On Communication Models and Best-Achievable Security in Two-Round MPC

Aarushi Goel[1], Abhishek Jain[1], Manoj Prabhakaran[2], and Rajeev Raghunath[2]

[1]Johns Hopkins University
{aarushig,abhishek}@cs.jhu.edu
[2]IIT Bombay
{mp,mrrajeev}@cse.iitb.ac.in

Abstract

Recently, a sequence of works have made strong advances in two-round (i.e., round-optimal) secure multi-party computation (MPC). In the *honest-majority* setting – the focus of this work – Ananth et al. [CRYPTO'18, EC'19], Applebaum et al. [TCC'18, EC'19] and Garg et al. [TCC'18] have established the feasibility of general two-round MPC in standard communication models involving broadcast ($\mathcal{BC}$) and private point-to-point ($\mathcal{P}2\mathcal{P}$) channels.

In this work, we set out to understand what features of the communication model are necessary for these results, and more broadly the design of two-round MPC. Focusing our study on the plain model – the most natural model for honest-majority MPC – we obtain the following results:

- **Dishonest majority from Honest majority:** In the two round setting, honest-majority MPC and dishonest-majority MPC are surprisingly close, and often *equivalent*. This follows from our results that the former implies 2-message oblivious transfer, in many settings. (i) We show that without private point-to-point ($\mathcal{P}2\mathcal{P}$) channels, i.e., when we use only broadcast ($\mathcal{BC}$) channels, *honest-majority MPC implies 2-message oblivious transfer*. (ii) Furthermore, this implication holds even when we use both $\mathcal{P}2\mathcal{P}$ and $\mathcal{BC}$, provided that the MPC protocol is robust against "fail-stop" adversaries.
- **The curious case of Identifiable Abort:** While security with guaranteed output delivery (and even fairness) against malicious adversaries is impossible in two rounds, nothing is known with regards to the "next best" security notion, namely, security with identifiable abort (IA). We show that IA is *impossible* to achieve with honest-majority even if we use both $\mathcal{P}2\mathcal{P}$ and $\mathcal{BC}$ channels. However, surprisingly, this lower bound can be overcome by replacing $\mathcal{P}2\mathcal{P}$ channels with a "bare" (i.e., untrusted) public-key infrastructure ($\mathcal{PKI}$).

These results "explain" that the reliance on $\mathcal{P}2\mathcal{P}$ channels (together with $\mathcal{BC}$) in the recent two-round protocols was in fact *necessary*, and that these protocols *couldn't* have achieved a stronger security guarantee, namely, IA. Overall, our results (put together with prior works) fully determine the best-achievable security for honest-majority MPC in different communication models in two rounds. As a consequence, they yield the following hierarchy of communication models:

$$\mathcal{BC} < \mathcal{P}2\mathcal{P} < \mathcal{BC} + \mathcal{P}2\mathcal{P} < \mathcal{BC} + \mathcal{PKI}.$$

This shows that contrary to common perception, $\mathcal{BC}$ channel is the *weakest* communication model, and that a bare $\mathcal{PKI}$ setup is strictly stronger than $\mathcal{P}2\mathcal{P}$ channels.

Contents

1 Introduction

Recently, a sequence of works [19, 9, 1, 31, 3, 18, 2, 4, 12] have made strong advances in *two-round* secure multi-party computation (MPC). These works have established the feasibility of general two-round (i.e., round-optimal) MPC, relying on essentially minimal computational assumptions.

Such round optimality is of both theoretical and practical interest. In particular, it opens up the possibility of using MPC in scenarios where more rounds of interaction leads to significant costs, or in tools where a third round is simply inadmissible (e.g., if the communication is over blockchains, or if the first round messages are to be interpreted as "public keys" used to create "ciphertexts" in the second round). On the theoretical front, the separation between 1, 2 or more round protocols is arguably as fundamental as the separation between minicrypt, cryptomania or obfustopia, in that they admit only some cryptographic tools and not others. Indeed, the round complexity of protocols (e.g., of zero-knowledge proofs [24] and MPC) has always been a central theoretical question.

The practical and theoretical significance of round complexity is intertwined with the specific communication models employed. There are two major models of communication channels – *broadcast* ($\mathcal{BC}$) channels and secure *point-to-point* ($\mathcal{P}2\mathcal{P}$) channels – that have been central in the MPC literature, starting from early results in the multi-party setting [22, 11, 8, 32]. In the *honest-majority* setting – the focus of this work – these channels can provide varying "powers": e.g., $\mathcal{P}2\mathcal{P}$ channels are necessary for achieving information-theoretic security [11, 8], and broadcast channels are necessary for achieving security against $t > n/3$ corruptions [17]. They can also provide different use cases, e.g., a protocol that solely uses $\mathcal{BC}$ would be applicable in scenarios where, say, the first round messages are to be interpreted as public keys.

Our Work. The focus of this work is on understanding the role of these channels in the two-round setting with honest majority, where their differences come into sharper contrast. We ask:

In two-round honest-majority MPC, in the different communication models involving $\mathcal{BC}$ and $\mathcal{P}2\mathcal{P}$, what levels of security are achievable for general computation, and under what assumptions?

That is, we seek to understand the best-achievable security and the necessary assumptions in different communication models. We focus our study on the *plain* model – the most natural model for honest-majority MPC.[1] We sometimes augment our model to include a "bare" (i.e., untrusted) public-key infrastructure (PKI) as a means for emulating $\mathcal{P}2\mathcal{P}$ channels over $\mathcal{BC}$.[2]

Background on Security Notions. Before presenting our results, we provide a brief discussion on the prominent security notions studied in the literature. The weakest of them all is *semi-honest* (SH) security that guarantees privacy against semi-honest (a.k.a. honest but curious) adversaries. The case of malicious adversaries is more complex, and a variety of security notions have been studied.[3]

- **Security with abort:** A suite of three increasingly stronger security notions allows a malicious adversary to prevent the honest parties from learning the output by prematurely

[1] Typically, the honest-majority assumption is viewed as an alternative to trusted setup assumptions such as a common reference string (CRS).

[2] In a *bare* PKI setup, the adversarial parties need not register their keys prior to the protocol. Moreover, a party does not need to prove knowledge of its secret key.

[3] The list of notions we discuss here is not exhaustive and some other notions have been studied that lie "in-between" the primary notions. This includes, e.g., semi-malicious security [5], which is a slight strengthening of SH, and fairness, which is a weakening of M-GoD. The lower and upper bounds for these notions tend to be similar to their respective "closest" notions; hence we do not explicitly discuss them.

aborting the protocol: (a) *selective abort* (SA), where the adversary may selectively force a subset of honest parties to abort, (b) *unanimous abort* (UA), where all the honest parties agree on whether or not to abort, and (c) *identifiable abort* (IA) [30], where the honest parties agree on the identity of a corrupted party in the case of an abort.

- **Security with guaranteed output delivery:** Security with *guaranteed output delivery* ensures that an adversary cannot prevent the honest parties from learning the output via premature aborts. This notion is meaningful, both against fully malicious adversaries, and *fail-stop* adversaries who behave like semi-honest adversaries, except that they may prematurely abort. We refer to security in these two cases as M-GoD and FS-GoD, respectively.

The relationship between all of these notions can be summarized as follows: SH < SA < UA < IA < M-GoD, and SH < FS-GoD < M-GoD (note that FS-GoD is incomparable to SA, UA and IA). It is already known that M-GoD is impossible in two-rounds [20, 31].[4] As such, we focus our study on the remaining security notions. Throughout this work, we focus on threshold adversary structures, where the adversary corrupts t out of n parties.

Summary of Our Contributions. We start by providing a high-level statement of the key conclusions from our study, while omitting some finer points and results. We sketch an overview (omitting the specifics of the computational assumptions involved) in Figure 1, which shows how our results fill in the gaps from prior work with regards to the feasibility of different security notions. A detailed description of our results in different communication models is given in Section 1.1.

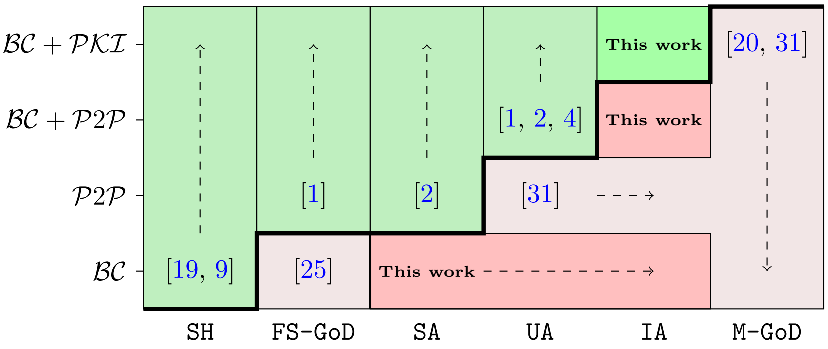

Figure 1: Hierarchy of communication models in two-round honest-majority MPC without trusted setup. Green denotes feasibility of a security level and red denotes impossibility. The security notions featured in the columns are explained below.

- **Necessity of Oblivious Transfer:** While honest-majority MPC without any round restrictions is possible information-theoretically, our first set of results show that in many cases two-round MPC implies the existence of a two-message two-party *oblivious transfer* (OT) protocol:

[4] There are two exceptions: first, in the corner case of exactly one corruption (i.e., $t = 1$) and $n \geq 4$, this impossibility result can be circumvented in the plain model [29, 27]. Alternatively, if we allow for *both* a common random string (CRS) setup and a public-key infrastructure, then the impossibility result can be circumvented (for any $t < n/2$) by relying on standard computational assumptions [25, 1]. In this work, we do not consider trusted setups.

- When the two-round honest-majority MPC protocol is over a $\mathcal{BC}$ channel only (no $\mathcal{P}2\mathcal{P}$ channels), then it implies a two-message OT protocol. If the original MPC protocol is semi-honest or malicious secure, and if it is in the plain model or uses a setup like a common reference string, the OT protocol inherits the same properties.
- Even if the honest-majority MPC protocol uses *both* a $\mathcal{BC}$ channel and $\mathcal{P}2\mathcal{P}$ channels, if it offers FS-GoD security, then it implies two-message semi-honest OT. Interestingly, this holds only when the corruption threshold is $n/3 \leq t < n/2$; for $t < n/3$, we show that minicrypt assumptions are in fact sufficient.

- **Equivalence of Honest Majority and Dishonest Majority:** An interesting consequence of the first of the above results is that it removes the qualitative difference between honest-majority and dishonest-majority in the two-round $\mathcal{BC}$-only setting. Specifically, in the semi-honest setting, an honest-majority protocol implies two-message semi-honest OT, which in turn implies two-round dishonest-majority MPC [19, 9]. On the other hand, in the malicious adversary setting, two-message OT is impossible in the plain model, and it follows that achieving malicious security is *impossible* in the honest-majority setting without $\mathcal{P}2\mathcal{P}$ channels (as was already known for dishonest majority [23]). In other words, removing $\mathcal{P}2\mathcal{P}$ channels "strips off" the advantages of the honest-majority model and places it on equal footing with dishonest-majority MPC – both in terms of necessary assumptions and feasibility.

- **The Curious Case of Identifiable Abort:** While M-GoD and fairness are impossible in two rounds without trusted setup [20, 31], nothing is known with regards to the "next best" security notion, namely, IA. We show that IA is also *impossible* in the plain model, even in the $\mathcal{BC} + \mathcal{P}2\mathcal{P}$ setting. However, surprisingly, if we replace $\mathcal{P}2\mathcal{P}$ channels with a bare $\mathcal{PKI}$, then IA security is in fact *possible.*

  This yields a separation between $\mathcal{BC} + \mathcal{PKI}$ and $\mathcal{BC} + \mathcal{P}2\mathcal{P}$, with only the former admitting secure protocols.[5]

These results "explain" that the reliance on $\mathcal{P}2\mathcal{P}$ channels (together with $\mathcal{BC}$) in the recent constructions of two-round honest-majority MPC protocols [1, 31, 3, 18, 2, 4] was in fact *necessary*, and that these protocols *couldn't* have achieved the stronger security guarantee of IA or achieved security with FS-GoD under weaker assumptions.

Overall, our results (put together with prior works) fully determine the best-achievable security notions in different communication models in two rounds in the honest-majority setting. Referring to Figure 1, we obtain the following hierarchy of communication models:

$$\mathcal{BC} \;<\; \mathcal{P}2\mathcal{P} \;<\; \mathcal{BC} + \mathcal{P}2\mathcal{P} \;<\; \mathcal{BC} + \mathcal{PKI}.$$

This shows that contrary to common perception, $\mathcal{BC}$ channel is the *weakest* communication model, and that a $\mathcal{PKI}$ setup is strictly stronger than $\mathcal{P}2\mathcal{P}$ channels.

1.1 Our Results in Detail

We conduct a comprehensive study of the role of communication channels in two-round honest-majority MPC. There are four natural communication models that one can consider: (i) $\mathcal{BC}$ *only*,

[5] Previously, [20, 25] also established a separation between $\mathcal{BC} + \mathcal{PKI}$ and $\mathcal{BC} + \mathcal{P}2\mathcal{P}$ w.r.t. M-GoD, albeit in the *CRS model*.

i.e., where the protocol only uses $\mathcal{BC}$ channels, (ii) $\mathcal{P2P}$ *only*, i.e., where the protocol only uses $\mathcal{P2P}$ channels, (iii) $\mathcal{BC}+\mathcal{P2P}$, where protocol uses both $\mathcal{BC}$ and $\mathcal{P2P}$ channels, and (iv) $\mathcal{BC}+\mathcal{PKI}$, where we replace $\mathcal{P2P}$ channels with a "bare" public-key infrastructure. Out of these four, the $\mathcal{P2P}$ only model is already pretty well-understood from prior work. Hence, we primarily focus on the remaining three models.

For each of these models, we obtain new results for two-round honest-majority MPC that we elaborate on below. See Figure 2 for a summary.

I. Broadcast only. We first investigate the feasibility of two-round honest-majority MPC without $\mathcal{P2P}$ channels, i.e., by relying only on $\mathcal{BC}$. In this model, we show that *two-round honest-majority MPC is equivalent to two-round dishonest-majority MPC*. In other words, without $\mathcal{P2P}$ channels, achieving security against dishonest minority is as hard as against dishonest majority.

Specifically, we show that any two-round honest-majority MPC for general functions in the $\mathcal{BC}$ only model can be transformed into two-round oblivious transfer (OT). Starting with an MPC with SH security yields semi-honest OT (sh-OT), while starting with one with SA (or stronger malicious) security yields malicious-receiver OT (mR-OT), where the view of a malicious receiver can be simulated.

Overall, in Section 4, we establish that sh-OT (resp., mR-OT) is *necessary* for SH (resp., SA, UA, IA), thereby yielding the following corollaries:

- SA, UA and IA are *impossible* in the plain model. This follows from the impossibility of two-round mR-OT in the plain model.

