On Achieving "Best-in-the-Multiverse" MPC

Anasuya Acharya[1⋆], Carmit Hazay[2], and Muthuramakrishnan Venkitasubramaniam[3]

[1] Aarhus University acharya@cs.au.dk
[2] Bar-Ilan University carmit.hazay@biu.ac.il
[3] Georgetown University mv783@georgetown.edu

Abstract. The notion of Best-of-Both-Worlds introduced in the work of Ishai et al. (CRYPTO 2006) investigated whether an MPC protocol can simultaneously provide two incomparable security guarantees: guaranteed output delivery against an honest majority and security with abort against a dishonest majority and provided tight upper and lower bounds in the presence of computationally bounded, i.e., PPT adversaries. Another line of works starting from the work of Chaum (CRYPTO 1989) considered protocols that simultaneously achieved security against an unbounded adversary corrupting a minority of the parties and security against arbitrary corruption by a PPT adversary.

In this work, we generalize previous work to investigate a fundamental challenge of designing an MPC in a *multiverse* where security is specified with respect to (1) GOD, (2) fairness, (3) security w.r.t. unbounded adversaries, and (4) security with abort. The work of Lucas et al. (PODC 2010) resolved this question when considering threshold adversaries; however, the case of general adversary structures remains open. Our main result completely characterizes when a protocol can simultaneously achieve all properties. Namely, given adversary structures $\mathcal{Z}_{\mathsf{GOD}}, \mathcal{Z}_{\mathsf{fair}}, \mathcal{Z}_{\mathsf{S}}$ and $\mathcal{Z}_{\mathsf{C}}$, we provide tight upper and lower bounds for when an MPC protocol can provide GOD, fairness, and security with abort respectively for unbounded and PPT adversaries w.r.t. these adversary structures.

Keywords: General Adversary · Feasibility · Secure Multiparty Computation

1 Introduction

Secure multiparty computation (MPC) allows participants with private inputs to collaboratively compute a function without disclosing their inputs. Moreover, the security is required to hold even if some participants are under the control of a coordinated adversary that can alter their behavior arbitrarily. MPC has been rigorously studied for the past four decades since its inception [13, 26] and remains a vibrant field of research, due to its ability to capture most security tasks. It is widely recognized as a cornerstone cryptographic primitive. In this

⋆ Work done while the author was a PhD student at Bar-Ilan University, Israel.

paper, we examine a novel aspect of MPC, highlighting the boundaries of the security that MPC can achieve.

The security of MPC is generally characterized by an adversary model which specifies the computational limits (bounded or unbounded), the subset of participants it can manipulate (honest or dishonest majority or more generally an arbitrary family of subsets), and the types of attacks it can initiate against the protocol (e.g., denial of service) or network conditions (synchronous or asynchronous with or without broadcast). The feasibility of MPC for different combinations of these attacks has been thoroughly studied, and we have a clear understanding of the achievable limits in each scenario and their feasibility. In this paper, our focus is on a fundamental challenge of creating a single protocol capable of addressing multiple combinations (or even all!).

This challenge has been studied in various forms in prior work. The works of Chaum et al. [6, 7] analyzed whether security against an unbounded adversary corrupting an honest majority can be achieved in a protocol while simultaneously providing security against a computationally bounded adversary against a dishonest majority. Introduced as a best-of-the-both-worlds protocol, the works of Katz [21] and Ishai et al. [18] studied if a single MPC protocol can guarantee output delivery (i.e. prevent denial of service) against a computationally bounded adversary corrupting a minority of participants while simultaneously providing security upto abort against a computationally bounded adversary corrupting an arbitrary subset of participants. More recently, extending the work of [18], Lucas et al. [23] come close to resolving our motivating question by identifying when MPC is feasible when guaranteed output delivery (GOD) and fairness[4] against unbounded adversaries and security upto abort against computationally bounded adversaries can be simultaneously achieved. However, all of these works have focused on the threshold setting, where a single number is used to capture the maximum number of participants that an adversary can control.

More generally, the set of participants controlled by the adversary can be specified via a family of subsets referred to as an access structure or general adversary model. Developed in the work of [15], designing protocols in this setting is harder to characterize and design. In the general adversary model, Hirt and Maurer [15] characterized $\mathcal{Q}^2$ adversary structures[5] as the set of structures where unconditionally secure MPC can be feasibly realized for any functionality. In this work, we want to characterize precisely when guaranteed output delivery, fairness, and security with abort can be achieved in the general adversary model, i.e. arbitrary access structures.

In a recent work, Acharya et al. [1] study the question of achieving simultaneously security against unbounded adversaries for one access structure $\mathcal{Z}_1$ and computationally bounded adversaries against a second access structure $\mathcal{Z}_2$. They show that for any access structure for which security against an unbounded

[4] A weaker property than guaranteed output where all participants including the adversary either all learn the result or none of them do.

[5] $\mathcal{Q}^2$ contains all families of subsets such that no two sets in the adversary structure cover the complete set of n parties.

adversary alone is feasible (i.e. for any $\mathcal{Z}_1 \in \mathcal{Q}^2$) can be compiled to a protocol that additionally achieves security against maximal corruptions (i.e. upto n-1 participants) by a computationally bounded adversary. However, this work only considered security with abort against both classes of adversaries.

In this work, we aim to design MPC protocols that can satisfy each of the security guarantees (GOD, fairness, and security with abort) across different adversarial resources (unbounded and PPT), depending on the set of parties corrupted. We are motivated by identifying a best-in-the-multiverse protocol.

Besides a fundamental theoretical challenge that expands our understanding of when MPC is feasible, here are some motivating scenarios for such a study:

- **Multiple Beliefs.** This could facilitate creating a protocol for a group of participants who hold *varied beliefs* about the adversary model. Namely, a subset of participants may require security against unbounded adversaries against some specific set of parties, and others may want just privacy and correctness (i.e., security w.r.t abort) against a larger set of parties.
- **Fall-Back Security.** Such protocols could enable some "fall-back" security in cases where some underlying assumptions fail to hold. Suppose unconditional security could hold against a specific subset of parties in addition to other guarantees (such as security w.r.t. abort), then even if the underlying cryptographic assumptions are broken (say through quantum attacks) by the specific subset, security continues to hold.
- **An Example.** A more concrete scenario can be an MPC between citizens and governments of multiple nations. Suppose that the US and UK governments and their citizens wanted to securely compute some statistic, we can consider four classes of participants: A = US government, B = UK government, C = US citizens, and D = UK citizens. One could want a protocol that provides some resistance to censorship against governments by providing GOD w.r.t. collusion of A and B, security w.r.t. abort of collusion among parties in $C \cup D$ (all citizens) or $A \cup C$ (US entities) and $B \cup D$ (UK entities).

As mentioned above, the feasibility of best-in-the-multiverse MPC has been resolved for the case of threshold adversary structures by Lucas et al. [23] where they design MPC protocols that achieve (1) GOD for unbounded adversaries for up to $t < \frac{n}{2}$ corruptions; (2) fairness for unbounded adversaries up to $\frac{n}{2}$ corruptions; and (3) security with abort for PPT adversaries up to s corruptions when $s + t < n$. Their protocols match the feasibility lower-bounds for threshold structures [18]. However, as we explain in Section 2, these do not directly help when considering general adversaries and cannot help build a protocol for the example listed above.

1.1 Our Contributions

Addressing this question in the general adversary model would involve considering four adversary structures: $\mathcal{Z}_{\mathsf{GOD}}$ for GOD, $\mathcal{Z}_{\mathsf{fair}}$ for fairness, $\mathcal{Z}_{\mathsf{S}}$ for security with abort for unbounded adversaries, and $\mathcal{Z}_{\mathsf{C}}$ for security with abort for PPT

adversaries. We fully characterize when any n-party functionality can be securely realized in a *multiverse* where security is specified with respect to four *general* adversary structures: $\mathcal{Z}_{\mathsf{GOD}}$ for GOD, $\mathcal{Z}_{\mathsf{fair}}$ for fairness, $\mathcal{Z}_{\mathsf{S}}$ for security with abort for unbounded adversaries, and $\mathcal{Z}_{\mathsf{C}}$ for security with abort for computationally bounded adversaries. For a set of n parties $\mathcal{P}$, these can be arbitrary subsets of the power set $2^{\mathcal{P}}$ that the adversary can corrupt.

Multiverse Characterization. It is well known that GOD, fairness, and security for unbounded adversaries cannot be individually achieved if the respective adversary structures are not $\mathcal{Q}^2$. Since GOD implies fairness, we will need $\mathcal{Z}_{\mathsf{GOD}} \cup \mathcal{Z}_{\mathsf{fair}}$ to additionally be a $\mathcal{Q}^2$ access structure for MPC to be feasible. Similarly security with abort subsumes all the other guarantees we have without loss of generality $\mathcal{Z}_{\mathsf{C}}$ contains all the other access structures. Our main result shows that MPC when $\mathcal{Z}_{\mathsf{GOD}} \cup \mathcal{Z}_{\mathsf{fair}}$ and $\mathcal{Z}_{\mathsf{S}}$ are $\mathcal{Q}^2$-structures, the maximal adversary structure $\mathcal{Z}_{\mathsf{C}}$ for which security with abort against PPT adversaries can be achieved is one that contains all the other access structures and for which at most one set in $\mathcal{Z}_{\mathsf{GOD}}$ has its complement set in $\mathcal{Z}_{\mathsf{C}}$. In other words, MPC is feasible if there exist *no* two distinct pairs $A_1, A_2 \in \mathcal{Z}_{\mathsf{GOD}}$ and $B_1, B_2 \in \mathcal{Z}_{\mathsf{C}}$ such that $A_1 \cup B_1 = A_2 \cup B_2 = \mathcal{P}$. More precisely,

$$\neg \exists Z_1 \neq Z_2 \in \mathcal{Z}_{\mathsf{GOD}}, \text{ s.t. } (\mathcal{P} - Z_1 \in \mathcal{Z}_{\mathsf{C}}) \text{ and } (\mathcal{P} - Z_2 \in \mathcal{Z}_{\mathsf{C}}) \tag{1}$$

Our lower bound proof holds even in the presence of only fail-stop adversaries[6] (as opposed to byzantine). Slightly more formally:

Theorem (Informal) 1.1 *Let $\mathcal{P} = \{P_i\}_{i \in [n]}$ be a set of n parties. Let $\mathcal{Z}_{\mathsf{GOD}} \in \mathcal{Q}^2$ be the adversary structure for GOD and $\mathcal{Z}_{\mathsf{C}} \subseteq 2^{\mathcal{P}}$ be the largest adversary structure for security with abort s.t. the following does* ***not*** *hold,*

$$\exists Z_1 \neq Z_2 \in \mathcal{Z}_{\mathsf{GOD}}, \textit{ s.t. } (\mathcal{P} - Z_1 \in \mathcal{Z}_{\mathsf{C}}) \textit{ and } (\mathcal{P} - Z_2 \in \mathcal{Z}_{\mathsf{C}})$$

Let $\mathcal{Z}_{\mathsf{fair}}, \mathcal{Z}_{\mathsf{S}} \in \mathcal{Q}^2$ be adversary structures for fairness and unbounded adversaries, respectively. Then MPC is feasible only when $\mathcal{Z}_{\mathsf{fair}} \cup \mathcal{Z}_{\mathsf{GOD}} \in \mathcal{Q}^2$ and,

$$\forall Z \in \mathcal{Z}_{\mathsf{GOD}}, \mathcal{Z}_{\mathsf{fair}}, \mathcal{Z}_{\mathsf{S}}, \exists Z' \in \mathcal{Z}_{\mathsf{C}}, Z \subseteq Z'$$

Observe that the threshold feasibility is a special case of the above theorem. Informally, the lower bound states that $\mathcal{Z}_{\mathsf{GOD}}$ must be a $\mathcal{Q}^2$ structure, and the complement of at most one set in $\mathcal{Z}_{\mathsf{GOD}}$ can be a member of $\mathcal{Z}_{\mathsf{C}}$. In the threshold setting, for threshold $s < \frac{n}{2}$, $\mathcal{Z}_{\mathsf{GOD}}$ contains all sets of size s. In order for $\mathcal{Z}_{\mathsf{C}}$ to be feasible with respect to $\mathcal{Z}_{\mathsf{GOD}}$, $\mathcal{Z}_{\mathsf{C}}$ cannot contain any ($n - s$ sized) complement of sets in $\mathcal{Z}_{\mathsf{GOD}}$ (except for maybe one set). Therefore the threshold t needs to be $< n - s$ for $\mathcal{Z}_{\mathsf{C}}$ to contain all sets of size t. This is implied in the lower bound $s + t < n$ for the threshold case.

[6] In the fail-stop corruption model, corrupt parties can abort at any point in the protocol but will semi-honestly follow the protocol specification before that.

Best-in-the-Multiverse Protocol. We show that our characterization is tight by presenting an n-party protocol that realizes any functionality in the presence of active adversaries in the best-in-the-multiverse setting, as long as the adversary structures are feasible according to Theorem 1.1. More formally:

Theorem (Informal) 1.2 *Let $\mathcal{Z}_{\mathsf{GOD}} \cup \mathcal{Z}_{\mathsf{fair}}, \mathcal{Z}_{\mathsf{S}} \in \mathcal{Q}^2$, where $\mathcal{Z}_{\mathsf{GOD}}$ and $\mathcal{Z}_{\mathsf{C}}$ satisfy Equation 1. Assuming oblivious transfer and secure point-to-point and broadcast channels, for any n-party functionality f, there exists a protocol that computes f in the presence of an active adversary with the following security guarantees:*

- *Security holds with GOD in the presence of an unbounded adversary w.r.t $\mathcal{Z}_{\mathsf{GOD}} \cap \mathcal{Z}_{\mathsf{S}}$ and a PPT adversary corrupting any subset in $\mathcal{Z}_{\mathsf{GOD}}$.*
- *Security holds with fairness in the presence of an unbounded adversary w.r.t $\mathcal{Z}_{\mathsf{fair}} \cap \mathcal{Z}_{\mathsf{S}}$ and a PPT adversary corrupting any subset in $\mathcal{Z}_{\mathsf{fair}}$.*
- *Security with abort holds in the presence of an unbounded adversary w.r.t $\mathcal{Z}_{\mathsf{S}}$ and a PPT adversary corrupting any subset in $\mathcal{Z}_{\mathsf{C}}$.*

Towards proving Theorem 1.2, we design protocols in two specific multiverses which we identify as an independent contribution. These are:

1. **No overlap scenario.** Let $\mathcal{Z}'_{\mathsf{GOD}}$ be an adversary structure such that $\mathcal{Z}'_{\mathsf{GOD}} \cup \mathcal{Z}_{\mathsf{fair}}, \mathcal{Z}_{\mathsf{S}} \in \mathcal{Q}^2$. We construct an n-party maliciously secure protocol that computes any functionality f in the multiverse as long as $\mathcal{Z}'_{\mathsf{GOD}}$ and $\mathcal{Z}_{\mathsf{C}}$ satisfy the following property:

$$\forall Z \in \mathcal{Z}'_{\mathsf{GOD}}, \forall Z' \in \mathcal{Z}_{\mathsf{C}}, Z \cup Z' \neq \mathcal{P} \tag{2}$$

   On a high-level, we rely on the protocol virtualization technique similar to [1, 14, 20]. Our starting point is a semi-honest unbounded secure n-party protocol that computes a $\mathcal{Z}_{\mathsf{fair}}$-verifiable-secret-sharing (implied in [17]) of the output of f, that is secure against unbounded corruptions in $\mathcal{Z}_{\mathsf{S}}$. We identify this protocol as a sequence of next-message functions, and each of these functions is emulated by a maliciously secure n-party protocol that satisfies GOD w.r.t $\mathcal{Z}'_{\mathsf{GOD}}$; security with abort w.r.t $\mathcal{Z}_{\mathsf{C}}$; and security against unbounded adversaries that can corrupt all but a *designated* party. For each round of the semi-honest protocol, the next-message function for the i^{th} party is performed by a protocol with party P_i being the designated party.
2. **Exactly one overlap scenario.** Let $\mathcal{Z}^*_{\mathsf{GOD}} = \{Z\}$ contain only one set of parties $Z \subset \mathcal{P}$, and let $\mathcal{Z}^*_{\mathsf{GOD}} \cup \mathcal{Z}_{\mathsf{fair}}, \mathcal{Z}_{\mathsf{S}} \in \mathcal{Q}^2$. We construct an n-party maliciously secure protocol that computes any functionality f in the multiverse where $\mathcal{Z}^*_{\mathsf{GOD}}$ and $\mathcal{Z}_{\mathsf{C}}$ satisfy:

$$\mathcal{Z}^*_{\mathsf{GOD}} = \{Z\}, \exists Z' \in \mathcal{Z}_{\mathsf{C}}, Z \cup Z' = \mathcal{P} \tag{3}$$

   We construct this by slightly modifying the previous protocol.

Complexities. The computation and communication complexities of all the protocols we design are polynomially proportional to that of the building-block protocols used – which, in turn, grow polynomially in the security parameter, the computed functionality, the number of parties, and the size of the representation of the adversary structures[7]. In particular, the building-blocks we require exist – admitting polynomial complexities in the security parameter, functionality size, and number of parties – for the cases of (1) threshold adversary structures and (2) adversary structures whose size grows polynomially in the number of parties. Therefore, plugging in the appropriate building-blocks, the protocols we design admit polynomial complexities if the adversary structures $\mathcal{Z}_{\mathsf{fair}}$, $\mathcal{Z}_{\mathsf{S}}$ and $\mathcal{Z}_{\mathsf{C}}$ are (1) either threshold structures, (2) or have size polynomial in the number of parties. Our protocols may admit super-polynomial complexities otherwise.

1.2 Related Work

Several notions of best-of-both-worlds MPC have been considered in the literature. In the concrete 2PC setting, Khurana and Mughees [22] studied the round complexity of designing a 2PC protocol that offers unconditional security against one party and computational against the other, providing round optimal protocols. This was improved by Badrinarayanan et al. [2] who improve the underlying cryptographic assumptions needed to design round-optimal protocols.

For n-party best-of-both-worlds secure MPC, the first works in this vein by Katz [21] and Ishai et al. [18] explored a notion that provides either fairness or guaranteed output delivery (GOD) when a strict minority of the parties are corrupted, and security with abort beyond that, for threshold computationally bounded adversaries. More formally, these works show that for threshold adversary structures, GOD can be achieved up to t corruptions and security with abort up to s corruptions if $t < n/2$ and $s + t < n$. In contrast, fairness can be achieved for $t < n/2$ corruptions simultaneously with security with abort up to n corruptions. Follow-up work by [24] design round-efficient protocols for these settings – presenting lower bounds and matching upper bound protocols for settings with and without setup. Regarding concrete functionalities, Fitzi et al. [10] designed secure n-party broadcast protocols. They considered two thresholds $t < T$ where for corruption $< t$, broadcast is achieved with GOD; for corruption $< T$, broadcast is achieved with unanimous abort; and for corruption beyond that, either broadcast is achieved if the dealer is honest, or all the parties only reach an agreement. As a corollary, [10] also characterized the conditions under which MPC is achievable without assuming broadcast.

A second notion of best-of-both-worlds security is that of n-party secure multi-party computation with security in the presence of an unbounded adversary corrupting a minority of parties and a PPT adversary corrupting any

[7] General adversary structures are represented by enumerating the set of subsets of $\mathcal{P}$ that an adversary is allowed to corrupt and the size is the cardinality of this set. In contrast, although having exponential cardinality, threshold adversary structures can be represented with just one number $t \in [n]$ denoting the threshold.

arbitrary subset of parties. This study was initiated by Chaum et al. [6, 7] presenting an n-party semi-honest secure protocol for threshold adversaries. They also provided a protocol for maliciously secure MPC with abort but for more restricted thresholds. In [11], Fitzi et al. designed secure n-party broadcast in the presence of a threshold adversary considering two thresholds: t for security in the presence of unbounded active adversaries and $T > t$ for PPT adversaries. Finally, Acharya et al. [1] address this problem for general adversary structures.

2 Technical Overview

In this section, we give a high-level overview of our upper and lower bounds characterizing the feasibility of MPC in a multiverse. We begin with an overview of the approach in the threshold setting by Lucas et al. [23].

Step 1. Lucas et al. [23] begin with a $n+1$-party protocol in which one party is considered a designated party, denoted by P_0, and the remaining are denoted by $\mathcal{P} = \{P_1, \ldots, P_n\}$. This protocol satisfies the following properties:

1. For a threshold $t < n/2$, the protocol satisfies GOD against an unbounded adversary corrupting t parties in $\mathcal{P}$. Moreover, if the party P_0 is corrupted in addition, then the protocol satisfies GOD against PPT adversaries.
2. If up to s parties in $\mathcal{P}$ such that $s + t < n$ are corrupted by an unbounded adversary, then the protocol satisfies security w.r.t abort. Moreover, if, in addition to the s parties in $\mathcal{P}$, the party P_0 is corrupted, then the protocol satisfies security w.r.t. abort against PPT adversaries.
3. If up to $n/2$ parties in $\mathcal{P}$ are corrupted by an unbounded adversary, then the protocol satisfies fairness. Further, if, in addition, the party P_0 is corrupted, then the protocol satisfies fairness against PPT adversaries.

Such a protocol can be designed by taking the n-party protocol Π of Ishai et al. [18] that guarantees GOD upto t and security w.r.t abort upto s with $t < n/2$ and $s + t < n$ against PPT adversaries and compile it to a $n + 1$ party protocol Π' where they emulate Π where P_i's algorithm in Π is emulated jointly by P_i and P_0 in the compiled protocol Π' via a 2-party maliciously secure protocol. If P_0 remains uncorrupted, the views are statistically hidden no matter how many parties the adversary corrupts in $\mathcal{P}$. The remaining properties essentially follow from the properties of the underlying protocol Π that is emulated. This protocol is enhanced to guarantee fairness up to $n/2$ parties in $\mathcal{P}$ by having the output secret shared using an $n/2$-out-of-n secret sharing protocol and then running a distributed reconstruction protocol that guarantees fairness up to $n/2$-parties.

Step 2. The final protocol is obtained by eliminating the designated party through player emulation. P_0 is emulated via a distributed protocol among the remaining n parties using a protocol with GOD for $n/2$ unbounded corruptions.

It is tempting to think that this approach can be extended to general adversaries. First, we remark that, as is observed in prior work [1], a general adversary typically requires a different approach. However, in our case, there is an added factor that the characterization is more complex than in the threshold case, as we

explain next. Consider the case where we only consider GOD and security with abort against computationally bounded adversaries that was considered in the work of [18,21] and the starting point of [23]. It was shown that MPC is feasible if $t < n/2$ and $s+t < n$. A first attempt to generalize this characterization would imply $\mathcal{Z}_{\mathsf{GOD}} \in \mathcal{Q}_2$ and $\mathcal{Z}_{\mathsf{GOD}} \subseteq \mathcal{Z}_{\mathsf{C}} \in \mathcal{Q}_2$, as generalizing $s + t < n$ would mean that we do not want to have two sets Z_1, Z_2 such that $Z_1 \in \mathcal{Z}_{\mathsf{C}}$ and $Z_2 \in \mathcal{Z}_{\mathsf{GOD}}$ where $Z_1 \cup Z_2 = \mathcal{P}$. Here is a concrete scenario where MPC is feasible and does not satisfy these conditions. Consider a three-party setting $\{P_1, P_2, P_3\}$ where the $\mathcal{Z}_{\mathsf{GOD}}$ access structure contains P_3 and $\mathcal{Z}_{\mathsf{C}}$ contains all subsets of $\{P_1, P_2\}$ and the set P_3. A simple protocol exists for this scenario through player emulation of a two-party protocol (A, B) where P_1, P_2 emulate A and P_3 emulates B. In the introduction, we provided another example with governments and citizens that would be feasible but does not satisfy the requirement stated above.

In summary, even generalizing the work of Ishai et al. [18] turns out to be non-trivial. Even if we extend this protocol to general adversaries, simply using the Lucas et al. approach directly will not work, as we need to consider $\mathcal{Z}_{\mathsf{S}}$, which requires security with abort against unbounded adversaries. The protocol of [23] could possibly be extended to the non-threshold case to yield a protocol for Equation 2 and this, combined with our protocol for Equation 3 like we do, would also yield the final feasibility (Theorem 5). In contrast, our techniques present a unified framework that results in protocols for both Equations 2 and 3 (Theorem 4) and is black-box in the underlying building blocks, unlike that of [23]. Next, we describe our approach starting with our lower bound and then our upper bound.

2.1 Characterizing MPC in the Multiverse

The multiverse considers four general adversary structures: $\mathcal{Z}_{\mathsf{GOD}}$ for guaranteed output delivery (GOD), $\mathcal{Z}_{\mathsf{fair}}$ for fairness, $\mathcal{Z}_{\mathsf{S}}$ for security with abort for unbounded adversaries, and $\mathcal{Z}_{\mathsf{C}}$ for security with abort for computationally bounded PPT adversaries. We design n-party protocols that can realize any functionality and provide these security guarantees only if the adversary structures given are *feasible*. That is, we require that $\mathcal{Z}_{\mathsf{GOD}} \cup \mathcal{Z}_{\mathsf{fair}}, \mathcal{Z}_{\mathsf{S}} \in \mathcal{Q}^2$. Furthermore, let $\mathcal{P}$ be the set of n parties and let $\mathcal{Z}^*_{\mathsf{C}} \subseteq 2^{\mathcal{P}}$ be the largest general adversary structure satisfying the following constraint with $\mathcal{Z}_{\mathsf{GOD}}$,

$$\neg\exists Z_1 \neq Z_2 \in \mathcal{Z}_{\mathsf{GOD}}, \text{ s.t. } (\mathcal{P} - Z_1 \in \mathcal{Z}^*_{\mathsf{C}}) \text{ and } (\mathcal{P} - Z_2 \in \mathcal{Z}^*_{\mathsf{C}}) \tag{1}$$

We show that any protocol in the multiverse that satisfies GOD tolerating $\mathcal{Z}_{\mathsf{GOD}}$ can only satisfy security with abort in the presence of PPT adversaries tolerating $\mathcal{Z}^*_{\mathsf{C}}$. For such protocols, no security is guaranteed against an adversary corrupting any set of parties not in $\mathcal{Z}^*_{\mathsf{C}}$. Therefore, for the adversary structures to be feasible, we additionally require that $\mathcal{Z}_{\mathsf{GOD}}, \mathcal{Z}_{\mathsf{fair}}, \mathcal{Z}_{\mathsf{S}}, \mathcal{Z}_{\mathsf{C}} \subseteq \mathcal{Z}^*_{\mathsf{C}}$. The property in Equation 1 serves as the feasibility lower-bound, discussed in the following text.

The Lower Bound. In Section 4, we show that the characterization given above is indeed the most general case for which best-in-the-multiverse MPC is

feasible. That is, security with GOD and abort cannot co-exist when,

$$\exists Z_1, Z_2 \in \mathcal{Z}_{\mathsf{GOD}} \text{ s.t. } (Z_1 - Z_2 \neq \emptyset) \text{ , } (Z_2 - Z_1 \neq \emptyset) \text{ and } (\mathcal{P} - Z_1, \mathcal{P} - Z_2 \in \mathcal{Z}_{\mathsf{C}})$$

This means that if $\mathcal{Z}_{\mathsf{GOD}}$ contains two or more distinct sets, where one does not completely contain the other, MPC with security with abort is *not* achievable if the adversary is allowed to corrupt both of their complement sets in $\mathcal{P}$. This statement and its proof is a generalization of the proof of [18], which only provides a lower bound for threshold adversaries, stating that MPC with GOD for t corruptions and security with abort for s corruptions can simultaneously only be achieved when $s+t < n$. Our proof for the general case follows from letting $\mathcal{P}$ be divided into three sets: $A = Z_1 \in \mathcal{Z}_{\mathsf{GOD}}$, $C = Z_2 \in \mathcal{Z}_{\mathsf{GOD}}$, and $B = \mathcal{P} - (A \cup C)$. Extending [18], we design a functionality f where a party $P_i \in A$ and $P_j \in C$ have the input bits b_i and b_j, and other parties do not have inputs. P_i and P_j both receive $b_i \oplus b_j$ as the output of f and other parties do not receive outputs. We then show that, if there exists a protocol that can securely compute f, then there exists an adversary corrupting parties in $A \cup B$, such that it can always distinguish the real world and ideal world with noticeable advantage simply by examining the view of the corrupt parties and strategically controlling when the parties in A abort in the protocol.

This condition on $\mathcal{Z}_{\mathsf{GOD}}$ and $\mathcal{Z}_{\mathsf{C}}$, together with the $\mathcal{Q}^2$ constraints on other adversary structures, constitutes a tight feasibility lower bound for general adversaries in this multiverse.

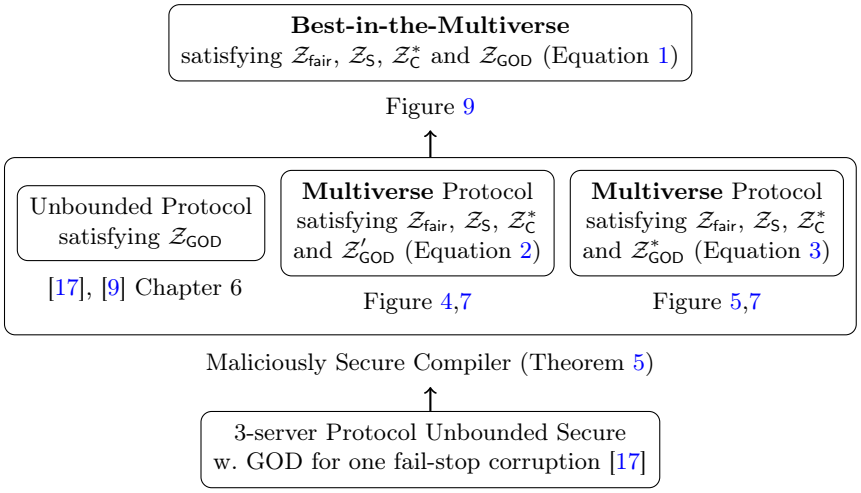


Fig. 1. Overview of our main construction. This figure is the roadmap for obtaining our Best-in-the-Multiverse protocol of Figure 9.

