in Cryptology – EUROCRYPT 2016, Part II. Lecture Notes in Computer Science, vol. 9666, pp. 420–447. Springer, Heidelberg, Germany, Vienna, Austria (May 8–12, 2016)
13. Damgård, I., Nielsen, J.B., Polychroniadou, A., Raskin, M.: On the communication required for unconditionally secure multiplication. In: Robshaw, M., Katz, J. (eds.) Advances in Cryptology – CRYPTO 2016, Part II. Lecture Notes in Computer Science, vol. 9815, pp. 459–488. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 14–18, 2016)

14. Dodis, Y., Halevi, S., Rothblum, R.D., Wichs, D.: Spooky encryption and its applications. In: Robshaw, M., Katz, J. (eds.) Advances in Cryptology – CRYPTO 2016, Part III. Lecture Notes in Computer Science, vol. 9816, pp. 93–122. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 14–18, 2016)
15. Garay, J.A., Ishai, Y., Ostrovsky, R., Zikas, V.: The price of low communication in secure multi-party computation. In: Katz, J., Shacham, H. (eds.) Advances in Cryptology – CRYPTO 2017, Part I. Lecture Notes in Computer Science, vol. 10401, pp. 420–446. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 20–24, 2017)
16. Garg, S., Gentry, C., Halevi, S., Raykova, M.: Two-round secure MPC from indistinguishability obfuscation. In: Lindell, Y. (ed.) TCC 2014: 11th Theory of Cryptography Conference. Lecture Notes in Computer Science, vol. 8349, pp. 74–94. Springer, Heidelberg, Germany, San Diego, CA, USA (Feb 24–26, 2014)
17. Garg, S., Srinivasan, A.: Garbled protocols and two-round MPC from bilinear maps. In: 58th Annual Symposium on Foundations of Computer Science. pp. 588–599. IEEE Computer Society Press (2017)
18. Garg, S., Srinivasan, A.: Two-round multiparty secure computation from minimal assumptions. In: Nielsen, J.B., Rijmen, V. (eds.) Advances in Cryptology – EUROCRYPT 2018, Part II. Lecture Notes in Computer Science, vol. 10821, pp. 468–499. Springer, Heidelberg, Germany, Tel Aviv, Israel (Apr 29 – May 3, 2018)
19. Goldreich, O., Micali, S., Wigderson, A.: How to play any mental game or A completeness theorem for protocols with honest majority. In: Aho, A. (ed.) 19th Annual ACM Symposium on Theory of Computing. pp. 218–229. ACM Press, New York City, NY, USA (May 25–27, 1987)
20. Gordon, S.D., Liu, F.H., Shi, E.: Constant-round MPC with fairness and guarantee of output delivery. In: Gennaro, R., Robshaw, M.J.B. (eds.) Advances in Cryptology – CRYPTO 2015, Part II. Lecture Notes in Computer Science, vol. 9216, pp. 63–82. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 16–20, 2015)
21. Halevi, S., Lindell, Y., Pinkas, B.: Secure computation on the web: Computing without simultaneous interaction. In: Rogaway, P. (ed.) Advances in Cryptology – CRYPTO 2011. Lecture Notes in Computer Science, vol. 6841, pp. 132–150. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 14–18, 2011)
22. Ishai, Y., Mittal, M., Ostrovsky, R.: On the message complexity of secure multiparty computation. In: Abdalla, M., Dahab, R. (eds.) PKC 2018: 21st International Conference on Theory and Practice of Public Key Cryptography, Part I. Lecture Notes in Computer Science, vol. 10769, pp. 698–711. Springer, Heidelberg, Germany, Rio de Janeiro, Brazil (Mar 25–29, 2018)
23. Mittal, M.: Necessary and sufficient conditions for general interaction patterns for mpc. UCLA Thesis for Master of Science in Computer Science (2017)
24. Mukherjee, P., Wichs, D.: Two round multiparty computation via multi-key FHE. In: Fischlin, M., Coron, J.S. (eds.) Advances in Cryptology – EUROCRYPT 2016, Part II. Lecture Notes in Computer Science, vol. 9666, pp. 735–763. Springer, Heidelberg, Germany, Vienna, Austria (May 8–12, 2016)
25. Yao, A.C.C.: How to generate and exchange secrets (extended abstract). In: 27th Annual Symposium on Foundations of Computer Science. pp. 162–167. IEEE Computer Society Press, Toronto, Ontario, Canada (Oct 27–29, 1986)