  Recently, two-round honest-majority MPC protocols with SH [1, 3, 18, 2], SA [2, 4] and UA [1, 2, 4] security were constructed for general circuits based on one-way functions (OWF) and for $\mathbf{NC}^1$ circuits unconditionally, i.e., with information-theoretic (IT) security. These protocols use (only) $\mathcal{P2P}$ channels for achieving SH and SA security, and $\mathcal{BC}+\mathcal{P2P}$ channels for achieving UA security. The above result establishes that the reliance on $\mathcal{P2P}$ channels in these protocols is *necessary*.

- We observe that our transformation in fact also works in the common reference string (CRS) model. In the CRS model, two-round dishonest-majority MPC with SA and UA security was established in [19, 9] based on mR-OT.[6] Recently, [12] extended these results to also capture IA security. A natural question is whether one could obtain similar feasibility results in the CRS model from weaker assumptions by assuming an honest majority. We establish that this is not the case; in particular, mR-OT is *necessary* even when we assume an honest majority.

II. Broadcast + P2P. We next investigate how the above landscape changes when we use $\mathcal{P2P}$ channels together with $\mathcal{BC}$. Recent works have already shown that SH, SA, UA and FS-GoD are achievable in this model. Our contribution here is in providing a more complete picture, both with regards to best-achievable security and the necessary computational assumptions.

1. **Identifiable Abort.** In light of the impossibility of M-GoD (as well as fairness), we investigate the feasibility of the "next best" security notion, namely, IA for which no prior results are known in the two-round setting (without trusted setup).

   In Section 5.1, we show that IA is *impossible* to achieve for general honest majority even in the $\mathcal{BC}+\mathcal{P2P}$ model.[7] This separates it from UA for which positive results are known in this model [1, 31, 2, 4].

[6] These works in fact rely on mR-OT in the CRS model with universally composable security [10].

[7] In the weaker $\mathcal{P2P}$ only model, honest-majority protocols with IA security are known to be impossible even if we allow for arbitrary rounds [13].

	SH	SA	UA	IA	FS-GoD	
	$t < n/2$			$t < n/2$	$t < n/3$	$t < n/2$
$\mathcal{BC}$	sh-OT [19, 9] ▼ Thm 4.1	✗ Cor 4.1			✗ [25]	
$\mathcal{P2P}$	OWF/IT [1, 3, 18, 2, 4]	OWF/IT [2, 4]	✗ [31]		OWF/IT Cor 6.1	sh-OT [1] ▼ Thm 6.1
$\mathcal{BC} + \mathcal{P2P}$	OWF/IT [1, 3, 18, 2, 4]			✗ Thm 5.1		
$\mathcal{BC} + \mathcal{PKI}$	PKE [1, 3, 18, 2, 4]			PKE+m-NIZK Cor 5.1	PKE [1]	

Figure 2: Feasibility of two-round honest-majority MPC. The symbol ✗ denotes impossibility and ▼ denotes necessity of an assumption.

2. **Fail-Stop Guaranteed Output Delivery.** On the one hand, FS-GoD is known to be impossible in two rounds in the $\mathcal{BC}$ only model [25] due to implications to general-purpose program obfuscation [7]. On the other hand, it was recently shown to be achievable in the $\mathcal{P2P}$ only model based on sh-OT [1] for any $t < n/2$. A natural question is whether it is possible to base it on weaker assumptions, possibly in the stronger $\mathcal{BC} + \mathcal{P2P}$ model. We find that the answer is mixed:

   - For $n/3 \leq t < n/2$, in Section 6.1, we show that sh-OT is *necessary* for FS-GoD in the $\mathcal{BC} + \mathcal{P2P}$ model.
   - For $t < n/3$, in Section 6.2, we observe that FS-GoD can be easily achieved for general circuits based on only OWFs (and for $\mathbf{NC}^1$ circuits, with IT security) in the $\mathcal{P2P}$ only model.

III. P2P vs PKI. Next, we consider the case where the protocol uses a bare $\mathcal{PKI}$ setup (instead of $\mathcal{P2P}$ channels) together with $\mathcal{BC}$. It is natural to expect that all results in the $\mathcal{BC} + \mathcal{P2P}$ model should be exportable to the $\mathcal{BC} + \mathcal{PKI}$ model since $\mathcal{P2P}$ channels can be emulated over $\mathcal{BC}$ via public-key encryption (PKE). We show that "real" $\mathcal{P2P}$ channels are *not* equivalent to $\mathcal{P2P}$ channels emulated over $\mathcal{BC}$ using $\mathcal{PKI}$:

- In Section 5.2, we show that the impossibility result for IA in the $\mathcal{BC} + \mathcal{P2P}$ model no longer holds in the $\mathcal{BC} + \mathcal{PKI}$ model. Specifically, we provide a transformation from any UA-secure two-round protocol to a IA-secure protocol for any $t < n/2$ based on PKE and *multi*-CRS non-interactive zero-knowledge (m-NIZK) [26].[8]

The above result implies that a $\mathcal{PKI}$ setup is *strictly stronger* than $\mathcal{P2P}$ channels. The main insight underlying this separation is that misbehavior by a party over a $\mathcal{P2P}$ channel emulated via

[8] m-NIZK proof systems for NP are known based on Zaps [15] (which in turn can be constructed from various standard assumptions, including trapdoor permutations and assumptions on bilinear maps) or learning with errors [6].

$\mathcal{PKI}$ can be "proven" to others in two rounds, but not when the misbehavior is over a "real" $\mathcal{P}2\mathcal{P}$ channel.

IV. P2P Only. The remaining case is when the parties have access to *only* P2P channels. A recent work of [31] established SA as the strongest achievable notion of security against malicious adversaries in this setting, and a matching positive result for computing general circuits was given by [2, 4] based on OWFs (and for $\mathbf{NC}^1$ circuits, with IT security). For FS-GoD, [1] showed that it is achievable for $t < n/2$ based on sh-OT. We have further sharpened this result by showing that for $t < n/3$, OWFs suffice, and for $n/3 \leq t < n/2$, sh-OT is necessary. Put together, these results complete the picture for the $\mathcal{P}2\mathcal{P}$ only model as well.

1.2 Related Work

As discussed earlier, we primarily focus on the plain model, i.e., a setting where the parties do not have access to any trusted setup. Note that this is in sharp contrast to a setting where the parties have access to a trusted setup (e.g., a CRS). Indeed, while we show that any form of malicious security is impossible in the $\mathcal{BC}$ only setting without any setup, feasibility of SA, UA and IA in the $\mathcal{BC}$ only setting is well known in the CRS model [19, 9, 12]. Moreover, if the parties have access to *both* a $\mathcal{PKI}$ and CRS, then M-GoD is also achievable in two-rounds [25].

In a concurrent and independent work, Damgård et al. [14], explore a related (but different) question in the setting where parties have access to both a *$\mathcal{PKI}$ and a trusted CRS setup*. As mentioned above, in this setting, M-GoD is achievable in two-rounds when a broadcast channel is available in both rounds. Therefore, they investigate the necessity of a broadcast channel in each individual round of a two-round MPC protocol. In contrast, we consider a setting *without any trusted setup* (i.e., either the plain model or the plain model augmented with a bare (i.e., untrusted) $\mathcal{PKI}$). Hence, their results are incomparable to ours.

2 Technical Overview

In this section, we discuss the main ideas underlying our results.

2.1 Lower Bounds in the $\mathcal{BC}$ only Model

In the $\mathcal{BC}$ only model, we show that 2-round honest-majority MPC implies the existence of 2-message oblivious transfer (sh-OT or mR-OT, depending on the level of security of the honest-majority MPC). This is in sharp contrast to the general setting, where without any restriction on the number of rounds or communication channels, honest-majority MPC (even with M-GoD security) is possible *unconditionally*.

To understand the source of this requirement, we consider an n-party variant of OT, denoted as $\mathcal{F}_{n\text{-OT}}$, in which there is a sender, a receiver, and $(n-2)$ "helper parties" (who do not have any inputs or outputs). Interestingly, by relying on $\mathcal{P}2\mathcal{P}$ channels, $\mathcal{F}_{n\text{-OT}}$ can be securely realized (with SH security) unconditionally in two rounds.[9] Further, even if we only use $\mathcal{BC}$ channels but allow for at least three rounds, then public-key encryption (rather than OT) is sufficient, by using the first round to send public keys for establishing private channels for the next two rounds. Thus the necessity of OT must stem from the combination of the two-round constraint and the restriction to $\mathcal{BC}$.

[9] Specifically, it can be implemented as OLE over a large field, using a protocol in which each helper party receives degree t Shamir shares of a and x from sender and receiver respectively, and degree $2t$ shares of b from sender, and sends degree $2t$ shares of $ax + b$ to the receiver.

Our strategy is to build a two-message (two-party) OT protocol from an honest-majority two-round protocol Π for $\mathcal{F}_{n\text{-}\mathsf{OT}}$, in the $\mathcal{BC}$ model. In this section, we only consider $n = 3$ (with the sender, the receiver and a single helper party), so that honest-majority translates to corruption of at most one party. The proof easily generalizes to an arbitrary number of parties and is shown in the technical section.

As a first attempt, one may hope that the helper party – who has no input and receives only publicly visible messages – can be implemented by either party (thus collapsing to a 2-party protocol), and the protocol will remain secure. Unfortunately, this is not true. For instance, suppose the receiver and the helper also broadcast a public key for encryption in the first round, and the sender's second round message also includes a 2-out-of-2 secret-sharing of its inputs, each share encrypted using one of these keys. In such a case, corrupting at most one party in Π does not reveal these inputs, but if the helper is implemented by the receiver, then the protocol is no longer secure. This attack is symmetric, and prevents clubbing the helper with either the sender or the receiver. On the other hand, the sender and the receiver *jointly* implementing the helper in a secure manner is not an option, as it leaves us with a harder problem than we set out to solve.

The key to resolving this conundrum is to *break the symmetry* between the receiver and the sender. We observe that Π can first be modified so that the receiver does not send any message in the second round. This is a legitimate modification, since the last round messages are only used for output generation, and the receiver is the only party with an output in the protocol. This modification to Π prevents the attack mentioned above *when the helper is implemented by the sender.* We go on to show that this in fact, leads to a protocol that is secure against all passive attacks. Clearly, security against corruption of the receiver follows from the same in Π. Security against corruption of the sender follows, informally, from the fact that even in Π, by corrupting the sender alone, the adversary can obtain the same view as in the transformed 2-party protocol, by internally simulating the helper party. Specifically, *since the honest receiver never responds to the helper's messages*, the internally simulated helper's view can be combined with the independently generated message of the receiver to obtain a valid simulation.

Thus the transformed protocol is a semi-honest secure 2-party OT protocol (i.e., sh-OT). Further, it can be cast as a *two message* protocol:

- **Round 1:** The first message from the receiver consists of its first round message in Π.
- **Round 2:** The second message from the sender consists of both first and second round messages from the sender and the helper in Π.

Note that we are able to "postpone" the first round messages of the sender and helper in Π to the second message of OT because an honest receiver is *non-rushing*; i.e., its first round message does not depend on the messages of the other parties.

This argument partly extends to the case when Π is secure against active corruptions. In this case, the transformed protocol will have the same security as Π against the corruption of the receiver, but only security against semi-honest corruption of the sender. When Π is secure w.r.t. straightline simulation (which is standard for security with honest majority) this yields a 2-party, 2-round OT protocol that is secure against passive corruption of senders, and active corruption of receivers, with straightline simulation in the latter case. We term such a protocol an mR-OT protocol.

These arguments readily extend to all $n \geq 3$. Thus two-round n-party honest-majority MPC over $\mathcal{BC}$ channels implies two-round sh-OT or two-round mR-OT, depending on the security level of the honest-majority protocol. In the latter case, we obtain an impossibility result for MPC in the plain model, by proving the impossibility of two-round mR-OT protocol (in the plain model), similar to the impossibility of UC security in the plain model. We give a formal proof in Section 4.

2.2 Identifiable Abort

2.2.1 Impossibility Result in $\mathcal{BC} + \mathcal{P}2\mathcal{P}$ Model

We next describe our ideas for proving the impossibility of 2-round honest-majority MPC with IA security in the $\mathcal{BC} + \mathcal{P}2\mathcal{P}$ model, without any setup. We focus on the case of $n = 3$ parties and $t = 1$ corruption.

From our first lower bound, we know that security with IA is impossible in two-rounds in the $\mathcal{BC}$ only model. In general, access to $\mathcal{P}2\mathcal{P}$ channels can often help in overcoming such impossibilities. Indeed, recent two-round protocols [1, 4, 2] that achieve SA/UA security crucially rely on the use of $\mathcal{P}2\mathcal{P}$ channels. An obvious advantage of using $\mathcal{P}2\mathcal{P}$ channels in the honest majority setting is "easy" (straight-line) extraction of the adversary's inputs during simulation. However, there is also a *potential disadvantage*: an adversary may use $\mathcal{P}2\mathcal{P}$ channels to create inconsistent views amongst the honest parties. For example, it may send honestly computed messages to one honest party, but not to the other.

While such attacks can usually be handled (by requiring the honest parties to output $\perp$ by default in case of any conflict or confusion) when we only require SA or UA security, it becomes a challenge in achieving IA security. Recall that in IA, if the honest parties output $\perp$, they *must* also be able to identify a corrupt party. In a *two round* protocol, even if an honest party – who does not receive a "valid" message in the first round from the adversary – tries to complain to another honest party in the second round, the latter party is left in a dilemma about whether the complaint is legitimate or fabricated (to frame the other party). As a result, it is unable to decide who amongst the other two parties is actually corrupt. This observation forms the basis of our impossibility result.

Consider a 3-party functionality $\mathcal{F}$ that takes inputs $b \in \{0, 1\}$ from P_2 and (x_0, x_1) from P_3 and outputs x_b to P_1. That is, $\mathcal{F}(\perp, b, (x_0, x_1)) = (x_b, \perp, \perp)$. Consider an adversary who corrupts P_2 in the following manner: it behaves honestly, except that it does not send any protocol specified private channel message to P_1 (i.e., simply drops them).[10] We argue that no protocol can achieve IA security against such an attack.

In particular, we argue that in this case, the honest parties can neither output $\perp$ nor a non-$\perp$ value. As discussed earlier, if the honest parties output $\perp$, they must also be able to identify the corrupt party. However, P_3's view in this case is indistinguishable from another execution where a corrupt P_1 falsely accuses an honest P_2 of not sending private channel messages. It is easy to see that this inherent "conflict" for P_3 about who amongst P_1 and P_2 is the corrupt party is impossible to resolve. Hence, the output of the honest parties cannot be $\perp$.

This leaves the possibility of the output being non-$\perp$. Consider P_2 using an input b in the protocol execution. In case the output of the honest parties is a non-$\perp$ value, there are two possible outcomes, corresponding to what a simulator extracts as P_2's input: (1) the simulator extracts b with probability (almost) 1 or (2) with at least a non-negligible probability, it extracts $1 - b$.