2.2 Best-in-the-Multiverse MPC Protocol

In Section 5, we present a best-in-the-multiverse MPC protocol that can realize any functionality as long as the adversary structures satisfy the feasibility constraints given above. Figure 1 outlines the building-blocks and roadmap for constructing a protocol that satisfies this characterization. Note that by Equation 1, any $\mathcal{Z}_{\mathsf{GOD}}$ that satisfies it can contain at most one set $Z \in \mathcal{Z}_{\mathsf{GOD}}$, where $\mathcal{P} - Z \in \mathcal{Z}^*_{\mathsf{C}}$. Going forward, we will split $\mathcal{Z}_{\mathsf{GOD}}$ into $\mathcal{Z}^*_{\mathsf{GOD}} = \{Z\}$, as in Equation 3, and $\mathcal{Z}'_{\mathsf{GOD}} = \mathcal{Z}_{\mathsf{GOD}} - \mathcal{Z}^*_{\mathsf{GOD}}$, which satisfies Equation 2, as explained in the following text and referred to accordingly in Figure 1. We reduce the problem of realizing any n-party functionality in the multiverse into three simpler problems that require protocols satisfying weaker security notions:

1. MPC protocols that are maliciously unbounded secure satisfying GOD tolerating $\mathcal{Z}_{\mathsf{GOD}}$. Note that assuming broadcast, such protocols for computing any n-party functionality exists as long as $\mathcal{Z}_{\mathsf{GOD}} \in \mathcal{Q}^2$, as implied in [17].
2. MPC protocols in the multiverse that realize any functionality as long as the following property holds:

$$\forall Z \in \mathcal{Z}'_{\mathsf{GOD}}, \forall Z' \in \mathcal{Z}_{\mathsf{C}}, Z \cup Z' \neq \mathcal{P} \tag{2}$$

   This provides security tolerating $\mathcal{Z}_{\mathsf{fair}}, \mathcal{Z}_{\mathsf{C}}, \mathcal{Z}_{\mathsf{S}}$ and $\mathcal{Z}'_{\mathsf{GOD}} = \mathcal{Z}_{\mathsf{GOD}} - \{Z\}$. We design this protocol and go into detail in the following text.
3. MPC protocols in the multiverse that realize any functionality as long as the following property holds:

$$\mathcal{Z}^*_{\mathsf{GOD}} = \{Z\}, \exists Z' \in \mathcal{Z}_{\mathsf{C}}, Z \cup Z' = \mathcal{P} \tag{3}$$

   We construct this by slightly modifying the protocol for the previous setting.

Our main result is a compiler that takes a 3-party fail-stop secure protocol as input and enhances it to obtain a maliciously secure best-in-the-multiverse protocol. In this compiler, the protocols from settings described above are used to compute the next-message functions of the fail-stop secure protocol with all n parties collectively emulating each party in the input protocol. We explain our approach in more detail below.

Compiling to Best-in-the-Multiverse MPC. The fail-stop secure 3-party virtual protocol that is input to the compiler can be used to build an MPC protocol in the client-server paradigm [20], where n clients with the functionality inputs secretly share them among 3 servers. This sharing ensures that at least two servers need to collude to derive the clients' inputs. The servers then interact with each other using the fail-stop secure protocol to compute a similar secret-share of each client's output. This protocol ensures GOD even when one of the servers terminate computation. These output shares are then given to the client as the last message of the protocol, and the client reconstructs its output from these shares. We require the complete protocol to be secure in the presence of an unbounded adversary who corrupts up to $n-1$ clients and 1 server.

Denoting the servers by $\{S_i\}_{i\in[3]}$, their actions in the protocol can be captured as next-message functions: one for each server, for each round. Compiling this to best-in-the-multiverse entails having all n parties together emulate the servers, while each party additionally takes on the role of one client. This emulation is done by first secret-sharing the initial state of each server among all parties. Corresponding to each next-message function, we define an n-party functionality that takes this shared view as input. It reconstructs the view, computes the next-messages, and secret-shares the output among all parties.

The compilation proceeds by first having the initial state of all the servers secret-shared among all parties; and then having all the next-message functionalities of S_1 (resp. S_2 and S_3) executed using MPC protocols in Setting (1) (resp (2) and (3)) as listed in the building-blocks above. All these building-block protocols are maliciously secure assuming broadcast. We show that this compilation yields a best-in-the-multiverse secure protocol in the following way:

- For a malicious unbounded adversary with adversary structure $\mathcal{Z}_{\mathsf{S}} \in \mathcal{Q}^2$, all the next-message functionalities corresponding to S_2 and S_3 are securely realized using protocols in the multiverse settings that tolerate $\mathcal{Z}_{\mathsf{S}}$. This, along with the secret-sharing of intermediate states, ensures that the view of the servers S_2 and S_3 are never compromised. While the view of S_1 may be compromised if $\mathcal{Z}_{\mathsf{S}} \not\subseteq \mathcal{Z}_{\mathsf{GOD}}$, this is simulatable in the virtual protocol.
- Similarly, for a malicious PPT adversary and $\mathcal{Z}_{\mathsf{C}} \subseteq \mathcal{Z}^*_{\mathsf{C}}$, the view of S_2 and S_3 are never compromised and that of S_1 is simulatable in the virtual protocol.
- For any malicious adversary with adversary structure $\mathcal{Z}_{\mathsf{fair}}$, fairness always holds since, again, the view of S_2 and S_3 are never compromised and that of S_1 is simulatable in the virtual protocol.
- For a malicious adversary with adversary structure $\mathcal{Z}^*_{\mathsf{GOD}}$, GOD holds for the views of S_1 and S_3. Since the virtual protocol ensures GOD in the presence of one fail-stop and the final outputs can be derived from 2-out-of-3 secret-sharings from the views of these servers, the compiled protocol satisfies GOD.
- Similarly, for adversary structure $\mathcal{Z}'_{\mathsf{GOD}}$, GOD holds for the views of S_1 and S_2. So the final outputs can be derived from the secret-shares within the views of these servers and the compiled protocol satisfies GOD.

Details of this compilation can be found in Section 5.4. It now remains to explain how the latter two building-block protocols are constructed. We start with a complete outline of the construction of the multiverse protocol that satisfies Equation 2 and then show how to tweak this to satisfy Equation 3.

Protocol for the Setting in Equation 2. We tackle the problem of realizing any n-party functionality in this multiverse setting in two stages; see Figure 2. Let $\mathcal{Z}'_{\mathsf{GOD}}$ and $\mathcal{Z}^*_{\mathsf{C}}$ satisfy Equation 2. In the first step, represented as Theorem 2, we create MPC protocols satisfying GOD tolerating $\mathcal{Z}'_{\mathsf{GOD}}$ and security with abort for PPT adversaries tolerating $\mathcal{Z}^*_{\mathsf{C}}$; where security holds for a computationally unbounded adversary that can corrupt all-but a *designated party*. In the second step, represented as Theorem 4, we build a compiler that takes a

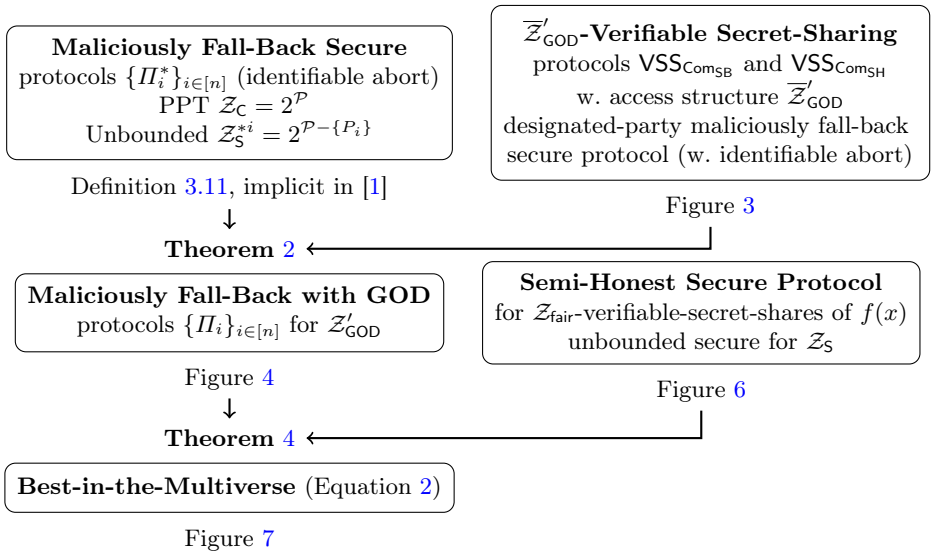


Fig. 2. Overview of our construction for the multiverse setting where $\mathcal{Z}'_{\mathsf{GOD}}$ and $\mathcal{Z}^*_{\mathsf{C}}$ satisfy Equation 2. This Figure 2 is the roadmap for obtaining our protocol of Figure 7.

semi-honest secure MPC protocol and compiles it to the multiverse setting. This is done by having the next-message functions in the virtual protocol executed using the designated party protocols from the first step. This is explained in more detail below.

The first of this two stage process entails constructing the protocols that will become the building-blocks in the compiler. This is formalized in Theorem 2 and detailed in Section 5.1. Here, for each $i \in [n]$, we start with an n-party protocol that is maliciously secure with identifiable abort[8] for arbitrary PPT corruption and for unbounded corruption of any set excluding party $P_i \in \mathcal{P}$. Then Theorem 2 provides a compiler that takes any such protocol and enhances it to a protocol Π_i providing GOD for any $\mathcal{Z}'_{\mathsf{GOD}}$ while restricting security with abort to $\mathcal{Z}^*_{\mathsf{C}}$. The conditions for security in the presence of a PPT and unbounded adversary are inherited from the base protocol, but limited to $\mathcal{Z}^*_{\mathsf{C}}$. This is done using player-emulation techniques [18] and verifiable secret sharing (VSS).

The second stage is the compiler formalized in Theorem 4 and detailed in Section 5.3. Inspired by [1], our main technique uses an MPC-in-the-head paradigm, similar to that of [20]. This compiler takes as input a virtual n-party protocol with semi-honest unbounded security tolerating $\mathcal{Z}_{\mathsf{S}}$. This protocol computes an unbounded secure $\mathcal{Z}_{\mathsf{fair}}$-verifiable-secret-sharing (implied in [17]) of the desired

[8] MPC protocols secure with *identifiable abort* satisfy security with abort, with the identity of the corrupt party causing the protocol to abort revealed to all parties.

functionality output. The compilation upgrades this to a maliciously secure MPC protocol in the multiverse setting satisfying Equation 2. It works by first secret-sharing the initial state of all virtual parties in such a way that any set $Z \in \mathcal{Z}_{\mathsf{C}}^*$ cannot reconstruct. The virtual protocol is parsed as a sequence of next-message functions – one for each party and each round. Each of these functions is computed using the designated party protocol resulting from the first step: all next-message functions of the i^{th} virtual party are computed using protocol Π_i.

This compilation ensures that an unbounded adversary who corrupts a set of parties Z in the compiled protocol will also corrupt exactly those parties in the virtual protocol. Thus, the compiled protocol 'inherits' the unbounded security of the virtual protocol. There are mechanisms to ensure that the outputs of such a protocol are actually used as inputs in the next round of the virtual protocol, if necessary. Finally, all parties reconstruct the output of the virtual protocol and then the $\mathcal{Z}_{\mathsf{fair}}$-verifiable-secret-sharing of the function output $f(x)$, ensuring fairness. We next discuss in detail the complete construction of our protocols in this multiverse setting, including each building-block (Figure 2).

Starting Point: Malicious Fall-Back Security. The input to the compiler in Theorem 2 is an n-party malicious fall-back secure protocol that can compute the functionality. This is a *designated party* protocol Π_i^* that satisfies security with identifiable abort for malicious PPT adversaries that corrupt any subset of the parties, and for unbounded adversaries that corrupt any subset not containing P_i. All parties can only derive outputs in the last round of the protocol. During compilation, this is enhanced to provide GOD for any $\mathcal{Z}'_{\mathsf{GOD}} \in \mathcal{Q}^2$, at the cost of restricting the adversary structure for security with abort.

Verifiable Secret-Sharing (VSS). Let $\mathcal{Z}_{\mathsf{C}}^*$ be the largest adversary structure such that it and $\mathcal{Z}'_{\mathsf{GOD}}$ satisfy Equation 2. Let $\overline{\mathcal{Z}}'_{\mathsf{GOD}}$ represent the set of all compliment sets of those in $\mathcal{Z}'_{\mathsf{GOD}}$. Our protocol requires a verifiable secret sharing (VSS) scheme for the access structure $\overline{\mathcal{Z}}'_{\mathsf{GOD}}$ (resp. adversary structure $\mathcal{Z}_{\mathsf{C}}^*$). Note that every set in $\mathcal{Z}_{\mathsf{C}}^*$ can at best only be a proper subset of one in $\overline{\mathcal{Z}}'_{\mathsf{GOD}}$. This implies that (1) for any corruption beyond $\mathcal{Z}_{\mathsf{C}}^*$, the message is leaked to the adversary; (2) for any corruption within $\mathcal{Z}_{\mathsf{C}}^*$, the adversary can never learn the message, but it may be the case that the honest parties do not have enough shares to reconstruct the secret among themselves; and (3) for any corruption within $\mathcal{Z}'_{\mathsf{GOD}}$, there will always exist a set $\overline{Z} \notin \mathcal{Z}_{\mathsf{C}}^*$ ($\overline{Z} \in \overline{\mathcal{Z}}'_{\mathsf{GOD}}$) of honest parties who can reconstruct the secret. This is constructed using a $\overline{\mathcal{Z}}'_{\mathsf{GOD}}$-secret-sharing scheme as a building-block.

Another property we need from the VSS protocol is 'strong commitment'. That is, if the shares of a message are not generated honestly, then upon reconstruction, the honest parties can still *agree* on the value of the message: either the original shared value or a default value. For this, we use commitment schemes like in the VSS scheme in [12]: secret-share the message, create a commitment for each share, and give each party $P_i \in \mathcal{P}$ the complete set of commitments, the i^{th} secret share, and the randomness for its corresponding commitment. How-

ever, commitment schemes cannot be both statistically hiding and binding at the same time, and this poses a challenge when dealing with unbounded adversaries.

We, therefore, instantiate our VSS scheme in two flavors: $\mathsf{VSS}_{\mathsf{Com}_{\mathsf{SB}}}$, where the underlying commitment scheme is statistically binding (computationally hiding), and $\mathsf{VSS}_{\mathsf{Com}_{\mathsf{SH}}}$, where the commitment scheme is statistically hiding (computationally binding). In compiling our designated-party protocol, we use the latter when sharing the input of this designated party and the former for inputs of any other party. This is explained in more detail in the ensuing text.

Incorporating GOD with Fall-Back Security. Starting with a base protocol that is only secure with identifiable abort, we use an idea similar to that in [18]: use VSS to share each party's inputs and randomness with all other parties. If a party causes the protocol to abort, its inputs and randomness are reconstructed, restarting the protocol; now, each party locally computes that party's messages. Let P_i be the designated party, we combine our base (malicious fall-back secure) protocol Π_i^* with VSS to obtain our protocol Π_i in Theorem 2. This is done in 3 phases:

1. In a 'Commit Phase,' party P_i samples randomness for Π_i^* and uses $\mathsf{VSS}_{\mathsf{Com}_{\mathsf{SH}}}$ to share it among all parties. Since this commitment scheme is statistically hiding, an unbounded adversary cannot violate privacy and any set of shares corresponding to $Z \in \mathcal{Z}_{\mathsf{C}}^*$ reveals no information about the secret. All other parties P_j do the same using $\mathsf{VSS}_{\mathsf{Com}_{\mathsf{SB}}}$.
2. In the 'Protocol Emulation' phase, all the parties execute the fall-back secure protocol Π_i^* using the same inputs and randomness as in the 'Commit Phase.'
3. If the adversary aborts the base protocol, the other parties use the VSS shares from the 'Commit Phase' to reconstruct the input and randomness of that party. If there are enough remaining parties for the VSS reconstruction to succeed, the protocol Π_i^* is restarted, with each party computing the messages of the aborted party locally. Otherwise, if the reconstruction fails, the Π_i protocol is aborted.

Here, the base protocol Π_i^* is malicious fall-back secure with identifiable abort. So if a PPT adversary corrupts $Z \in \mathcal{Z}'_{\mathsf{GOD}}$ and party P_i aborts Π_i^*, privacy is preserved and there are always enough VSS shares with honest parties to reconstruct that party's secrets. If any other party P_j aborts, its randomness can be reconstructed by all honest parties and its masked input can be recovered using P_i. When an unbounded adversary corrupts any set $Z \in \mathcal{Z}'_{\mathsf{GOD}}$ that includes P_i, note that the honest parties can still recover the initial state of the corrupted parties, and so correctness is satisfied even though the privacy of inputs is compromised.[9] If a PPT (or unbounded) adversary corrupts a set of parties $Z \in \mathcal{Z}_{\mathsf{C}}^*$ (or excluding P_i), security with abort follows from the fall-back security of the base protocol and the VSS adversary structure $\mathcal{Z}_{\mathsf{C}}^*$. The details can be found in Section 5.1.

[9] Looking ahead, we will rely on this property in our security analysis.

Compiling to Security in the Multiverse. Given the building-block protocols as above, it remains to describe how the n-party virtual protocol is compiled to be secure in the multiverse. This is formalized in Theorem 4. As mentioned before, the virtual protocol is semi-honest $\mathcal{Z}_{\mathsf{S}}$-secure and computes an unbounded secure $\mathcal{Z}_{\mathsf{fair}}$-verifiable-secret-sharing (implied in [17]) of the output $f(\vec{x})$. It can be rewritten as a sequence of next-message functions that are implemented using the protocols previously described. In addition, to ensure that the inputs of the next-message function are consistent with the previous virtual round in the virtual protocol, the parties use commitment schemes. This is done by having the designated party receive a commitment to each party's input (resp. each party receives a commitment to the designated party's input) to the next-message functionality, while the other party has the corresponding randomness. The next-message functionality works by first verifying the correctness of these commitments before proceeding to compute the next-message and generate commitments to be used in subsequent functionalities. The compiled n-party malicious secure protocol is defined in 3 phases:

1. In the 'Initialize Shared State' phase, each party samples randomness for the virtual protocol and $\mathcal{Z}_{\mathsf{C}}^*$-secret-shares its input and randomness among all parties. All parties now have a secret-share of each virtual party's state.
2. In the 'Compute the Virtual Protocol' phase, each next-message function is executed using the protocol in Theorem 2, with all functions for the virtual party V_i being executed using the protocol Π_i for each $i \in [n]$.
3. Finally, in the 'Derive the Output' phase, each party gives all other parties the shares of the output of the virtual protocol. All parties reconstruct the $\mathcal{Z}_{\mathsf{C}}^*$-secret-shares, and this leaves them with a $\mathcal{Z}_{\mathsf{fair}}$-VSS-share of the function output $f(\vec{x})$ that they can reconstruct with.

Details of this protocol can be found in Section 5.3. Its security is argued for six different cases, depending on the adversary and security guarantee we consider:

1. The first two cases are that of GOD for the PPT and unbounded adversary.
   (a) For the PPT case, security stems from all the commitment schemes being computationally hiding and the protocol implementing all next-message functions providing GOD for a PPT adversary tolerating $\mathcal{Z}'_{\mathsf{GOD}}$.
   (b) For the unbounded case, it follows from two separate arguments. If an unbounded adversary corrupts parties $Z \in \mathcal{Z}'_{\mathsf{GOD}}$, security is broken for all next-message protocols corresponding to these parties in the virtual protocol. However, correctness for honest parties still holds and the views of these parties are simulatable in the virtual protocol. For all parties that the adversary does not corrupt, security follows from the protocol in Theorem 2 satisfying GOD tolerating this unbounded corruption.
2. The next two cases are for fairness for PPT and unbounded adversaries for corruption within $\mathcal{Z}_{\mathsf{fair}}$. This follows from (1) all next-message protocols providing security with abort for corruption beyond $\mathcal{Z}'_{\mathsf{GOD}}$; and (2) the secret-sharing of the output of the virtual protocol. Thus, as in the GOD case, aborting any next-message protocol does not compromise the privacy of the

honest parties. If the protocol runs to completion and the corrupted set lies in $\mathcal{Z}_{\mathsf{fair}}$, there will always be enough honest shares to reconstruct the output.
3. Security with abort for PPT and unbounded adversaries follows from the security of all next-message protocols and secret-sharing of intermediate views.

Protocol for the Setting in Equation 3. In this setting, $\mathcal{Z}^*_{\mathsf{GOD}}$ consists of a single set $Z \subset \mathcal{P}$ whose complement is in $\mathcal{Z}^*_{\mathsf{C}}$. Here the VSS-based approach in the previous setting fails since the adversary can have enough shares to reconstruct the view. However, since $\mathcal{Z}^*_{\mathsf{GOD}}$ contains only one set, the input consistency that needs to hold between different sets is no longer required.

A protocol for this setting can be constructed in two steps. In the first stage, we show in Theorem 3 that one can compile a fall-back secure protocol with identifiable abort and unbounded security for a designated party to one that provides GOD in the presence of $\mathcal{Z}^*_{\mathsf{GOD}}$ without affecting $\mathcal{Z}^*_{\mathsf{C}}$ at all. This is done by running the protocol as is, and when a party within $\mathcal{Z}^*_{\mathsf{GOD}}$ aborts the computation, the protocol is simply re-started, assuming default inputs and outputs for that party. Details of this compilation can be found in Section 5.2. Then in the second stage such protocols are used for compiling a semi-honest secure protocol to one in the multiverse exactly as in the previous setting (Theorem 4).

3 Preliminaries

NOTATION. We denote by κ a computational security parameter and s a statistical security parameter that captures a statistical error of up to 2^{-s}. We assume $s \leq \kappa$. We denote by $\mathcal{P} = \{P_i\}_{i\in[n]}$ the set of n parties participating in MPC and by $\mathcal{A}$ the adversary. We denote by $Z \in \mathcal{P}$, a set of corrupt parties and by $\mathcal{Z} \in 2^{\mathcal{P}}$ an adversary structure. Throughout, we denote by $\mathcal{Z}_{\mathsf{GOD}}$, $\mathcal{Z}_{\mathsf{fair}}$, $\mathcal{Z}_{\mathsf{S}}$ and $\mathcal{Z}_{\mathsf{C}}$ respectively the adversary structures for guaranteed output delivery (GOD), fairness, security with abort for unbounded adversaries and security with abort for PPT adversaries. We denote by $\mathcal{Z}'_{\mathsf{GOD}}$ and $\mathcal{Z}^*_{\mathsf{GOD}}$ two partitions of $\mathcal{Z}_{\mathsf{GOD}}$ such that $\mathcal{Z}^*_{\mathsf{GOD}} = \{Z\}$ contains a single set. We denote by $\mathcal{Z}^*_{\mathsf{C}}$ the largest adversary structure (containing $\mathcal{Z}_{\mathsf{GOD}}$, $\mathcal{Z}_{\mathsf{fair}}$, $\mathcal{Z}_{\mathsf{S}}$ and $\mathcal{Z}_{\mathsf{C}}$) that does not contain the complement of any set in $\mathcal{Z}'_{\mathsf{GOD}}$. By 'fail-stop' corruption we refer to the case where an adversary can cause a corrupt party to abort protocol execution at will, but corrupt parties behave otherwise semi-honestly during the protocol execution.

We attribute the definitions of the MPC models and other building-blocks to [1].

Definition 3.1 *Two probability ensembles $X = \{X_n\}_{n\in\mathbb{N}}$ and $Y = \{Y_n\}_{n\in\mathbb{N}}$ defined over a finite domain D are* ***statistically indistinguishable****, denoted $X \overset{\mathrm{s}}{\equiv} Y$, if every positive polynomial $p(\cdot)$ and all sufficiently large n's,*

$$\Delta(X_n, Y_n) < \frac{1}{p(n)}$$

where,

$$\Delta(X_n, Y_n) = \frac{1}{2} \cdot \sum_{\alpha \in D} |\Pr[X_n = \alpha] - \Pr[Y_n = \alpha]|$$

Definition 3.2 *Two probability ensembles $X = \{X_n\}_{n\in\mathbb{N}}$ and $Y = \{Y_n\}_{n\in\mathbb{N}}$ are **computationally indistinguishable**, denoted $X \overset{c}{\equiv} Y$, if for every PPT distinguisher D, every positive polynomial $p(\cdot)$ and all sufficiently large n's,*

$$|\Pr[D(X_n, 1^n) = 1] - \Pr[D(Y_n, 1^n) = 1]| < \frac{1}{p(n)}$$

Commitment Schemes. A commitment scheme Com is a 2-party protocol between a committer C with an input x and a receiver R in which C samples randomness r and creates a 'commitment' to the value x. This commitment should be hidden: no information about x should be revealed. At a later point in time, C can send a 'decommit' message to R showing that the value committed to was indeed x. This property by which C cannot deceive R into believing that the value committed to was something different $x' \neq x$ is called the binding property of the commitment. Our protocols use statistically binding commitments to ensure that an unbounded malicious adversary cannot deviate from the protocol.

Definition 3.3 *$\mathsf{Com}_{\mathsf{SB}}(m; r)$ is a **Statistically Binding Commitment** for messages $m \in \{0,1\}^\ell$ if it satisfies:*

- ***Statistical Binding:*** *$\forall$ adversary $\mathcal{A}$ there exists a negligible function μ s.t.,*

$$\Pr[(c, m_0, r_0, m_1, r_1) \leftarrow \mathcal{A}(1^s) | c = \mathsf{Com}_{\mathsf{SB}}(m_0; r_0); \; c = \mathsf{Com}_{\mathsf{SB}}(m_1; r_1); \; m_0, m_1 \in \{0,1\}^\ell \;\;] < \mu(s)$$

- ***Computationally Hiding:*** *For all choice of $m_0, m_1 \in \{0,1\}^\ell$ it holds that,*

$$\{\mathsf{Com}_{\mathsf{SB}}(m_0; r)\}_{r\in\{0,1\}^*} \overset{c}{\equiv} \{\mathsf{Com}_{\mathsf{SB}}(m_1; s)\}_{s\in\{0,1\}^*}$$

- ***Decommitment*** *to a commitment τ is (m, r) such that $\tau = \mathsf{Com}_{\mathsf{SB}}(m; r)$*

Since the binding and hiding properties of commitments contradict, a commitment scheme cannot be statistically binding and hiding. But there exist commitment schemes that are either statistically hiding and computationally binding or statistically binding and computationally hiding. In our work, we require both kinds of commitment schemes.

Definition 3.4 *A non-interactive **Statistically Hiding Commitment** scheme for a message space $\{0,1\}^\ell$ is a PPT algorithm $\mathsf{Com}_{\mathsf{SH}}$ satisfying:*

– ***Computationally Binding:*** *$\forall$ PPT adversary $\mathcal{A}$ there exists a negligible function μ such that,*

$$\Pr[(c, m_0, r_0, m_1, r_1) \leftarrow \mathcal{A}(1^\kappa) | c = \mathsf{Com}_{\mathsf{SH}}(m_0; r_0); \\ c = \mathsf{Com}_{\mathsf{SH}}(m_1; r_1); \\ m_0, m_1 \in \{0,1\}^\ell \quad] < \mu(\kappa)$$

– ***Statistically Hiding:*** *For all choice of $m_0, m_1 \in \{0,1\}^\ell$ it holds that,*

$$\{\mathsf{Com}_{\mathsf{SH}}(m_0; r)\}_{r \in \{0,1\}^*} \stackrel{\text{s}}{\equiv} \{\mathsf{Com}_{\mathsf{SH}}(m_1; s)\}_{s \in \{0,1\}^*}$$

– ***Decommitment*** *to a commitment c is (m, r) such that $c = \mathsf{Com}_{\mathsf{SH}}(m; r)$*

Secure Multiparty Computation. Let $\mathcal{P} = \{P_i\}_{i \in [n]}$ be a set of n parties. MPC involves the computation of a random process that maps a tuple of inputs to a tuple of outputs (one for each party). This random process is referred to as a functionality $f : \{0,1\}^* \times \cdots \times \{0,1\}^* \to \{0,1\}^* \times \cdots \times \{0,1\}^*$ where $f = (f_1, \ldots, f_n)$. Each party P_i has input x_i. For simplicity and without loss of generality, we work over functionalities for which $x_i \in \{0,1\}$. For every tuple of inputs $\vec{x} = (x_i, \ldots, x_n)$, the vector output by f is a random variable $(f_1(\vec{x}), \ldots, f_n(\vec{x}))$ ranging over tuples of strings where P_i receives $f_i(\vec{x})$. The following notation describes a functionality:

$$(x_1, \ldots, x_n) \mapsto (f_1(\vec{x}), \ldots, f_n(\vec{x}))$$

We prove the security of our MPC protocols in the presence of malicious adversaries in the dishonest majority setting.

In the semi-honest setting, an adversary can control a subset of the participating parties and always follows the protocol specification but may try to learn additional information from the transcript of messages received and its internal state. Let $f = (f_1, \ldots, f_n)$ be a multiparty functionality and Π be an n-party protocol computing f in t rounds. The view of party P_i in an execution of Π on inputs $\vec{x} = (x_1, \ldots, x_n)$ is,

$$\mathbf{View}_{\Pi,i}(\kappa, \vec{x}) = (\kappa, x_i, r_i, \{m^j_{k \to i}\}_{j \in [t], k \neq i \in [n]})$$

where κ is the security parameter, r_i is the content of P_i's internal random tape, and each $m^j_{k \to i}$ represents the message received from party P_k in round j. The output of P_i in an execution of Π on $\vec{x}$ is denoted $\mathsf{output}_{\Pi,i}(\vec{x})$ and can be computed from $\mathbf{View}_{\Pi,i}(\kappa, \vec{x})$. The set of corrupted parties is denoted by $Z \subset \mathcal{P}$ and the set of honest parties by $\overline{Z}$. We extend the view notation to capture any subset of parties, denoting by $\mathbf{View}_{\Pi,T}(\kappa, \vec{x})$ the joint view of all parties in $T \subseteq \mathcal{P}$ on $(\kappa, \vec{x})$.

Definition 3.5 *Let f and Π be as above, and let $\mathcal{Z}_{\mathsf{C}} \subseteq 2^{\mathcal{P}}$ be an adversary structure. The protocol Π is said to securely compute f in the presence of a*

PPT semi-honest adversary *tolerating adversary structure* $\mathcal{Z}_{\mathsf{C}}$ *if there exists a PPT simulator* Sim *such that for every* $Z \in \mathcal{Z}_{\mathsf{C}}$,

$$\left\{\mathsf{Sim}(1^{\kappa}, Z, \{x_i, f_i(\vec{x})\}_{i\in Z}), \{f_i(\vec{x})\}_{i\notin Z}\right\}_{\kappa\in\mathbb{N},\vec{x}\in\{0,1\}^*}$$
$$\stackrel{\mathrm{c}}{\equiv}\left\{\mathbf{View}_{\Pi,Z}(\kappa,\vec{x}), \mathsf{output}_{\Pi,\overline{Z}}(\kappa,\vec{x})\right\}_{\kappa\in\mathbb{N},\vec{x}\in\{0,1\}^*}$$

where κ *is the computational security parameter. The distribution is considered over the randomness of the simulator and random tapes of parties in the protocol.*

Definition 3.6 *Let* f *and* Π *be as above, and let* $\mathcal{Z}_{\mathsf{S}} \subseteq 2^{\mathcal{P}}$ *be an adversary structure. The protocol* Π *is said to securely compute* f *in the presence of an* ***unbounded semi-honest adversary*** *tolerating adversary structure* $\mathcal{Z}_{\mathsf{S}}$ *if there exists a PPT simulator* Sim *such that for every* $Z \in \mathcal{Z}_{\mathsf{S}}$,

$$\left\{\mathsf{Sim}(1^{s}, Z, \{x_i, f_i(\vec{x})\}_{i\in Z}), \{f_i(\vec{x})\}_{i\notin Z}\right\}_{s\in\mathbb{N},\vec{x}\in\{0,1\}^*}$$
$$\stackrel{\mathrm{s}}{\equiv}\left\{\mathbf{View}_{\Pi,Z}(s,\vec{x}), \mathsf{output}_{\Pi,\overline{Z}}(s,\vec{x})\right\}_{s\in\mathbb{N},\vec{x}\in\{0,1\}^*}$$

where s *is the statistical security parameter. The distribution is considered over the simulator's randomness and the parties' random tapes in the protocol.*

In the malicious setting, an adversary controlling a subset of the participating parties can have them deviate arbitrarily from the protocol specification. Security in this setting is defined by comparing an execution of the functionality in an ideal model and an execution of the protocol in the real model. Depending on the security guarantee required, the ideal model execution is defined differently for the case of security with abort, fairness, and guaranteed output delivery (GOD).

Execution in the ideal model (Security with Abort). An ideal execution is an interaction between all the parties in $\mathcal{P}$ and a trusted party $\mathcal{F}$, wherein the parties submit their input to the trusted party that computes the output and returns it. An honest party receives input for the computation and forwards it to the trusted party. In contrast, a corrupt party can replace its input with an arbitrary value of the same length. On receiving these, $\mathcal{F}$ computes the output and first sends the outputs of the corrupt parties to the adversary. The adversary then decides whether the honest parties will receive their outputs from $\mathcal{F}$ or an abort symbol $\perp$. In the case of identifiable abort, in the latter case, the honest parties additionally receive an index $i \in [n]$ of the corrupt party that the adversary provides to $\mathcal{F}$. For a multiparty functionality $f = (f_1, \ldots, f_n)$, let $\mathcal{A}$ be a non-uniform probabilistic machine and $Z \subset \mathcal{P}$ be the set of corrupted parties. Then the ideal execution of f on inputs $(\kappa, \vec{x})$, auxiliary input z to $\mathcal{A}$ and security parameter κ is defined as the output tuple of the honest parties and

the adversary from this execution and is denoted as $\mathbf{IDEAL}^{\perp}_{f,\mathcal{A}(z),Z}(\kappa,\vec{x})$.

Execution in the ideal model (Fairness). This is a similar ideal execution as described above. Here again, honest parties receive inputs for the computation and forward it to the trusted party. A corrupt party can replace its input with an arbitrary value of the same length. If these inputs contain an abort symbol $\perp$, $\mathcal{F}$ does not compute the output, and none of the parties receive it. Instead, $\mathcal{F}$ sends $\perp$ to the honest parties. Otherwise, $\mathcal{F}$ computes the output and simultaneously sends it to *both the adversary and the honest parties*. For a multiparty functionality $f = (f_1, \ldots, f_n)$, let $\mathcal{A}$ be a non-uniform probabilistic machine and $Z \subset \mathcal{P}$ be the set of corrupted parties. Then the ideal execution of f on inputs $(\kappa,\vec{x})$, auxiliary input z to $\mathcal{A}$ and security parameter κ is defined as the output tuple of the honest parties and the adversary from this execution and is denoted as $\mathbf{IDEAL}^{\mathsf{fair}}_{f,\mathcal{A}(z),Z}(\kappa,\vec{x})$.

Execution in the ideal model (GOD). In this ideal execution, honest parties receive inputs for the computation and forward it to the trusted party. A corrupt party can replace its input with an arbitrary value of the same length. On receiving all inputs, $\mathcal{F}$ *always* computes the output and simultaneously sends it to both the adversary and the honest parties. For a multiparty functionality $f = (f_1, \ldots, f_n)$, let $\mathcal{A}$ be a non-uniform probabilistic machine and $Z \subset \mathcal{P}$ be the set of corrupted parties. Then the ideal execution of f on inputs $(\kappa,\vec{x})$, auxiliary input z to $\mathcal{A}$ and security parameter κ is defined as the output tuple of the honest parties and the adversary from this execution and is denoted as $\mathbf{IDEAL}^{\mathsf{GOD}}_{f,\mathcal{A}(z),Z}(\kappa,\vec{x})$.

Execution in the real model. This is an interaction directly among the parties in $\mathcal{P}$ wherein the honest parties follow the instructions in the protocol Π. The adversary $\mathcal{A}$ sends all messages in place of the corrupted parties and may follow any arbitrary strategy. Letting Π be a protocol that computes f as above, $\mathcal{A}$ be a non-uniform probabilistic machine, and let $Z \subset \mathcal{P}$ be the set of corrupted parties, the real execution of Π on inputs $(\kappa,\vec{x})$, auxiliary input z to $\mathcal{A}$ and security parameter κ is defined as the output tuple of the honest parties and the adversary from this execution and is denoted as $\mathbf{REAL}_{\Pi,\mathcal{A}(z),Z}(\kappa,\vec{x})$.

Definition 3.7 *Let f and Π be as above, and let $\mathcal{Z}_{\mathsf{C}} \subseteq 2^{\mathcal{P}}$ be an adversary structure. The protocol Π is said to securely compute f with abort (resp. fairness, GOD) in the presence of **PPT malicious adversaries** tolerating adversary structure $\mathcal{Z}_{\mathsf{C}}$ if there exists a PPT adversary Sim in the ideal model such that for every PPT $\mathcal{A}$ in the real model and $Z \in \mathcal{Z}_{\mathsf{C}}$,*

$$\left\{\mathbf{IDEAL}^{\perp}_{f,\mathsf{Sim}(z),Z}(\kappa,\vec{x})\right\}_{\kappa\in\mathbb{N},\vec{x},z\in\{0,1\}^*} \overset{\mathrm{c}}{\equiv} \left\{\mathbf{REAL}_{\Pi,\mathcal{A}(z),Z}(\kappa,\vec{x})\right\}_{\kappa\in\mathbb{N},\vec{x},z\in\{0,1\}^*}$$

resp.,

$$\left\{\mathbf{IDEAL}^{\mathsf{fair}}_{f,\mathsf{Sim}(z),Z}(s,\vec{x})\right\}_{s\in\mathbb{N},\vec{x},z\in\{0,1\}^*} \overset{\mathrm{c}}{\equiv} \left\{\mathbf{REAL}_{\Pi,\mathcal{A}(z),Z}(s,\vec{x})\right\}_{s\in\mathbb{N},\vec{x},z\in\{0,1\}^*}$$

$$\left\{\mathbf{IDEAL}^{\mathsf{GOD}}_{f,\mathsf{Sim}(z),Z}(s,\vec{x})\right\}_{s\in\mathbb{N},\vec{x},z\in\{0,1\}^*} \stackrel{\mathrm{c}}{\equiv} \left\{\mathbf{REAL}_{\Pi,\mathcal{A}(z),Z}(s,\vec{x})\right\}_{s\in\mathbb{N},\vec{x},z\in\{0,1\}^*}$$

where κ is the computational security parameter. The distribution is considered over the adversaries' randomness and all the parties' random tapes.

Definition 3.8 *Let f and Π be as above, and let $\mathcal{Z}_{\mathsf{S}} \subseteq 2^{\mathcal{P}}$ be an adversary structure. The protocol Π is said to securely compute f with abort (resp. fairness, GOD) in the presence of **unbounded malicious adversaries** tolerating adversary structure $\mathcal{Z}_{\mathsf{S}}$ if there exists an adversary Sim in the ideal model such that for every computationally unbounded $\mathcal{A}$ in the real model and every $Z \in \mathcal{Z}_{\mathsf{S}}$,*

$$\left\{\mathbf{IDEAL}^{\perp}_{f,\mathsf{Sim}(z),Z}(s,\vec{x})\right\}_{s\in\mathbb{N},\vec{x},z\in\{0,1\}^*} \stackrel{\mathrm{s}}{\equiv} \left\{\mathbf{REAL}_{\Pi,\mathcal{A}(z),Z}(s,\vec{x})\right\}_{s\in\mathbb{N},\vec{x},z\in\{0,1\}^*}$$

resp.,

$$\left\{\mathbf{IDEAL}^{\mathsf{fair}}_{f,\mathsf{Sim}(z),Z}(s,\vec{x})\right\}_{s\in\mathbb{N},\vec{x},z\in\{0,1\}^*} \stackrel{\mathrm{s}}{\equiv} \left\{\mathbf{REAL}_{\Pi,\mathcal{A}(z),Z}(s,\vec{x})\right\}_{s\in\mathbb{N},\vec{x},z\in\{0,1\}^*}$$

$$\left\{\mathbf{IDEAL}^{\mathsf{GOD}}_{f,\mathsf{Sim}(z),Z}(s,\vec{x})\right\}_{s\in\mathbb{N},\vec{x},z\in\{0,1\}^*} \stackrel{\mathrm{s}}{\equiv} \left\{\mathbf{REAL}_{\Pi,\mathcal{A}(z),Z}(s,\vec{x})\right\}_{s\in\mathbb{N},\vec{x},z\in\{0,1\}^*}$$

where s is the statistical security parameter. The distribution is considered over the adversaries' randomness and all the parties' random tapes.

Definition 3.9 (Correctness [19]) *We say that Π realizes a deterministic n-party functionality $f(\vec{x})$ with perfect correctness if for all inputs $\vec{x}$ the probability that the output of some player is different from the output of f is 0, where the probability is over the independent choices of the random inputs $r_1, ..., r_n$.*

Definition 3.10 (Robustness [19]) *We say that Π realizes f with perfect robustness for access structure $\mathcal{Z} \subseteq 2^{\mathcal{P}}$ if it is perfectly correct (Definition 3.9) in the presence of any computationally unbounded malicious adversary corrupting a set $Z \in \mathcal{Z}$ of players, and for any inputs $\vec{x}$, the following robustness property holds: if there is no $\vec{x}'$ such that $f(\vec{x}') = 1$, then the probability that some honest player outputs 1 in an execution of Π in which the inputs of the honest players are consistent with $\vec{x}$ is 0.*

MPC WITH FALL-BACK SECURITY. Informally, a multi-party protocol satisfies fall-back security if it is secure when certain subsets of parties are corrupted by a computationally unbounded adversary. However, if more parties than that are corrupted, computational security is still guaranteed.

Definition 3.11 (Malicious Fall-Back Security) *For parties $\mathcal{P} = \{P_i\}_{i\in[n]}$ where P_i has input x_i, let $\Pi = \langle\mathcal{P}\rangle$ be an n-party protocol computing a function f with input $\vec{x} = \{x_i\}_{i\in[n]}$. Let $\mathcal{Z}_{\mathsf{C}} = 2^{\mathcal{P}}$ and $\mathcal{Z}_{\mathsf{S}} \subseteq \mathcal{Z}_{\mathsf{C}}$ be adversary structures*

for PPT and computationally unbounded malicious adversaries, respectively. The protocol Π satisfies ***fall-back security in the malicious setting tolerating*** $\mathcal{Z}_{\mathsf{S}}$ *if there exists a PPT adversary* Sim *in the ideal model such that for every PPT* $\mathcal{A}$ *in the real model and every* $Z \in \mathcal{Z}_{\mathsf{C}}$*,*

$$\left\{\mathbf{IDEAL}^{\perp}_{f,\mathsf{Sim}(z),Z}(\kappa,\vec{x})\right\}_{\kappa\in\mathbb{N},\vec{x},z\in\{0,1\}^*} \overset{\mathrm{c}}{\equiv} \left\{\mathbf{REAL}_{\Pi,\mathcal{A}(z),Z}(\kappa,\vec{x})\right\}_{\kappa\in\mathbb{N},\vec{x},z\in\{0,1\}^*}$$

where κ *is the computational security parameter; and there exists an adversary* Sim *in the ideal model such that for every computationally unbounded* $\mathcal{A}$ *in the real model and every* $Z \in \mathcal{Z}_{\mathsf{S}}$*,*

$$\left\{\mathbf{IDEAL}^{\perp}_{f,\mathsf{Sim}(z),Z}(s,\vec{x})\right\}_{s\in\mathbb{N},\vec{x},z\in\{0,1\}^*} \overset{\mathrm{s}}{\equiv} \left\{\mathbf{REAL}_{\Pi,\mathcal{A}(z),Z}(s,\vec{x})\right\}_{s\in\mathbb{N},\vec{x},z\in\{0,1\}^*}$$

where s *is the statistical security parameter. The distribution is considered over the randomness of the simulator and all random tapes of parties in the protocol.*

Secret Sharing. A secret-sharing scheme [25] is a tuple of PPT functions $\mathsf{SS} = (\mathsf{Share}, \mathsf{Recon})$. In its complete generality, it is defined with security against an adversary structure $\mathcal{Z} \subset 2^{\mathcal{P}}$ over n parties. For a message m, the function $\mathsf{Share}_{\mathcal{Z}}(m)$ returns n values $\{[m]_i\}_{i\in[n]}$ such that for any set $Z \in \mathcal{Z}$, the set of shares $\{[m]_i\}_{P_i\in Z}$ reveal no information about m, even to a computationally unbounded adversary. However, for any set $Z \notin \mathcal{Z}$, the set of shares $\{[m]_i\}_{P_i\in Z}$ reveal the message by computing $m = \mathsf{Recon}_{\mathcal{Z}}(\{[m]_i\}_{P_i\in Z})$. Such a scheme exists for any general adversary structure $\mathcal{Z}$ [4].

A n-party Verifiable Secret Sharing (VSS) scheme defined over an adversary structure $\mathcal{Z}$ is a tuple of PPT functions $\mathsf{VSS}^{\mathcal{Z}} = (\mathsf{Share}, \mathsf{Recon})$. It involves a possibly malicious dealer D with input x, whose honest actions are captured by $\{[m]_i\}_{i\in[n]} \leftarrow \mathsf{Share}_{\mathcal{Z}}(m)$. This is such that for any set $Z \in \mathcal{Z}$, the set of shares $\{[m]_i\}_{P_i\in Z}$ reveals no information about m, which may hold in the presence of a PPT or a computationally unbounded adversary. For any set $Z \notin \mathcal{Z}$, the corresponding parties can pool their set of shares $\{[m]_i\}_{P_i\in Z}$ and reveal the message by computing $m = \mathsf{Recon}_{\mathcal{Z}}(\{[m]_i\}_{P_i\in Z})$. Similar to secret-sharing, these properties must hold when these algorithms are honestly executed. However, VSS differs from secret-sharing described earlier in the security guarantees that are required even in the presence of malicious parties. Namely, a malicious dealer D' cannot make a set of honest parties *disagree* on the reconstructed value, even if it colludes with other parties, for any $Z \in \mathcal{Z}$. Any set of honest parties $\overline{Z} \subseteq \mathcal{P}$ will either reconstruct a unique value m' if they compute $\mathsf{Recon}_{\mathcal{Z}}(\{[m]^*_i\}_{P_i\in\overline{Z}})$, or unanimously abort with output $\perp$.

Definition 1 ($\mathcal{Z}$**-Secure Verifiable Secret Sharing [5]).** *Let* $\mathsf{VSS}^{\mathcal{Z}} = (\mathsf{Share}, \mathsf{Recon})$ *be a pair of protocols for the parties in* $\mathcal{P}$*, where a designated dealer* $D \in \mathcal{P}$ *has some private input* $s \in \mathcal{S}$*, where* $\perp \in \mathcal{S}$*, for the protocol* Share*. Then* $(\mathsf{Share}, \mathsf{Recon})$ *is called a perfectly-secure VSS scheme if the following requirements hold:*

– ***Privacy***: *If D is honest, then the view of $\mathcal{A}$ with adversary structure $\mathcal{Z}$ during* Share *is distributed independent of s.*
– ***Correctness***: *If D is honest, then all honest parties output s after* Recon.
– ***Strong Commitment***: *Even if D is corrupt, in any execution of* Share*, the joint view of the honest parties defines a unique value $s^* \in \mathcal{S}$ (which could be different from s), such that all honest parties output s^* at the end of* Recon*, irrespective of the behavior of $\mathcal{A}$.*

Looking ahead, whenever we refer to a $\mathcal{Z}$-secure VSS, the corresponding $\mathcal{Z}$ is the adversary structure, as indicated in Definition 1 above. On the other hand, when we refer to $\mathcal{Z}$ VSS, we refer to an *access structure* $\mathcal{Z}$ that contains all the sets that are a strict super-set of the members in the adversary structure. In our results, we use two types of VSS protocols. The first is a VSS protocol secure in the presence of computationally unbounded adversaries. Such protocols exist only for $\mathcal{Q}^2$ adversary structures and are implied by [17]. The second kind of VSS we require needs to satisfy adversary structures beyond $\mathcal{Q}^2$. We design two variants of such a protocol: one satisfying *privacy* in the presence of an unbounded adversary; and another satisfying *strong commitment* for unbounded adversaries. Note that VSS protocols for adversary structures beyond $\mathcal{Q}^2$ cannot unbounded-securely provide both these guarantees simultaneously. We construct these by extending the VSS scheme (PPT secure for threshold adversaries) in Chapter 7 of [12] to general adversary structures (beyond $\mathcal{Q}^2$).

4 Feasibility of General MPC in the Multiverse

In this section, we provide lower-bound conditions under which MPC is simultaneously feasible in the presence of general non-threshold adversaries across multiple characterizations. In Section 5, we construct an upper-bound protocol that matches this lower-bound feasibility. We assume the existence of secure point-to-point and broadcast channels between all parties.

In the plain model with stand-alone security, it is already known from [15] that MPC is only achievable if $\mathcal{Z}_{\mathsf{GOD}}, \mathcal{Z}_{\mathsf{fair}}, \mathcal{Z}_{\mathsf{S}} \in \mathcal{Q}^2$. We also know that fairness is a strictly more general guarantee than GOD, and therefore, it must hold that every set $Z \in \mathcal{Z}_{\mathsf{GOD}}$ is a subset of some set $Z' \in \mathcal{Z}_{\mathsf{fair}}$. It is also known from [8] that, assuming broadcast, GOD is achievable for any $\mathcal{Z}_{\mathsf{fair}}$. Further, security for unbounded adversaries is achievable for any feasible $\mathcal{Z}_{\mathsf{fair}}$ and $\mathcal{Z}_{\mathsf{GOD}}$. However, these structures pose no restrictions on how large $\mathcal{Z}_{\mathsf{S}}$ can be with respect to security with abort. For an adversary structure for MPC with security with abort for PPT adversaries $\mathcal{Z}_{\mathsf{C}}$, any corruption beyond this renders the protocol insecure. Therefore, it follows from this that all sets $Z \in \mathcal{Z}_{\mathsf{GOD}}, \mathcal{Z}_{\mathsf{fair}}, \mathcal{Z}_{\mathsf{S}}$ need to be a subset of a set $Z' \in \mathcal{Z}_{\mathsf{C}}$. This summarizes the restrictions on the structures $\mathcal{Z}_{\mathsf{GOD}}, \mathcal{Z}_{\mathsf{fair}}, \mathcal{Z}_{\mathsf{S}}$ as applicable in the multiverse.

It now remains to characterize the feasibility of $\mathcal{Z}_{\mathsf{C}} \subseteq 2^{\mathcal{P}}$ with respect to the other adversary structures. It is known from [1] that $\mathcal{Z}_{\mathsf{S}}$ poses no restriction on how large $\mathcal{Z}_{\mathsf{C}}$ can be and vice-versa. We can also extend the result in [24]

for the threshold case to derive that $\mathcal{Z}_{\mathsf{fair}}$ similarly poses no restriction on $\mathcal{Z}_{\mathsf{C}}$. However, [18] shows that in the threshold case $\mathcal{Z}_{\mathsf{GOD}}$ already imposes a restriction on the size of $\mathcal{Z}_{\mathsf{C}}$. That is, letting t and s be the thresholds for GOD and security with abort respectively, MPC is not possible when $\mathsf{t}+\mathsf{s} \geq n$. We generalize this to the non-threshold case to yield the following result.

Lemma 1. *Let $\mathcal{P} = \{P_i\}_{i\in[n]}$ be the set of n parties. Let $\mathcal{Z}_{\mathsf{GOD}} \subseteq 2^{\mathcal{P}}$ be the adversary structure for MPC with guaranteed output delivery (Definitions 3.7,3.8 – GOD) and $\mathcal{Z}_{\mathsf{C}} \subseteq 2^{\mathcal{P}}$ be the largest adversary structure for MPC with PPT security with abort (Definitions 3.7,3.8 – abort) such that the following holds:*

$$\exists Z_1, Z_2 \in \mathcal{Z}_{\mathsf{GOD}} \text{ s.t. } (Z_1 - Z_2 \neq \phi) \text{ , } (Z_2 - Z_1 \neq \phi) \text{ and } (\mathcal{P} - Z_1, \mathcal{P} - Z_2 \in \mathcal{Z}_{\mathsf{C}})$$

Then there exists an n-party functionality f for which no protocol Π can compute f, even in the presence of fail-stop adversaries, tolerating both $\mathcal{Z}_{\mathsf{GOD}}$ and $\mathcal{Z}_{\mathsf{C}}$.

Lemma 1 states that if $\mathcal{Z}_{\mathsf{GOD}}$ contains two or more distinct sets, where one does not completely contain the other, security with abort is *not* achievable if the adversary is allowed to corrupt one of both their complementary sets in $\mathcal{P}$.

Proof Outline. We extend the proof technique used by [18] for the threshold case. We first define an n party functionality, which we claim cannot be computed if the above condition holds. We split the parties in $\mathcal{P}$ into 3 sets: $A = Z_1$, $C = Z_2$, and $B = \mathcal{P} - (A \cup C)$. Note that an adversary is able to corrupt $A \cup B \subseteq Z_2'$ or $B \cup C \subseteq Z_1'$. We argue that for any round k in the protocol, if an adversary who corrupts $A \cup B$ (resp. $B \cup C$) causes all parties in A (resp. C) to abort, then the protocol must satisfy GOD. However, then the joint view of the adversary corrupting all parties in B, in real execution together with the output of the honest party in C (resp. in A), is distinguishable from that in the ideal execution, violating privacy.

The complete proof can be found in Appendix A. As a result of the above, we also have the following.

Theorem 1. *Let $\mathcal{P} = \{P_i\}_{i\in[n]}$ be n parties. Let $\mathcal{Z}_{\mathsf{GOD}} \in \mathcal{Q}^2$ be the adversary structure for GOD and $\mathcal{Z}_{\mathsf{C}}$ be the largest adversary structure for PPT security with abort such that Lemma 1 holds. Let $\mathcal{Z}_{\mathsf{fair}} \in \mathcal{Q}^2$ and $\mathcal{Z}_{\mathsf{S}} \in \mathcal{Q}^2$ be adversary structures for MPC with fairness and unbounded adversaries, respectively. Then MPC is feasible only when,*

$$\forall Z \in \mathcal{Z}_{\mathsf{GOD}}, \mathcal{Z}_{\mathsf{fair}}, \mathcal{Z}_{\mathsf{S}}, \exists Z' \in \mathcal{Z}_{\mathsf{C}}, Z \subseteq Z'$$
$$\forall Z \in \mathcal{Z}_{\mathsf{GOD}}, \exists Z' \in \mathcal{Z}_{\mathsf{fair}}, Z \subseteq Z'$$

Proof. The proof for this follows from the fact that firstly, following Lemma 1, there is no security for corruption beyond $\mathcal{Z}_{\mathsf{C}}$. Therefore, secure MPC for unbounded adversaries, with fairness or GOD, is only possible when the set of corrupted parties $Z \subseteq \mathcal{P}$ is contained within some set $Z' \in \mathcal{Z}_{\mathsf{C}}$. Secondly, the fairness security guarantee generalizes GOD. If an MPC protocol guarantees fairness only for the adversary structure $\mathcal{Z}_{\mathsf{fair}}$ and not beyond this, it follows that GOD cannot be achieved for any $Z \subseteq \mathcal{P}$ that is not contained in $\mathcal{Z}_{\mathsf{fair}}$.

5 Best in the Multiverse Protocol

In this section, we present a protocol that computes arbitrary n-party functionalities in the presence of active adversaries that achieves best-in-the-multiverse security, taking into account the following general adversary structures taken over $\mathcal{P}$: $\mathcal{Z}_{\mathsf{GOD}}$ for guaranteed output delivery (GOD), $\mathcal{Z}_{\mathsf{fair}}$ for fairness, $\mathcal{Z}_{\mathsf{S}}$ for security with abort in the presence of unbounded adversaries, and $\mathcal{Z}_{\mathsf{C}}$ for security with abort in the presence of PPT adversaries. We assume the existence of secure point-to-point and broadcast channels between all parties. Let $\mathcal{P} = \{P_i\}_{i\in[n]}$ be the set of n parties, and let $\mathcal{Z}^*_{\mathsf{C}} \subseteq 2^{\mathcal{P}}$ be the largest general adversary structure for security with abort in the presence of PPT adversaries where,

$$\neg\exists Z_1, Z_2 \in \mathcal{Z}_{\mathsf{GOD}},\ \text{s.t. } (\mathcal{P} - Z_1 \in \mathcal{Z}^*_{\mathsf{C}}) \text{ and } (\mathcal{P} - Z_2 \in \mathcal{Z}^*_{\mathsf{C}}) \tag{1}$$

We can realize such a best-in-the-multiverse protocol as long as $\mathcal{Z}_{\mathsf{S}}, \mathcal{Z}_{\mathsf{fair}} \cup \mathcal{Z}_{\mathsf{GOD}} \in \mathcal{Q}^2$ and all adversary structures are contained in $\mathcal{Z}^*_{\mathsf{C}}$.