- In the first case, note that the simulator's view of P_2's messages only involves messages visible to P_3. Then, since the simulator is a straight-line simulator, and the protocol is in the plain

[10] If the protocol does not require any P2P message from P_2 to P_1, then the corrupted P_2 is simply behaving honestly since there is no message to be dropped. In this case, the protocol must result in a not-$\perp$ output. This case is addressed below.

model, a corrupt P_3 can violate privacy by running the same simulator to extract an *honest* P_2's input. Hence this case is not possible.

- In the second case, consider another instance where P_1 is corrupt, while P_2 and P_3 are honest. Consider an execution where P_1 follows the protocol honestly and learns the output x_b. Later it launches an "offline reset attack," by recomputing its second round messages pretending that it did not receive a message from party P_2 in the first round. Upon recomputing the output using this alternate view (where P_2's private messages were not received), it learns, with non-negligible probability, x_{1-b}. Hence, P_1 can distinguish between the case $x_0 = x_1$ and $x_0 \neq x_1$ with a non-negligible advantage, thereby violating P_3's privacy. Hence, this case is also not possible.

We present a formal proof in Section 5.1.

2.2.2 Positive Result in $\mathcal{BC} + \mathcal{PKI}$ Model

We now discuss how replacing $\mathcal{P}2\mathcal{P}$ channels with a bare PKI can help in overcoming the above impossibility result. Specifically, we describe the main ideas underlying our compiler from a two-round UA-secure protocol in the $\mathcal{BC}+\mathcal{P}2\mathcal{P}$ model to a two-round IA-secure protocol in the $\mathcal{BC}+\mathcal{PKI}$ model.

Given a two-round protocol Π that achieves security with UA in the $\mathcal{BC} + \mathcal{P}2\mathcal{P}$ model, a natural idea to strengthen its security to IA (in the $\mathcal{BC} + \mathcal{PKI}$ model) is to simply require each party to prove honest behavior using the standard "commit and prove" approach: the parties encrypt their private channel messages under the public-keys of the recipient parties, broadcast them in the first round and attach a proof of having computed all of these messages honestly in each round. If a party cheats, then its proof will fail verification, and *all* the honest parties will be able to identify that corrupt party. While this idea can be easily implemented using NIZKs, it would result in a protocol in the CRS model.

Since we are in the honest majority setting, we can attempt to replace standard NIZKs with *multi*-CRS NIZKs (m-NIZKs)[26], where the setup consists of multiple CRS strings (as opposed to a single CRS) and soundness holds as long as a majority of the CRS are honestly generated. In our setting, the CRS strings can be generated by the parties in the first round of the protocol and the honest majority assumption implies that a majority of the CRS are computed honestly. Using m-NIZKs, the parties can still prove honest behavior in the second round of the protocol. However, a proof of honest behavior in the first round can no longer be sent in the first round itself (since the CRS strings are not known at that point); instead it can only be sent (belatedly) in the second round. In this case, we need to ensure that it is not "too late" for the honest parties to detect and identify a cheating party.

We implement this idea in the following manner. If the parties are able to compute their second round messages – given the first round messages from all the other parties – they give a single proof in the second round to prove that they computed all their (first and second round) messages honestly.

In case a corrupt party does not compute and encrypt its first round private channel messages honestly, there are two possibilities: (1) the honest recipient of the malformed private message is able to detect that the message is not "well-formed" (e.g. if the message is an empty string or it does not satisfy the syntax specified by underlying protocol, etc.) and is unable to use this message to compute its second round message, or (2) the honest recipient does not detect any issues with the message and is able to compute its second round message as per the specification of the underlying protocol. We handle these two scenarios differently.

In the first case, the recipient party simply reveals the decrypted malformed message to all other parties in the second round and gives a proof to convince them that its (respective) public key was honestly generated and that the corrupt party did indeed send them an encryption of this malformed message. Given the decrypted message, the remaining parties can perform the same (public) verification as the recipient party to determine whether or not the message is well-formed and identify the corrupt party. In the second case, we will rely on the soundness of the proof given by the corrupt party. In case the corrupt party did not encrypt its first round private channel messages honestly, it will not be able to give a convincing proof in the second round, and will be easily identified. We give a formal description of this construction in Section 5.2.

2.3 Necessity of sh-OT for FS-GoD in the $\mathcal{BC} + \mathcal{P}2\mathcal{P}$ Model

In the $\mathcal{BC} + \mathcal{P}2\mathcal{P}$ model, we show that 2-round honest-majority that achieves FS-GoD security implies the existence of 2-message sh-OT. This implication holds for $n/3 \leq t < n/2$; for $t < n/3$, we describe a simple FS-GoD protocol in the technical sections based on weaker assumptions.

Recall that in the transformation from a two-round $\mathcal{BC}$ only protocol for $\mathcal{F}_{\mathsf{3\text{-}OT}}$ to a secure protocol for OT (discussed in Section 2.1), the sender implements the helper party. Security against a semi-honest sender follows from the fact that in the $\mathcal{BC}$ only model, the view of an adversary who corrupts the sender and the helper in the transformed protocol is no different from the view of an adversary who only corrupts the sender in the original protocol. It is easy to see that this argument *fails* (even in the semi-honest setting) when the protocol additionally uses $\mathcal{P}2\mathcal{P}$ channels. Consider, for example, the case where the receiver is required to send a private message to the helper in the first round. An adversary who corrupts both the sender and the helper now gets this additional information, which it does not get by corrupting the sender alone. Indeed, since two-round protocols [1, 3, 18, 4, 2] that achieve security with SA or UA in the $\mathcal{BC} + \mathcal{P}2\mathcal{P}$ model are already known, we know that the above approach *must* fail.

Our key insight is that if the two-round protocol achieves FS-GoD security, then it means that some private channel messages are "redundant," and can be removed if one only cares about security against semi-honest adversaries. This observation allows us to start with with a "truncated" version of the underlying FS-GoD protocol (which only achieves SH security) and then use a similar strategy as in Section 2.1 to construct two-message sh-OT. We first focus on the setting with $n = 3$ parties and $t = 1$ corruption. Later we discuss how this argument can be extended for arbitrary n and $n/3 \leq t < n/2$.

As earlier, we consider the functionality $\mathcal{F}_{\mathsf{3\text{-}OT}}$ involving a sender, a receiver and a helper party. Let Π be a 3-party protocol for this functionality with FS-GoD security. Note that FS-GoD security implies that even if the helper does *not* send its second round message, the protocol must still remain (at the very least) semi-honest secure. Furthermore, if the helper is not required to send any messages in the second round, the sender and receiver do not need to send any messages to the helper in the first round (except the broadcast channel messages, which are received by everyone). Combining these observations with the observation from Section 2.1 that the receiver (by virtue of being the only output party) does not need to send a message in the second round, and that the sender and helper can send all their messages in the second round, we obtain the following two-message protocol:

- **Round 1:** The receiver computes and sends its first round broadcast message and its private message for the sender.

- **Round 2:** The sender computes and sends its first and second round broadcast messages and its private channel messages for the receiver. It also computes and sends the first round broadcast message and the private channel message of the helper for the receiver.

Security against a semi-honest sender and receiver in the transformed OT protocol can be argued similarly as before, although we need to be slightly more careful in handling private channel messages of each party in the underlying three-party protocol.

The above idea can be generalized to n parties and $n/3 \leq t < n/2$ corruptions for the n-party functionality $\mathcal{F}_{n\text{-}\mathsf{OT}}$ (described earlier). In this case, the first $2t$ parties are emulated by the sender and the remaining $n-2t$ are emulated by the receiver. Since $n/3 \leq t < n/2$, we know that $n-2t \leq t$. Security against a semi-honest receiver in this case follows exactly as before. For security against a semi-honest sender, we rely on the fact that since t out of the $2t$ parties emulated by the sender do not send second round messages, the receiver parties do not need to send them private channel messages in the first round. We can now rely on the semi-honest security of (the truncated version of) Π to show that an adversary who corrupts the sender does not gain any more advantage over an adversary who corrupts the first t parties in Π. We defer further details to Section 6.1.

3 Preliminaries

Throughout the paper, we use λ to denote the security parameter. We recall some standard cryptographic definitions in this section. Apart from the primitives defined in this section, we also use some other standard building blocks like non-interactive commitments and public key encryption. We omit their definitions here.

3.1 Oblivious Transfer (OT)

In this paper, we consider the standard notion of 1-out-of-2 oblivious transfer [16]; where one party (the sender) has inputs (m_0, m_1) in some domain (say $\{0,1\}^*$), and another party (the receiver) has a choice bit $b \in \{0,1\}$. At the end, the receiver should learn m_b and nothing more while the sender should learn nothing about b.

We consider two variants of this OT protocol, a *semi-honest* version called $\mathsf{sh\text{-}OT}$ and one that is secure against a *malicious receiver* called $\mathsf{mR\text{-}OT}$. For $\mathsf{mR\text{-}OT}$, we require an efficient straight-line simulator for a maliciously corrupt receiver.

We define the syntax and the security guarantees of a two-message OT protocol in the plain model. The definition can be naturally extended to the CRS model.

Definition 3.1 (2 Message OT). *A two-message oblivious transfer between a receiver R and a sender S is defined by a tuple of 3 PPT algorithms $(\mathsf{OT}_R, \mathsf{OT}_S, \mathsf{OT}_{\mathsf{out}})$. Let λ be the security parameter. The receiver computes msg_R, ρ as the evaluation of $\mathsf{OT}_R(1^\lambda, b)$, where $b \in \{0,1\}$ is the receiver's input. The receiver sends msg_R to the sender. The sender computes msg_S as the evaluation of $OT_S(1^\lambda, \mathsf{msg}_R, (m_0, m_1))$, where $m_0, m_1 \in \{0,1\}^*$ are the sender's input. The sender sends msg_S to the receiver. Finally the receiver computes m_b by evaluating $OT_{\mathsf{out}}(\rho, \mathsf{msg}_R, \mathsf{msg}_S)$.*

A sh-OT protocol satisfies correctness, security against semi-honest receiver and semi-honest sender*, while a mR-OT satisfies* correctness, security against semi-honest sender and malicious receiver*, which are defined as follows:*

- ***Correctness**: For each $m_0, m_1 \in \{0,1\}^*$, $b \in \{0,1\}$, it holds that*

$$\Pr\left[\begin{array}{r} (\rho, \mathsf{msg}_R) \leftarrow \mathsf{OT}_R\left(1^\lambda, b\right) \\ \mathsf{msg}_S \leftarrow OT_S\left(1^\lambda, \mathsf{msg}_R, (m_0, m_1)\right) \end{array} \middle| \ \mathsf{OT}_{\mathsf{out}}\left(\rho, \mathsf{msg}_R, \mathsf{msg}_S\right) = m_b \right] = 1,$$

- ***Security against Semi-Honest Sender**: It holds that,*

$$\left\{(\mathsf{msg}_R^0, \rho^0) \leftarrow \mathsf{OT}_R\left(1^\lambda, 0\right) \;\middle|\; \mathsf{msg}_R^0\right\} \approx_c \left\{(\mathsf{msg}_R^1, \rho^1) \leftarrow \mathsf{OT}_R\left(1^\lambda, 1\right) \mid \mathsf{msg}_R^1\right\}$$

- ***Security against Semi-Honest Receiver**: it holds that for each $b \in \{0,1\}$, $m_0, m_1, m_0', m_1' \in \{0,1\}^*$, and $m_b = m_b'$,*

$$\left\{\mathsf{OT}_S\left(1^\lambda, \mathsf{msg}_R, (m_0, m_1)\right)\right\} \approx_c \left\{\mathsf{OT}_S\left(1^\lambda, \mathsf{msg}_R, (m_0', m_1')\right)\right\}$$

  where $(\mathsf{msg}_R, \rho) \leftarrow OT_R(1^\lambda, b)$.

- ***Security against a Malicious Receiver**: For every PPT adversary $\mathcal{A}$, there exists a PPT simulator $\mathcal{S}_R = (\mathcal{S}_R^1, \mathcal{S}_R^2)$ for any choice of $m_0, m_1 \in \{0,1\}^*$ such that the following holds*

$$\left| \Pr\left[\mathsf{IDEAL}_{\mathcal{S}_R, \mathcal{F}_{\mathsf{OT}}}(1^\lambda, m_0, m_1) = 1\right] - \Pr\left[\mathsf{REAL}_{\mathcal{A}, \mathsf{OT}}(1^\lambda, m_0, m_1) = 1\right] \right| \leq \frac{1}{2} + \mathsf{negl}(\lambda).$$

  Where experiments $\mathsf{IDEAL}_{\mathcal{S}_R, \mathcal{F}_{\mathsf{OT}}}$ and $\mathsf{REAL}_{\mathcal{A}, \mathsf{OT}}$ are defined as follows:

```
Exp $\boxed{\mathsf{IDEAL}_{\mathcal{S}_R, \mathcal{F}_{\mathsf{OT}}}(1^\lambda, m_0, m_1) :}$
$\mathsf{msg}_R \leftarrow \mathcal{A}\left(1^\lambda\right)$
    $b \leftarrow \mathcal{S}_R^1(1^\lambda, \mathsf{msg}_R)$
    $m_b \leftarrow \mathcal{F}_{\mathsf{OT}}(m_0, m_1, b)$
$\mathsf{msg}_S \leftarrow \mathcal{S}_R^2(1^\lambda, m_b, \mathsf{msg}_R)$
Out $\mathcal{A}(\mathsf{msg}_S)$
```

```
Exp $\boxed{\mathsf{REAL}_{\mathcal{A}, \mathsf{OT}}(1^\lambda, m_0, m_1) :}$
$\mathsf{msg}_R \leftarrow \mathcal{A}\left(1^\lambda\right)$
$\mathsf{msg}_S \leftarrow \mathsf{OT}_S\left(1^\lambda, \mathsf{msg}_R, (m_0, m_b)\right)$
Out $\mathcal{A}(\mathsf{msg}_S)$
```

3.2 Secure Multiparty Computation

A secure multi-party computation protocol (MPC) is a protocol executed by n parties $\mathcal{P} = \{P_1, \cdots, P_n\}$ for a functionality $\mathcal{F}$. We allow for parties to exchange messages simultaneously. In every round, every party is allowed to exchange messages with other parties using different communication channels, depending on the model. A protocol is said to have *k rounds* if it proceeds in k distinct and interactive rounds.

3.2.1 Adversarial Behavior

One of the primary goals in MPC is to protect the honest parties against dishonest behavior of 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 the case where the adversary can only control a minority of the parties in the protocol. We discuss the following adversarial models in detail:

1. **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.

2. **Fail-Stop Adversaries:** A non-rushing 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 abort at any time in the protocol. The decision to abort or not may depend on its view. Note that a fail-stop adversary does not selectively abort over private channels, i,e, it either sends all private messages in a round or does not send any message in that particular round.