We realize this protocol by constructing a compiler that takes as input a protocol in the client-server paradigm that is unbounded secure for semi-honest client corruption and provides GOD for one fail-stop server[10]. This is upgraded to a best-in-the-multiverse protocol in the presence of active adversaries. A detailed description of this compiler and the final protocol can be found in Section 5.4. This compiler uses as building-blocks n-party protocols for general functionalities, designed for two other, more restricted, classes of adversary structures. We describe these settings next and show constructions of such protocols:

1. Setting A: Let $\mathcal{Z}_{\mathsf{S}}, \mathcal{Z}_{\mathsf{fair}} \cup \mathcal{Z}'_{\mathsf{GOD}} \in \mathcal{Q}^2$ and let $\mathcal{Z}^*_{\mathsf{C}} \subseteq 2^{\mathcal{P}}$ be the largest general adversary structure such that it and $\mathcal{Z}'_{\mathsf{GOD}}$ satisfy,
$$\forall Z \in \mathcal{Z}'_{\mathsf{GOD}}, \forall Z' \in \mathcal{Z}^*_{\mathsf{C}}, Z \cup Z' \neq \mathcal{P} \tag{2}$$
We construct n-party protocols for general functionalities in the multiverse satisfying $\mathcal{Z}'_{\mathsf{GOD}}$, $\mathcal{Z}_{\mathsf{fair}}$, $\mathcal{Z}_{\mathsf{S}}$ and $\mathcal{Z}_{\mathsf{C}}$ when all of these are contained in $\mathcal{Z}^*_{\mathsf{C}}$.
2. Setting B: For a subset of parties $Z \subset \mathcal{P}$, let $\mathcal{Z}^*_{\mathsf{GOD}} = \{Z\}$, $\mathcal{Z}_{\mathsf{S}}, \mathcal{Z}_{\mathsf{fair}} \cup \mathcal{Z}^*_{\mathsf{GOD}} \in \mathcal{Q}^2$ and let $\mathcal{Z}^*_{\mathsf{C}}$ be the largest general adversary structure such that,
$$\mathcal{Z}^*_{\mathsf{GOD}} = \{Z\}, \exists Z' \in \mathcal{Z}^*_{\mathsf{C}}, Z \cup Z' = \mathcal{P} \tag{3}$$
Without loss of generality, in this setting $\mathcal{Z}^*_{\mathsf{C}} = 2^{\mathcal{P}}$. We construct n-party protocols for general functionalities in the multiverse satisfying $\mathcal{Z}^*_{\mathsf{GOD}}$, $\mathcal{Z}_{\mathsf{fair}}$, $\mathcal{Z}_{\mathsf{S}}$ and $\mathcal{Z}_{\mathsf{C}}$ when all adversary structures are contained in $\mathcal{Z}^*_{\mathsf{C}}$.

Protocols for both settings are constructed in two steps. In the first step, we take a maliciously secure n-party protocol, which is unboundedly secure against the corruption of all but a *designated-party*, and otherwise only PPT secure (implied in [1]). Depending on the setting, this protocol is extended to satisfy GOD, which tolerates either $\mathcal{Z}'_{\mathsf{GOD}}$ or $\mathcal{Z}^*_{\mathsf{GOD}}$. In the former case, this comes at

[10] In the fail-stop corruption model, corrupt parties can abort at any point in the protocol but will semi-honestly follow the protocol specification before that.

the cost of limiting the achievable security with abort to $\mathcal{Z}_{\mathsf{C}}^*$. The details of this step for setting A are discussed in Section 5.1, and for setting B in Section 5.2.

In the second step, we present a compiler that uses instances of this designated-party protocol from the first step as its building-blocks. The compiler takes as input any n-party semi-honest unbounded secure protocol that tolerates $\mathcal{Z}_{\mathsf{S}}$ and computes an unbounded secure $\mathcal{Z}_{\mathsf{fair}}$-verifiable-secret-sharing of the function output. When compiled, this semi-honest protocol is emulated with all next-message functions of the i^{th} party executed as the malicious-secure protocol with party P_i as the designated-party for which unbounded security is guaranteed. Depending on the setting (either A or B) whose designated-party protocol is used, this protocol becomes maliciously secure, satisfying that multiverse setting. Section 5.3 elaborates on this step.

Finally, our best-in-the-multiverse protocol for the most general feasibility results from a compiler that takes as input a semi-honest unbounded secure protocol in the client-server paradigm [20] providing GOD for one fail-stop server corruption. This is a protocol for n clients and 3 servers, secure under corruption of $n-1$ clients and 1 server (implied in [3,17]). In the compilation, this is emulated by n parties, with each party playing the role of one client and all the parties collectively emulating the servers. Each next-message function of the first server is executed using an n-party maliciously unbounded secure protocol satisfying $\mathcal{Z}_{\mathsf{GOD}}$ (implied in [17]). Each such function of the second and third servers is executed using the multiverse protocols of Setting A and B, respectively. Details can be found in Section 5.4.

5.1 Step A.1: Incorporating Guaranteed Output Delivery

In this section, we construct n-party protocols for computing any functionality with unbounded security for a *designated-party* that satisfy GOD tolerating any $\mathcal{Z}'_{\mathsf{GOD}} \in \mathcal{Q}^2$. More precisely, for a set of n parties $\mathcal{P}$, let $\mathcal{Z}_{\mathsf{C}}^*$ be the largest adversary structure such that $\mathcal{Z}_{\mathsf{C}}^*$ and $\mathcal{Z}'_{\mathsf{GOD}}$ satisfy,

$$\forall Z \in \mathcal{Z}'_{\mathsf{GOD}}, \forall Z' \in \mathcal{Z}_{\mathsf{C}}^*, Z \cup Z' \neq \mathcal{P}.$$

Let $P_i \in \mathcal{P}$ be a designated-party. Then the protocol we construct, denoted as Π_i, satisfies security with abort for PPT adversaries tolerating $\mathcal{Z}_{\mathsf{C}}^*$ and for unbounded adversaries tolerating $\mathcal{Z}_{\mathsf{S}}^i$, which contains all sets in $\mathcal{Z}_{\mathsf{C}}^*$ that do not contain P_i. In addition, GOD is satisfied with respect to $\mathcal{Z}'_{\mathsf{GOD}}$. There may be one such designated-party protocol for each party $P_i \in \mathcal{P}$.

Our starting point is a malicious fall-back secure protocol, denoted Π_i^*, that can compute any functionality with unbounded security for that *designated-party* P_i. Such fall-back secure protocols exist that tolerate any $\mathcal{Q}^2$ adversary structure [1]. The protocol Π_i^* satisfies security with identifiable abort against a PPT adversary who corrupts any subset of parties, and against an unbounded adversary who corrupts any subset not including party P_i. Let this adversary structure containing all subsets of parties, not including party P_i, be $\mathcal{Z}_{\mathsf{S}}^{i*}$. Then we say that Π_i^* is malicious fall-back secure with abort with respect to $\mathcal{Z}_{\mathsf{S}}^{i*}$.

We upgrade such a designated-party fall-back secure protocol to incorporate GOD for a limited adversary structure $\mathcal{Z}'_{\mathsf{GOD}}$, but this comes at the cost of not achieving complete fall-back security: no security guarantees can be provided for corruption beyond $\mathcal{Z}^*_{\mathsf{C}}$, for both unbounded and PPT adversaries. The basic idea for this will follow that in [18], though we extend it to the non-threshold case. Consider the verifiable secret sharing (VSS) functionality shown in Figure 3. We require a designated-party fall-back secure protocol (satisfying security with identifiable abort) that realizes this functionality – ensuring a correct sharing of a secret, if it executes to completion. If the protocol aborts, default inputs and randomness are assumed for the identified corrupt party and then the sharing protocol is re-executed. As a result reconstruction of a secret input is possible if, at most, any subset of parties in $\mathcal{Z}'_{\mathsf{GOD}}$ are corrupt. If the reconstruction fails, all the parties detect this and set the output to a default value. This VSS protocol uses a secret-sharing scheme and a commitment scheme as building-blocks, and can be parameterized by the underlying commitment scheme (statistically hiding $\mathsf{Com}_{\mathsf{SH}}$ or statistically binding $\mathsf{Com}_{\mathsf{SB}}$).

Lemma 2. *Let* $\mathsf{SS}_{\overline{\mathcal{Z}}'_{\mathsf{GOD}}} = (\mathsf{Share}_{\overline{\mathcal{Z}}'_{\mathsf{GOD}}}, \mathsf{Recon}_{\overline{\mathcal{Z}}'_{\mathsf{GOD}}})$ *be a secret sharing scheme with access structure* $\overline{\mathcal{Z}}'_{\mathsf{GOD}}$. *Let* $\mathsf{com}(x; r)$ *be a commitment scheme where* $\mathsf{com} \in \{\mathsf{Com}_{\mathsf{SB}}, \mathsf{Com}_{\mathsf{SH}}\}$ *can be either statistically hiding or binding. Then* $\mathsf{VSS}_{\mathsf{Com}_{\mathsf{SB}}}$ *(resp.* $\mathsf{VSS}_{\mathsf{Com}_{\mathsf{SH}}}$*) as in Figure 3 is a* $\mathcal{Z}^*_{\mathsf{C}}$*-secure VSS scheme (Definition 1), where* $\forall Z \in \mathcal{Z}^*_{\mathsf{C}}, \neg\exists Z' \in \overline{\mathcal{Z}}'_{\mathsf{GOD}}$ *s.t.* $Z \supseteq Z'$.

The protocol described in Figure 3 is identical to the VSS protocol in Chapter 7 of [12], with the difference that we replace their underlying threshold secret-sharing protocol with a $\overline{\mathcal{Z}}'_{\mathsf{GOD}}$-Secret Sharing scheme. The proof for our protocol therefore follows from that of [12].

Given such a VSS scheme, we can modify the protocols Π^*_i to that one shown in Figure 4. In this protocol, first, all parties P_j s.t. $j \neq i$ sample the randomness to be used in Π^*_i and share their randomness using the $\mathcal{Z}^*_{\mathsf{C}}$-Secure VSS with statistically binding commitments $\mathsf{Com}_{\mathsf{SB}}$. They then use this randomness to mask their input and send it to P_i. Party P_i shares its input, randomness, and all these masked values with all other parties using $\mathcal{Z}^*_{\mathsf{C}}$-Secure VSS with statistically hiding commitments $\mathsf{Com}_{\mathsf{SH}}$. The protocol then runs as it originally would. If, at any point in the protocol, a party aborts the computation, there are mechanisms to satisfy GOD that come into play depending on the set of parties that have already aborted the computation. (1) If this set is not in $\mathcal{Z}'_{\mathsf{GOD}}$, there are not enough shares with active parties to reconstruct the VSS secrets, and the protocol terminates. (2) Otherwise, if the set is covered by $\mathcal{Z}'_{\mathsf{GOD}}$, the nature of our protocol allows the identification of the last malicious party that aborted, and different actions are taken depending on the identity of this party. (a) If this is the designated-party P_i, all other parties pool in their VSS shares and reconstruct the input and randomness. Then, they complete the protocol with all other parties computing their messages. However, this is only secure against a PPT adversary. (b) If a different party causes the abort, regardless of the

$\overline{\mathcal{Z}}'_{\mathsf{GOD}}$ **Verifiable Secret Sharing**

Setting: Let $\mathcal{P} = \{P_i\}_{i\in[n]}$ be such that P_1 has input x. For an access structure $\overline{\mathcal{Z}}'_{\mathsf{GOD}}$, let $\mathsf{SS}_{\overline{\mathcal{Z}}'_{\mathsf{GOD}}} = (\mathsf{Share}_{\overline{\mathcal{Z}}'_{\mathsf{GOD}}}, \mathsf{Recon}_{\overline{\mathcal{Z}}'_{\mathsf{GOD}}})$ be a secret sharing scheme where only any set of shares indexed with $Z \in \overline{\mathcal{Z}}'_{\mathsf{GOD}}$ (or super-set thereof) can reconstruct the secret. This means that the scheme is secure against the collusion of any subset of parties in $\mathcal{Z}^*_{\mathsf{C}}$ with one share each. Let $\mathsf{com}(x; r)$ be a commitment scheme where $\mathsf{com} \in \{\mathsf{Com}_{\mathsf{SB}}, \mathsf{Com}_{\mathsf{SH}}\}$ can be either statistically hiding or binding. Then the sharing functionality $\mathcal{F}_{\mathsf{VSS.Share}}$ is defined as follows:

- $\mathcal{F}_{\mathsf{VSS.Share}}$ accepts the input x from P_1
- $\mathcal{F}_{\mathsf{VSS.Share}}$ computes $\{[x]_i\}_{i\in[n]} \leftarrow \mathsf{SS}_{\overline{\mathcal{Z}}'_{\mathsf{GOD}}}.\mathsf{Share}_{\overline{\mathcal{Z}}'_{\mathsf{GOD}}}(x)$
- $\mathcal{F}_{\mathsf{VSS.Share}}$ samples $\vec{r} = \{r_i\}_{i\in[n]}$
- $\mathcal{F}_{\mathsf{VSS.Share}}$ sets $\vec{c} = \{c_i\}_{i\in[n]}$, where $\forall i \in [n], c_i = \mathsf{com}([x]_i; r_i)$
- $\forall i \in [n]$, $\mathcal{F}_{\mathsf{VSS.Share}}$ sends $(\vec{c}, [x]_i, r_i)$ to P_i as its VSS share

Let $\Pi^i_{\mathsf{VSS.Share}}$ denote an n-party protocol that securely realizes the above functionality. For $i \in [n]$, the protocol satisfies designated-party malicious fall-back security with identifiable abort where party P_i is the designated party. The commitments internally used are $\mathsf{Com}_{\mathsf{SH}}$ if P_i is the dealer, and $\mathsf{Com}_{\mathsf{SB}}$ otherwise. The corresponding reconstruction protocol $\Pi_{\mathsf{VSS.Recon}}$ operates as follows:

- for any $Z \subseteq \mathcal{P}$, each $P_i \in Z$ broadcasts from its share $([x]_i, r_i)$
- each party verifies $\forall P_i \in Z$, $c_i == \mathsf{com}([x]_i, r_i)$
- if all checks pass, accept $x = \mathsf{SS}_{\overline{\mathcal{Z}}'_{\mathsf{GOD}}}.\mathsf{Recon}_{\overline{\mathcal{Z}}'_{\mathsf{GOD}}}(\{[x]_i\}_{P_i\in Z})$

Let $\mathsf{VSS}_{\mathsf{Com}_{\mathsf{SB}}}$ (resp. $\mathsf{VSS}_{\mathsf{Com}_{\mathsf{SH}}}$) denote $\overline{\mathcal{Z}}'_{\mathsf{GOD}}$ VSS instantiated with a statistically binding (resp. statistically hiding) commitment scheme.

Fig. 3. $\mathcal{Z}^*_{\mathsf{C}}$-Secure Verifiable Secret Sharing

adversary's computational power, the protocol remains secure if the remaining parties reconstruct the randomness, use P_i's help to reconstruct the input, and continue the protocol on its behalf.

Theorem 2. *Let $\mathcal{Z}'_{\mathsf{GOD}} \in \mathcal{Q}^2$ and $\mathcal{Z}^*_{\mathsf{C}}$ be two adversary structures that satisfy*

$$\forall Z \in \mathcal{Z}'_{\mathsf{GOD}}, \forall Z' \in \mathcal{Z}^*_{\mathsf{C}}, Z \cup Z' \neq \mathcal{P}$$

Let $\mathcal{Z}^{i,\mathsf{GOD}}_{\mathsf{S}}$ be the set of all subsets in $\mathcal{Z}'_{\mathsf{GOD}}$ not including P_i, and $\mathcal{Z}^i_{\mathsf{S}}$ be the set of all subsets in $\mathcal{Z}^*_{\mathsf{C}}$ not including P_i. Let $\mathsf{VSS}_{\mathsf{Com}_{\mathsf{SB}}}$ and $\mathsf{VSS}_{\mathsf{Com}_{\mathsf{SH}}}$ be $\mathcal{Z}^*_{\mathsf{C}}$-Secure Verifiable Secret Sharing schemes (Figure 3 – Definition 1). Assuming oblivious transfer and secure point-to-point and broadcast channels, let Π^*_i be a protocol that computes any n-party functionality f with malicious fall-back security with identifiable abort (Definition 3.11)tolerating $\mathcal{Z}^{i*}_{\mathsf{S}}$, where $\mathcal{Z}^{i*}_{\mathsf{S}}$ contains all subsets of parties not including P_i. Then protocol Π_i in Figure 4 securely computes f in the presence of malicious adversaries with the following guarantees:

Maliciously Best-of-both-Worlds Fall-Back Secure Protocol Π_i

Setting: Let $\mathcal{P} = \{P_j\}_{j\in[n]}$ be such that P_j has input x_j. Let f be a function, $\vec{x} = \{x_j\}_{j\in[n]}$ be its input and the protocol outputs $f(\vec{x})$ to all parties in $\mathcal{P}$. Let $\mathsf{Com}_{\mathsf{SB}}$ and $\mathsf{Com}_{\mathsf{SH}}$ be statistically binding and statistically hiding commitment schemes respectively. For an adversary structure $\mathcal{Z}'_{\mathsf{GOD}} \in \mathcal{Q}^2$, let $\mathcal{Z}^*_{\mathsf{C}}$ be the largest adversary structure s.t. $\mathcal{Z}^*_{\mathsf{C}}$ and $\mathcal{Z}'_{\mathsf{GOD}}$ satisfy

$$\forall Z \in \mathcal{Z}'_{\mathsf{GOD}}, \forall Z' \in \mathcal{Z}^*_{\mathsf{C}}, Z \cup Z' \neq \mathcal{P}$$

For each set $Z \in \mathcal{Z}'_{\mathsf{GOD}}$, there exists a set $Z' \in \mathcal{Z}^*_{\mathsf{C}}$ s.t. $Z \subseteq Z'$. Let $\mathsf{VSS}_{\mathsf{Com}_{\mathsf{SB}}}$ and $\mathsf{VSS}_{\mathsf{Com}_{\mathsf{SH}}}$ be $\mathcal{Z}^*_{\mathsf{C}}$-Secure Verifiable Secret Sharing schemes (Figure 3). Let Π^*_i be a protocol that computes f with malicious fall-back security with identifiable abort tolerating $\mathcal{Z}^{i*}_{\mathsf{S}}$ that contains all subsets of parties not including P_i.

1. **Commit Phase:** Let $\mathcal{P}_{\mathsf{N}} = \{P_j\}_{j\neq i\in[n]}$.
   - $\forall P_j \in \mathcal{P}_{\mathsf{N}}$, let r_j and x_j be the randomness and input P_j uses in Π^*_i,
     - P_j computes $\Lambda_j = r_j \oplus x_j$ and sends to P_i
     - compute $\{s^j_k\}_{k\in[n]} \leftarrow \mathsf{VSS}_{\mathsf{Com}_{\mathsf{SB}}}(r_j)$ and $\forall k \in [n]$ P_k receives s^j_k
   - Let r_i be the randomness and x_i be the input of P_i
     - compute $\{s^i_k\}_{k\in[n]} \leftarrow \mathsf{VSS}_{\mathsf{Com}_{\mathsf{SH}}}(r_i, x_i, \{\Lambda_j\}_{j\neq i\in[n]})$ and $\forall k \in [n]$ P_k receives s^i_k
2. **Protocol Emulation:** Compute the protocol Π^*_i.
3. **Abort Handling:** If a party $P_j \in \mathcal{P}$ causes the protocol above to abort, letting Z be the set of parties that have aborted the execution till now,
   - All active parties in $P_k \in \mathcal{P}$ broadcast VSS shares s^j_k and attempt reconstruction, that succeeds if $\exists Z' \in \mathcal{Z}'_{\mathsf{GOD}}$ s.t. $Z \subseteq Z'$,
     - If $P_j = P_i$, the active parties reconstruct (r_i, x_i), and emulate the protocol on P_i's behalf
     - If $P_j \neq P_i$, the active parties reconstruct r_j, get Λ_j from P_i and compute x_j; then continues protocol emulation on P_j's behalf
   - Otherwise, ABORT the protocol

Fig. 4. Maliciously Best-of-both-Worlds Fall-Back Secure Protocol (Equation 2)

- *GOD tolerating an unbounded adversary w.r.t. $\mathcal{Z}^{i,\mathsf{GOD}}_{\mathsf{S}}$ (Definition 3.8 – GOD) and PPT adversary w.r.t. $\mathcal{Z}'_{\mathsf{GOD}}$ (Definition 3.7 – GOD).*
- *Security with abort tolerating an unbounded adversary w.r.t. $\mathcal{Z}^i_{\mathsf{S}}$ (Definition 3.8 – abort) and PPT adversary w.r.t. $\mathcal{Z}^*_{\mathsf{C}}$ (Definition 3.7 – abort).*
- *Correctness (Definition 3.9) for an unbounded adversary w.r.t. $\mathcal{Z}'_{\mathsf{GOD}}$.*

The complete proof for this can be found in Appendix B.

5.2 Step B.1: Incorporating GOD for a Designated Subset

In this section, as above, we construct n protocols, each of which provides unbounded security for a designated party. Additionally, for a given subset of parties $Z \subset \mathcal{P}$, this protocol provides guaranteed output delivery if parties in Z

Maliciously Best-of-both-Worlds Fall-Back Secure Protocol Π_i

Setting: Let $\mathcal{P} = \{P_j\}_{j\in[n]}$ be such that P_j has input x_j. Let f be a function, $\vec{x} = \{x_j\}_{j\in[n]}$ be its input and the protocol outputs $f(\vec{x})$ to all parties in $\mathcal{P}$. For $Z \subset \mathcal{P}$, let $\mathcal{Z}^*_{\mathsf{GOD}} = \{Z\}$ and $\mathcal{Z}^*_{\mathsf{C}}$ be any adversary structure. Let Π^*_i be a protocol that computes f with malicious fall-back security with identifiable abort tolerating $\mathcal{Z}^{i*}_{\mathsf{S}}$ that contains all subsets of parties not including P_i.

1. **Protocol Emulation:** Compute the protocol Π^*_i.
2. **Abort Handling:** If a party $P_j \in \mathcal{P}$ causes the protocol to abort,
   - If $P_j \in Z$, all active parties in $P_k \in \mathcal{P}$ restart the protocol assuming default inputs and randomness for P_j, locally emulating its messages.
   - Otherwise, ABORT the protocol

Fig. 5. Maliciously Best-of-both-Worlds Fall-Back Secure Protocol (Equation 3)

are corrupted. This protocol uses a player emulation technique as above, except that VSS is no longer used.

Theorem 3. *Assume the existence of oblivious transfer and secure point-to-point and broadcast channels. For $Z \subset \mathcal{P}$, let $\mathcal{Z}^*_{\mathsf{GOD}} = \{Z\}$ and $\mathcal{Z}^*_{\mathsf{C}}$ be any adversary structure. Let $\mathcal{Z}^{i,\mathsf{GOD}}_{\mathsf{S}}$ be the set of all subsets in $\mathcal{Z}^*_{\mathsf{GOD}}$ not including P_i, and $\mathcal{Z}^i_{\mathsf{S}}$ be the set of all subsets in $\mathcal{Z}^*_{\mathsf{C}}$ not including P_i. Assuming Π^*_i is a protocol that computes any n-party functionality f with malicious fall-back security with identifiable abort (Definition 3.11) tolerating $\mathcal{Z}^{i*}_{\mathsf{S}}$, where $\mathcal{Z}^{i*}_{\mathsf{S}}$ contains all subsets of parties not including P_i, protocol Π_i in Figure 5 securely computes f in the presence of malicious adversaries with the following guarantees:*

- *GOD in the presence of an unbounded adversary w.r.t. $\mathcal{Z}^{i,\mathsf{GOD}}_{\mathsf{S}}$ (Definition 3.8 – GOD) and a PPT adversary tolerating $\mathcal{Z}^*_{\mathsf{GOD}}$ (Definition 3.7 – GOD).*
- *Security with abort in the presence of an unbounded adversary w.r.t. $\mathcal{Z}^i_{\mathsf{S}}$ (Definition 3.8 – abort)and a PPT adversary corrupting any subset of parties within $\mathcal{Z}^*_{\mathsf{C}}$ (Definition 3.7 – abort).*
- *Correctness (Definition 3.9)for honest parties holds when an unbounded adversary corrupts any subset of parties in $\mathcal{Z}^*_{\mathsf{GOD}}$.*

The complete proof for this can be found in Appendix C.

5.3 Step 2: Compiling to the Multiverse Setting

In this section we use the protocols $\{\Pi_i\}_{i\in[n]}$ from either Section 5.1 or Section 5.2 to compile a semi-honest secure virtual protocol into a protocol satisfying Setting A or B in the multiverse, respectively. We start by describing the properties required from the semi-honest protocol that is input to the compiler.

Semi-Honest Virtual Protocol. Let $\mathcal{V} = \{V_i\}_{i\in[n]}$ be a set of n virtual parties. Let f be the function to be computed among these parties and let

$\vec{x} = \{x_i\}_{i\in[n]}$ be the inputs. In the final compiled protocol, let $\mathcal{Z}_{\mathsf{S}} \in \mathcal{Q}^2$ be an adversary structure for which security, at least with abort, is required in the presence of a computationally unbounded adversary. Let $\mathcal{Z}_{\mathsf{fair}} \in \mathcal{Q}^2$ be an adversary structure for which MPC with fairness is required of the compiled protocol. Let $\mathcal{F}$ be the n-party functionality computing an unbounded secure $\mathcal{Z}_{\mathsf{fair}}$-verifiable-secret-sharing [17] of the output of f among the n parties. Then the virtual protocol $\Pi_{\mathcal{V}}$ that we consider needs to realize $\mathcal{F}$ in the presence of a semi-honest unbounded adversary tolerating $\mathcal{Z}_{\mathsf{S}}$.

For a party $V_i \in \mathcal{V}$, we refer to its complete view in $\Pi_{\mathcal{V}}$ as $\mathbf{View}_{\Pi_{\mathcal{V}},i}$ and by $\mathbf{View}^j_{\Pi_{\mathcal{V}},i}$, we refer to the partial view of V_i in the protocol up to round j. This includes the security parameter κ, the input x_i, contents of the random tape r_i, and all the messages received from all parties in $\mathcal{V}$ up to round j.

$$\mathbf{View}^j_{\Pi_{\mathcal{V}},i} = (\kappa, x_i, r_i, \{m^{j'}_{k\to i}\}_{j'\in[j],k\neq i\in[n]})$$

Here $m^{j'}_{k\to i}$ is the message that party V_k sends V_i in round j'. Let NxtMsg be the next message function such that,

$$\{m^j_{i\to k}\}_{k\neq i\in[n]} = \mathsf{NxtMsg}^j_i(\mathbf{View}^{j-1}_{\Pi_{\mathcal{V}},i})$$

This function is parameterized by $i \in [n]$ that indexes the virtual party $V_i \in \mathcal{V}$. It takes as input the view up to round $j-1$ and creates all the messages to be delivered to all the parties in round j. Note that after each round,

$$\mathbf{View}^j_{\Pi_{\mathcal{V}},i} = \mathbf{View}^{j-1}_{\Pi_{\mathcal{V}},i} \cup \{m^j_{k\to i}\}_{k\neq i\in[n]}$$

Letting r be the number of rounds in $\Pi_{\mathcal{V}}$, the complete protocol $\Pi_{\mathcal{V}}$ itself can be described as a set of the next message functions,

$$\Pi_{\mathcal{V}} = \{\mathsf{NxtMsg}^j_i(\mathbf{View}^{j-1}_{\Pi_{\mathcal{V}},i})\}_{i\in[n],j\in[r]}$$

Finally, for any subset of parties $Z \subseteq \mathcal{V}$ we denote by $\mathbf{View}_{\Pi_{\mathcal{V}},Z}$ the combined view of $\Pi_{\mathcal{V}}$ for all the parties in Z and similarly, we denote by $\mathbf{View}^j_{\Pi_{\mathcal{V}},Z}$ the combined view up to round j in the protocol.

Next-Message Functionality. Corresponding to each next message function $\mathsf{NxtMsg}^j_i \in \Pi_{\mathcal{V}}$, we describe an n-party functionality $\mathcal{F}_{\mathsf{NxtMsg}^j_i}$ as given in Figure 6. For $\mathcal{Z}_{\mathsf{GOD}} \in \{\mathcal{Z}'_{\mathsf{GOD}}, \mathcal{Z}^*_{\mathsf{GOD}}\}$ as required, let $\mathcal{Z} = \mathcal{Z}^*_{\mathsf{C}} \cup \mathcal{Z}_{\mathsf{GOD}}$. This functionality takes as input a $\mathcal{Z}$-secret-sharing of the partial view of the protocol that would be the input to the next message function. It's primary purpose is to reconstruct the view from the share, evaluate the next message function, and re-share the output among the parties using a $\mathcal{Z}$-secret-sharing scheme again.

Since the virtual protocol is semi-honest secure and the compiled protocol needs to be maliciously secure, we require mechanisms to ensure that the outputs of a previous next message functionality are the inputs to future such functionalities. We do so using statistically hiding and binding commitment schemes. For the next message function NxtMsg^j_i, the party P_i receives statistically binding commitments to the outputs of all other parties. All other parties receive a statistically hiding commitment of the output of party P_i.

The Compiled Protocol. Given a semi-honest virtual protocol $\Pi_{\mathcal{V}}$ that computes any functionality in such a way that the output is shared among the n-parties and any set of parties beyond that in $\mathcal{Z}_{\mathsf{fair}}$ can reconstruct it, this can be

Next Message Functionality $\mathcal{F}_{\mathsf{NxtMsg}_i^j}$

Setting: Let $\mathcal{P} = \{P_i\}_{i\in[n]}$ be n parties. Let $\mathcal{Z}_{\mathsf{GOD}} \in \mathcal{Q}^2$ where $\mathcal{Z}_{\mathsf{GOD}} \in \{\mathcal{Z}'_{\mathsf{GOD}}, \mathcal{Z}^*_{\mathsf{GOD}}\}$ and $\mathcal{Z}^*_{\mathsf{C}}$ be the largest access structure s.t. $\forall Z \in \mathcal{Z}'_{\mathsf{GOD}}, \forall Z' \in \mathcal{Z}^*_{\mathsf{C}}, Z \cup Z' \neq \mathcal{P}$. Let $\mathcal{Z} = \mathcal{Z}^*_{\mathsf{C}} \cup \mathcal{Z}_{\mathsf{GOD}}$. Let $\mathsf{SS}_{\mathcal{Z}} = (\mathsf{Share}_{\mathcal{Z}}, \mathsf{Recon}_{\mathcal{Z}})$ be a $\mathcal{Z}$-secret-sharing scheme. Let $\mathsf{Com}_{\mathsf{SB}}$ and $\mathsf{Com}_{\mathsf{SH}}$ be statistically binding and statistically hiding commitment schemes respectively. For a protocol $\Pi_{\mathcal{V}}$ computing $\mathcal{F}$ let $\{m^j_{i\to k}\}_{k\neq i\in[n]} = \mathsf{NxtMsg}^j_i(\mathbf{View}^{j-1}_{\Pi_{\mathcal{V}},i})$ be the next-message function for party V_i in round j. The n-party functionality $\mathcal{F}_{\mathsf{NxtMsg}_i^j}$ operates as follows:

- **Inputs:** Each party $P_k \in \mathcal{P}$ has:
  - Share of the view of V_i until round $(j-1)$ in $\Pi_{\mathcal{V}}$,
  $$[\mathbf{View}^{j-1}_{\Pi_{\mathcal{V}},i}]_k = \left(\kappa, [x_i]_k, [r_i]_k, \{[m^{j'}_{i'\to i}]_k\}_{j'\in[j-1], i'\neq i\in[n]}\right)$$
  - If $P_k = P_i$, statistically binding commitments to all other parties' partial views $\{c_{k'}\}_{k'\neq i\in[n]}$ and randomness r_i.
  - If $P_k \neq P_i$, statistically hiding commitment to P_i's partial view c_i and randomness r_k.
  - Sample a random mask $M_k \leftarrow \{0,1\}^{\ell'}$ where $\ell' = |\mathsf{Output}_k|$.

  P_k gives to $\mathcal{F}_{\mathsf{NxtMsg}_i^j}$ the inputs,
  $$[\mathbf{View}^{j-1}_{\Pi_{\mathcal{V}},i}]_k, M_k, r_k \text{ and } \begin{cases} \{c_{k'}\}_{k'\neq i\in[n]} & \text{if } i = k \\ c_i & \text{otherwise} \end{cases}$$
- **Computation:** The functionality $\mathcal{F}_{\mathsf{NxtMsg}_i^j}$ computes:
$$\begin{aligned}
\forall i' \neq i \in [n], c_{i'} \neq \mathsf{Com}_{\mathsf{SB}}([\mathbf{View}^{j-1}_{\Pi_{\mathcal{V}},i}]_{i'}; r_{i'}),\ & \vec{c} = \mathsf{ABORT}_{i'} \\
c_i \neq \mathsf{Com}_{\mathsf{SH}}([\mathbf{View}^{j-1}_{\Pi_{\mathcal{V}},i}]_i; r_i),\ & \vec{c} = \mathsf{ABORT}_i \\
\text{otherwise,}\quad & \mathbf{View}^{j-1}_{\Pi_{\mathcal{V}},i} \leftarrow \mathsf{Recon}_{\mathcal{Z}}(\{[\mathbf{View}^{j-1}_{\Pi_{\mathcal{V}},i}]_k\}_{k\in[n]}) \\
& \{m^j_{i\to i'}\}_{i'\neq i\in[n]} = \mathsf{NxtMsg}^j_i(\mathbf{View}^{j-1}_{\Pi_{\mathcal{V}},i}) \\
\forall i' \neq i \in [n],\quad & \{[m^j_{i\to i'}]_k\}_{k\in[n]} \leftarrow \mathsf{Share}_{\mathcal{Z}}(m^j_{i\to i'}) \\
\forall k \neq i \in [n],\ \text{sample } R_k,\quad & c_k = \mathsf{Com}_{\mathsf{SB}}(\{[m^j_{i\to i'}]_k\}_{i'\neq i\in[n]}; R_k) \\
\text{sample } R_i,\quad & c_i = \mathsf{Com}_{\mathsf{SH}}(\{[m^j_{i\to i'}]_i\}_{i'\neq i\in[n]}; R_i) \\
\forall k \in [n],\quad & \mathsf{Output}_k = R_k, \{[m^j_{i\to i'}]_k\}_{i'\neq i\in[n]}, \\
& \qquad \begin{cases} \{c_{k'}\}_{k'\neq i\in[n]} & \text{if } i = k \\ c_i & \text{otherwise} \end{cases} \\
& \vec{c} = \{M_k \oplus \mathsf{Output}_k\}_{k\in[n]}
\end{aligned}$$
- **Output:** $\mathcal{F}_{\mathsf{NxtMsg}_i^j}$ gives $\vec{c}$ to each $P_k \in \mathcal{P}$.

Fig. 6. Next Message Functionality

compiled into our final protocol secure in the presence of malicious adversaries. This is done using the next-message functionalities of $\Pi_{\mathcal{V}}$ as in Figure 6 and implementing them using the best-of-both-worlds fall-back secure protocol as in

Figure 4. First, each party $P_i \in \mathcal{P}$ secret shares the inputs and randomness x_i and r_i that V_i would use in $\Pi_{\mathcal{V}}$ using a $\mathcal{Z}$-secret-sharing, among all the parties. They also create and distribute the commitments as required to be inputs in the next-message functionalities. Then, for each round, for virtual party $V_i \in \mathcal{V}$, each functionality $\mathcal{F}_{\mathsf{NxtMsg}_i^j}$ is executed using the protocol Π_i from Step 1. If all these protocols execute till completion, then in the end, all the parties attempt to reconstruct the $\mathcal{Z}$-secret-sharing of their respective inputs from the views. If this succeeds, then all the parties end up with a $\mathcal{Z}_{\mathsf{fair}}$-verifiable-secret-sharing of the required function output. Finally, fairness is guaranteed for the honest parties that reconstruct. The complete protocol can be found in Figure 7.

Theorem 4. *Assume the existence of oblivious transfer and secure point-to-point and broadcast channels. Let $\mathcal{P} = \{P_i\}_{i\in[n]}$ be a set of n parties where each party P_i has input x_i. Let $\mathcal{Z}'_{\mathsf{GOD}} \in \mathcal{Q}^2$ (resp. $\mathcal{Z}^*_{\mathsf{GOD}} = \{Z \subset \mathcal{P}\} \in \mathcal{Q}^2$) be an adversary structure and $\mathcal{Z}^*_{\mathsf{C}}$ be the largest adversary structure such that*

$$\forall Z \in \mathcal{Z}'_{\mathsf{GOD}}, \forall Z' \in \mathcal{Z}^*_{\mathsf{C}}, Z \cup Z' \neq \mathcal{P}$$

Let $\mathcal{Z}_{\mathsf{fair}}, \mathcal{Z}_{\mathsf{S}} \in \mathcal{Q}^2$ and $\mathcal{Z}_{\mathsf{C}}$ be adversary structures such that $\mathcal{Z}^*_{\mathsf{fair}} = \mathcal{Z}'_{\mathsf{GOD}} \cup \mathcal{Z}_{\mathsf{fair}} \in \mathcal{Q}^2$ (resp. $\mathcal{Z}^*_{\mathsf{fair}} = \mathcal{Z}^*_{\mathsf{GOD}} \cup \mathcal{Z}_{\mathsf{fair}} \in \mathcal{Q}^2$). Let it hold that for all $Z \in \mathcal{Z}'_{\mathsf{GOD}}, \mathcal{Z}_{\mathsf{fair}}, \mathcal{Z}_{\mathsf{S}}, \mathcal{Z}_{\mathsf{C}}$, (resp. also $Z \in \mathcal{Z}^*_{\mathsf{GOD}}$) there exists $Z' \in \mathcal{Z}^*_{\mathsf{C}}$ such that $Z \subseteq Z'$.

- *Let $\mathsf{VSS}^{\mathcal{Z}_{\mathsf{fair}}} = (\mathsf{Share}, \mathsf{Recon})$ be a VSS scheme tolerating $\mathcal{Z}_{\mathsf{fair}}$ [17].*
- *For an n-party functionality f, let $\Pi_{\mathcal{V}}$ semi-honest unbounded securely compute $\mathsf{VSS}^{\mathcal{Z}_{\mathsf{fair}}}.\mathsf{Share}(f(\vec{x}))$ tolerating $\mathcal{Z}_{\mathsf{S}}$ (Definition 3.6).*
- *For each $i \in [n]$, let Π_i be a protocol satisfying Theorem 2 (resp. Theorem 3).*

The compiled protocol in Figure 7 maliciously securely computes f satisfying:

- ***GOD** tolerating an unbounded adversary w.r.t $\mathcal{Z}'_{\mathsf{GOD}} \cap \mathcal{Z}_{\mathsf{S}}$ (resp. $\mathcal{Z}^*_{\mathsf{GOD}} \cap \mathcal{Z}_{\mathsf{S}}$, Definition 3.8 – GOD) and PPT adversary corrupting any subset in $\mathcal{Z}'_{\mathsf{GOD}}$ (resp. $\mathcal{Z}^*_{\mathsf{GOD}}$, Definition 3.7 – GOD).*
- ***Fairness** tolerating an unbounded adversary w.r.t $\mathcal{Z}^*_{\mathsf{fair}} \cap \mathcal{Z}_{\mathsf{S}}$ (Definition 3.8 – fairness) and PPT adversary w.r.t $\mathcal{Z}^*_{\mathsf{fair}}$ (Definition 3.7 – fairness).*
- ***Security with abort** tolerating an unbounded adversary w.r.t $\mathcal{Z}_{\mathsf{S}}$ (Definition 3.8 – abort) and PPT adversary w.r.t $\mathcal{Z}^*_{\mathsf{C}}$ (Definition 3.7 – abort).*

The complete proof for this can be found in Appendix D.

5.4 Best-in-the-Multiverse Protocol for General Feasibility

In this section, we construct a best-in-the-multiverse protocol for the most general adversary structures feasible. So far the previous subsections cover constructions for multiverse protocols separately for the following characterizations:

A. The structures $\mathcal{Z}_{\mathsf{S}}, \mathcal{Z}'_{\mathsf{GOD}} \cup \mathcal{Z}_{\mathsf{fair}} \in \mathcal{Q}^2$. $\mathcal{Z}^*_{\mathsf{C}}$ is the largest structure such that,
$$\forall Z \in \mathcal{Z}'_{\mathsf{GOD}}, \forall Z' \in \mathcal{Z}^*_{\mathsf{C}}, Z \cup Z' \neq \mathcal{P}$$
and for each $Z \in \mathcal{Z}_{\mathsf{S}} \cup \mathcal{Z}^*_{\mathsf{fair}}$ there exists a $Z' \in \mathcal{Z}^*_{\mathsf{C}}$ s.t. $Z \subseteq Z'$.

Multiverse Setting Protocol Π

Setting: Let $\mathcal{P} = \{P_i\}_{i\in[n]}$ be n parties. For function f on $\vec{x} = \{x_i\}_{i\in[n]}$, P_i has input x_i. Let $\Pi_{\mathcal{V}} = \{\mathsf{NxtMsg}_i^j\}_{i\in[n],j\in[r]}$ be a protocol for the functionality $\mathcal{F}$ computing $\mathsf{VSS}^{\mathcal{Z}_{\mathsf{fair}}}.\mathsf{Share}(f(\vec{x}))$ among $\mathcal{V} = \{V_i\}_{i\in[n]}$. $\Pi_{\mathcal{V}}$ is secure against a semi-honest unbounded adversary tolerating $\mathcal{Z}_{\mathsf{S}}$. Let $\mathsf{Com}_{\mathsf{SB}}$ and $\mathsf{Com}_{\mathsf{SH}}$ be statistically binding and statistically hiding commitment schemes respectively (Definition 3.4,3.3). For $\mathcal{Z}'_{\mathsf{GOD}}$ (resp. $\mathcal{Z}^*_{\mathsf{GOD}}$) and the maximal $\mathcal{Z}^*_{\mathsf{C}}$ satisfying Equation 2 (resp. Equation 3), for each $i \in [n]$, let Π_i be a protocol satisfying Theorem 2 (resp. Theorem 3). Let $\mathcal{Z} = \mathcal{Z}^*_{\mathsf{C}} \cup \mathcal{Z}'_{\mathsf{GOD}}$ (resp. $\mathcal{Z} = \mathcal{Z}^*_{\mathsf{C}} \cup \mathcal{Z}^*_{\mathsf{GOD}}$).

1. **Initialize shared state.** Each party $P_i \in \mathcal{P}$ does the following:
   - Sample $r_i \leftarrow \mathcal{R}(\kappa)$ and compute $\{[x_i]_{i'}, [r_i]_{i'}\}_{i'\in[n]} \leftarrow \mathsf{Share}_{\mathcal{Z}}(x_i, r_i)$
   - $\forall i' \neq i \in [n]$, send $([x_i]_{i'}, [r_i]_{i'})$ to $P_{i'}$
   - $\forall k \in [n]$, set the share of the initial state of virtual party $V_k \in \mathcal{V}$ as,
   $$[\mathbf{View}^0_{\Pi_{\mathcal{V}},k}]_i = ([x_k]_i, [r_k]_i)$$
   - For all $k \in [n]$, for $\mathcal{F}_{\mathsf{NxtMsg}^1_k}$ (Figure 6), sample $r_i^{k,0}$ and
     - If $i = k$, compute and broadcast $c_i^{k,0} = \mathsf{Com}_{\mathsf{SH}}(r_i; [\mathbf{View}^0_{\Pi_{\mathcal{V}},k}]_i)$
     - Otherwise, send $c_i^{k,0} = \mathsf{Com}_{\mathsf{SB}}(r_i; [\mathbf{View}^0_{\Pi_{\mathcal{V}},k}]_i)$ to P_k
2. **Compute virtual protocol.** For each $\mathsf{NxtMsg}_i^j \in \Pi_{\mathcal{V}}$, let $\Pi_{\mathsf{NxtMsg}_i^j}$ be a protocol for $\mathcal{F}_{\mathsf{NxtMsg}_i^j}$ (Figure 4) operating as Π_i. Each $P_{i'} \in \mathcal{P}$ executes:
   - $P_{i'}$ has input $[\mathbf{View}^{j-1}_{\Pi_{\mathcal{V}},i}]_{i'}$, commitments, either $c_i^{i,j}$ or $\{c_{i*}^{i,j}\}_{i*\neq i\in[n]}$, and randomness $r_{i'}^{i,j}$; and samples $M_{i'} \leftarrow \{0,1\}^{\ell'}$
   - $P_{i'}$ participates in an execution of $\Pi_{\mathsf{NxtMsg}_i^j}$ with input
   $$[\mathbf{View}^{j-1}_{\Pi_{\mathcal{V}},i}]_{i'}, M_{i'}, r_{i'}^{i,j} \text{ and } \begin{cases} \{c_{i*}^{i,j}\}_{i*\neq i\in[n]} & \text{if } i = i' \\ c_i^{i,j} & \text{otherwise} \end{cases}$$
   Let $\mathbf{View}_{\mathsf{NxtMsg}_i^j,i'}$ be the view of $P_{i'}$ in this protocol.
   - $P_{i'}$ gets the output $\vec{c} = \{c_k = M_k \oplus (R_k, \{[m^j_{i\to k'}]_k\}_{k'\neq i\in[n]})\}_{k\in[n]}$
   - $P_{i'}$ extracts $\{[m^j_{i\to k'}]_{i'}\}_{k'\neq i\in[n]}, R_{i'} = c_{i'} \oplus M_{i'}$
   - $P_{i'}$ updates the view for each virtual party $V_k \in \mathcal{V} - \{V_i\}$,
   $$[\mathbf{View}^j_{\Pi_{\mathcal{V}},k}]_{i'} = [\mathbf{View}^j_{\Pi_{\mathcal{V}},k}]_{i'} \cup \{[\mathbf{View}^{j-1}_{\Pi_{\mathcal{V}},k}]_{i'}, [m^j_{i\to k}]_{i'}\}$$

   $\Pi_{\mathsf{NxtMsg}_i^j}$ is computed for all $V_i \in \mathcal{V}$ and rounds indexed $j \in [r]$ in $\Pi_{\mathcal{V}}$
3. **Derive the output.** Each $P_i \in \mathcal{P}$ has a share $\{[\mathbf{View}_{\Pi_{\mathcal{V}},j}]_i\}_{j\in[n]}$
   - Each party $P_i \in \mathcal{P}$ sends output shares $[f_j(\vec{x})]_i$ to $P_j \in \mathcal{P}$ where:
   $$\{[f_j(\vec{x})]_i\}_{j\in[n]} \leftarrow \{[\mathbf{View}_{\Pi_{\mathcal{V}},j}]_i\}_{j\in[n]}$$
   - Each P_i computes $f_i(\vec{x}) = [f(\vec{x})]_i = \mathsf{Recon}_{\mathcal{Z}}(\{[f_i(\vec{x})]_j\}_{j\in[n]})$
   - Each P_i broadcast shares to compute $f(\vec{x}) = \mathsf{Recon}_{\mathcal{Z}_{\mathsf{fair}}}(\{[f(\vec{x})]_i\}_{i\in[n]})$

Fig. 7. Multiverse Setting Protocol

B. The structures $\mathcal{Z}^*_{\mathsf{GOD}} = \{Z\}$ where $\mathcal{Z}_{\mathsf{S}}, \mathcal{Z}^*_{\mathsf{GOD}} \cup \mathcal{Z}_{\mathsf{fair}} \in \mathcal{Q}^2$. $\mathcal{Z}^*_{\mathsf{C}}$ is such that,

$$\exists Z' \in \mathcal{Z}^*_{\mathsf{C}}, Z \cup Z' = \mathcal{P}$$

and for each $Z \in \mathcal{Z}_{\mathsf{S}} \cup \mathcal{Z}^*_{\mathsf{fair}}$ there exists a $Z' \in \mathcal{Z}^*_{\mathsf{C}}$ s.t. $Z \subseteq Z'$.

In this section, we construct protocols that combine the two settings above. In order to achieve this for general n-party functionalities f, the starting point is to consider an n-client 3-server virtual protocol that realizes f. Then each server functionality is emulated among the n parties using multiverse setting protocols.

Unbounded Secure Protocols providing GOD. In our protocol satisfying best-in-the-multiverse corruption, in addition to protocols for the above stated settings, we use MPC protocols for general functionalities f that are only maliciously unbounded secure. Such protocols satisfy GOD tolerating any $\mathcal{Z}_{\mathsf{GOD}} \in \mathcal{Q}^2$ and already exist in literature [16,17].

The Virtual Protocol. For an n-party functionality $f(x_1, \cdots, x_n) = \{y_i\}_{i\in[n]}$ there exists a 3-party functionality F that receives from each of the 3 parties a 2-out-of-3 threshold secret share of all the inputs $\{x_i\}_{i\in[n]}$. This functionality reconstructs the inputs from the shares, computes f and secret-shares its outputs. Each of the 3 parties receives a 2-out-of-3 threshold secret share of all the outputs $\{y_i\}_{i\in[n]}$. There exist unbounded secure 3-party protocols that can realize such an F with GOD where a fail-stop adversary can corrupt one party [17].

Let us denote this protocol by $\Pi_{\mathcal{V}}$ and the three parties by $\mathcal{V}_1$, $\mathcal{V}_2$ and $\mathcal{V}_3$. For each party $\mathcal{V}_i$ we refer to its complete view as $\mathbf{View}_{\Pi_{\mathcal{V}},i}$, and $\mathbf{View}^j_{\Pi_{\mathcal{V}},i}$ is its partial view in the protocol up to round j. This includes the security parameter κ, the input X_i, contents of the random tape r_i, and all messages received from all the other parties up to this round.

$$\mathbf{View}^j_{\Pi_{\mathcal{V}},i} = (\kappa, X_i, r_i, \{m^{j'}_{k\to i}\}_{j'\in[j],k\neq i\in[n]})$$

Here $m^{j'}_{k\to i}$ is the message that party $\mathcal{V}_k$ sends $\mathcal{V}_i$ in round j'. Let NxtMsg be the next message function such that,

$$\{m^j_{i\to k}\}_{k\neq i\in[3]} = \mathsf{NxtMsg}^j_i(\mathbf{View}^{j-1}_{\Pi_{\mathcal{V}},i})$$

This function is parameterized by $i \in [3]$ that indexes the virtual party. It takes as input the view up to round $j-1$ and creates all the messages to be delivered to all the parties in round j. Note that after each round,

$$\mathbf{View}^j_{\Pi_{\mathcal{V}},i} = \mathbf{View}^{j-1}_{\Pi_{\mathcal{V}},i} \cup \{m^j_{k\to i}\}_{k\neq i\in[3]}$$

Letting r be the number of rounds in $\Pi_{\mathcal{V}}$, the complete protocol $\Pi_{\mathcal{V}}$ itself can be described as a set of the next message functions,

$$\Pi_{\mathcal{V}} = \{\mathsf{NxtMsg}^j_i(\mathbf{View}^{j-1}_{\Pi_{\mathcal{V}},i})\}_{i\in[3],j\in[r]}$$

Corresponding to each next-message function, there exists an n-party functionality $\mathcal{F}_{\mathsf{NxtMsg}^j_i}$ as in Figure 8. We use pairwise information-theoretic MACs for checking consistency of inputs and outputs of different next-message functions.

Next Message Functionality $\mathcal{F}_{\mathsf{NxtMsg}_i^j}$

Setting: Let $\mathcal{P} = \{P_i\}_{i\in[n]}$ be n parties. Let $\mathcal{Z} = \mathcal{Z}_{\mathsf{C}}^* \cup \mathcal{Z}_{\mathsf{GOD}}$, and $\mathsf{SS}_{\mathcal{Z}} = (\mathsf{Share}_{\mathcal{Z}}, \mathsf{Recon}_{\mathcal{Z}})$ be a $\mathcal{Z}$-secret-sharing scheme. For a 3-party protocol $\Pi_{\mathcal{V}}$ computing $\mathcal{F}$ let $\{m^j_{i\to k}\}_{k\neq i\in[3]} = \mathsf{NxtMsg}_i^j(\mathbf{View}^{j-1}_{\Pi_{\mathcal{V}},i})$ be the next-message function for party V_i in round j. The instructions marked in grey apply for rounds $j > 1$. The n-party functionality $\mathcal{F}_{\mathsf{NxtMsg}_i^j}$ operates as follows:

- **Inputs:** Each party $P_k \in \mathcal{P}$ has:
  - Share of the view of V_i until round $(j-1)$ in $\Pi_{\mathcal{V}}$,
  $$[\mathbf{View}^{j-1}_{\Pi_{\mathcal{V}},i}]_k = \left(\kappa, [x_i]_k, [r_i]_k, \{[m^{j'}_{i'\to i}]_k\}_{j'\in[j-1], i'\neq i\in[3]}\right)$$
  - $\forall k' \neq k \in [n]$, the MAC keys $K_k^{*k'} = (a_k^{*k'}, b_k^{*k'})$ and $K_k^{k'} = (a_k^{k'}, b_k^{k'})$
  - Information-theoretic MACs on all other $n-1$ parties' partial views $\{c^k_{k'}\}_{k'\neq k\in[n]}$ using the keys $\{K_k^{k'}\}_{k'\neq k\in[n]}$
  - Sample a random mask $M_k \leftarrow \{0,1\}^{\ell'}$ where $\ell' = |\mathsf{Output}_k|$.

  P_k gives to $\mathcal{F}_{\mathsf{NxtMsg}_i^j}$ the inputs,
  $$[\mathbf{View}^{j-1}_{\Pi_{\mathcal{V}},i}]_k, M_k, \{K_k^{*k'}\}_{k'\neq k\in[n]}, \{K_k^{k'}, c^k_{k'}\}_{k'\neq k\in[n]}$$
- **Computation:** The functionality $\mathcal{F}_{\mathsf{NxtMsg}_i^j}$ computes:

$$\forall k \in [n], \forall i' \neq k \in [n], \text{ if } c^k_{i'} \neq a_k^{i'} \cdot [\mathbf{View}^{j-1}_{\Pi_{\mathcal{V}},i}]_{i'} + b_k^{i'}, \quad \vec{c} = \mathsf{ABORT}_{i'}$$

otherwise,
$$\mathbf{View}^{j-1}_{\Pi_{\mathcal{V}},i} \leftarrow \mathsf{Recon}_{\mathcal{Z}}(\{[\mathbf{View}^{j-1}_{\Pi_{\mathcal{V}},i}]_k\}_{k\in[n]})$$
$$\{m^j_{i\to i'}\}_{i'\neq i\in[3]} = \mathsf{NxtMsg}_i^j(\mathbf{View}^{j-1}_{\Pi_{\mathcal{V}},i})$$
$$\forall i' \neq i \in [3], \quad \{[m^j_{i\to i'}]_k\}_{k\in[n]} \leftarrow \mathsf{Share}_{\mathcal{Z}}(m^j_{i\to i'})$$
$$\forall k \in [n], \forall k' \neq k \in [n], \quad c^k_{k'} = a_k^{*k'} \cdot \{[m^j_{i\to i'}]_k\}_{i'\neq i\in[3]} + b_k^{*k'}$$
$$\forall k \in [n], \quad \mathsf{Output}_k = \{[m^j_{i\to i'}]_k\}_{i'\neq i\in[3]}, \{K_k^{*k'}, c^k_{k'}\}_{k'\neq k\in[n]}$$
$$\vec{c} = \{M_k \oplus \mathsf{Output}_k\}_{k\in[n]}$$

- **Output:** $\mathcal{F}_{\mathsf{NxtMsg}_i^j}$ gives $\vec{c}$ to each $P_k \in \mathcal{P}$.

Fig. 8. Next Message Functionality for the General Feasibility

The Complete Protocol. The virtual 3-party protocol $\Pi_{\mathcal{V}}$ can be compiled into our final protocol for best-in-the-multiverse MPC. This is done by realizing each next-message functionality corresponding to the functions in $\Pi_{\mathcal{V}}$ using maliciously secure protocols. Let $\mathcal{Z}_{\mathsf{GOD}} = \mathcal{Z}'_{\mathsf{GOD}} \cup \mathcal{Z}^*_{\mathsf{GOD}}$ such that,

$$\forall Z \in \mathcal{Z}'_{\mathsf{GOD}}, \forall Z' \in \mathcal{Z}^*_{\mathsf{C}}, Z \cup Z' \neq \mathcal{P} \tag{2}$$

$$\mathcal{Z}^*_{\mathsf{GOD}} = \{Z\}, \exists Z' \in \mathcal{Z}^*_{\mathsf{C}}, Z \cup Z' = \mathcal{P} \tag{3}$$

All functionalities corresponding to next-message functions of party $\mathcal{V}_1$ are realized using an n-party protocol satisfying $\mathcal{Z}_{\mathsf{GOD}}$ as described before. All functionalities corresponding to next-message functions of party $\mathcal{V}_2$ are realized using an

n-party best-in-the-multiverse protocol where $\mathcal{Z}'_{\mathsf{GOD}}$ and $\mathcal{Z}^*_{\mathsf{C}}$ satisfy Equation 2. All functionalities corresponding to next-message functions of party $\mathcal{V}_3$ are realized using an n-party best-in-the-multiverse protocol where $\mathcal{Z}^*_{\mathsf{GOD}}$ and $\mathcal{Z}^*_{\mathsf{C}}$ satisfy Equation 3. Letting $\mathcal{Z} = \mathcal{Z}^*_{\mathsf{C}} \cup \mathcal{Z}_{\mathsf{GOD}}$, all intermediate views and states of all these virtual parties are $\mathcal{Z}$-secret-shared among the n parties. The complete protocol can be found in Figure 9.

Theorem 5. *Assuming oblivious transfer and secure point-to-point and broadcast channels, let* $\mathcal{Z}_{\mathsf{S}}, \mathcal{Z}^*_{\mathsf{fair}} = \mathcal{Z}_{\mathsf{fair}} \cup \mathcal{Z}_{\mathsf{GOD}} \in \mathcal{Q}^2$*, and* $\mathcal{Z}_{\mathsf{GOD}}$ *and* $\mathcal{Z}_{\mathsf{C}} \subseteq \mathcal{Z}^*_{\mathsf{C}}$ *satisfy*

$$\neg\exists Z_1, Z_2 \in \mathcal{Z}_{\mathsf{GOD}}, \exists Z'_1, Z'_2 \in \mathcal{Z}^*_{\mathsf{C}} \text{ s.t. } (Z_1 \cup Z'_1 = \mathcal{P}) \text{ and } (Z_2 \cup Z'_2 = \mathcal{P})$$

For any n-party functionality f,

- *Let $\Pi_{\mathcal{V}}$ be a 3-party protocol computing $\mathcal{F}$ that outputs a 2-out-of-3 secret-sharing of $f(\vec{x})$ among $\mathcal{V} = \{\mathcal{V}_i\}_{i\in[3]}$. $\Pi_{\mathcal{V}}$ securely computes $\mathcal{F}$ with GOD against an unbounded fail-stop adversary corrupting any one party.*
- *Let Π_1 be an n-party protocol satisfying unbounded GOD tolerating $\mathcal{Z}_{\mathsf{GOD}}$ [17]*
- *Let Π_2 be an n-party multiverse protocol satisfying Equation 2 (Theorem 2,4)*
- *Let Π_3 be an n-party multiverse protocol satisfying Equation 3 (Theorem 3,4)*

The compiled protocol in Figure 9 maliciously securely computes f satisfying:

- ***GOD** holds in the presence of an unbounded adversary w.r.t $\mathcal{Z}_{\mathsf{GOD}} \cap \mathcal{Z}_{\mathsf{S}}$ and a PPT adversary corrupting any subset in $\mathcal{Z}_{\mathsf{GOD}}$.*
- ***Fairness** holds in the presence of an unbounded adversary w.r.t $\mathcal{Z}^*_{\mathsf{fair}} \cap \mathcal{Z}_{\mathsf{S}}$ and a PPT adversary corrupting any subset in $\mathcal{Z}^*_{\mathsf{fair}}$.*
- ***Security with abort** holds in the presence of an unbounded adversary w.r.t $\mathcal{Z}_{\mathsf{S}}$ and a PPT adversary corrupting any subset in $\mathcal{Z}_{\mathsf{C}}$.*

The complete proof for this can be found in Appendix E.