3. **Malicious Adversaries:** A malicious adversary can deviate from the protocol and instruct the corrupted parties to follow any arbitrary strategy.

We provide the basic definitions for secure multiparty computation according to the real/ideal paradigm [21]. Informally, a protocol is considered secure if whatever an adversary can do in the real execution of protocol, can be done also in an ideal computation, in which an uncorrupted trusted party assists the computation.

3.2.2 Security Definitions

Real World. The real world execution of a protocol $\Pi = (P_1, \ldots, P_n)$ begins by an adversary $\mathcal{A}$ selecting any arbitrary subset of parties $\mathcal{I} \subset [n]$ to corrupt. The parties then engage in an execution of a real n-party protocol Π. Throughout the execution of Π, the adversary $\mathcal{A}$ sends all messages on behalf of the corrupted parties, and may follow an arbitrary polynomial-time strategy. In contrast, the honest parties follow the instructions of Π. At the conclusion of the protocol, each honest party outputs all the outputs it obtained in the computations. Malicious parties may output an arbitrary PPT function of the view of $\mathcal{A}$. This joint execution of Π under $(\mathcal{A}, \mathcal{I})$ in the real model, on input vector $\vec{x} = (x_1, \ldots, x_n)$, auxiliary input z and security parameter λ, denoted by $\mathsf{REAL}_{\Pi,\mathcal{I},\mathcal{A}(z)}\left(1^\lambda, \vec{x}\right)$, is defined as the output vector of $P_1, \ldots, P_n$ and $\mathcal{A}(z)$ resulting from this protocol interaction.

Ideal World. We now present standard definitions of ideal-model computations that are used to define security with `SA`/`UA`, with `IA`, with `FS-GoD`. We start by presenting the ideal-model computation for security with `UA`, where the adversary may abort the computation either before or after it has learned the output; other ideal-model computations are defined either by allowing the adversary to selectively abort to some parties but not to others or by restricting the power of the adversary either by forcing the adversary to identify a corrupted party in case of abort, or no abort (guaranteed output delivery).

Ideal Computation with `UA`. An ideal computation with `UA` of an n-party functionality $\mathcal{F}$ on input $\vec{x} = (x_1, \ldots, x_n)$ for parties $(P_1, \ldots, P_n)$ in the presence of an ideal-model adversary $\mathcal{A}$ controlling the parties indexed by $\mathcal{I} \subset [n]$, proceeds via the following steps.

Sending inputs to trusted party: For each $i \notin \mathcal{I}$, P_i sends its input x_i to the trusted party. If $i \in \mathcal{I}$, the adversary may send to the trusted party any arbitrary input for the corrupted party P_i. Let x'_i be the value actually sent as the i^{th} party's input.

Early abort: The adversary $\mathcal{A}$ can abort the computation by sending an abort message to the trusted party. In case of such an abort, the trusted party sends $\perp$ to all parties and halts.

Trusted party answers adversary: The trusted party computes $(y_1, \ldots, y_n) = \mathcal{F}(x'_1, \ldots, x'_n)$ and sends y_i to party P_i for every $i \in \mathcal{I}$.

Late abort: The adversary $\mathcal{A}$ can abort the computation (after seeing the outputs of corrupted parties) by sending an abort message to the trusted party. In case of such abort, the trusted party sends $\perp$ to all honest parties and halts. Otherwise, the adversary sends a $\mathsf{continue}$ message to the trusted party.

Trusted party answers remaining parties: The trusted party sends y_i to P_i for every $i \notin \mathcal{I}$.

Outputs: Honest parties always output the message received from the trusted party and the corrupted parties output nothing. The adversary $\mathcal{A}$ outputs an arbitrary function of the initial inputs x_i s.t. $i \in \mathcal{I}$, the messages received by the corrupted parties from the trusted party and its auxiliary input.

We now define the following variants of this ideal computation:

- **Ideal computation with $\texttt{SA}$.** This ideal model proceeds as in Definition 3.2, with the exception that during *late abort*, the adversary is allowed to choose which honest parties get the output and which ones get abort.

- **Ideal computation with $\texttt{IA}$.** This ideal model proceeds as in Definition 3.2, with the exception that in order to abort the computation, the adversary chooses an index of a corrupted party $i^* \in \mathcal{I}$ and sends (abort, i^*) to the trusted party. In this case the trusted party responds with $(\perp, i^*)$ to all parties.

- **Ideal computation with $\texttt{FS-GoD}$.** This ideal model proceeds as in Definition 3.2, with the exceptions that the adversary can only send $x'_i \in \{x_i, \perp\}$ (for each $i \in \mathcal{I}$) as input for the corrupted parties and is additionally not allowed to send abort to the trusted party at any point.

Definition 3.2 (Ideal-model computation)**.** *Let* $\mathsf{type} \in \{\texttt{UA}, \texttt{SA}, \texttt{IA}, \texttt{FS-GoD}\}$ *and* $\mathcal{F} : (\{0,1\}^*)^n \to (\{0,1\}^*)^n$ *be an n-party functionality. let $\mathcal{I} \subset [n]$ be the set of indices of the corrupted parties, and let λ be the security parameter. Then, the joint execution of f under $(\mathcal{A}, \mathcal{I})$ in the ideal model, on input vector $\vec{x} = (x_1, \dots, x_n)$, auxiliary input z to $\mathcal{A}$ and security parameter λ, denoted* $\mathsf{IDEAL}^{\mathsf{type}}_{\mathcal{F},\mathcal{I},\mathcal{A}(z)}(1^\lambda, \vec{x})$*, is defined as the output vector of $P_1, \dots, P_n$ and $\mathcal{A}$ resulting from the above described ideal process.*

Security Having defined the real and ideal models, we can now define security of protocols according to the real/ideal paradigm.

Definition 3.3. *Let* $\mathsf{type} \in \{\texttt{UA}, \texttt{SA}, \texttt{IA}, \texttt{FS-GoD}\}$*. Let $\mathcal{F} : (\{0,1\}^*)^n \to (\{0,1\}^*)^n$ be an n-party functionality and let Π be a probabilistic polynomial-time protocol computing $\mathcal{F}$. The protocol Π t-securely computes f with "*type*", if for every probabilistic polynomial-time real-model adversary $\mathcal{A}$, there exists a probabilistic polynomial-time simulator $\mathcal{S}$ for the ideal model, such that for every $\mathcal{I} \subset [n]$ of size at most t, it holds that*

$$\left\{\mathsf{REAL}_{\Pi,\mathcal{I},\mathcal{A}(z)}\left(1^\lambda, \vec{x}\right)\right\}_{(\vec{x},z)\in(\{0,1\}^*)^{n+1},\lambda\in\mathbb{N}} \approx_c \left\{\mathsf{IDEAL}^{\mathsf{type}}_{\mathcal{F},\mathcal{I},\mathcal{S}(z)}(1^\lambda, \vec{x})\right\}_{(\vec{x},z)\in(\{0,1\}^*)^{n+1},\lambda\in\mathbb{N}}$$

We also consider protocols where the adversary $\mathcal{A}$ is allowed to be unbounded and the running time of $\mathcal{S}$ is polynomial in the running time of $\mathcal{A}$. We say that such a protocol is statistically (or perfectly resp.) secure if the above distributions are statistically (or perfectly resp.) indistinguishable.

For all our lower bounds and positive results, we only work with MPC protocols that are simulatable by (efficient) straight-line simulators. We note that this is standard in the honest majority MPC literature, since all known honest majority protocols admit straight-line simulation.

3.3 Multi-CRS Non-Interactive Zero Knowledge (m-NIZK)

We use the definition from [6], which is adapted from [26]. Let R be an efficiently computable binary relation and L an NP-language of statements x such that $(x, w) \in R$ for some witness w.

Definition 3.4 (Multi-CRS NIZK). *A multi-CRS NIZK for a language L is a tuple of PPT algorithms $\mathsf{m\text{-}NIZK} = (\mathsf{m\text{-}NIZK.Gen}, \mathsf{m\text{-}NIZK.Prove}, \mathsf{m\text{-}NIZK.Verify})$ satisfying the following specifications:*

- $\mathsf{m\text{-}NIZK.Gen}(1^\lambda)$*: It takes as input the security parameter λ and outputs a uniformly random string* crs.
- $\mathsf{m\text{-}NIZK.Prove}(\mathsf{crs}, x, w)$*: It takes as input a set of n random strings $\overrightarrow{\mathsf{crs}}$, a statement x, and a witness w and outputs a* proof.
- $\mathsf{m\text{-}NIZK.Verify}(\overrightarrow{\mathsf{crs}}, x, \mathsf{proof})$*: It takes as input a set of n random strings $\overrightarrow{\mathsf{crs}}$, a statement x, and a* proof*. It outputs 1 if it accepts the* proof *and 0 if it rejects it.*

We require that the algorithms satisfy the following properties for all non uniform PPT adversaries $\mathcal{A}$:

- ***Perfect Completeness.***

$$\Pr\left[\begin{array}{r} s = \emptyset; (\overrightarrow{\mathsf{crs}}, x, w) \leftarrow \mathcal{A}^{\mathsf{m\text{-}NIZK.Gen}} \\ \mathsf{proof} \leftarrow \mathsf{m\text{-}NIZK.Prove}(\overrightarrow{\mathsf{crs}}, x, w) \end{array} \middle| \begin{array}{r} \mathsf{m\text{-}NIZK.Verify}(\overrightarrow{\mathsf{crs}}, x, \mathsf{proof}) = 0 \\ \textit{and } (x, w) \in R \end{array}\right] = 0,$$

where $\mathsf{m\text{-}NIZK.Gen}$ *is an oracle that when queried, outputs* $\mathsf{crs} \leftarrow \mathsf{m\text{-}NIZK.Gen}(1^\lambda)$ *and sets* $\overrightarrow{\mathsf{crs}} = \overrightarrow{\mathsf{crs}} \cup \mathsf{crs}$*. Note that this says that even if the adversary arbitrarily picks all the random strings, perfect completeness still holds.*

- ***Soundness.***

$$\Pr\left[S = \emptyset; (\overrightarrow{\mathsf{crs}}, x, \mathsf{proof}) \leftarrow \mathcal{A}^{\mathsf{m\text{-}NIZK.Gen}} \middle| \begin{array}{r} \mathsf{m\text{-}NIZK.Verify}(\overrightarrow{\mathsf{crs}}, x, \mathsf{proof}) = 0 \textit{ and} \\ x \notin L \textit{ and } |\overrightarrow{\mathsf{crs}} \cap S| > n/2 \end{array}\right] \leq \mathsf{negl}(\lambda)$$

where $\mathsf{m\text{-}NIZK.Gen}$ *is an oracle that when queried, outputs* $\mathsf{crs} \leftarrow \mathsf{m\text{-}NIZK.Gen}(1^\lambda)$ *and sets* $S = S \cup \mathsf{crs}_q$*. Note that this says that as long as at least half of the random strings are honestly generated, the adversary cannot forge a proof except with negligible probability.*

- ***Zero-Knowledge.*** *There exist PPT algorithms $\mathcal{S}_{\mathsf{Gen}}$, $\mathcal{S}_{\mathsf{Prove}}$ such that*

$$\Pr[\mathsf{crs} \leftarrow \mathsf{m\text{-}NIZK.Gen}(1^\lambda) \mid \mathcal{A}(\mathsf{crs}) = 1] \approx \Pr[(\mathsf{crs}, \tau) \leftarrow \mathcal{S}_{\mathsf{Gen}}(1^\lambda) : \mathcal{A}(\mathsf{crs}) = 1]$$

and

$$\Pr\left[\begin{array}{r} s = \emptyset; (\overrightarrow{\mathsf{crs}}, x, \mathsf{proof}) \leftarrow \mathcal{A}^{\mathcal{S}_{\mathsf{Gen}}} \\ \mathsf{proof} \leftarrow \mathsf{m\text{-}NIZK.Prove}(\overrightarrow{\mathsf{crs}}, x, w) \end{array} \middle| \begin{array}{r} \mathcal{A}(\mathsf{proof}) = 1 \textit{ and } (x, w) \in R \\ \textit{and } |\overrightarrow{\mathsf{crs}} \cap S| > n/2 \end{array}\right]$$

$$\approx \Pr\left[\begin{array}{r} s = \emptyset; (\overrightarrow{\mathsf{crs}}, x, \mathsf{proof}) \leftarrow \mathcal{A}^{\mathcal{S}_{\mathsf{Gen}}} \\ \mathsf{proof} \leftarrow \mathcal{S}_{\mathsf{Prove}}(\overrightarrow{\mathsf{crs}}, x, \overrightarrow{\tau}) \end{array} \middle| \begin{array}{r} \mathcal{A}(\mathsf{proof}) = 1 \textit{ and } (x, w) \in R \\ \textit{and } |\overrightarrow{\mathsf{crs}} \cap S| > n/2 \end{array}\right]$$

where $\overrightarrow{\tau}$ is the set containing all simulation trapdoors τ generated by $\mathcal{S}_{\mathsf{Gen}}$.

4 Broadcast Model

In this section, we investigate the minimal assumptions required to enable two-round honest-majority secure MPC protocols over only a $\mathcal{BC}$ channel. In Section 4.1, we show that any two-round honest majority MPC for general functionalities that achieves either semi-honest security or security against malicious adversaries, over a $\mathcal{BC}$ channel can be transformed into a two-message oblivious transfer protocol. In the semi-honest case, this yields a semi-honest OT protocol ($\mathsf{sh\text{-}OT}$), while in the malicious setting, this yields a malicious receiver OT protocol ($\mathsf{mR\text{-}OT}$). Later in Section 4.2, we show that such a two-round malicious receiver OT is impossible in the plain model, thereby showing that maliciously secure, two-round MPC is impossible in the plain model given only broadcast channels.

4.1 Lower Bound for $t = 1$

We start by formally stating the observation that for functionalities where only a single party receives an output, the output party need not send any messages in the last round.

Observation 1. *Let $\mathcal{F}$ be any n-input functionality and let Π be a secure MPC protocol that computes $\mathcal{F}$, such that only one party P_{out} receives the output of $\mathcal{F}$. Then Π can be transformed into a protocol Π', where the output party does not send any message in the last round. Moreover, Π' achieves the same security as Π in the same communication/setup model.*

Indeed, the above observation holds w.l.o.g. If P_{out} simply drops its last round message, then by virtue of being the only output party, the output of all other parties remains unaffected. While P_{out} can still compute its output by first locally computing its last round message in Π and then running the output reconstruction algorithm of Π on the protocol transcript and this locally computed message. It is easy to see that the security of this modified protocol follows from the security of Π.

Given this observation, we now show that any two-round protocol in the $\mathcal{BC}$ model can be transformed into a two-message OT in the same setting.

Theorem 4.1. *If there exists a 2-round, n-party protocol over $\mathcal{BC}$ channels for general functions, in the plain model, that is secure against $t = 1$ semi-honest corruption, then there exists a 2-message semi-honest OT protocol in the plain model.*

If there exists a 2-round, n-party protocol over $\mathcal{BC}$ channels for general functions, in the plain model, that achieves security with abort ($\mathtt{SA}, \mathtt{UA}, \mathtt{IA}$) against $t = 1$ malicious corruption, then there exists a 2-message malicious receiver OT protocol in the plain model.

Looking ahead, in Section 4.2, we show that two-message $\mathsf{mR\text{-}OT}$ in the plain model is impossible, thereby proving impossibility of $\mathtt{SA}, \mathtt{UA}$ and $\mathtt{IA}$ in the plain model over only $\mathcal{BC}$ channels. We remark that while Theorem 4.1 is stated for the plain model, it will be easy to see that this implication from two-round $\mathcal{BC}$ only protocols to two-message OT also holds in the *CRS model*. As discussed in the Introduction, since $\mathsf{mR\text{-}OT}$ is achievable in two-rounds in the CRS model, this implication complements the two-round protocols based on two-message $\mathsf{mR\text{-}OT}$ for $\mathtt{SA}, \mathtt{UA}$ and $\mathtt{IA}$ from [19, 9, 12] in the CRS model.

The proof of Theorem 4.1 is organised as follows: We first give a common transformation from an n-party protocol Π to a two-message OT protocol. Then in Lemma 4.1, we show that if Π is semi-honest secure, then the resulting OT protocol is also semi-honest secure. Finally, in Lemma 4.2, we show that if Π achieves security with abort ($\mathtt{SA}$,$\mathtt{UA}$,$\mathtt{IA}$) against a malicious adversary, then the resulting OT protocol achieves malicious receiver security.

Proof of Theorem 4.1. Consider the following functionality involving a set of n parties, $\mathcal{P} = \{P_1, \ldots, P_n\}$:

$$\mathcal{F}_{n\text{-}\mathsf{OT}}((m_0, m_1), \{\perp\}_{i \in [n-2]}, b) = (\{\perp\}_{i \in [n-1]}, m_b)$$

where the input of the first party P_1 is $(m_0, m_1) \in \{0,1\}^*$, parties $P_2, \ldots, P_{n-1}$ have no inputs and the input of the last party P_n is a bit $b \in \{0,1\}$. Party P_n is the only output party in this functionality.

Let Π be a protocol for $\mathcal{F}_{n\text{-}\mathsf{OT}}$ that operates over a $\mathcal{BC}$ channel. From Observation 1, we know that any MPC protocol with a single output party can be transformed into one where the output party does not send any message in the last round. In Figure 3, we show how such a protocol (where P_n does not participate in the second round) for $\mathcal{F}_{n\text{-}\mathsf{OT}}$ can be used to design a two-message OT protocol Π_{OT} in the same setup/communication model as Π. We assume Π^r to be the r^{th} round next message function in Π that takes the index of a party P_i among other values as input and outputs msg_i^r, ρ_i^r (internal state). We use $\overrightarrow{\mathsf{msg}}^r$ to denote the set of all the messages sent by the parties in round r. For simplicity of notation, we do not specify the randomness used in these functions explicitly. We specify the input of a party as part of the input to Π^1, and internal state as part of the input to Π^r, for $r > 1$.

Two-message OT from Two-round MPC for $\mathcal{F}_{n\text{-}\mathsf{OT}}$ over $\mathcal{BC}$

Receiver Message

The receiver computes $(\mathsf{msg}_n^1, \rho_n^1) \leftarrow \Pi^1(n, b)$ and sends msg_n^1 to the sender.

Sender Message

The sender computes $(\mathsf{msg}_1^1, \rho_1^1) \leftarrow \Pi^1(1, (m_0, m_1))$, and for each $j \in [n-1] \setminus \{1\}$ it computes $(\mathsf{msg}_j^1, \rho_j^1) \leftarrow \Pi^1(j, \perp)$ and for each $j \in [n-1]$, it computes $\mathsf{msg}_j^2 \leftarrow \Pi^2(j, \rho_j^1, \overrightarrow{\mathsf{msg}}^1)$. It sends $\{\mathsf{msg}_j^1, \mathsf{msg}_j^2\}_{j \in [n-1]}$ to the receiver.

Receiver Output

The receiver computes and outputs $\mathsf{out} = \Pi^{\mathsf{out}}(n, \rho_n^1, \overrightarrow{\mathsf{msg}}^1, \overrightarrow{\mathsf{msg}}^2)$, where $\overrightarrow{\mathsf{msg}}^1 = \{\mathsf{msg}_1^1, ..., \mathsf{msg}_n^1\}$, and $\overrightarrow{\mathsf{msg}}^2 = \{\mathsf{msg}_1^2, ..., \mathsf{msg}_{n-1}^2\}$.

Figure 3: A transformation from a two-round MPC Π for $\mathcal{F}_{n\text{-}\mathsf{OT}}$ that achieves `SH/SA/UA/IA` over a $\mathcal{BC}$ channel to a two-message OT protocol Π_{OT}.

Lemma 4.1. *Let Π be a two-round n-party protocol for $\mathcal{F}_{n\text{-}\mathsf{OT}}$, secure against a single semi-honest corruption over $\mathcal{BC}$ in the plain (or CRS resp.) model, then the protocol Π_{OT} in figure 3 is a two-message sh-OT in the plain (or CRS resp.) model.*

Proof. Correctness of Π_{OT} follows directly from the correctness of the protocol Π for functionality $\mathcal{F}_{n\text{-}\mathsf{OT}}$. We now argue sender and receiver security. Let $\mathcal{E}$ be an execution of Π, where $P_1's$ input is (m_0, m_1) and $P_n's$ input is b.

1. **Security against semi-honest receiver:** From the semi-honest security of Π, we know that there exists a simulator $\mathcal{S}_n$ corresponding to the real world execution $\mathcal{E}$ where the adversary

corrupts party P_n, such that the following holds:

$$\left\{\mathcal{S}_n(b, m_b), \{\bot\}_{i\in[n-1]}\right\} \approx_c \{\mathsf{view}_n(\mathcal{E}), \mathsf{out}_1(\mathcal{E}), \ldots, \mathsf{out}_{n-1}(\mathcal{E})\}$$
$$\implies \{\mathcal{S}_n(b, m_b)\} \approx_c \{\mathsf{view}_n(\mathcal{E})\}$$

where $\mathsf{view}_i(\mathcal{E}), \mathsf{out}_i(\mathcal{E})$ denote the view and output of party P_i in the real world execution $\mathcal{E}$.

Let $\mathcal{E}'$ be another execution of Π, where $P_1's$ input is (m_0', m_1') and $P_n's$ input is b and let $m_b = m_b'$. Then it also holds that $\{\mathcal{S}_n(b, m_b)\} \approx_c \{\mathsf{view}_n(\mathcal{E}')\}$. From transitivity of the indistinguishability property,

$$\{\mathsf{view}_n(\mathcal{E})\} \approx_c \left\{\mathsf{view}_n(\mathcal{E}')\right\} \implies \{\mathsf{view}_R(\mathcal{E})\} \approx_c \left\{\mathsf{view}_R(\mathcal{E}')\right\}$$

where $\mathsf{view}_n = \mathsf{view}_R$. Thus, sender security holds.

2. **Security against semi-honest sender:** From the semi-honest security of Π, we know that there exists a simulator $\mathcal{S}_1$ corresponding to $\mathcal{E}$ where the adversary corrupts party P_1, such that the following holds:

$$\left\{\mathcal{S}_1((m_0, m_1), \bot), \{\bot\}_{i\in[n-2]}, m_b\right\} \approx_c \{\mathsf{view}_1(\mathcal{E}), \mathsf{out}_2(\mathcal{E}), \ldots, \mathsf{out}_n(\mathcal{E})\}$$
$$\left\{\overline{\mathsf{msg}}_n^1\right\} \approx_c \left\{\mathsf{msg}_n^1\right\}$$

where $\overline{\mathsf{msg}}_n^1$ is the first round message of party P_n simulated by $\mathcal{S}_1((m_0, m_1), \bot)$.

Let $\mathcal{E}'$ be another execution of Π, where $P_1's$ input is (m_0, m_1) and $P_n's$ input is $b' \neq b$. Then it also holds that $\left\{\overline{\mathsf{msg}}_n^1\right\} \approx_c \left\{\mathsf{msg}_n'^1\right\}$. Receiver security now follows from transitivity of the indistinguishability property

$$\left\{\mathsf{msg}_n^1\right\} \approx_c \left\{\mathsf{msg}_n'^1\right\} \implies \{\mathsf{view}_S(\mathcal{E})\} \approx_c \left\{\mathsf{view}_S(\mathcal{E}')\right\}$$

□

Lemma 4.2. *Let Π be a two-round n-party protocol for $\mathcal{F}_{n\text{-}\mathsf{OT}}$, that achieves security with abort ($\mathtt{SA}, \mathtt{UA}, \mathtt{IA}$) against a single malicious corruption over $\mathcal{BC}$ in the plain (or CRS resp.) model, then the protocol Π_{OT} in figure 3 is a two-message mR-OT in the plain (or CRS resp.) model.*

Proof. Correctness of the OT protocol follows directly from the correctness of the underlying protocol Π. Receiver security against a semi-honest sender follows exactly as in Lemma 4.1. We proceed to argue simulation-based sender security against a malicious receiver. Let the adversary corrupt party P_n in the underlying protocol Π. From security of Π, we know that there exists a stateful PPT simulator $\mathcal{S}_n$, that can simulate an indistinguishable view for this adversary in the ideal world.

Given $\mathcal{S}_n$, the simulator $\mathcal{S}_R$ for the OT protocol first computes $\{\mathsf{msg}_i^1\}_{i\in[n-1]} \leftarrow \mathcal{S}_n$. Upon receiving the OT receiver message $\mathsf{msg}_R = \mathsf{msg}_n^1$, it invokes $\mathcal{S}_n$ on this message. At some point, while running $\mathcal{S}_n$, when $\mathcal{S}_n$ queries the ideal functionality on input b of party P_n (receiver), the simulator $\mathcal{S}_R$ of the OT protocol forwards this query to its ideal functionality $\mathcal{F}_{\mathsf{OT}}$. Upon receiving the output m_b from its ideal functionality, it forwards it to the simulator $\mathcal{S}_n$. At the end, $\mathcal{S}_n$ also outputs simulated second round messages $\{\mathsf{msg}_i^2\}_{i\in[n-1]}$. It sends $\mathsf{msg}_S = \{\mathsf{msg}_i^1, \mathsf{msg}_i^2\}_{i\in[n-1]}$ to the adversary.

Indistinguishability of the real and ideal world executions of the OT protocol follow from security of protocol Π. We note that we do not need to explicitly consider the output of honest parties in the real and ideal experiments in this case, because the output of an honest sender in this case is $\bot$. □

This completes the proof of Theorem 4.1. □

4.2 Impossibility of Two-message mR-OT in the Plain Model

In this section we show that a two-message malicious receiver OT is impossible in the plain model. We prove this impossibility by showing that if there exists a simulator that can simulate an indistinguishable view for a malicious receiver, then a malicious/semi-honest sender can run the same simulator to extract the input of an honest receiver.

Lemma 4.3. *There does not exist a 2-message* OT *with one-sided efficient straight-line simulation security against a corrupt receiver.*

Proof. Suppose there exists a 2-round protocol which securely realizes such an OT, i.e. for each PPT $\mathcal{A}$, there exists a PPT $\mathcal{S}_R = (\mathcal{S}_R^1, \mathcal{S}_R^2)$ s.t for each $m_0, m_1 \in \{0,1\}^*$:

$$\left| \Pr\left[\mathsf{IDEAL}_{\mathcal{S}_R,\mathcal{F}_{\mathsf{OT}}}(1^\lambda, m_0, m_1) = 1\right] - \Pr\left[\mathsf{REAL}_{\mathcal{A},\mathsf{OT}}(1^\lambda, m_0, m_1) = 1\right] \right| \leq \frac{1}{2} + \mathsf{negl}(\lambda).$$

where experiments $\mathsf{IDEAL}_{\mathcal{S}_R,\mathcal{F}_{\mathsf{OT}}}$ and $\mathsf{REAL}_{\mathcal{A},\mathsf{OT}}$ are as defined in Definition 3.1. Let b be the input on which $\mathcal{S}_R$ queries the functionality $\mathcal{F}_{OT}(m_0, m_1)$. Then, we construct an adversary $\mathcal{A}_S$ who corrupts the sender as follows: $\mathcal{A}_S$ receives msg_R from an honest receiver, runs $\mathcal{S}_R^1\left(1^\lambda, \mathsf{msg}_R\right)$ and computes b. This enables $\mathcal{A}_S$ to extract an honest receiver's input with a high probability. Note that $\mathcal{A}_S$ is a semi-honest adversary since it does not need to send any message before extracting the receiver's input. This contradicts the assumption that the protocol is secure against a semi-honest sender. □

Combining Theorem 4.1 with the above Lemma, we get the following corollary.

Corollary 4.1. *There exists a functionality $\mathcal{F} \in P/Poly$, for which there does not exist a two-round n-party protocol over $\mathcal{BC}$ that achieves security with* SA/UA/IA *against $t = 1$ malicious corruption with straight-line simulation in the plain model.*

We note that all known honest majority protocols have straight-line simulation.

Another interesting consequence of Theorem 4.1, is an equivalence between a two-round honest-majority MPC and a two-round dishonest majority MPC over broadcast channels. We note that the above reduction from 2-round honest majority MPC for general functionalities to mR-OT compliments the protocols in [19, 9], where they show that OT is complete for two-round MPC over $\mathcal{BC}$ in the CRS model.

5 Broadcast and P2P (or PKI): Identifiable Abort

In this section, we investigate the feasibility of a two round IA protocol with general honest majority in different communication models. We show that while IA for general honest majority is impossible over $\mathcal{BC} + \mathcal{P2P}$, it is possible over $\mathcal{BC} + \mathcal{PKI}$.

5.1 $\mathcal{BC} + \mathcal{P2P}$: Impossibility Result

In this section, we show that there does not exist a two-round IA protocol for general functionalities and general honest majority over $\mathcal{BC} + \mathcal{P2P}$ in the plain model. To prove this result, it suffices to show that there exists a three-party functionality that cannot be securely realized with IA security, over $\mathcal{BC} + \mathcal{P2P}$ in the plain model, in two-rounds, against a single corrupt party.