Acknowledgments. The first author was supported by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme under grant agreement No 101124977 (DECRYPSYS). The second author was supported by the United States-Israel Binational Science Foundation (BSF) through Grant No.2020277. The third author was supported by a JPMorgan Chase Faculty Research Award, Technology, and Humanity Fund from the McCourt School of Public Policy at Georgetown University, and a Google Research Award.

References

1. Acharya, A., Hazay, C., Poburinnaya, O., Venkitasubramaniam, M.: Best of both worlds - revisiting the spymasters double agent problem. In: CRYPTO 2023. pp. 328–359 (2023), https://doi.org/10.1007/978-3-031-38557-5_11
2. Badrinarayanan, S., Patranabis, S., Sarkar, P.: Statistical security in two-party computation revisited. In: TCC 2022. pp. 181–210 (2022), https://doi.org/10.1007/978-3-031-22365-5_7

Best-in-the-Multiverse Protocol Π

Setting: Let $\mathcal{P} = \{P_i\}_{i\in[n]}$ be n parties. For function f on $\vec{x} = \{x_i\}_{i\in[n]}$, let P_i has input x_i. Let $\Pi_{\mathcal{V}} = \{\mathsf{NxtMsg}_i^j\}_{i\in[3],j\in[r]}$ be a 3-party protocol computing the functionality $\mathcal{F}$ outputting a 2-out-of-3 secret-sharing of $f(\vec{x})$ among $\mathcal{V} = \{\mathcal{V}_i\}_{i\in[3]}$. $\Pi_{\mathcal{V}}$ securely computes $\mathcal{F}$ with GOD against an unbounded fail-stop adversary corrupting any one party. Let Π_1 be an n-party protocol satisfying $\mathcal{Z}_{\mathsf{GOD}}$. Let Π_2 and Π_3 be n-party best-in-the-multiverse protocols satisfying Equation 2 and 3 respectively. Let $\mathcal{Z} = \mathcal{Z}_{\mathsf{C}}^* \cup \mathcal{Z}_{\mathsf{GOD}}$.

1. **Initialize shared state.** Each party $P_i \in \mathcal{P}$ does the following:
   - Compute 2-out-of-3 sharing $\{[x_i]_k\}_{k\in[3]} \leftarrow \mathsf{Share}_{2,3}(x_i)$
   - Sample $r_i \leftarrow \mathcal{R}(\kappa)$ and compute $\{[r_i]_k\}_{k\in[3]} \leftarrow \mathsf{Share}_{2,3}(r_i)$
   - For each $k \in [3]$, the initial state of each virtual party $\mathcal{V}_k \in \mathcal{V}$ is,
     $$\mathbf{View}^0_{\Pi_{\mathcal{V}},k} = \{[x_i]_k, [r_i]_k\}_{i\in[n]}$$
     compute $\{[\mathbf{View}^0_{\Pi_{\mathcal{V}},k}]_i\}_{i\in[n]} \leftarrow \mathsf{Share}_{\mathcal{Z}}(\mathbf{View}^0_{\Pi_{\mathcal{V}},k})$ by having each party locally compute the sharing of the elements that they have
   - The parties then distribute these shares s.t. P_i has $\{[\mathbf{View}^0_{\Pi_{\mathcal{V}},k}]_i\}_{k\in[3]}$
   - $\forall k \in [3], j \in [r]$, corresponding to each $\mathcal{F}_{\mathsf{NxtMsg}_k^j}$ (Figure 8), $\forall i' \neq i \in [n]$, sample MAC keys $K_{i,k}^{i',j}$
2. **Compute virtual protocol.** For each $\mathsf{NxtMsg}_k^j \in \Pi_{\mathcal{V}}$, let $\Pi_{\mathsf{NxtMsg}_k^j}$ be an n-party protocol computing $\mathcal{F}_{\mathsf{NxtMsg}_k^j}$ (Figure 4) and operating as Π_k, for $k \in [3]$. Each party $P_i \in \mathcal{P}$:
   - Has input $[\mathbf{View}^{j-1}_{\Pi_{\mathcal{V}},k}]_i$, keys $\{K_{i,k}^{i',j}\}_{i'\neq i\in[n]}$; for $j > 1$, MACs with corresponding keys $\{c_{i',k}^{i,j-1}, K_{i,k}^{i',j-1}\}_{i'\neq i\in[n]}$; and samples $M_i \leftarrow \{0,1\}^{\ell'}$
   - Participates in an execution of $\Pi_{\mathsf{NxtMsg}_k^j}$ with input
     $$[\mathbf{View}^{j-1}_{\Pi_{\mathcal{V}},k}]_i, M_i, \{c_{i',k}^{i,j-1}, K_{i,k}^{i',j-1}, K_{i,k}^{i',j}\}_{i'\neq i\in[n]}$$
     Let $\mathbf{View}_{\mathsf{NxtMsg}_k^j,i}$ be the view of P_i in this protocol.
   - Gets $\vec{c} = \{M_{k*} \oplus (\{K_{k*,k}^{i',j}, c_{i',k}^{k*,j}\}_{i'\neq k*\in[n]}, \{[m_{k\to k'}^j]_{k*}\}_{k'\neq k\in[3]})\}_{k*\in[n]}$
   - Extracts $\{[m_{k\to k'}^j]_i\}_{k'\neq k\in[3]}, \{K_{i,k}^{i',j}, c_{i',k}^{i,j}\}_{i'\neq i\in[n]}$
   - Updates the view for each virtual party $\mathcal{V}_{k'} \in \mathcal{V} - \{\mathcal{V}_k\}$,
     $$[\mathbf{View}^j_{\Pi_{\mathcal{V}},k'}]_i = [\mathbf{View}^j_{\Pi_{\mathcal{V}},k'}]_i \cup \{[\mathbf{View}^{j-1}_{\Pi_{\mathcal{V}},k'}]_i, [m_{k\to k'}^j]_i\}$$

   $\Pi_{\mathsf{NxtMsg}_k^j}$ is computed for all $\mathcal{V}_k \in \mathcal{V}$ and rounds indexed $j \in [r]$ in $\Pi_{\mathcal{V}}$
3. **Derive the output.** Each $P_i \in \mathcal{P}$ has a share of $\{[\mathbf{View}_{\Pi_{\mathcal{V}},k}]_i\}_{k\in[3]}$
   - Each party $P_i \in \mathcal{P}$ extracts the shares of the output:
     $$\{[[f(\vec{x})]_k]_i\}_{k\in[3]} \leftarrow \{[\mathbf{View}_{\Pi_{\mathcal{V}},k}]_i\}_{k\in[3]}$$
   - All parties reconstruct the 2-out-of-3 and $\mathcal{Z}$-secret-sharing to get $f(\vec{x})$.

Fig. 9. Best-in-the-Multiverse for General Feasibility

3. Ben-Or, M., Goldwasser, S., Wigderson, A.: Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In: Proceedings of the 20th Annual ACM Symposium on Theory of Computing. pp. 1–10 (1988), https://doi.org/10.1145/62212.62213
4. Blakley, G.R., Kabatianskii, G.: General perfect secret sharing schemes. In: CRYPTO 1995. pp. 367–371. Springer (1995), https://doi.org/10.1007/3-540-44750-4_29
5. Chandramouli, A., Choudhury, A., Patra, A.: A survey on perfectly secure verifiable secret-sharing. ACM Comput. Surv. **54**(11s), 232:1–232:36 (2022), https://doi.org/10.1145/3512344
6. Chaum, D.: The spymasters double-agent problem: Multiparty computations secure unconditionally from minorities and cryptographically from majorities. In: CRYPTO 1989. pp. 591–602 (1989), https://doi.org/10.1007/0-387-34805-0_52
7. Chaum, D., Damgård, I., van de Graaf, J.: Multiparty computations ensuring privacy of each party's input and correctness of the result. In: CRYPTO 1987. pp. 87–119 (1987), https://doi.org/10.1007/3-540-48184-2_7
8. Cohen, R., Lindell, Y.: Fairness versus guaranteed output delivery in secure multiparty computation. In: ASIACRYPT 2014. pp. 466–485 (2014), https://doi.org/10.1007/978-3-662-45608-8_25
9. Cramer, R., Damgård, I., Nielsen, J.B.: Secure Multiparty Computation and Secret Sharing (2015), http://www.cambridge.org/de/academic/subjects/computer-science/cryptography-cryptology-and-coding/secure-multiparty-computation-and-secret-sharing?format=HB&isbn=9781107043053
10. Fitzi, M., Hirt, M., Holenstein, T., Wullschleger, J.: Two-threshold broadcast and detectable multi-party computation. In: EUROCRYPT. pp. 51–67 (2003), https://doi.org/10.1007/3-540-39200-9_4
11. Fitzi, M., Holenstein, T., Wullschleger, J.: Multi-party computation with hybrid security. In: EUROCRYPT 2004. pp. 419–438 (2004), https://doi.org/10.1007/978-3-540-24676-3_25
12. Goldreich, O.: The Foundations of Cryptography - Volume 2: Basic Applications. Cambridge University Press (2004). https://doi.org/10.1017/CBO9780511721656
13. Goldreich, O., Micali, S., Wigderson, A.: How to prove all np-statements in zero-knowledge, and a methodology of cryptographic protocol design. In: CRYPTO 1986. pp. 171–185 (1986), https://doi.org/10.1007/3-540-47721-7_11
14. Hazay, C., Venkitasubramaniam, M., Weiss, M.: The price of active security in cryptographic protocols. In: EUROCRYPT 2020. pp. 184–215 (2020), https://doi.org/10.1007/978-3-030-45724-2_7
15. Hirt, M., Maurer, U.M.: Complete characterization of adversaries tolerable in secure multi-party computation (extended abstract). In: PODC 1997. pp. 25–34 (1997), https://doi.org/10.1145/259380.259412
16. Hirt, M., Maurer, U.M.: Player simulation and general adversary structures in perfect multiparty computation. J. Cryptol. **13**(1), 31–60 (2000), https://doi.org/10.1007/s001459910003
17. Hirt, M., Tschudi, D.: Efficient general-adversary multi-party computation. In: ASIACRYPT 2013. pp. 181–200 (2013), https://doi.org/10.1007/978-3-642-42045-0_10

18. Ishai, Y., Katz, J., Kushilevitz, E., Lindell, Y., Petrank, E.: On achieving the "best of both worlds" in secure multiparty computation. SIAM J. Comput. **40**(1), 122–141 (2011), https://doi.org/10.1137/100783224
19. Ishai, Y., Kushilevitz, E., Ostrovsky, R., Sahai, A.: Zero-knowledge proofs from secure multiparty computation. SIAM J. Comput. **39**, 1121–1152 (2009), https://doi.org/10.1137/080725398
20. Ishai, Y., Prabhakaran, M., Sahai, A.: Founding cryptography on oblivious transfer - efficiently. In: CRYPTO 2008. pp. 572–591 (2008), https://doi.org/10.1007/978-3-540-85174-5_32
21. Katz, J.: On achieving the "best of both worlds" in secure multiparty computation. In: Proceedings of the 39th Annual ACM Symposium on Theory of Computing, San Diego, California, USA, June 11-13, 2007. pp. 11–20 (2007), https://doi.org/10.1145/1250790.1250793
22. Khurana, D., Mughees, M.H.: On statistical security in two-party computation. In: TCC 2020. pp. 532–561 (2020), https://doi.org/10.1007/978-3-030-64378-2_19
23. Lucas, C., Raub, D., Maurer, U.M.: Hybrid-secure MPC: trading information-theoretic robustness for computational privacy. In: PODC 2010. pp. 219–228 (2010), https://doi.org/10.1145/1835698.1835747
24. Patra, A., Ravi, D., Singla, S.: On the exact round complexity of best-of-both-worlds multi-party computation. In: ASIACRYPT 2020. pp. 60–91 (2020), https://doi.org/10.1007/978-3-030-64840-4_3
25. Shamir, A.: How to share a secret. Commun. ACM **22**(11), 612–613 (1979), http://doi.acm.org/10.1145/359168.359176
26. Yao, A.C.: How to generate and exchange secrets (extended abstract). In: FoCS. pp. 162–167 (1986)

A Proof of Lemma 1

Proof. The proof for this lemma extends the lower bound result in [18] to the non-threshold case. In their proof, for $t+s=n$ and $t<\frac{n}{2}$ they define a function f that cannot be securely computed when either (1) the first t parties abort at any round and the adversary has the view of the first s parties, or (2) the last t parties abort and the adversary has the view of the last s parties. We extend this to the non-threshold case by generalizing this functionality to one that cannot be computed when, for $Z_1, Z_2 \in \mathcal{Z}_{\mathsf{GOD}}$, either (1) a set $Z_1 \in \mathcal{Z}_{\mathsf{GOD}}$ of the parties abort at any round and the adversary has the view of $\mathcal{P}-Z_2$, or (2) the parties in Z_2 abort and the adversary adversary has the view of $\mathcal{P}-Z_1$. The rest of the analysis remains the same as in the original proof.

Let $\mathcal{P}=\{P_i\}_{i\in[n]}$ be the set of n parties. Let $\mathcal{Z}_{\mathsf{GOD}}$ be the adversary structure for MPC with Guaranteed Output Delivery (GOD) and $\mathcal{Z}_{\mathsf{C}}$ be the adversary structure for MPC with security with abort. We need to show that when the above condition in Theorem 1 is not satisfied, there exists an n-party function f that cannot be securely computed simultaneously with GOD tolerating $\mathcal{Z}_{\mathsf{GOD}}$ and security with abort tolerating $\mathcal{Z}_{\mathsf{C}}$, even in the presence of fail-stop adversaries.

Firstly, note that when $\mathcal{Z}_{\mathsf{GOD}} \notin \mathcal{Q}^2$, the proof of this theorem follows from the fact that there exist functionalities that cannot be computed tolerating this, even in the plain model [15]. So it remains to consider only the case for any $\mathcal{Z}_{\mathsf{GOD}} \in \mathcal{Q}^2$. Let $\mathcal{Z}_{\mathsf{C}}$ be an adversary structure such that,

$$\exists Z_1, Z_2 \in \mathcal{Z}_{\mathsf{GOD}} \text{ s.t. } (Z_1 - Z_2 \neq \phi) \cap (Z_2 - Z_1 \neq \phi)$$
$$\exists Z_1', Z_2' \in \mathcal{Z}_{\mathsf{C}} \text{ s.t. } (Z_1 \cup Z_1' = \mathcal{P}) \cap (Z_2 \cup Z_2' = \mathcal{P})$$

It follows from the fact that $\mathcal{Z}_{\mathsf{GOD}} \in \mathcal{Q}^2$ that $Z_1, Z_2 \subset \mathcal{P}$ and $Z_1 \cup Z_2 \neq \mathcal{P}$. So there exists a party $P_j \in \mathcal{P}$ such that $P_j \notin Z_1$ and $P_j \in Z_2 \subset Z_1'$. Similarly, let party $P_i \in \mathcal{P}$ be such that $P_i \in Z_1 \subset Z_2'$ and $P_i \notin Z_2$. Let us define a function f as follows:

- parties P_i and P_j have input bits b_i and b_j respectively and all other parties have no inputs
- P_i and P_j both receive $b_i \oplus b_j$ and all other parties do not receive any outputs.

Let π be a protocol that securely computes f in the presence of fail-stop adversaries tolerating $\mathcal{Z}_{\mathsf{GOD}}$ and $\mathcal{Z}_{\mathsf{C}}$. Similarly like in [18], we assume that π operates in a fixed number r of *segments*, each with n rounds where only a party P_k sends a message in the k^{th} round in a segment. We assume that if π runs honestly, then the outputs of P_i and P_j agree and are correct with probability at least $\frac{7}{8}$.

Let us define the sets of parties $A, B, C \subset \mathcal{P}$ such that, $A = Z_1 \in \mathcal{Z}_{\mathsf{GOD}}$, $B = \mathcal{P} - (Z_1 \cup Z_2)$ and $C = Z_2 \in \mathcal{Z}_{\mathsf{GOD}}$. Note that $P_i \in A$ and $P_j \in C$ and these parties lie in no other set. Consider the real-world execution in which the environment E chooses inputs for P_i and P_j uniformly and independently. Then all the parties run the protocol honestly, except that the parties in A or C can possibly abort. For $k \in [r]$, let v_i^k denote the final output of P_i when all parties in C abort in segment $k+1$. Similarly, let v_j^k denote the final output of P_j when

all parties in A abort in segment $k+1$. Note that v_i^k (resp. v_j^k) can be computed from the joint view of the parties in $(A-C)\cup B$ (resp. $B\cup(C-A)$) as soon as all the parties in C (resp. A) have sent their messages in the k^{th} segment.

Security of π implies that for all $k\in[r]$, it holds that $v_i^k, v_j^k \in \{0,1\}$ (and not $\perp$) except with negligible probability. This must hold since π satisfies GOD tolerating $\mathcal{Z}_{\mathsf{GOD}}$. For simplicity, assuming that $v_i^k, v_j^k \in \{0,1\}$ with probability 1, consider the following summation with all probabilities with respect to the real-world execution:

$$\big(\Pr[v_i^0=1\cap v_j^0=1]+\Pr[v_i^0=0\cap v_j^r=1]-\frac{1}{2}\big) \tag{4}$$

$$+\big(\Pr[v_i^0=0\cap v_j^0=0]+\Pr[v_i^0=1\cap v_j^r=0]-\frac{1}{2}\big) \tag{5}$$

$$+\sum_{k=0}^{r-1}\Big[\big(\Pr[v_i^k=0\cap v_j^{k+1}=0]+\Pr[v_i^k=1\cap v_j^k=0]-\frac{1}{2}\big) \tag{6}$$

$$+\big(\Pr[v_i^k=1\cap v_j^{k+1}=1]+\Pr[v_i^k=0\cap v_j^k=1]-\frac{1}{2}\big) \tag{7}$$

$$+\big(\Pr[v_i^{k+1}=0\cap v_j^{k+1}=0]+\Pr[v_i^k=0\cap v_j^{k+1}=1]-\frac{1}{2}\big) \tag{8}$$

$$+\big(\Pr[v_i^{k+1}=1\cap v_j^{k+1}=1]+\Pr[v_i^k=1\cap v_j^{k+1}=0]-\frac{1}{2}\big)\Big], \tag{9}$$

which evaluates to:

$$\begin{aligned}
&\Pr[v_i^0=1\cap v_j^0=1]+\Pr[v_i^0=0\cap v_j^r=1]\\
&+\Pr[v_i^0=0\cap v_j^0=0]+\Pr[v_i^0=1\cap v_j^r=0]-1\\
&+\sum_{k=0}^{r-1}\big[\Pr[v_i^k=1\cap v_j^k=0]+\Pr[v_i^k=0\cap v_j^k=1]\\
&\quad+\Pr[v_i^{k+1}=0\cap v_j^{k+1}=0]+\Pr[v_i^{k+1}=1\cap v_j^{k+1}=1]-1\big]
\end{aligned}$$

$$\begin{aligned}
&=\Pr[v_i^0=1\cap v_j^0=1]+\Pr[v_i^0=0\cap v_j^r=1]+\Pr[v_i^0=0\cap v_j^0=0]\\
&+\Pr[v_i^0=1\cap v_j^r=0]+\Pr[v_i^0=1\cap v_j^0=0]+\Pr[v_i^0=0\cap v_j^0=1]\\
&+\Pr[v_i^r=0\cap v_j^r=0]+\Pr[v_i^r=1\cap v_j^r=1]-2
\end{aligned}$$

$$=\Pr[v_i^0=0\cap v_j^r=1]+\Pr[v_i^0=1\cap v_j^r=0]+\Pr[v_i^r=v_j^r]-1$$

$$\begin{aligned}
&\geq\Pr[v_i^0=0\cap b_i\neq b_j]-\Pr[v_j^r\neq 1\cap b_i\neq b_j]+\Pr[v_i^0=1\cap b_i=b_j]\\
&-\Pr[v_j^r\neq 0\cap b_i=b_j]+\Pr[v_i^r=v_j^r]-1
\end{aligned}$$

$$\geq\Pr[b_i\neq b_j\big|v_i^0=0]\cdot\Pr[v_i^0=0]+\Pr[b_i=b_j\big|v_i^0=1]\cdot\Pr[v_i^0=1]+\frac{3}{4}-1$$

using the assumed correctness of π when run honestly to completion. Since v_i^0 is independent of P_j's input bit b_j, we have,

$$\Pr[b_i \neq b_j \big| v_i^0 = 0] = \Pr[b_i = b_j \big| v_i^0 = 1] = \frac{1}{2}$$

It follows that the above sum is at least $\frac{1}{4}$ and so at least one of the summands (4) – (9) is at least $p(\kappa) = \frac{1}{4(4r(\kappa)+2)}$, which is noticeable. It remains to show that this would imply that π does not securely compute f with abort in the presence of a fail-stop adversary with adversary structure $\mathcal{Z}_{\mathsf{C}}$. Consider the following cases:

1. Consider the case where,

   $$\Pr[v_i^0 = 1 \cap v_j^0 = 1] + \Pr[v_i^0 = 0 \cap v_j^r = 1] - \frac{1}{2} \geq p(\kappa)$$

   Here, consider an adversary that corrupts the parties in $A \cup B \in \mathcal{Z}_{\mathsf{C}}$ and does the following: it computes v_i^0 and if this is 1, it aborts all parties in A and has parties in B run π honestly with the rest of the honest parties. If $v_i^0 = 0$, the adversary has all the parties it controls run the protocol honestly. The probability that P_j outputs 1 in this real execution is,

   $$\Pr[v_i^0 = 1 \cap v_j^0 = 1] + \Pr[v_i^0 = 0 \cap v_j^r = 1] \geq \frac{1}{2} + p(\kappa)$$

   However, in an ideal execution, when an adversary corrupts $A \cup B$, the honest P_j will not output 1 with probability greater than $\frac{1}{2}$. Therefore π does not securely compute f with abort when a fail-stop adversary has adversary structure $\mathcal{Z}_{\mathsf{C}}$.
   A similar argument holds when,

   $$\Pr[v_i^0 = 0 \cap v_j^0 = 0] + \Pr[v_i^0 = 1 \cap v_j^r = 0] - \frac{1}{2} \geq p(\kappa)$$

   for when a real-world adversary corrupts the parties in $A \cup B \in \mathcal{Z}_{\mathsf{C}}$ and forces P_j to output 0 instead, with probability noticeably greater than $\frac{1}{2}$.
2. Consider the case where there exists an index $k \in \{0, \ldots, r(\kappa)-1\}$ for which,

   $$\Pr[v_i^k = 0 \cap v_j^{k+1} = 0] + \Pr[v_i^k = 1 \cap v_j^k = 0] - \frac{1}{2} \geq p(\kappa)$$

   The adversary is given k as an auxiliary input and corrupts the parties in $A \cup B \in \mathcal{Z}_{\mathsf{C}}$. It runs the protocol honestly up to the end of segment k and computes v_i^k. If this is 1, the adversary aborts all the parties in A. If $v_i^k = 0$, the parties in A compute and send their messages for the next segment and then abort. In both cases, the parties in B continue to run the rest of the protocol honestly until completion. The probability that P_j outputs 0 in this real execution is,

   $$\Pr[v_i^k = 0 \cap v_j^{k+1} = 0] + \Pr[v_i^k = 1 \cap v_j^k = 0] \geq \frac{1}{2} + p(\kappa)$$

However, as before, in an ideal execution, when an adversary corrupts $A \cup B$, the honest P_j will not output 0 with probability greater than $\frac{1}{2}$.
A similar argument holds when,

$$\Pr[v_i^k = 1 \cap v_j^{k+1} = 1] + \Pr[v_i^k = 0 \cap v_j^k = 1] - \frac{1}{2} \geq p(\kappa)$$

for when a real-world adversary corrupts the parties in $A \cup B \in \mathcal{Z}_{\mathsf{C}}$ and forces P_j to output 1 with probability noticeably greater than $\frac{1}{2}$.

3. Consider the case where there exists an index $k \in \{0, \ldots, r(\kappa)\}$ for which,

$$\Pr[v_i^{k+1} = 0 \cap v_j^{k+1} = 0] + \Pr[v_i^k = 0 \cap v_j^{k+1} = 1] - \frac{1}{2} \geq p(\kappa)$$

The adversary is given k as an auxiliary input and corrupts the parties in $C \cup B \in \mathcal{Z}_{\mathsf{C}}$. It runs the protocol honestly up to segment k where the parties in C have not yet sent their messages. It computes v_j^k and if this is 1, the adversary aborts all the parties in C. If $v_j^k = 0$, the parties in C compute and send their messages for this segment and then abort. In both cases, the parties in B continue to run the rest of the protocol honestly until completion. The probability that P_i outputs 0 in this real execution is,

$$\Pr[v_i^{k+1} = 0 \cap v_j^{k+1} = 0] + \Pr[v_i^k = 0 \cap v_j^{k+1} = 1] \geq \frac{1}{2} + p(\kappa)$$

However, as before, in an ideal execution, when an adversary corrupts $C \cup B$, the honest P_i will not output 0 with probability greater than $\frac{1}{2}$.
A similar argument holds when,

$$\Pr[v_i^{k+1} = 1 \cap v_j^{k+1} = 1] + \Pr[v_i^k = 1 \cap v_j^{k+1} = 0] - \frac{1}{2} \geq p(\kappa)$$

for when a real-world adversary corrupts the parties in $C \cup B \in \mathcal{Z}_{\mathsf{C}}$ and forces P_i to output 1 with probability noticeably greater than $\frac{1}{2}$.

B Proof of Theorem 2

Proof. Let $\mathcal{Z}_{\mathsf{GOD}} \in \mathcal{Q}^2$ and $\mathcal{Z}_{\mathsf{C}}^*$ be the maximal adversary structure such that both of these satisfy Theorem 1. For the protocol Π_i in Figure 4, the proof for Theorem 2 follows from analysing the following cases separately:

1. We need to show that security with Guaranteed Output Delivery (GOD) holds in the presence of an unbounded adversary when it corrupts any set of parties in the adversary structure $\mathcal{Z}_{\mathsf{S}}^{i,\mathsf{GOD}}$, where $\mathcal{Z}_{\mathsf{S}}^{i,\mathsf{GOD}}$ contains all sets in $\mathcal{Z}_{\mathsf{GOD}}$ not including party P_i.
2. We need to show that GOD is also satisfied for a PPT adversary corrupting any set of parties in $\mathcal{Z}_{\mathsf{GOD}}$, including those containing party P_i.
3. We show that security with abort holds for an unbounded adversary when it corrupts any set of parties in $\mathcal{Z}_{\mathsf{S}}^i$, an adversary structure that contains all sets in $\mathcal{Z}_{\mathsf{C}}^*$ without P_i as a member.

4. We also show that security with abort holds for a PPT adversary when it corrupts any set of parties in $\mathcal{Z}_{\mathsf{C}}^*$, even all the sets containing P_i.
5. Finally, we show that when an unbounded adversary corrupts any set of parties in $\mathcal{Z}_{\mathsf{GOD}}$, even those including party P_i, the protocol will still execute until completion with all honest parties receiving the correct outputs of the functionality, although the privacy of the honest parties' inputs is completely compromised.

In protocol Π_i in Figure 4, let Π_i^* be the base protocol that computes f with maliciously fall-back security with identifiable abort tolerating $\mathcal{Z}_{\mathsf{S}}^{i'}$, where $\mathcal{Z}_{\mathsf{S}}^{i'}$ is the adversary structure containing all subsets of parties not including P_i. For this protocol, let $\mathsf{Sim}_{\mathsf{S}}$ be the PPT machine that simulates the view produced by an interaction with an unbounded malicious adversary corrupting any set in $\mathcal{Z}_{\mathsf{S}}^{i'}$. Similarly, let $\mathsf{Sim}_{\mathsf{C}}$ be the PPT simulator for the view of a PPT malicious adversary corrupting any set of parties, even including P_i.