Theorem 5.1. *There exists a functionality $\mathcal{F} \in P/Poly$, for which there does not exist a three-party protocol that achieves security with* `IA` *against a single malicious corruption over $\mathcal{BC} + \mathcal{P2P}$ with straight-line simulation in the plain model.*

Proof. Let $\mathcal{F}$ be a 3-party functionality in which party P_1 has no input, P_2's input is $b \in \{0, 1\}$ and P_3's input is (x_0, x_1). P_1 receives an output x_b, while P_2 and P_3 do not receive any output. That is, $\mathcal{F}(\perp, b, (x_0, x_1)) = (x_b, \perp, \perp)$. Let Π be a three-party protocol over $\mathcal{BC} + \mathcal{P2P}$ channels, realises $\mathcal{F}$ with `IA` security and straight line simulation. Let $\mathcal{E}^1$ be an execution of the protocol Π computing $\mathcal{F}$. Also, let Π be such that the parties do not send any private messages in the second round (this holds w.l.o.g.). Let $\mathcal{A}$ be an adversary who corrupts party P_2 and works as follows; it behaves like an honest party except that it does not send its private channel message to party P_1 in the first round.

We consider the following three cases:

1. **Output of the honest parties is $\perp$:** We know that in security with `IA`, if the output of the honest parties is $\perp$, then they must identify at least one corrupted party. Since by assumption Π achieves security with `IA`, it must be the case that both P_1 and P_3 correctly identify P_2 as the corrupt party. Let $\mathsf{view}_3(\mathcal{E}^1)$ be the view of party P_3 in execution $\mathcal{E}^1$.

   Consider another execution $\mathcal{E}^2$ for the same functionality with the same set of inputs, where the adversary corrupts party P_1 and works as follows. It behaves honestly in the first round. In the second round, it lies about not having received a message from party P_2 in the first round and computes its second round messages accordingly. Let $\mathsf{view}_3(\mathcal{E}^2)$ be the view of party P_3 in execution $\mathcal{E}^2$. Clearly, the view of party P_3 in this case is indistinguishable from its view in execution $\mathcal{E}^1_\Pi$, i.e., $\mathsf{view}_3(\mathcal{E}^1) \approx_c \mathsf{view}_3(\mathcal{E}^2)$. Since the output of P_3 in $\mathcal{E}^1$ was $(\perp, P_2)$, it must be the case that the output of party P_3 in execution $\mathcal{E}^2$ is also $(\perp, P_2)$. However, since P_2 is an honest party, this violates the requirements of security with `IA`.

   Hence either Π does not achieve `IA` or the output of the honest parties in $\mathcal{E}^1$ cannot be $\perp$.

2. **The simulator extracts b as P_2's input with probability (almost) 1:** In this case, simulator $\mathcal{S}_2$'s view of P_2's messages only involves the broadcast message (say bmsg_2^1) and the private message (say $\mathsf{pmsg}_{2\to3}^1$) that was sent to P_3. The simulator $\mathcal{S}_2$, it straight-line, it is able to extract P_2's input b *only* using $(\mathsf{bmsg}_2^1, \mathsf{pmsg}_{2\to3}^1)$. Note that both of these messages are visible to P_3, i.e., $(\mathsf{bmsg}_2^1, \mathsf{pmsg}_{2\to3}^1) \in \mathsf{view}_3(\mathcal{E}^1)$.

   Consider another execution $\mathcal{E}^2$, where the adversary passively corrupts P_3 and all parties (including P_3) compute and send their messages honestly. Let $(\overline{\mathsf{bmsg}_2^1}, \overline{\mathsf{pmsg}}_{2\to3}^1)$ be the messages sent by an honest P_2 to P_3 in execution $\mathcal{E}^2$. Since the simulator $\mathcal{S}_2$ is straight-line, a corrupt P_3 can now simply run $\mathcal{S}_2$ on $(\overline{\mathsf{bmsg}_2^1}, \overline{\mathsf{pmsg}}_{2\to3}^1)$ to extract an honest P_2's input. This would clearly break privacy of an honest P_2's input. Hence, either Π does not achieve `IA` or there does not exist a straight-line simulator that extracts P_2's correct input b.

3. **The simulator extracts $1 - b$ as P_2's input with some non-negligible probability.** Consider another execution $\mathcal{E}^2$ for the same functionality $\mathcal{F}$, with the same set of inputs, where the adversary passively corrupts party P_1 and behaves honestly throughout the protocol execution. Let $\{\mathsf{bmsg}_i^1, \mathsf{bmsg}_i^2, \{\mathsf{pmsg}_{i\to j}^1\}_{j\in[3]}\}_{i\in[3]}$ be the set of messages exchanged between the parties. From correctness of protocol Π, it follows that P_1 learns the output $x'_{b'}$, where $x'_{b'}$ is P_3's input in $\mathcal{E}_2$ and b' is P_2's input.

   A semi-honest P_1 can now launch the following offline resetting attack: It computes a new second round message while assuming that it did not receive a message from P_2 in the first round, i.e.,

$$\overline{\mathsf{bmsg}_1^2} \leftarrow \Pi^2(1, \overline{\mathtt{T}}_1^1),$$

where $\overline{\mathtt{T}}_1^1$ is the truncated first round transcript $(\mathsf{bmsg}_2^1, \mathsf{bmsg}_3^1, \mathsf{pmsg}_{3\to1}^1)$ of party P_1. Note that the transcript of P_1 is now similar to the one in $\mathcal{E}^1$ and hence outcome of the protocol (output of P_1) in this case must be $x'_{1-b'}$ with non-negligible probability. As a result of this attack, P_1 is able to learn both $x'_{b'}$ and $x'_{1-b'}$, which clearly violates the privacy of P_3's input. Hence, either Π does not achieve IA or there does not exist a straight-line simulator that extracts $1-b$ with non-negligible probability.

Since all 3 cases above are impossible, protocol Π cannot be a secure implementation of functionality $\mathcal{F}$, tolerating a single corruption with IA. □

5.2 $\mathcal{BC}+\mathcal{PKI}$: Positive Result

In this section we give a generic compiler from any two-round UA protocol over $\mathcal{BC}+\mathcal{P}2\mathcal{P}$ channels to a two-round IA protocol over $\mathcal{BC}+\mathcal{PKI}$. Our transformation relies on two building blocks: public-key encryption (PKE) and multi-CRS non-interactive zero-knowledge (m-NIZK) proof systems. We refer the reader to Appendix 3.3 in the Supplementary Material for a formal definition of m-NIZKs.

Formally, we prove the following theorem.

Theorem 5.2. *Assuming the existence of PKE and m-NIZK, there exists a generic transformation from any two-round, n-party* UA *protocol that operates over $\mathcal{BC}+\mathcal{P}2\mathcal{P}$ channels for $t < n/2$, to a two-round n-party* IA *protocol in the $\mathcal{BC}+\mathcal{PKI}$ model for $t < n/2$.*

Instantiating the above theorem using the protocol from [1], we get the following corollary:

Corollary 5.1. *Assuming the existence of PKE and m-NIZKs, there exists an n-party protocol in the $\mathcal{BC}+\mathcal{PKI}$ mode that achieves security with* IA *against $t < n/2$ malicious corruptions for any $\mathcal{F} \in P/Poly$.*

5.2.1 Protocol Description

Let $\mathcal{P} = \{P_1, \ldots, P_n\}$ be the set of parties with inputs $\mathsf{X}_1, \ldots, \mathsf{X}_n$. We start by listing the building blocks and establishing some notations:

1. **Protocol Π:** A two-round n-party MPC protocol $\Pi = (\Pi^1, \Pi^2, \Pi^{\mathsf{out}})$ in the plain model that operates over $\mathcal{BC}+\mathcal{P}2\mathcal{P}$ channels[11] and achieves security with UA against $t < n/2$. We assume that these are stateful algorithms and only indicate the actual input and randomness explicitly for computing the first round messages in Π; for subsequent computations, we assume these to be implicit. We use $\mathsf{pmsg}_{i\to j}$ to denote the first round private message sent by party P_i to party P_j and bmsg_i^r to denote the broadcast message of party P_i in round r. We assume that there exists an additional PPT algorithm Valid_Π that checks if the first round messages $(\mathsf{bmsg}_i^1, \mathsf{pmsg}_{i\to j})$ sent by P_i to P_j are well-formed. As discussed in the technical overview, algorithm Valid_Π corresponds to a simple check (e.g., with regards to the syntax of the messages) that can be easily performed by the party receiving these messages.

[11] We assume that this protocol uses $\mathcal{BC}+\mathcal{P}2\mathcal{P}$ channels in the first round and only a $\mathcal{BC}$ channel in the second round. Note that this holds without loss of generality, since any such protocol can be transformed into one that only uses a $\mathcal{BC}$ channel in the second round, by allowing each pair of parties to exchange sufficiently long time pads in the first round and then encrypting and broadcasting the second round private channel messages using these pads.

2. **PKE:** Public key encryption scheme $(\mathsf{PKE.Gen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ with perfect completeness.

3. **m-NIZK:** Multi-string NIZK $(\mathsf{m\text{-}NIZK.Gen}, \mathsf{m\text{-}NIZK.Prove}, \mathsf{m\text{-}NIZK.Verify})$ (see Definitions 3.3). We assume the randomness used in these algorithms to be implicit and do not specify them.

At the start of the protocol, each party P_i samples a sufficiently long random tape ρ_i to use in the various sub-parts of the protocol; let ρ_i^{key} be the randomness used for generating keys $(\mathsf{pk}_i, \mathsf{sk}_i)$, ρ_i^{Π} for generating messages in protocol Π and $\rho_{i,j}^{\mathsf{enc}}$ to encrypt the private message intended for P_j. Whenever necessary, we augment our notation with subscript $i \to j$ to indicate that the message is sent by P_i and is intended for P_j. We use the vector notation along with a $\bullet$ symbol to refer to a set of n messages, for instance, $\overrightarrow{\mathsf{pmsg}}_{\bullet \to i} = \mathsf{pmsg}_{1 \to i}, \ldots, \mathsf{pmsg}_{n \to i}$. The remaining notations are borrowed from previous sections. A full description of our protocol appears in Figure 4.

5.2.2 Security.

We now proceed to give a description of simulator. Let $\mathcal{A}$ be the real world adversary and $\mathcal{H}$ be the set of honest parties. The simulator $\mathcal{S}$ for this protocol makes use of the simulator $\mathcal{S}_\Pi$ of the underlying protocol Π and the simulator of the m-NIZK scheme $(\mathcal{S}_{\mathsf{Gen}}, \mathcal{S}_{\mathsf{Prove}})$ and proceeds as follows:

Round 1: $\mathcal{S} \to \mathcal{A}$

- Compute $\left\{\mathsf{bmsg}_i^1, \left\{\mathsf{pmsg}_{i \to j}\right\}_{j \in \mathcal{I}}\right\}_{i \in [\mathcal{H}]} \leftarrow \mathcal{S}_\Pi$.

- For each $i \in \mathcal{H}$,$j \in \mathcal{I}$, compute $\mathsf{ct}_{i \to j} \leftarrow \mathsf{PKE.Enc}\left(\mathsf{pmsg}_{i \to j}, \mathsf{pk}_j\right)$, and $\mathsf{crs}_{i \to j} \leftarrow \mathsf{m\text{-}NIZK.Gen}$.

- For each $i, j \in \mathcal{H}$, compute $\mathsf{ct}_{i \to j}^1 \leftarrow \mathsf{PKE.Enc}\left(0, \mathsf{pk}_j\right)$ and $(\mathsf{crs}_{i \to j}, \tau_{i,j}) \leftarrow \mathcal{S}_{\mathsf{Gen}}$.

- It sends $\left\{\mathsf{bmsg}_i^1, \overrightarrow{\mathsf{ct}}_{i \to \bullet}, \overrightarrow{\mathsf{crs}}_{i \to \bullet}\right\}_{i \in \mathcal{H}}$ to the adversary parties.

Round 2.

- Upon receiving first round messages from the adversary, the simulator checks if the messages are well-formed as specified in the protocol and computes $\mathcal{C}_i$ for each $i \in \mathcal{H}$.

- Compute $\left\{\mathsf{bmsg}_i^2\right\}_{i \in \mathcal{H}} \leftarrow \mathcal{S}_\Pi^2\left(\left\{\mathsf{bmsg}_j^1, \left\{\mathsf{pmsg}_{j \to i}\right\}_{i \in \mathcal{H}}\right\}_{j \in \mathcal{I}}\right)$. For each $i \in \mathcal{H}$:

  1. **Normal Mode.** If $\mathcal{C}_i = \emptyset$, compute $\mathsf{proof}_i \leftarrow \mathcal{S}_{\mathsf{Prove}}\left(\overrightarrow{\mathsf{crs}}_{\bullet \to i}, y_i, \overrightarrow{\tau_i}\right)$ where $y_i = \left(\overrightarrow{\mathsf{pk}}_\bullet, \overrightarrow{\mathsf{bmsg}}_\bullet^1, \overrightarrow{\mathsf{ct}}_{i \to \bullet}, \overrightarrow{\mathsf{ct}}_{\bullet \to i}, \mathsf{bmsg}_i^2\right)$ and $\overrightarrow{\tau_i} = \{\tau_{j,i}\}_{j \in \mathcal{H}}$ using language L_{norm} (Figure 5).

  2. **Complaint Mode.** If $\mathcal{C}_i \neq \emptyset$, compute $\mathsf{proof}_i^j \leftarrow \mathsf{m\text{-}NIZK.Prove}\left(\overrightarrow{\mathsf{crs}}_{\bullet \to i}, y_i^j, w_i^j\right)$, where $y_i^j = \left(\mathsf{pk}_i, \mathsf{ct}_{j \to i}, \mathsf{pmsg}_{j \to i}^*\right)$ and $w_i^j = \left(\mathsf{sk}_i, \rho_i^{\mathsf{key}}\right)$ using language L_{comp} (Figure 5).

Two-Round $\mathtt{IA}$ Protocol for $t < n/2$ in the $\mathcal{BC} + \mathcal{PKI}$ Model

Party P_i in Round 1

- **Protocol Π:** Compute $\left(\mathsf{bmsg}^1_i, \overrightarrow{\mathsf{pmsg}}_{i\to\bullet}\right) \leftarrow \Pi^1\left(i, \mathsf{X}_i; \rho^\Pi_i\right)$ and broadcast bmsg^1_i.
- **Ciphertexts:** For each $j \in [n]$, compute and broadcast $\mathsf{ct}_{i\to j} \leftarrow \mathsf{PKE.Enc}\left(\mathsf{pmsg}_{i\to j}, \mathsf{pk}_j; \rho^{\mathsf{enc}}_{i,j}\right)$.
- **m-NIZKs:** For each $j \in [n]$, compute and broadcast $\mathsf{crs}_{i\to j} \leftarrow \mathsf{m\text{-}NIZK.Gen}$.

Party P_i in Round 2

Initialize a cheater list $\mathcal{C}_i = \emptyset$ and compute the following for each $j \in [n]$:

1. Compute $\mathsf{pmsg}^*_{j\to i} \leftarrow \mathsf{PKE.Dec}\left(\mathsf{ct}_{j\to i}, \mathsf{sk}_i\right)$. If $\mathsf{pmsg}^*_{j\to i} = \bot$, append j to the cheater list $\mathcal{C}_i$ and skip the next check.
2. Check if $\mathsf{Valid}_\Pi\left(\mathsf{bmsg}^1_j, \mathsf{pmsg}^*_{j\to i}\right) = 0$. If so, append j to the cheater list $\mathcal{C}_i$.

Normal Mode - If $\mathcal{C}_i = \emptyset$:

1. **Protocol Π:** Compute $\mathsf{bmsg}^2_i \leftarrow \Pi^2\left(i, \mathtt{T}^1_i\right)$, where $\mathtt{T}^1_i = \left(\overrightarrow{\mathsf{bmsg}}^1_\bullet, \overrightarrow{\mathsf{pmsg}}^*_{\bullet\to i}\right)$.
2. **m-NIZK:** Compute $$\mathsf{proof}_i \leftarrow \mathsf{m\text{-}NIZK.Prove}\left(\overrightarrow{\mathsf{crs}}_{\bullet\to i}, y_i, w_i\right),$$ where $y_i = \left(\overrightarrow{\mathsf{pk}}_\bullet, \overrightarrow{\mathsf{bmsg}}^1_\bullet, \overrightarrow{\mathsf{ct}}_{i\to\bullet}, \overrightarrow{\mathsf{ct}}_{\bullet\to i}, \mathsf{bmsg}^2_i\right)$ and $w_i = \left(\mathsf{sk}_i, \rho^{\mathsf{key}}_i, \mathsf{X}_i, \rho^\Pi_i, \overrightarrow{\rho}^{\mathsf{enc}}_{i,\bullet}\right)$ using language L_{norm} (Figure 5).
3. **Broadcast** $\left(\mathtt{normal}, \mathsf{bmsg}^2_i, \mathsf{proof}_i\right)$ to all other parties.

Complaint Mode - If $\mathcal{C}_i \neq \emptyset$:

1. **m-NIZK:** For each $j \in \mathcal{C}_i$, compute $$\mathsf{proof}^j_i \leftarrow \mathsf{m\text{-}NIZK.Prove}\left(\overrightarrow{\mathsf{crs}}_{\bullet\to i}, y^j_i, w^j_i\right),$$ where $y^j_i = \left(\mathsf{pk}_i, \mathsf{ct}_{j\to i}, \mathsf{pmsg}^*_{j\to i}\right)$ and $w^j_i = \left(\mathsf{sk}_i, \rho^{\mathsf{key}}_i\right)$ using language L_{comp} (Figure 5).
2. **Broadcast** $\left(\mathtt{complaint}, \left\{j, \mathsf{pmsg}^*_{j\to i}, \mathsf{proof}^j_i\right\}_{j\in\mathcal{C}_i}\right)$.

Output Reconstruction.

- For each pair $i, j \in [n]$, if P_i raises a $\mathtt{complaint}$ a party P_j, check if $\mathsf{Valid}_\pi\left(\mathsf{bmsg}^1_j, \mathsf{pmsg}^*_{j\to i}\right) = 1$ and $\mathsf{m\text{-}NIZK.Verify}\left(y^j_i, \mathsf{proof}^j_i\right) = 1$. If both the above checks pass, output $(\bot, P_j)$. Otherwise, output $(\bot, P_i)$.
- Else, if no $\mathtt{complaints}$ are raised, check if $\mathsf{m\text{-}NIZK.Verify}\left(y_i, \mathsf{proof}_i\right) = 1$, $\forall i \in [n]$, where $y_i = \left(\overrightarrow{\mathsf{pk}}_\bullet, \overrightarrow{\mathsf{bmsg}}^1_\bullet, \overrightarrow{\mathsf{ct}}_{i\to\bullet}, \overrightarrow{\mathsf{ct}}_{\bullet\to i}, \mathsf{bmsg}^2_i\right)$. If this check fails, output $(\bot, P_i)$.
- Compute and output $\Pi^{\mathsf{out}}\left(i, \overrightarrow{\mathsf{bmsg}}^2_\bullet\right)$.

Figure 4: A transformation from a two-round $\mathtt{UA}$ protocol for $t < n/2$ in the $\mathcal{BC} + \mathcal{P2P}$ model to a two-round $\mathtt{IA}$ protocol for $t < n/2$ in the $\mathcal{BC} + \mathcal{PKI}$ model.

L_{norm}: NP Language for Normal Mode

Statement $y_i = \left(\overrightarrow{\mathsf{pk}}_\bullet, \overrightarrow{\mathsf{bmsg}}^1_\bullet, \overrightarrow{\mathsf{ct}}_{i\to\bullet}, \overrightarrow{\mathsf{ct}}_{\bullet\to i}, \mathsf{bmsg}^2_i\right)$

Witness $w_i = \left(\mathsf{sk}_i, \rho_i^{\mathsf{key}}, \mathsf{X}_i, \rho_i^{\Pi}, \overrightarrow{\rho}^{\mathsf{enc}}_{i,\bullet}\right)$

Relation $R_{\mathsf{norm}}(y_i, w_i) = 1$, if *all* of the following conditions hold:

1. The keys $\mathsf{pk}_i, \mathsf{sk}_i$ were honestly generated using randomness ρ_i^{key}.
2. $\left(\mathsf{bmsg}^1_i, \overrightarrow{\mathsf{pmsg}}_{i,\bullet}\right)$ were computed honestly using input X_i and randomness ρ_i^{Π}.
3. For each $j \in [n]$, $\mathsf{ct}_{i\to j}$ is an honest encryption of $\mathsf{pmsg}_{i\to j}$ using key pk_j and randomness $\rho^{\mathsf{enc}}_{i,j}$.
4. bmsg^2_i was honestly computed using transcript $\mathsf{T}^1_i = \left(\overrightarrow{\mathsf{bmsg}}^1_\bullet, \overrightarrow{\mathsf{pmsg}}^*_{\bullet\to i}\right)$ and for each $j \in [n]$, $\mathsf{pmsg}^*_{j\to i}$ is the decryption of $\mathsf{ct}_{j\to i}$ using secret key sk_i.

L_{comp}: NP Language for Complaint Mode

Statement $y_i^j : \left(\mathsf{pk}_i, \mathsf{ct}_{j\to i}, \mathsf{pmsg}_{j\to i}\right)$

Witness $w_i^j : \left(\mathsf{sk}_i, \rho_i^{\mathsf{key}}\right)$

Relation $R_{\mathsf{comp}}(y_i^j, w_i^j) = 1$, if *all* of the following conditions hold:

1. The keys $\mathsf{pk}_i, \mathsf{sk}_i$ were honestly generated using randomness ρ_i^{key}.
2. $\mathsf{pmsg}_{j\to i}$ is the decryption of $\mathsf{ct}_{j\to i}$ using secret key sk_i.

Figure 5: NP Languages used in the protocol description in Figure 4.

At the end of Round 2. If $\exists i \in \mathcal{H}$, such that $|\mathcal{C}_i| \neq 0$. Then the simulator simply chooses the corrupted party with the smallest index (say j^*) who was identified as a cheater and sends $(\perp, j^*)$ to the ideal functionality as output for the honest parties. Else, if for each $i \in \mathcal{H}$, $|\mathcal{C}_i| = 0$ then for each $j \in \mathcal{I}$, the simulator checks if $1 =^? = \mathsf{m\text{-}NIZK.Verify}\left(y_j, \mathsf{proof}_j\right)$, where $y_j = \left(\overrightarrow{\mathsf{pk}}_\bullet, \overrightarrow{\mathsf{bmsg}}^1_\bullet, \overrightarrow{\mathsf{ct}}_{j\to\bullet}, \overrightarrow{\mathsf{ct}}_{\bullet\to j}, \mathsf{bmsg}^2_j\right)$. If this check fails for any party P_j, then the simulator sends outputs $(\perp, P_j)$ to the ideal functionality. Else, it simply sends $\mathsf{continue}$ to the ideal functionality, signaling it to send the correct output to the honest parties.

Proof of Indistinguishability. We prove indistinguishability of the adversary's view and the output of the honest parties in the real and ideal worlds using hybrid arguments. We consider the following hybrids:

$\mathcal{H}_0$: This hybrid is same as the real world execution.

$\mathcal{H}_1$: This hybrid is similar to hybrid $\mathcal{H}_0$, except that we change the way the honest parties encrypt private channel messages for other honest parties.

- $\mathcal{H}_0 \approx_c \mathcal{H}_1$: Indistinguishability of hybrids $\mathcal{H}_0$ and $\mathcal{H}_1$ follows from semantic security of the public-key encryption scheme.

$\mathcal{H}_2$: This hybrid is similar to hybrid $\mathcal{H}_1$, except that we change the way the honest parties' proofs are computed. In particular, all the honest parties use the simulator of the multi-string NIZK proof system to simulated their respective crs's for each of the other honest parties, who then also simulate their proofs using these simulated crs's in the second round.

- $\mathcal{H}_1 \approx_c \mathcal{H}_2$: Indistinguishability of hybrids $\mathcal{H}_1$ and $\mathcal{H}_2$ follows from composable zero knowledge of the multi-string NIZK proof system.

$\mathcal{H}_3$: This hybrid is similar to hybrid $\mathcal{H}_2$, except that the honest parties use the simulator $\mathcal{S}_\Pi$ to compute their messages in the first and second rounds, i.e.,

- $\mathcal{H}_2 \approx_c \mathcal{H}_3$: Indistinguishability of hybrids $\mathcal{H}_2$ and $\mathcal{H}_3$ follows from the security of protocol Π against malicious corruptions.

$\mathcal{H}_4$: This hybrid is similar to hybrid $\mathcal{H}_3$, except for the way the output of the honest parties is decided.

- $\mathcal{H}_3 \approx_c \mathcal{H}_4$: We consider the following cases:
  1. If the output of the honest parties is a non-$\perp$ value in $\mathcal{H}_4$, then it follows from soundness of the multi-string NIZKs that an adversarial party will not be able to raise a false complaint against an honest party in the real world to influence the output of other honest parties.
  2. If the output of the honest parties is $P_j, \perp$ in in $\mathcal{H}_4$, for some $j \in \mathcal{A}$, then we can rely on the completeness of the multi-string NIZKs to guarantee that the party some honest party will be able to raise a convincing complaint against P_j and therefore all honest parties will identify it as the cheater.

6 Broadcast and P2P: Fail-Stop Guaranteed Output Delivery

FS-GoD is known to be impossible [25] in the plain/CRS models in the absence of private channels in two rounds. In this section, we investigate the minimal assumptions that are required to a realize such protocols in the presence of private channels. More specifically, we show that for $n/3 \leq t < n/2$, sh-OT is necessary for achieving FS-GoD for general functionalities in the plain model[12], while OWF suffice for $t < n/3$.

6.1 $\mathcal{BC} + \mathcal{P}2\mathcal{P}$: Necessity of sh-OT for $(t < n/2)$

In this section, we show that any n-party FS-GoD protocol for general functionalities with $n/3 \leq t < n/2$ implies sh-OT.

Theorem 6.1. *If there exists a 2-round n-party FS-GoD protocol for any $\mathcal{F} \in P/Poly$ in the plain model for $n/3 \leq t < n/2$, then there exists a two-message sh-OT protocol in the plain model.*

Proof. Let Φ be a n-party FS-GoD protocol over $\mathcal{BC} + \mathcal{P}2\mathcal{P}$ for the following functionality:

$$\mathcal{F}_{n\text{-OT}}((m_0, m_1), \{\perp\}_{i\in[n-2]}, b) = (\{\perp\}_{i\in[n-1]}, m_b)$$

where, input of P_1 is $(m_0, m_1) \in \{0,1\}^*$, parties $P_2, \ldots, P_{n-1}$ have no inputs, input of P_n is a bit $b \in \{0,1\}$; and output of P_n is m_b.

From Observation 1, we assume that P_n does not send any message in the last round. Additionally, the remaining parties only need to send private channel messages to P_n in the second round. Now, since Φ achieves FS-GoD, even if t parties, say $P_{t+1}, \ldots, P_{2t}$ fail-stop after sending their first round messages, an honest P_n will still be able to learn the output. Let Π be a slightly modified version of Φ, which forces $P_{t+1}, \ldots, P_{2t}$ to stop after sending their first round messages, as follows:

[12] We note that this lower bound complements the protocol designed by Ananth et al. in [1]

- No messages are sent to $P_{t+1}, \ldots, P_{2t}$ in the first round.
- $P_{t+1}, \ldots, P_{2t}$ do not send any messages in the second round.

Note that Π is not only a correct protocol (based on $\texttt{FS-GoD}$ security of Φ), but also a semi-honest secure protocol against corruption of any t parties. This is true since an adversary in protocol Φ corrupting any t parties can further pretend to not have received the messages omitted in Π, thus simulating the view in protocol Π.

In Figure 6, we show how Π for $\mathcal{F}_{n\text{-}\mathsf{OT}}$ can be used to design a two-message $\mathsf{sh\text{-}OT}$ in the same setup/communication model as Π, where the first $2t$ parties act as the sender and the remaining parties act as the receiver. We use $\mathtt{T}_i^r$ to denote the transcript of party P_n, at the end of the round r. We borrow the remaining notations from previous sections.

Two-message sh-OT from n-Party $\texttt{FS-GoD}$ Protocol over $\mathcal{BC} + \mathcal{P2P}$

Receiver Message

- Compute $\left(\mathsf{bmsg}_n^1, \left\{\mathsf{pmsg}_{n\to j}^1\right\}_{j\in\{1,\ldots,t,2t+1,\ldots,n\}}\right) \leftarrow \Pi^1(n, b)$.
- For $i \in [2t+1, n]$, compute $\left(\mathsf{bmsg}_i^1, \left\{\mathsf{pmsg}_{i\to j}^1\right\}_{j\in\{1,\ldots,t,2t+1,\ldots,n\}}\right) \leftarrow \Pi^1(i, \perp)$.

Send $\left\{\mathsf{bmsg}_i^1, \mathsf{pmsg}_{i\to j}^1\right\}_{i\in[2t+1,n],j\in[1,t]}$ to the sender.

Sender Message

- Compute $\left(\mathsf{bmsg}_1^1, \{\mathsf{pmsg}_{1\to j}^1\}_{j\in[n]}\right) \leftarrow \Pi^1\left(1, (m_0, m_1)\right)$.
- For each $i \in [2t]$, compute $\left(\mathsf{bmsg}_i^1, \{\mathsf{pmsg}_{i\to j}^1\}_{j\in\{1,\ldots,t,2t+1,\ldots,n\}}\right) \leftarrow \Pi^1(i, \perp)$.
- For each $i \in [t]$, compute $\left(\mathsf{bmsg}_i^2, \mathsf{pmsg}_{i\to n}^2\right) \leftarrow \Pi^2(i, \mathtt{T}_i^1)$, where $\mathtt{T}_i^1 = \left\{\mathsf{bmsg}_j^1, \mathsf{pmsg}_{j\to i}^1\right\}_{j\in[n]}$.