GOD for Unbounded Adversaries. The adversary structure $\mathcal{Z}_{\mathsf{S}}^{i,\mathsf{GOD}}$ contains all sets in $\mathcal{Z}_{\mathsf{GOD}}$ that do not have party P_i as its member. Note that this structure is a subset of $\mathcal{Z}_{\mathsf{S}}^{i'}$. We need to show that security with GOD holds in the presence of a malicious unbounded adversary $\mathcal{A}$ corrupting a set $Z \in \mathcal{Z}_{\mathsf{S}}^{i,\mathsf{GOD}}$. For this, consider the following PPT simulator $\mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}$ that interacts with $\mathcal{A}$ in the ideal world for Guaranteed Output Delivery. It makes black-box calls to the simulator $\mathsf{Sim}_{\mathsf{S}}$ of the underlying fall-back secure protocol Π_i^*. $\mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}$ gets as input the statistical security parameter 1^s and would operate as follows:

– **Commit Phase:** The simulator $\mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}$ acts as follows:
  - $\forall$ corrupt parties $P_j \in Z$, receive $(\Lambda_j, \{s_k^j\}_{k\in\mathcal{P}-Z})$ from $\mathcal{A}$
  - $\forall$ honest parties $P_j \notin Z$ that are not party P_i, sample r_j at random, compute the VSS with statistically binding commitments $\{s_k^j\}_{k\in[n]} \leftarrow \mathsf{VSS}_{\mathsf{Com}_{\mathsf{SB}}}(r_j)$ and send $\{s_k^j\}_{k\in Z}$ to $\mathcal{A}$
  - on behalf of the honest P_i, sample (r_i, x_i), honestly compute the VSS with statistically hiding commitments and send $\{s_k^i\}_{k\in Z}$ to $\mathcal{A}$
– **Protocol Emulation:** $\mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}$ invokes the simulator $\mathsf{Sim}_{\mathsf{S}}$ for protocol Π_i^* with input 1^s and interacts with the adversary $\mathcal{A}$.
  - In this interaction, $\mathsf{Sim}_{\mathsf{S}}$ extracts the adversary's inputs to Π_i^* to give to the ideal functionality computing f.
  - The simulator $\mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}$, emulating this ideal functionality, accepts these extracted inputs from $\mathsf{Sim}_{\mathsf{S}}$.
  - If this protocol Π_i^* executes to completion without aborting, then $\mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}$ gives these inputs to the ideal functionality $\mathcal{F}$ and receives $f(x)$ from it. It then, emulating the functionality, passes this to $\mathsf{Sim}_{\mathsf{S}}$ for it to complete the protocol simulation.
  - For the j^{th} execution of the 'Protocol Emulation' phase, let $Z_j^* \subseteq Z$ be the set of corrupt parties that have already aborted before this phase has started. Then $\mathsf{Sim}_{\mathsf{S}}$ outputs the following from this interaction:

$$\hat{\mathbf{View}}_{\Pi_i^*}^{Z-Z_j^*} \leftarrow \langle \mathsf{Sim}_{\mathsf{S}}(1^s, \{x_k, r_k\}_{P_k\in Z_j^*}), \mathcal{A}\rangle$$

– **Abort Handling:** If $\mathcal{A}$ causes the protocol to abort on behalf of any corrupted party P_j, $\mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}$, receiving j from $\mathsf{Sim}_{\mathsf{S}}$, honestly participates in the reconstruction of the input and randomness of this party in the 'Commit Phase' with all the values $(c_j, \Lambda_j, \{s_k^j\}_{k\in\mathcal{P}-Z})$ received on behalf of the honest parties. Then 'Protocol Emulation' is re-started with each party locally computing the messages of the aborted party with these inputs and randomness.

Note that when an adversary corrupts a set of parties $Z \in \mathcal{Z}_{\mathsf{S}}^{i,\mathsf{GOD}}$, the VSS access structure is such that all the parties in the set $\overline{Z} = \mathcal{P} - Z$ can always reconstruct the secret. Therefore, for any corrupt party P_j that aborts, its randomness r_j can always be reconstructed using the shares $\{s_k^j\}_{k\in\mathcal{P}-Z}$ with $\mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}$ alone. Further, its input $x_j = \Lambda_j \oplus r_j$ can also be reconstructed since the simulator receives all the messages from $\mathcal{A}$ on party P_i's behalf. So it follows that for every corrupt party that aborts, the protocol emulation can be re-started with these values and the protocol will finally securely terminate with the honest parties receiving their desired outputs. Therefore, Guaranteed Output Delivery is satisfied.

Let Z^* be a set of indices where the j^{th} index denotes the j^{th} protocol emulation. The joint view produced by $\mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}$ with the honest parties' outputs is distributed as,

$$\Bigg\{ \Big\{\Lambda_j, \{s_k^j\}_{k\in\mathcal{P}-Z}\Big\}_{P_j\in Z}, \Big\{\{s_k^{*j}\}_{k\in Z}\Big\}_{P_j\in\mathcal{P}-Z}, \\ \Big\{\hat{\mathbf{View}}_{\Pi_i^*}^{Z-Z_j^*}, \mathbf{View}_{\Pi_i^*,Z_j^*}(\{r_k, x_k\}_{P_k\in Z_j^*})\Big\}_{j\in Z^*}, f(x)\Bigg\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n}$$

It remains to argue that the above view is statistically close to the real view in the protocol Π_i. For this, consider the following hybrids:

– Hybrid H_0: This is the joint distribution with the output of the simulator $\mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}$ as given above.
– Hybrid H_1: This is a hybrid experiment where, in the adversary's view, the inputs and randomness of the honest parties that are used in the 'Commit Phase' is that of the real execution of the protocol. The rest of the protocol emulation is performed as in the simulation $\mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}$.

$$\Bigg\{ \Big\{\Lambda_j, \{s_k^j\}_{k\in\mathcal{P}-Z}\Big\}_{P_j\in Z}, \Big\{\{s_k^{j}\}_{k\in Z}\Big\}_{P_j\in\mathcal{P}-Z}, \\ \Big\{\hat{\mathbf{View}}_{\Pi_i^*}^{Z-Z_j^*}, \mathbf{View}_{\Pi_i^*,Z_j^*}(\{r_k, x_k\}_{P_k\in Z_j^*})\Big\}_{j\in Z^*}, f(x)\Bigg\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n}$$

This distribution differs from hybrid H_0 in that the VSS shares of the honest parties correspond to the real randomness of the honest parties, and not that in the simulation, although they are distributed identically. Furthermore, for the honest P_i, the VSS shares of the inputs correspond to the real input and not arbitrary values picked uniformly at random as in the simulation.

– Hybrid H_2: This hybrid distribution is the same as the joint view in the real execution. This differs from hybrid H_1 in that the 'Protocol Emulation' is carried out as in real executions of Π_i^* instead of using calls to Sim_S.

It remains to show that this set of adjacent hybrids are statistically close.

Claim. In the VSS protocol (Figure 3) instantiated with statistically hiding commitments $\mathsf{VSS}_{\mathsf{Com}_\mathsf{SH}}$ for the adversary structure $\mathcal{Z}_\mathsf{C}^*$, for $Z \in \mathcal{Z}_\mathsf{S}^{i,\mathsf{GOD}}$ s.t. $Z \in \mathcal{Z}_\mathsf{C}^*$, assuming that the statistical distance between the distributions of the set of shares $\{s_k^i\}_{k\in Z}$ and $\{s_k^{*i}\}_{k\in Z}$, derived from two different inputs, is a negligible value ϵ, the statistical distance between the distributions H_0 and H_1 is also ϵ. Hence,

$$\mathsf{H}_0 \overset{\mathrm{s}}{\equiv} \mathsf{H}_1$$

Proof. Let $\overline{Z} \subseteq \mathcal{P}$ denote the set of honest parties in this protocol execution and let $s = |\overline{Z}|$ be the size of this set. This set contains the party P_i. First, note that for all other parties $P_j \in \overline{Z}$ s.t. $P_j \neq P_i$, the simulator, like in the real execution, samples r_j on behalf of this party uniformly at random and creates VSS shares with statistically binding commitments. Then the corrupt parties' shares are sent to the adversary $\mathcal{A}$. Hence these shares are distributed identically in the real and simulated view.

It follows that the only difference in the hybrid distributions H_0 and H_1 originates from the set of corrupt parties' shares generated using the VSS protocol with statistically hiding commitments. These correspond to the input x_i and randomness r_i of the party P_i. Again, in both executions the distribution of r_i is identical. However, in hybrid distribution H_0, this input is an arbitrarily picked x_i^*, while in the real execution this is the honest party's real input x_i. Note that since $Z \in \mathcal{Z}_\mathsf{C}^*$, and the VSS uses a $\mathcal{Z}_\mathsf{C}^*$-secret-sharing, the adversary will never have enough shares to reconstruct the input. Further, each VSS share contains a set of statistically hiding commitments. It follows that if the statistical distance between these distributions of shares $\{s_k^i\}_{k\in Z}$ and $\{s_k^{*i}\}_{k\in Z}$ is a value ϵ, that is negligible in s, since this is the only difference between the two hybrid distributions, H_0 and H_1 are also ϵ apart.

Claim. Let the protocol Π_i^* is maliciously fall-back secure tolerating $\mathcal{Z}_\mathsf{S}^{i'}$. Then, for all 'Protocol Emulation' executions $j \in Z_j^*$, assuming $\left\{\hat{\mathbf{View}}_{\Pi_i^*}^{Z-Z_j^*} \leftarrow \langle \mathsf{Sim}_\mathsf{S}(1^s, \{x_k, r_k\}_{P_k\in Z_j^*}), \mathcal{A}\rangle\right\}_{r\in\vec{r}}$ is distributed statistically close to the distribution of the real view $\left\{\mathbf{View}_{\Pi_i^*}^{Z-Z_j^*}\right\}_{r\in\vec{r}}$ with statistical distance ϵ, negligible in s, then the statistical distance between the distributions H_1 and H_2 is upper bounded by $|Z^*|\epsilon$. Hence,

$$\mathsf{H}_1 \overset{\mathrm{s}}{\equiv} \mathsf{H}_2$$

Proof. Letting Z^* be the set of indices denoting the executions of the 'Protocol Emulation' phase and $s = |Z^*|$ be the size of this set, the proof of the above claim follows from considering the following set of hybrids:

– Hybrid H'_0 : this distribution is the same as that of hybrid H_1 where the view in the 'Commit Phase' is identical to that in the real distribution but the complete protocol emulation is distributed as in the simulation $\mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}$.
– $\forall j \in Z^*, \mathsf{H}'_j$: in this distribution, up to the j^{th} protocol emulation, the view is generated exactly as in the real execution. Beyond this, each k^{th} protocol emulation view is generated as in the simulation:

$$\hat{\mathbf{View}}_{\Pi_i^*}^{Z-Z_k^*} \leftarrow \langle \mathsf{Sim}_{\mathsf{S}}(1^s, \{x_{k'}, r_{k'}\}_{P_{k'} \in Z_k^*}), \mathcal{A} \rangle$$

– The last of these hybrids H'_s is distributed the same as H_2 which is the real distribution.

It remains to show that for each $j \in Z^*$, the adjacent hybrid distributions H'_j and H'_{j-1} are statistically close. Note that each such set of adjacent hybrids only differ in the view of the j^{th} execution of the 'Protocol Emulation' phase. In hybrid H'_j, this is as in the real protocol execution, while in H'_{j-1} this is a simulated view generated using an interaction with $\mathsf{Sim}_{\mathsf{S}}$. For the protocol Π_i^*, the real view for the adversary and that generated using $\mathsf{Sim}_{\mathsf{S}}$ are statistically close with distance ϵ. Since this is the only difference between these adjacent hybrids, it follows that the complete hybrids are also statistically close. Hence, the hybrid distributions H_1 and H_2 are also at most $s\epsilon$ apart.

GOD for PPT Adversaries. Let $\mathcal{A}$ be a malicious PPT adversary corrupting a set $Z \in \mathcal{Z}_{\mathsf{GOD}}$. Consider a PPT simulator $\mathsf{Sim}_{\mathsf{C},\mathsf{GOD}}$ that simulates the view of $\mathcal{A}$ in the ideal world for GOD. Note that if $Z \in \mathcal{Z}_{\mathsf{S}}^{i,\mathsf{GOD}}$, then $\mathsf{Sim}_{\mathsf{C},\mathsf{GOD}}$ can just interact like $\mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}$ as above. However, if $P_i \in Z$, then it would need to make black-box calls to the simulator $\mathsf{Sim}_{\mathsf{C}}$ of the underlying fall-back secure protocol Π_i^*. This $\mathsf{Sim}_{\mathsf{C},\mathsf{GOD}}$ gets the computational security parameter 1^κ as input and would operate as follows:

– **Commit Phase:** The simulator $\mathsf{Sim}_{\mathsf{C},\mathsf{GOD}}$ acts as follows:
  • $\forall$ corrupt parties $P_j \in Z$, receive $\{s_k^j\}_{k \in \mathcal{P}-Z}$ from $\mathcal{A}$
  • $\forall$ honest parties $P_j \in \mathcal{P} - Z$, sample (x_j, r_j) at random, then honestly compute the VSS with computationally hiding (statistically binding) commitments for the randomness, and give $(\Lambda_j, \{s_k^j\}_{k \in Z})$ to $\mathcal{A}$
– **Protocol Emulation:** $\mathsf{Sim}_{\mathsf{C},\mathsf{GOD}}$ invokes the simulator $\mathsf{Sim}_{\mathsf{C}}$ for protocol Π_i^* with input 1^κ and interacts with the adversary $\mathcal{A}$.
  • In this interaction, $\mathsf{Sim}_{\mathsf{C}}$ extracts the adversary's inputs to Π_i^* to give to the ideal functionality computing f.
  • The simulator $\mathsf{Sim}_{\mathsf{C},\mathsf{GOD}}$, emulating this ideal functionality, accepts these extracted inputs from $\mathsf{Sim}_{\mathsf{C}}$.
  • If this protocol Π_i^* executes to completion without aborting, then $\mathsf{Sim}_{\mathsf{C},\mathsf{GOD}}$ gives these inputs to the ideal functionality $\mathcal{F}$ and receives $f(x)$ from it. It then, emulating the functionality, passes this to $\mathsf{Sim}_{\mathsf{C}}$ for it to complete the protocol simulation.

- For the j^{th} execution of the 'Protocol Emulation' phase, let $Z_j^* \subseteq Z$ be the set of corrupt parties that have already aborted before this phase has started. Then $\mathsf{Sim_C}$ outputs the following from this interaction:

$$\hat{\mathbf{View}}_{\Pi_i^*}^{Z-Z_j^*} \leftarrow \langle \mathsf{Sim_C}(1^\kappa, \{x_k, r_k\}_{P_k \in Z_j^*}), \mathcal{A} \rangle$$

– **Abort Handling:** If $\mathcal{A}$ causes the protocol to abort on behalf of any corrupted party P_j, $\mathsf{Sim_{C,GOD}}$, receiving j from $\mathsf{Sim_C}$, honestly participates in the reconstruction of the input and randomness of this party in the 'Commit Phase' with all the values $\{s_k^j\}_{k \in \mathcal{P}-Z}$ received on behalf of the honest parties. Then 'Protocol Emulation' is re-started with each party locally computing the messages of the aborted party with these inputs and randomness.

Note that when an adversary corrupts a set $Z \in \mathcal{Z}_{\mathsf{GOD}}$, the VSS access structure is such that all the parties in the set $\overline{Z} = \mathcal{P} - Z$ can always reconstruct the secret. Therefore, for any corrupt party P_j that aborts, its randomness r_j can always be reconstructed using the shares $\{s_k^j\}_{k \in \mathcal{P}-Z}$ with $\mathsf{Sim_{C,GOD}}$ alone. Further, its input $x_j = \Lambda_j \oplus r_j$ can also be reconstructed, either with the help of P_i if it has not caused an abort, or from an emulation of P_i after its input and randomness has been recovered. So it follows that for every corrupt party that aborts, the protocol emulation can be re-started with these values and the protocol will finally securely terminate with the honest parties receiving their desired outputs. Therefore, Guaranteed Output Delivery is satisfied.

Let $Z^* \subseteq Z$ be the set of indices where the j^{th} index denotes the j^{th} protocol emulation. The joint view produced by $\mathsf{Sim_{C,GOD}}$ with the honest parties' outputs is distributed as,

$$\Big\{ \big\{\Lambda_j^*, \{s_k^{*j}\}_{k \in \mathcal{P}-Z}\big\}_{P_j \in Z}, \big\{\{s_k^j\}_{k \in Z}\big\}_{P_j \in \mathcal{P}-Z}, \\ \big\{\hat{\mathbf{View}}_{\Pi_i^*}^{Z-Z_j^*}, \mathbf{View}_{\Pi_i^*, Z_j^*}(\{r_k, x_k\}_{P_k \in Z_j^*})\big\}_{j \in Z^*}, f(x) \Big\}_{\kappa \in \mathbb{N}, \vec{r} \in \mathcal{R}^n}$$

It remains to argue that the above view is computationally indistinguishable from the real view in the protocol Π_i. For this, consider the following hybrids:

– Hybrid H_0: This is the joint distribution with the output of the simulator $\mathsf{Sim_{C,GOD}}$ as given above.
– Hybrid H_1: This is a hybrid experiment where, in the adversary's view, the inputs and randomness of the honest parties that are used in the 'Commit Phase' is that of the real execution of the protocol. The rest of the protocol emulation is performed as in the simulation $\mathsf{Sim_{C,GOD}}$.

$$\Big\{ \big\{\Lambda_j, \{s_k^j\}_{k \in \mathcal{P}-Z}\big\}_{P_j \in Z}, \big\{\{s_k^j\}_{k \in Z}\big\}_{P_j \in \mathcal{P}-Z}, \\ \big\{\hat{\mathbf{View}}_{\Pi_i^*}^{Z-Z_j^*}, \mathbf{View}_{\Pi_i^*, Z_j^*}(\{r_k, x_k\}_{P_k \in Z_j^*})\big\}_{j \in Z^*}, f(x) \Big\}_{\kappa \in \mathbb{N}, \vec{r} \in \mathcal{R}^n}$$

This differs from H_0 in that the VSS shares of the honest parties correspond to the real randomness of the honest parties, and not values picked uniformly at random in the simulation. This is also true of the commitment and the masked input that is made using the real input and not one arbitrarily picked by the simulator.
- Hybrid H_2: This hybrid distribution is the same as the view in the real execution. This differs from H_1 in that the 'Protocol Emulation' is carried out as in real executions of Π_i^* instead of using calls to Sim_C.

It now remains to show that this set of adjacent hybrids are computationally indistinguishable.

Claim. In the VSS protocol (Figure 3) instantiated with computationally hiding (statistically binding) commitments $\mathsf{VSS}_{\mathsf{Com}_{\mathsf{SB}}}$ for the adversary structure $\mathcal{Z}_\mathsf{C}^*$, for $Z \in \mathcal{Z}_{\mathsf{GOD}}$ s.t. $Z \in \mathcal{Z}_\mathsf{C}^*$, assuming that all the commitments generated are made from an underlying commitment scheme that is computationally hiding for a PPT distinguisher $\mathsf{D}_{\mathsf{Com}_{\mathsf{SB}}}$, then the hybrid distributions H_0 and H_1 are also computationally indistinguishable.

$$\mathsf{H}_0 \stackrel{\mathrm{c}}{\equiv} \mathsf{H}_1$$

Proof. Let $\overline{Z} \subseteq \mathcal{P}$ denote the set of honest parties in this protocol execution and let $s = |\overline{Z}|$ be the size of this set. The proof of this claim follows from considering the following set of $s^2 + 1$ intermediate hybrid distributions:

- Hybrid H'_0 : this distribution is the same as hybrid H_0, the simulated distribution and contains all the inputs of the honest parties used in the 'Commit Phase' as sampled by the simulator.
- $\forall j \in [s^2], \mathsf{H}'_j$: in this distribution, on behalf of the honest parties, in the commitment vectors within the VSS shares, the first j statistically binding (computationally hiding) commitments corresponding to the shares where honest parties are recipients are created using values as in the 'Commit Phase' in the real protocol execution. The rest, including all commitments to shares where corrupt parties are recipients, are as in the simulation.
- The last of these hybrids H'_{s^2} is distributed the same as H_1 as described above with all the inputs of the honest parties used in the 'Commit Phase' being that in the real protocol execution.

It now remains to argue that adjacent hybrid distributions as described above are computationally indistinguishable. For each $j \in [s^2]$, note that the only difference between the hybrid distributions H'_j and H'_{j-1} is that in the 'Commit Phase', the input for the j^{th} computationally hiding commitment is different. In both cases this is a commitment of a share in a $\mathcal{Z}_\mathsf{C}^*$-secret-sharing of a message and this value committed to does not exist in the view of the adversary. First, note that due to the access structure of the secret sharing, it holds that for any honest party $P_{j'}$, the set $\{s_k^{j'}\}_{k \in Z}$ does not contain enough shares to reconstruct the secret. Due to the underlying secret-sharing scheme, the reconstructed value

is completely determined by shares that the adversary does not hold and the adversary's own shares reveals no information about the secret. In H'_{j-1}, this is a share that allows reconstruction to a value chosen by the simulator, while in H'_j this allows reconstruction to the value in the real execution.

Now, if there exists a PPT distinguisher D that can distinguish the distributions H'_j and H'_{j-1} with non-negligible advantage ϵ, then we can show that this can be used in a black-box way to create a PPT machine $\mathsf{D}_{\mathsf{Com}_{\mathsf{SB}}}$ that can violate the computational hiding property of this underlying commitment scheme. $\mathsf{D}_{\mathsf{Com}_{\mathsf{SB}}}$ would operate in the computational hiding security game as follows:

- First, $\mathsf{D}_{\mathsf{Com}_{\mathsf{SB}}}$ uses the real inputs and randomness of all the parties and creates the view of the adversary up to the $j-1^{th}$ statistically binding commitment in the 'Commit Phase' exactly as in the real distribution.
- For the j^{th} share to be committed to, it samples a simulated message and secret-shares it under the constraint that the shares that the adversary receives are the same as in the real distribution. Then this j^{th} share of the simulated value s_0 and the real share s_1 are given to the challenger $\mathcal{C}$.
- $\mathcal{C}$ samples a bit b uniformly at random and creates a commitment c to s_b using the computationally hiding commitment scheme $\mathsf{Com}_{\mathsf{SB}}$. This commitment c is returned to $\mathsf{D}_{\mathsf{Com}_{\mathsf{SB}}}$.
- $\mathsf{D}_{\mathsf{Com}_{\mathsf{SB}}}$ uses this and completes the view of the adversary as generated by the simulator. Note that if $b = 0$, this view would be distributed as in H'_j and this would be H'_{j-1} otherwise.
- $\mathsf{D}_{\mathsf{Com}_{\mathsf{SB}}}$ passes this view onto D and outputs whatever it outputs.

Note that $\mathsf{D}_{\mathsf{Com}_{\mathsf{SB}}}$ here has the same advantage as D which is non-negligible. However, since the commitment schemes are computationally hiding, no such $\mathsf{D}_{\mathsf{Com}_{\mathsf{SB}}}$ can exist, so no such D can exist.

In the above, we have argued that we can switch from simulated to real randomness of the honest parties. Given this, switching from simulated to real inputs in the creation of Λ_i induces an identical distribution due to the one-time-pad effect of the randomness on the input. Therefore, all the adjacent hybrids H'_j and H'_{j-1} are computationally indistinguishable.

Claim. Let the protocol Π_i^* be maliciously fall-back secure tolerating $\mathcal{Z}_{\mathsf{S}}^{i'}$. Then, for all 'Protocol Emulation' executions $j \in Z_j^*$, assuming $\left\{\hat{\mathbf{View}}_{\Pi_i^*}^{Z-Z_j^*} \leftarrow \langle \mathsf{Sim}_{\mathsf{C}}(1^\kappa, \{x_k, r_k\}_{P_k \in Z_j^*}), \mathcal{A}\rangle\right\}_{r\in\vec{r}}$ is distributed computationally indistinguishably to the distribution of the real view $\left\{\mathbf{View}_{\Pi_i^*}^{Z-Z_j^*}\right\}_{r\in\vec{r}}$, then distributions H_1 and H_2 are also computationally indistinguishable. Hence,

$$\mathsf{H}_1 \stackrel{\mathrm{c}}{\equiv} \mathsf{H}_2$$

Proof. Letting $Z' \subseteq Z$ be the set of parties that aborted the computation and $s = |Z'|$ be the size of this set, the proof of the above claim follows from considering the following set of hybrids:

- Hybrid H'_0 : this distribution is the same as that of hybrid H_1 where the view in the 'Commit Phase' is identical to that in the real distribution but the complete protocol emulation is distributed as in the simulation using Sim_C.
- $\forall j \in [s], \mathsf{H}'_j$: in this distribution, up to the j^{th} protocol emulation, the view is generated exactly as in the real execution. Beyond this, all the protocol emulation views are generated as in the simulation using Sim_C.
- The last of these hybrids H'_s is distributed the same as H_2 which is the real distribution.

It remains to show that for each $j \in [s]$, the adjacent hybrid distributions H'_j and H'_{j-1} are computationally indistinguishable. Note that each such set of adjacent hybrids only differ in the view of the j^{th} execution of the 'Protocol Emulation' phase. In hybrid H'_j, this is as in the real protocol execution, while in H'_{j-1} this is a simulated view generated using an interaction with Sim_C. For the protocol Π^*_i, the real view for the malicious PPT adversary corrupting any subset of parties and that generated using Sim_C are computationally indistinguishable.

If there exists a PPT distinguisher D that can distinguish the distributions H_j and H_{j-1} with non-negligible advantage ϵ then we can show that this can be used in a black-box way to create a PPT machine D^* that can distinguish between a real and simulated view of $\mathcal{A}$ in Π^*_i. D^* would work by creating the view up to that of the $j-1^{th}$ 'Protocol Emulation' phase as in the real execution. Then it would give all the inputs and randomness to the challenger and receive a challenge view for the j^{th} execution, either real or simulated. D^* would use this and complete the view of the protocol Π_i as would be produced by the simulation. This is then given to D and then D^* outputs whatever D outputs. Note that D^* here has the same advantage as D which is non-negligible. However, since Π^*_i is maliciously fall-back secure, no such D^* can exist, so no such D can exist. Therefore, all the adjacent hybrids H_j and H_{j-1} are computationally indistinguishable.

Security with Abort for Unbounded Adversaries. The protocol Π_i satisfies security with abort in the presence of a malicious unbounded adversary $\mathcal{A}$ tolerating $\mathcal{Z}^i_\mathsf{S}$ where $\mathcal{Z}^i_\mathsf{S}$ is the adversary structure containing all sets in $\mathcal{Z}^*_\mathsf{C}$ that do not have party P_i as a member. Intuitively, this holds because the VSS scheme tolerates $\mathcal{Z}^*_\mathsf{C}$. Let $Z \in \mathcal{Z}^i_\mathsf{S}$ be the set of corrupted parties. Note that Z is also contained within $\mathcal{Z}^{i'}_\mathsf{S}$, that is the adversary structure for which the base protocol Π^*_i is fall-back secure. Let $\mathsf{Sim}_{\mathsf{S},\perp}$ be the PPT simulator that interacts with the malicious adversary and simulates its view in the ideal world for this corruption case. This simulator would operate exactly as $\mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}$ described previously, but with the following difference:

- In the 'Abort Handling' phase, along with the functions of $\mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}$, if $\mathcal{A}$ causes the protocol emulation to abort on behalf of a corrupted party P_j where now the set of aborted parties $Z^* \notin \mathcal{Z}^{i,\mathsf{GOD}}_\mathsf{S}$, then send $\perp$ to the ideal functionality $\mathcal{F}$ on behalf of the adversary and terminate the protocol execution.

The view of the protocol produced by $\mathsf{Sim}_{\mathsf{S},\perp}$ would be the same as that produced by $\mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}$, but with the last protocol emulation possibly aborted, as opposed to running to completion. It can be argued as being statistically close to the adversary's view in the real execution of the protocol following the same set of hybrids and arguments as the unbounded GOD case.

Security with Abort for PPT Adversaries. The protocol Π_i satisfies security with abort in the presence of a malicious PPT adversary $\mathcal{A}$ tolerating $\mathcal{Z}^*_{\mathsf{C}}$. Intuitively, this holds because the VSS scheme tolerates $\mathcal{Z}^*_{\mathsf{C}}$. Note that this includes sets that may have P_i as its member. Let $Z \in \mathcal{Z}^*_{\mathsf{C}}$ be the set of corrupted parties. Let $\mathsf{Sim}_{\mathsf{C},\perp}$ be the PPT simulator that interacts with the malicious adversary and simulates its view in the ideal world for this corruption case. This simulator would operate exactly as $\mathsf{Sim}_{\mathsf{C},\mathsf{GOD}}$ described previously, but with the following difference:

- In the 'Abort Handling' phase, along with the functions of $\mathsf{Sim}_{\mathsf{C},\mathsf{GOD}}$, if $\mathcal{A}$ causes the protocol emulation to abort on behalf of a corrupted party P_j where now the set of aborted parties $Z^* \notin \mathcal{Z}_{\mathsf{GOD}}$, then send $\perp$ to the ideal functionality $\mathcal{F}$ on behalf of the adversary and terminate the protocol execution.

The view of the protocol produced by $\mathsf{Sim}_{\mathsf{C},\perp}$ would be the same as that produced by $\mathsf{Sim}_{\mathsf{C},\mathsf{GOD}}$, but with the last protocol emulation possibly aborted, as opposed to running to completion. It can be argued as being computationally indistinguishable from the adversary's view in the real execution of the protocol following the same set of hybrids and reductions as the PPT GOD case.

Unbounded Adversary corrupting $\mathcal{Z}_{\mathsf{GOD}}$. We consider here only the special case where an unbounded malicious adversary $\mathcal{A}$ corrupts a subset of parties $Z \in \mathcal{Z}_{\mathsf{GOD}}$, specifically one that includes the party P_i. Note that for this case, no privacy of inputs of the honest parties can be guaranteed. However, the VSS access structure in the protocol is such that all the parties in the set $\overline{Z} = \mathcal{P} - Z$ can always reconstruct the secret. Therefore, for any corrupt party P_j that aborts, its inputs x_j and randomness r_j can always be reconstructed by the shares with the honest parties. So it follows that for every corrupt party that aborts, the protocol emulation can be re-started with these values and the protocol will finally terminate with the honest parties receiving their desired outputs of the functionality. This will hold regardless of the actions of the malicious adversary and the fact that the adversary learns the honest parties' inputs.