Send $\left\{\mathsf{bmsg}_i^1, \mathsf{pmsg}_{i\to j}^1\right\}_{i\in[2t],j\in[2t+1,n]}, \left\{\mathsf{bmsg}_i^2, \mathsf{pmsg}_{i\to n}^2\right\}_{i\in[t]}$ to the receiver.

Receiver Output

- For each $i \in [2t+1, n]$, compute $\left(\mathsf{bmsg}_i^2, \mathsf{pmsg}_{i\to n}^2\right) \leftarrow \Pi^2(i, \mathtt{T}_i^1)$, where $\mathtt{T}_i^1 = \left\{\mathsf{bmsg}_j^1, \mathsf{pmsg}_{j\to i}^1\right\}_{j\in[n]}$.
- Compute and output $\mathsf{out} = \Pi^{\mathsf{out}}(n, \mathtt{T}_n^2)$, where $\mathtt{T}_n^2 = \left\{\mathsf{bmsg}_j^1, \mathsf{bmsg}_j^2, \mathsf{pmsg}_{j\to n}^1, \mathsf{pmsg}_{j\to n}^2\right\}_{j\in\{1,\ldots,t,2t+1,\ldots,n\}}$ ..

Figure 6: A transformation from an n-party $\texttt{FS-GoD}$ protocol Φ with $n/3 \leq t < n/2$ over $\mathcal{BC} + \mathcal{P2P}$ for $\mathcal{F}_{n\text{-}\mathsf{OT}}$ to a two-message $\mathsf{sh\text{-}OT}$. Π refers to a truncated $\texttt{SH}$ variant of Φ, where parties $P_2, \ldots, P_{t+1}$ and P_n do not send any messages in the second round.

Correctness of the OT protocol in figure 6 follows directly from the correctness of the underlying protocol Π for functionality $\mathcal{F}_{n\text{-}\mathsf{OT}}$. The proof for security against semi-honest receiver follows from semi-honest security of Π, since, any adversary corrupting the receiver in OT protocol can be viewed as an adversary corrupting the last $n - 2t$ parties in the underlying protocol Π (where $n - 2t < t$). We now argue security against semi-honest sender.

Security Against Semi-Honest Sender. Recall that, we need to show that the distribution of the first message by the receiver on input $b = 0$ is indistinguishable from that on input $b = 1$. The message sent by the receiver is $\{\mathsf{bmsg}^1_j, \{\mathsf{pmsg}^1_{j\to i}\}_{i\in[2t]}\}_{j\in[2t+1,n]}$. But, since the parties do not send any messages to $P_t, \ldots, P_{2t}$ in the underlying protocol Π, the first message is in fact $\{\mathsf{bmsg}^1_j, \{\mathsf{pmsg}^1_{j\to i}\}_{i\in[t]}\}_{j\in[2t+1,n]}$. This however, is part of the view of a semi-honest adversary corrupting the first t parties in the underlying protocol Π. Hence by the semi-honest security guarantee of Π, this view remains indistinguishable between $b = 0$ and $b = 1$.

□

6.2 $\mathcal{P}2\mathcal{P}$: Positive Result for $(t < n/3)$

In this section, we construct a two-round `FS-GoD` protocol for $t < n/3$. Our construction is based on one-way functions for general functionalities in $P/Poly$ and achieves information-theoretic security for functions in $\mathbf{NC}^1$. We obtain this result by using the compiler from [4], who show that the task of securely computing any arbitrary polynomial function can be non-interactively reduced to securely computing arbitrary quadratic functions in the multi-party setting. An important property of their reduction is that the resulting protocol for arbitrary polynomial functions achieves the same security as the protocol for quadratic functions. We leverage this observation and focus on constructing an `FS-GoD` protocol for quadratic functionalities and prove the following theorem.

Theorem 6.2. *There exists a perfectly secure two-round `FS-GoD` protocol for quadratic functionalities with $t < n/3$ unbounded fail-stop corruptions over $\mathcal{P}2\mathcal{P}$ channels in the plain model.*

Instantiating the Master Theorem from [4] using the protocol from the above theorem, we get the following results.

Corollary 6.1. *Assuming the existence of OWF, there exists a two round `FS-GoD` protocol for $t < n/3$ over $\mathcal{P}2\mathcal{P}$ channels in the plain model for any $f \in P/Poly$.*

There exists a statistically secure two round `FS-GoD` protocol for $t < n/3$ over $\mathcal{P}2\mathcal{P}$ channels in the plain model for any $f \in \mathbf{NC}^1$.

Proof of Theorem 6.2. We observe that a slightly modified version of the semi-honest protocol in [28], achieves `FS-GoD` with $t < n/3$ for quadratic functionalities. The protocol in [28] is based on the standard "share-evaluate-reconstruct" approach, where the parties compute t-out-of-n threshold secret shares [33] of their inputs in the first round. In the second round all the parties evaluate the functionality (that they wish to compute) on their respective shares and send the evaluated share to all other parties, who can then run the reconstruction algorithm of the secret sharing scheme to reconstruct the output. We observe that pre-mature aborts by a fail-stop adversary can be handled in this protocol for $t < n/3$ as follows:

- **Abort in Round 1:** If a corrupt party P_i aborts in the first round and does not send any messages, the remaining parties can evaluate the functionality by simply setting the shares that they were expecting from P_i to 0 and proceed as normal, without any disruption.

- **Abort in Round 2:** Since there are $> 2t$ honest parties and evaluated shares in the second round correspond to a $2t$-out-of-n secret sharing, the shares of the honest parties are sufficient to reconstruct the output. Therefore, aborts in the second round do not disrupt the computation.

For the sake of completeness, we give a description of this protocol in Figure 7. The correctness and security of this modified protocol follows trivially and hence we omit it. □

A two-round `FS-GoD` protocol for any quadratic functionality with $t < n/3$ over $\mathcal{P}2\mathcal{P}$ channels

Let $\mathcal{P} = \{P_1, \ldots, P_n\}$ be the set of parties and $\mathcal{F}$ be the function that they wish to jointly compute. Let X_i be the input held by party P_i. We say that a party is 'active', if it does not abort in the first round. Let $\mathsf{active} \subseteq [n]$ be the subset of parties that are active in the last round of the protocol. Let $(\mathsf{Share}, \mathsf{Recon})$ be a threshold secret sharing scheme [33].

Party P_i in Round 1

1. Compute $\{[\mathsf{X}_i]_1, \ldots, [\mathsf{X}_i]_n\} \leftarrow \mathsf{Share}((t,n), \mathsf{X}_i)$ and send $[\mathsf{X}_i]_j$ to party P_j.
2. Compute $\{[\mathsf{Y}_i]_1, \ldots, [\mathsf{Y}_i]_n\} \leftarrow \mathsf{Share}((t,n), 0)$ and send $[\mathsf{Y}_i]_j$ to party P_j.

Party P_i in Round 2

Compute $[\mathsf{Z}]_i = \mathcal{F}([\mathsf{X}_1]_i, \ldots, [\mathsf{X}_n]_i) + \sum_{j \in [n]} [\mathsf{Y}_j]_i$, where $[\mathsf{X}_j]_i = [\mathsf{Y}_j]_i = 0$, if $P_j \notin \mathsf{active}$.

Output Evaluation

Compute and output $\mathsf{Z} = \mathsf{Recon}((2t, n), \{[\mathsf{Z}]_i\}_{i \in [n]})$.

Figure 7: A two round `FS-GoD` protocol for quadratic functionalities with $t < n/3$ over $\mathcal{P}2\mathcal{P}$ channels.

.

Acknowledgements. The first and second authors were supported in part by an NSF CNS grant 1814919, NSF CAREER award 1942789 and Johns Hopkins University Catalyst award. The second author was additionally supported in part by an Office of Naval Research grant N00014- 19-1-2294. The third author is supported by the joint Indo-Israel Project DST/INT/ISR/P-16/2017 and Ramanujan Fellowship of Dept. of Science and Technology, India.

References

[1] Prabhanjan Ananth, Arka Rai Choudhuri, Aarushi Goel, and Abhishek Jain. Round-optimal secure multiparty computation with honest majority. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part II*, volume 10992 of *LNCS*, pages 395–424. Springer, Heidelberg, August 2018.

[2] Prabhanjan Ananth, Arka Rai Choudhuri, Aarushi Goel, and Abhishek Jain. Two round information-theoretic MPC with malicious security. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part II*, volume 11477 of *LNCS*, pages 532–561. Springer, Heidelberg, May 2019.

[3] Benny Applebaum, Zvika Brakerski, and Rotem Tsabary. Perfect secure computation in two rounds. In Amos Beimel and Stefan Dziembowski, editors, *TCC 2018, Part I*, volume 11239 of *LNCS*, pages 152–174. Springer, Heidelberg, November 2018.

[4] Benny Applebaum, Zvika Brakerski, and Rotem Tsabary. Degree 2 is complete for the round-complexity of malicious MPC. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part II*, volume 11477 of *LNCS*, pages 504–531. Springer, Heidelberg, May 2019.

[5] Gilad Asharov, Abhishek Jain, Adriana López-Alt, Eran Tromer, Vinod Vaikuntanathan, and Daniel Wichs. Multiparty computation with low communication, computation and interaction via threshold FHE. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 483–501. Springer, Heidelberg, April 2012.

[6] Saikrishna Badrinarayanan, Aayush Jain, Nathan Manohar, and Amit Sahai. Secure MPC: Laziness leads to GOD. Cryptology ePrint Archive, Report 2018/580, 2018. https://eprint.iacr.org/2018/580.

[7] Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil P. Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. In Joe Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 1–18. Springer, Heidelberg, August 2001.

[8] Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In *20th ACM STOC*, pages 1–10. ACM Press, May 1988.

[9] Fabrice Benhamouda and Huijia Lin. k-round multiparty computation from k-round oblivious transfer via garbled interactive circuits. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part II*, volume 10821 of *LNCS*, pages 500–532. Springer, Heidelberg, April / May 2018.

[10] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *42nd FOCS*, pages 136–145. IEEE Computer Society Press, October 2001.

[11] David Chaum, Claude Crépeau, and Ivan Damgård. Multiparty unconditionally secure protocols (abstract) (informal contribution). In Carl Pomerance, editor, *CRYPTO'87*, volume 293 of *LNCS*, page 462. Springer, Heidelberg, August 1988.

[12] Ran Cohen, Juan Garay, and Vasillis Zikas. Broadcast-optimal two-round mpc. In *Advances in Cryptology – EUROCRYPT 2020*, 2020.

[13] Ran Cohen and Yehuda Lindell. Fairness versus guaranteed output delivery in secure multiparty computation. In Palash Sarkar and Tetsu Iwata, editors, *ASIACRYPT 2014, Part II*, volume 8874 of *LNCS*, pages 466–485. Springer, Heidelberg, December 2014.

[14] Ivan Damgård, Bernardo Magri, Luisa Siniscalchi, and Sophia Yakoubov. Broadcast-optimal two round mpc with an honest majority. Cryptology ePrint Archive, Report 2020/1254, 2020. https://eprint.iacr.org/2020/1254.

[15] Cynthia Dwork and Moni Naor. Zaps and their applications. In *41st FOCS*, pages 283–293. IEEE Computer Society Press, November 2000.

[16] Shimon Even, Oded Goldreich, and Abraham Lempel. A randomized protocol for signing contracts. In David Chaum, Ronald L. Rivest, and Alan T. Sherman, editors, *CRYPTO'82*, pages 205–210. Plenum Press, New York, USA, 1982.

[17] Michael J. Fischer, Nancy A. Lynch, and Michael Merritt. Easy impossibility proofs for distributed consensus problems. In Michael A. Malcolm and H. Raymond Strong, editors, *4th ACM PODC*, pages 59–70. ACM, August 1985.

[18] Sanjam Garg, Yuval Ishai, and Akshayaram Srinivasan. Two-round MPC: Information-theoretic and black-box. In Amos Beimel and Stefan Dziembowski, editors, *TCC 2018, Part I*, volume 11239 of *LNCS*, pages 123–151. Springer, Heidelberg, November 2018.

[19] Sanjam Garg and Akshayaram Srinivasan. Two-round multiparty secure computation from minimal assumptions. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part II*, volume 10821 of *LNCS*, pages 468–499. Springer, Heidelberg, April / May 2018.

[20] Rosario Gennaro, Yuval Ishai, Eyal Kushilevitz, and Tal Rabin. On 2-round secure multiparty computation. In Moti Yung, editor, *CRYPTO 2002*, volume 2442 of *LNCS*, pages 178–193. Springer, Heidelberg, August 2002.

[21] Oded Goldreich. *Foundations of Cryptography: Volume 2 – Basic Applications.* Cambridge University Press, 2004.

[22] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or A completeness theorem for protocols with honest majority. In Alfred Aho, editor, *19th ACM STOC*, pages 218–229. ACM Press, May 1987.

[23] Oded Goldreich and Yair Oren. Definitions and properties of zero-knowledge proof systems. *Journal of Cryptology*, 7(1):1–32, December 1994.

[24] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof-systems (extended abstract). In *17th ACM STOC*, pages 291–304. ACM Press, May 1985.

[25] S. Dov Gordon, Feng-Hao Liu, and Elaine Shi. Constant-round MPC with fairness and guarantee of output delivery. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *CRYPTO 2015, Part II*, volume 9216 of *LNCS*, pages 63–82. Springer, Heidelberg, August 2015.

[26] Jens Groth and Rafail Ostrovsky. Cryptography in the multi-string model. In Alfred Menezes, editor, *CRYPTO 2007*, volume 4622 of *LNCS*, pages 323–341. Springer, Heidelberg, August 2007.

[27] Yuval Ishai, Ranjit Kumaresan, Eyal Kushilevitz, and Anat Paskin-Cherniavsky. Secure computation with minimal interaction, revisited. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *CRYPTO 2015, Part II*, volume 9216 of *LNCS*, pages 359–378. Springer, Heidelberg, August 2015.

[28] Yuval Ishai and Eyal Kushilevitz. Randomizing polynomials: A new representation with applications to round-efficient secure computation. In *41st FOCS*, pages 294–304. IEEE Computer Society Press, November 2000.

[29] Yuval Ishai, Eyal Kushilevitz, and Anat Paskin. Secure multiparty computation with minimal interaction. In Tal Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 577–594. Springer, Heidelberg, August 2010.

[30] Yuval Ishai, Rafail Ostrovsky, and Vassilis Zikas. Secure multi-party computation with identifiable abort. In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO 2014, Part II*, volume 8617 of *LNCS*, pages 369–386. Springer, Heidelberg, August 2014.

[31] Arpita Patra and Divya Ravi. On the exact round complexity of secure three-party computation. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part II*, volume 10992 of *LNCS*, pages 425–458. Springer, Heidelberg, August 2018.

[32] Tal Rabin and Michael Ben-Or. Verifiable secret sharing and multiparty protocols with honest majority (extended abstract). In *21st ACM STOC*, pages 73–85. ACM Press, May 1989.

[33] Adi Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, 1979.