C Proof of Theorem 3

Proof. For $Z \subset \mathcal{P}$, let $\mathcal{Z}_{\mathsf{GOD}} = \{Z\}$ and $\mathcal{Z}^*_{\mathsf{C}}$ be any adversary structure. For the protocol Π_i in Figure 5, the proof for Theorem 3 follows from analysing the following cases separately:

1. We need to show that security with Guaranteed Output Delivery (GOD) holds in the presence of an unbounded adversary when it corrupts any set of parties in the adversary structure $\mathcal{Z}_{\mathsf{S}}^{i,\mathsf{GOD}}$, where $\mathcal{Z}_{\mathsf{S}}^{i,\mathsf{GOD}}$ contains $Z - \{P_i\}$.
2. We need to show that GOD is also satisfied for a PPT adversary corrupting any subset of Z, including P_i.
3. We show that security with abort holds for an unbounded adversary when it corrupts any set of parties in $\mathcal{Z}_{\mathsf{S}}^{i}$, an adversary structure that contains all sets in $\mathcal{Z}_{\mathsf{C}}^{*}$ without P_i as a member.
4. We also show that security with abort holds for a PPT adversary when it corrupts any set of parties in $\mathcal{Z}_{\mathsf{C}}^{*}$, even all the sets containing P_i.
5. Finally, we show that when an unbounded adversary corrupts parties in $\mathcal{Z}_{\mathsf{GOD}}$, even including party P_i, the protocol will still execute until completion with all honest parties receiving the correct outputs of the functionality, although the privacy of the honest parties' inputs is completely compromised.

In protocol Π_i in Figure 5, let Π_i^* be the base protocol that computes f with maliciously fall-back security with identifiable abort tolerating $\mathcal{Z}_{\mathsf{S}}^{i'}$, where $\mathcal{Z}_{\mathsf{S}}^{i'}$ is the adversary structure containing all subsets of parties not including P_i. For this protocol, let $\mathsf{Sim}_{\mathsf{S}}$ be the PPT machine that simulates the view produced by an interaction with an unbounded malicious adversary corrupting any set in $\mathcal{Z}_{\mathsf{S}}^{i'}$. Similarly, let $\mathsf{Sim}_{\mathsf{C}}$ be the PPT simulator for the view of a PPT malicious adversary corrupting any set of parties, even including P_i.

GOD for Unbounded Adversaries. The adversary structure $\mathcal{Z}_{\mathsf{S}}^{i,\mathsf{GOD}}$ contains $Z - \{P_i\}$. Note that this structure is a subset of $\mathcal{Z}_{\mathsf{S}}^{i'}$. We need to show that security with GOD holds in the presence of a malicious unbounded adversary $\mathcal{A}$ corrupting a set $Z' \in \mathcal{Z}_{\mathsf{S}}^{i,\mathsf{GOD}}$. For this, consider the following PPT simulator $\mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}$ that interacts with $\mathcal{A}$ in the ideal world for Guaranteed Output Delivery. It makes black-box calls to the simulator $\mathsf{Sim}_{\mathsf{S}}$ of the underlying fall-back secure protocol Π_i^*. $\mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}$ gets as input the statistical security parameter 1^s and would operate as follows:

– **Protocol Emulation:** $\mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}$ invokes the simulator $\mathsf{Sim}_{\mathsf{S}}$ for protocol Π_i^* with input 1^s and interacts with the adversary $\mathcal{A}$.
  - In this interaction, $\mathsf{Sim}_{\mathsf{S}}$ extracts the adversary's inputs to Π_i^* to give to the ideal functionality computing f.
  - The simulator $\mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}$, emulating this ideal functionality, accepts these extracted inputs from $\mathsf{Sim}_{\mathsf{S}}$.
  - If this protocol Π_i^* executes to completion without aborting, then $\mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}$ gives these inputs to the ideal functionality $\mathcal{F}$ and receives $f(x)$ from it. It then, emulating the functionality, passes this to $\mathsf{Sim}_{\mathsf{S}}$ for it to complete the protocol simulation in the last round.
  - For the j^{th} execution of the 'Protocol Emulation' phase, let $Z_j^* \subseteq Z$ be the set of corrupt parties that have already aborted before this phase has started. Then $\mathsf{Sim}_{\mathsf{S}}$ outputs the following from this interaction:

$$\hat{\mathbf{View}}_{\Pi_i^*}^{Z-Z_j^*} \leftarrow \langle \mathsf{Sim}_{\mathsf{S}}(1^s, \{x_k, r_k\}_{P_k \in Z_j^*}), \mathcal{A} \rangle$$

– **Abort Handling:** If $\mathcal{A}$ causes the protocol to abort on behalf of any corrupted party P_j, $\mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}$, receiving j from $\mathsf{Sim}_{\mathsf{S}}$, assumes default inputs and randomness for P_j. Then 'Protocol Emulation' is re-started with each party locally computing the messages of the aborted party with these inputs and randomness.

Note that the protocol will finally securely terminate with the honest parties receiving their outputs. Therefore, Guaranteed Output Delivery is satisfied.

Let Z^* be a set of indices where the j^{th} index denotes the j^{th} protocol emulation. The joint view produced by $\mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}$ with the honest parties' outputs is distributed as,

$$\left\{ \left\{ \hat{\mathbf{View}}_{\Pi_i^*}^{Z-Z_j^*}, \mathbf{View}_{\Pi_i^*, Z_j^*}(\{r_k, x_k\}_{P_k \in Z_j^*}) \right\}_{j \in Z^*}, f(x) \right\}_{\kappa \in \mathbb{N}, \vec{r} \in \mathcal{R}^n}$$

It remains to argue that the above view is statistically close to the real view in the protocol Π_i.

Claim. Let the protocol Π_i^* is maliciously fall-back secure tolerating $\mathcal{Z}_{\mathsf{S}}^{i'}$. Then, for all 'Protocol Emulation' executions $j \in Z_j^*$, assuming $\left\{ \hat{\mathbf{View}}_{\Pi_i^*}^{Z-Z_j^*} \leftarrow \langle \mathsf{Sim}_{\mathsf{S}}(1^s, \{x_k, r_k\}_{P_k \in Z_j^*}), \mathcal{A} \rangle \right\}_{r \in \vec{r}}$ is distributed statistically close to the distribution of the real view $\left\{ \mathbf{View}_{\Pi_i^*}^{Z-Z_j^*} \right\}_{r \in \vec{r}}$ with statistical distance ϵ, negligible in s, then the statistical distance between the distributions of the real and simulated view is upper bounded by $|Z^*|\epsilon$.

Proof. Letting Z^* be the set of indices denoting the executions of the 'Protocol Emulation' phase and $s = |Z^*|$ be the size of this set, the proof of the above claim follows from considering the following set of hybrids:

– Hybrid H'_0 : this distribution where the complete protocol emulation is distributed as in the simulation $\mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}$.
– $\forall j \in Z^*, \mathsf{H}'_j$: in this distribution, up to the j^{th} protocol emulation, the view is generated exactly as in the real execution. Beyond this, each k^{th} protocol emulation view is generated as in the simulation:

$$\hat{\mathbf{View}}_{\Pi_i^*}^{Z-Z_k^*} \leftarrow \langle \mathsf{Sim}_{\mathsf{S}}(1^s, \{x_{k'}, r_{k'}\}_{P_{k'} \in Z_k^*}), \mathcal{A} \rangle$$

– The last of these hybrids H'_s is distributed the same as the real view.

It remains to show that for each $j \in Z^*$, the adjacent hybrid distributions H'_j and H'_{j-1} are statistically close. Note that each such set of adjacent hybrids only differ in the view of the j^{th} execution of the 'Protocol Emulation' phase. In hybrid H'_j, this is as in the real protocol execution, while in H'_{j-1} this is a simulated view generated using an interaction with $\mathsf{Sim}_{\mathsf{S}}$. For the protocol Π_i^*, the real view for the adversary and that generated using $\mathsf{Sim}_{\mathsf{S}}$ are statistically close with distance ϵ. Since this is the only difference between these adjacent hybrids, it follows that the complete hybrids are also statistically close. Hence, the real and simulated distributions are also at most $s\epsilon$ apart.

GOD FOR PPT ADVERSARIES. Let $\mathcal{A}$ be a malicious PPT adversary corrupting a set $Z' \in \mathcal{Z}_{\mathsf{GOD}}$. Consider a PPT simulator $\mathsf{Sim}_{\mathsf{C},\mathsf{GOD}}$ that simulates the view of $\mathcal{A}$ in the ideal world for GOD. Note that if $Z' \in \mathcal{Z}_{\mathsf{S}}^{i,\mathsf{GOD}}$, then $\mathsf{Sim}_{\mathsf{C},\mathsf{GOD}}$ can just interact like $\mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}$ as above. However, if $P_i \in Z$, then it would need to make black-box calls to the simulator $\mathsf{Sim}_{\mathsf{C}}$ of the underlying fall-back secure protocol Π_i^*. This $\mathsf{Sim}_{\mathsf{C},\mathsf{GOD}}$ gets the computational security parameter 1^κ as input and would operate as follows:

- **Protocol Emulation:** $\mathsf{Sim}_{\mathsf{C},\mathsf{GOD}}$ invokes the simulator $\mathsf{Sim}_{\mathsf{C}}$ for protocol Π_i^* with input 1^κ and interacts with the adversary $\mathcal{A}$.
  - In this interaction, $\mathsf{Sim}_{\mathsf{C}}$ extracts the adversary's inputs to Π_i^* to give to the ideal functionality computing f.
  - The simulator $\mathsf{Sim}_{\mathsf{C},\mathsf{GOD}}$, emulating this ideal functionality, accepts these extracted inputs from $\mathsf{Sim}_{\mathsf{C}}$.
  - If this protocol Π_i^* executes to completion without aborting, then $\mathsf{Sim}_{\mathsf{C},\mathsf{GOD}}$ gives these inputs to the ideal functionality $\mathcal{F}$ and receives $f(x)$ from it. It then, emulating the functionality, passes this to $\mathsf{Sim}_{\mathsf{C}}$ for it to complete the protocol simulation.
  - For the j^{th} execution of the 'Protocol Emulation' phase, let $Z_j^* \subseteq Z$ be the set of corrupt parties that have already aborted before this phase has started. Then $\mathsf{Sim}_{\mathsf{C}}$ outputs the following from this interaction:

$$\hat{\mathbf{View}}_{\Pi_i^*}^{Z-Z_j^*} \leftarrow \langle \mathsf{Sim}_{\mathsf{C}}(1^\kappa, \{x_k, r_k\}_{P_k \in Z_j^*}), \mathcal{A} \rangle$$

- **Abort Handling:** If $\mathcal{A}$ causes the protocol to abort on behalf of any corrupted party P_j, $\mathsf{Sim}_{\mathsf{C},\mathsf{GOD}}$, receiving j from $\mathsf{Sim}_{\mathsf{C}}$, assumes default input and randomness for this party. Then 'Protocol Emulation' is re-started with each party locally computing the messages of the aborted party with these inputs and randomness.

Note that for every corrupt party that aborts, the protocol emulation can be re-started with these values and the protocol will finally securely terminate with the honest parties receiving their desired outputs. Therefore, Guaranteed Output Delivery is satisfied.

Let $Z^* \subseteq Z$ be the set of indices where the j^{th} index denotes the j^{th} protocol emulation. The joint view produced by $\mathsf{Sim}_{\mathsf{C},\mathsf{GOD}}$ with the honest parties' outputs is distributed as,

$$\left\{ \left\{ \hat{\mathbf{View}}_{\Pi_i^*}^{Z-Z_j^*}, \mathbf{View}_{\Pi_i^*, Z_j^*}(\{r_k, x_k\}_{P_k \in Z_j^*}) \right\}_{j \in Z^*}, f(x) \right\}_{\kappa \in \mathbb{N}, \vec{r} \in \mathcal{R}^n}$$

It remains to argue that the above view is computationally indistinguishable from the real view in the protocol Π_i.

Claim. Let the protocol Π_i^* be maliciously fall-back secure tolerating $\mathcal{Z}_{\mathsf{C}}^*$. Then, for all 'Protocol Emulation' executions $j \in Z_j^*$, assuming $\left\{ \hat{\mathbf{View}}_{\Pi_i^*}^{Z-Z_j^*} \leftarrow \langle \mathsf{Sim}_{\mathsf{C}}(1^\kappa, \{x_k, r_k\}_{P_k \in Z_j^*}), \mathcal{A} \rangle \right\}_{r \in \vec{r}}$ is distributed computationally indistinguishably to

the distribution of the real view $\left\{\mathbf{View}_{\Pi_i^*}^{Z-Z_j^*}\right\}_{r\in\vec{r}}$, then distributions of the real and simulated view are also computationally indistinguishable.

Proof. Letting $Z' \subseteq Z$ be the set of parties that aborted the computation and $s = |Z'|$ be the size of this set, the proof of the above claim follows from considering the following set of hybrids:

- Hybrid H'_0 : this distribution is the same as that of the simulation using Sim_C.
- $\forall j \in [s], \mathsf{H}'_j$: in this distribution, up to the j^{th} protocol emulation, the view is generated exactly as in the real execution. Beyond this, all the protocol emulation views are generated as in the simulation using Sim_C.
- The last of these hybrids H'_s is distributed the same as the real distribution.

It remains to show that for each $j \in [s]$, the adjacent hybrid distributions H'_j and H'_{j-1} are computationally indistinguishable. Note that each such set of adjacent hybrids only differ in the view of the j^{th} execution of the 'Protocol Emulation' phase. In hybrid H'_j, this is as in the real protocol execution, while in H'_{j-1} this is a simulated view generated using an interaction with Sim_C. For the protocol Π_i^*, the real view for the malicious PPT adversary corrupting any subset of parties and that generated using Sim_C are computationally indistinguishable.

If there exists a PPT distinguisher D that can distinguish the distributions H'_j and H'_{j-1} with non-negligible advantage ϵ then we can show that this can be used in a black-box way to create a PPT machine D^* that can distinguish between a real and simulated view of $\mathcal{A}$ in Π_i^*. D^* would work by creating the view up to that of the $j-1^{th}$ 'Protocol Emulation' phase as in the real execution. Then it would give all the inputs and randomness to the challenger and receive a challenge view for the j^{th} execution, either real or simulated. D^* would use this and complete the view of the protocol Π_i as would be produced by the simulation. This is then given to D and then D^* outputs whatever D outputs. Note that D^* here has the same advantage as D which is non-negligible. However, since Π_i^* is maliciously fall-back secure, no such D^* can exist, so no such D can exist. Therefore, all the adjacent hybrids H'_j and H'_{j-1} are computationally indistinguishable.

Security with Abort for Unbounded Adversaries. The protocol Π_i satisfies security with abort in the presence of a malicious unbounded adversary $\mathcal{A}$ tolerating $\mathcal{Z}_\mathsf{S}^i$ where $\mathcal{Z}_\mathsf{S}^i$ is the adversary structure containing all sets in $\mathcal{Z}_\mathsf{C}^*$ that do not have party P_i as a member. Let $Z \in \mathcal{Z}_\mathsf{S}^i$ be the set of corrupted parties. Note that Z is also contained within $\mathcal{Z}_\mathsf{S}^{i'}$, that is the adversary structure for which the base protocol Π_i^* is fall-back secure. Let $\mathsf{Sim}_{\mathsf{S},\perp}$ be the PPT simulator that interacts with the malicious adversary and simulates its view in the ideal world for this corruption case. This simulator would operate exactly as $\mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}$ described previously, but with the following difference:

- In the 'Abort Handling' phase, along with the functions of $\mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}$, if $\mathcal{A}$ causes the protocol emulation to abort on behalf of a corrupted party P_j

where now the set of aborted parties $Z^* \notin \mathcal{Z}_{\mathsf{S}}^{i,\mathsf{GOD}}$, then send $\perp$ to the ideal functionality $\mathcal{F}$ on behalf of the adversary and terminate the protocol execution.

The view of the protocol produced by $\mathsf{Sim}_{\mathsf{S},\perp}$ would be the same as that produced by $\mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}$, but with the last protocol emulation possibly aborted, as opposed to running to completion. It can be argued as being statistically close to the adversary's view in the real execution of the protocol following the same set of hybrids and arguments as the unbounded GOD case.

Security with Abort for PPT Adversaries. The protocol Π_i satisfies security with abort in the presence of a malicious PPT adversary $\mathcal{A}$ tolerating $\mathcal{Z}_{\mathsf{C}}^*$. Note that this includes sets that may have P_i as its member. Let $Z \in \mathcal{Z}_{\mathsf{C}}^*$ be the set of corrupted parties. Let $\mathsf{Sim}_{\mathsf{C},\perp}$ be the PPT simulator that interacts with the malicious adversary and simulates its view in the ideal world for this corruption case. This simulator would operate exactly as $\mathsf{Sim}_{\mathsf{C},\mathsf{GOD}}$ described previously, but with the following difference:

– In the 'Abort Handling' phase, along with the functions of $\mathsf{Sim}_{\mathsf{C},\mathsf{GOD}}$, if $\mathcal{A}$ causes the protocol emulation to abort on behalf of a corrupted party P_j where now the set of aborted parties $Z^* \notin \mathcal{Z}_{\mathsf{GOD}}$, then send $\perp$ to the ideal functionality $\mathcal{F}$ on behalf of the adversary and terminate the protocol.

The view of the protocol produced by $\mathsf{Sim}_{\mathsf{C},\perp}$ would be the same as that produced by $\mathsf{Sim}_{\mathsf{C},\mathsf{GOD}}$, but with the last protocol emulation possibly aborted, as opposed to running to completion. It can be argued as being computationally indistinguishable from the adversary's view in the real execution of the protocol following the same set of hybrids and reductions as the PPT GOD case.

Unbounded Adversary corrupting $\mathcal{Z}_{\mathsf{GOD}}$. We consider here only the special case where an unbounded malicious adversary $\mathcal{A}$ corrupts a subset of parties $Z \in \mathcal{Z}_{\mathsf{GOD}}$, specifically one that includes the party P_i. Note that for this case, no privacy of inputs of the honest parties can be guaranteed. However, for any corrupt party P_j that aborts, the protocol emulation can always be re-started with default inputs and randomness and the protocol will finally terminate with the honest parties receiving their desired outputs of the functionality. This will hold regardless of the actions of the malicious adversary and the fact that the adversary learns the honest parties' inputs.

D Proof of Theorem 4

Proof. Let $\mathcal{Z}_{\mathsf{GOD}} \in \mathcal{Q}^2$ and $\mathcal{Z}_{\mathsf{C}}^*$ be the maximal adversary structure such that both of these satisfy Equation 1. Let $\mathcal{Z}_{\mathsf{fair}} \in \mathcal{Q}^2$ be the adversary structure for which fairness is required and it holds that $\mathcal{Z}_{\mathsf{fair}}^* = \mathcal{Z}_{\mathsf{GOD}} \cup \mathcal{Z}_{\mathsf{fair}} \in \mathcal{Q}^2$. Let $\mathcal{Z}_{\mathsf{S}} \in \mathcal{Q}^2$ be the adversary structure for which security, at least with abort, in the presence of an unbounded adversary is required. Additionally, let it hold that

$\mathcal{Z}^*_{\mathsf{fair}}, \mathcal{Z}_{\mathsf{S}} \subset \mathcal{Z}^*_{\mathsf{C}}$. Let Π be a protocol compiled as indicated in Section 5.3. For this protocol, the proof for Theorem 4 follows from analysing the following cases separately:

1. We need to show that security with Guaranteed Output Delivery (GOD) holds in the presence of an unbounded adversary when it corrupts any set of parties in the adversary structure $\mathcal{Z}_{\mathsf{S}} \cap \mathcal{Z}_{\mathsf{GOD}}$.
2. We need to show that GOD is also satisfied for a PPT adversary corrupting any set of parties in $\mathcal{Z}_{\mathsf{GOD}}$, even for the sets not included above case.
3. We need to show that security with fairness holds in the presence of an unbounded adversary when it corrupts any set of parties in the adversary structure $\mathcal{Z}_{\mathsf{S}} \cap \mathcal{Z}_{\mathsf{fair}}$.
4. We need to show that fairness is also satisfied for a PPT adversary corrupting any set of parties in $\mathcal{Z}_{\mathsf{fair}}$, even for the sets not included above case.
5. We show that security with abort holds for an unbounded adversary when it corrupts any set of parties in $\mathcal{Z}_{\mathsf{S}}$, in particular, for those not in $\mathcal{Z}^*_{\mathsf{fair}}$.
6. We also show that security with abort holds for a PPT adversary when it corrupts any set of parties in $\mathcal{Z}^*_{\mathsf{C}}$, even for those not included in $\mathcal{Z}^*_{\mathsf{fair}}$.

In order to examine the security of Π in the plain model, it becomes necessary to describe the following properties of its building blocks:

– For the semi-honest secure virtual n-party protocol $\Pi_{\mathcal{V}}$, let $\mathsf{Sim}^{\mathcal{V}}_{\mathsf{S}}$ be the simulator for an adversary $\mathcal{A}$ corrupting a set of parties $Z \in \mathcal{Z}_{\mathsf{S}}$.

$$\hat{\mathbf{View}}^{Z}_{\mathcal{V}} \leftarrow \mathsf{Sim}^{\mathcal{V}}_{\mathsf{S}}(1^s, \{x_i\}_{V_i \in Z}, f(\vec{x}))$$

$$\left\{\hat{\mathbf{View}}^{Z}_{\mathcal{V}}\right\}_{\kappa \in \mathbb{N}, \vec{r} \in \mathcal{R}^n} \overset{\mathrm{s}}{\equiv} \left\{\mathbf{View}^{Z}_{\mathcal{V}}\right\}_{\kappa \in \mathbb{N}, \vec{r} \in \mathcal{R}^n}$$

Letting r be the number of rounds in this protocol, recall that the final protocol Π consists of nr executions of protocols computing next-message-functions of this virtual protocol. Much like how $\Pi_{\mathcal{V}}$ can be written as the set of functions $\{\mathsf{NxtMsg}^j_i\}_{i \in [n], j \in [r]}$, the simulator $\mathsf{Sim}^{\mathcal{V}}_{\mathsf{S}}$ can be written as,

$$\mathsf{Sim}^{\mathcal{V}}_{\mathsf{S}} = \{\mathsf{NxtMsg}^{j*}_i\}_{i \in [n], j \in [r]}$$

Here, for each corrupted party $V_i \in Z$, the functions $\{\mathsf{NxtMsg}^{j*}_i\}_{j \in [r]} = \{\mathsf{NxtMsg}^{j}_i\}_{j \in [r]}$ are as in the real protocol $\Pi_{\mathcal{V}}$. However, for each honest party $V_i \notin Z$, for each round $j \in [r]$, the next-message function has the same structure as the real function but produces simulated messages that do not depend on the party's input:

$$\{m^{j'}_{i \to k}\}_{k \neq i \in [n]} \leftarrow \mathsf{NxtMsg}^{j*}_i(\hat{\mathbf{View}}^{j-1}_{\mathcal{V},i})$$

Here $\hat{\mathbf{View}}^{j-1}_{\mathcal{V},i}$ is the simulated view with all the messages that the corrupt party sends to V_i up to round $j-1$. The set $\{m^{j'}_{i \to k}\}_{k \neq i \in [n]}$ is the set of simulated messages output on behalf of V_i to the other parties in round j.

– The real protocol Π is composed of the set of protocols $\{\Pi_{\mathsf{NxtMsg}_i^j}\}_{i\in[n],j\in[r]}$ corresponding to the $\Pi_{\mathcal{V}}$ in question. For each $i \in [n]$, protocol Π_i in Figure 4 is used to execute all next-message-function sub-protocol $\Pi_{\mathsf{NxtMsg}_i^j}$ of virtual party V_i, Due to Theorem 2, we know that the following exist:
  - Let $\mathsf{Sim}_{\mathsf{S,GOD}}^{\mathsf{NxtMsg}_i^j}$ be the PPT simulator for the view of an unbounded malicious adversary $\mathcal{A}$ corrupting any set $Z \in \mathcal{Z}_{\mathsf{S}}^{i,\mathsf{GOD}}$. This interacts with the adversary and produces a view that is statistically close to the real view of the protocol execution:

$$\hat{\mathbf{View}}_{\mathsf{NxtMsg}_i^j}^{Z,\mathsf{GOD}} \leftarrow \langle \mathsf{Sim}_{\mathsf{S,GOD}}^{\mathsf{NxtMsg}_i^j}(1^s), \mathcal{A}\rangle$$

$$\left\{\hat{\mathbf{View}}_{\mathsf{NxtMsg}_i^j}^{Z,\mathsf{GOD}}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n} \stackrel{\mathrm{s}}{\equiv} \left\{\mathbf{View}_{\mathsf{NxtMsg}_i^j}^{Z,\mathsf{GOD}}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n}$$

    Similarly to the above, if this simulator is given the outputs based on $\{[m_{i\to i'}^{j'}]_k\}_{i'\neq i\in[n]}\}_{k\in[n]}$, as output by the simulator of the virtual protocol $\mathsf{NxtMsg}_i^{j*} \in \mathsf{Sim}_{\mathsf{S}}^{\mathcal{V}}$, the view simulated is a view that is still statistically close to that in the real execution of $\Pi_{\mathsf{NxtMsg}_i^j}$:

$$\hat{\mathbf{View}}_{\mathsf{NxtMsg}_i^{j*}}^{Z,\mathsf{GOD}} \leftarrow \langle \mathsf{Sim}_{\mathsf{S,GOD}}^{\mathsf{NxtMsg}_i^j}(1^s), \mathcal{A}\rangle$$

$$\left\{\hat{\mathbf{View}}_{\mathsf{NxtMsg}_i^{j*}}^{Z,\mathsf{GOD}}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n} \stackrel{\mathrm{s}}{\equiv} \left\{\mathbf{View}_{\mathsf{NxtMsg}_i^j}^{Z,\mathsf{GOD}}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n}$$

  - Let $\mathsf{Sim}_{\mathsf{C,GOD}}^{\mathsf{NxtMsg}_i^j}$ be the PPT simulator for the view of a PPT malicious adversary $\mathcal{A}$ corrupting any set of parties in $\mathcal{Z}_{\mathsf{GOD}}$, even including P_i. This interacts with the adversary and produces a view that is computationally indistinguishable from the real view of the protocol execution:

$$\hat{\mathbf{View}}_{\mathsf{NxtMsg}_i^j}^{Z,\mathsf{GOD}} \leftarrow \langle \mathsf{Sim}_{\mathsf{C,GOD}}^{\mathsf{NxtMsg}_i^j}(1^\kappa), \mathcal{A}\rangle$$

$$\left\{\hat{\mathbf{View}}_{\mathsf{NxtMsg}_i^j}^{Z,\mathsf{GOD}}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n} \stackrel{\mathrm{c}}{\equiv} \left\{\mathbf{View}_{\mathsf{NxtMsg}_i^j}^{Z,\mathsf{GOD}}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n}$$

  - Let $\mathsf{Sim}_{\mathsf{S},\perp}^{\mathsf{NxtMsg}_i^j}$ be the PPT simulator for the view of an unbounded malicious adversary $\mathcal{A}$ corrupting any set $Z \in \mathcal{Z}_{\mathsf{S}}^{i}$. This interacts with the adversary and produces a view that is statistically close to the real view of the protocol execution:

$$\hat{\mathbf{View}}_{\mathsf{NxtMsg}_i^j}^{Z,\perp} \leftarrow \langle \mathsf{Sim}_{\mathsf{S},\perp}^{\mathsf{NxtMsg}_i^j}(1^s), \mathcal{A}\rangle$$

$$\left\{\hat{\mathbf{View}}_{\mathsf{NxtMsg}_i^j}^{Z,\perp}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n} \stackrel{\mathrm{s}}{\equiv} \left\{\mathbf{View}_{\mathsf{NxtMsg}_i^j}^{Z,\perp}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n}$$

    Similarly to the above, if this simulator is given the outputs based on $\{[m_{i\to i'}^{j'}]_k\}_{i'\neq i\in[n]}\}_{k\in[n]}$, as output by the simulator of the virtual protocol $\mathsf{NxtMsg}_i^{j*} \in \mathsf{Sim}_{\mathsf{S}}^{\mathcal{V}}$, the view simulated is a view that is still statistically close to that in the real execution of $\Pi_{\mathsf{NxtMsg}_i^j}$:

$$\hat{\mathbf{View}}_{\mathsf{NxtMsg}_i^{j*}}^{Z,\perp} \leftarrow \langle \mathsf{Sim}_{\mathsf{S},\perp}^{\mathsf{NxtMsg}_i^j}(1^s), \mathcal{A}\rangle$$

$$\left\{\hat{\mathbf{View}}^{Z,\perp}_{\mathsf{NxtMsg}_i^{j*}}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n} \stackrel{\mathrm{s}}{\equiv} \left\{\mathbf{View}^{Z,\perp}_{\mathsf{NxtMsg}_i^{j}}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n}$$

- Let $\mathsf{Sim}^{\mathsf{NxtMsg}_i^j}_{\mathsf{C},\perp}$ be the PPT simulator for the view of a PPT malicious adversary $\mathcal{A}$ corrupting any set of parties $Z \in \mathcal{Z}^*_{\mathsf{C}}$, even including P_i. This interacts with the adversary and produces a view that is computationally indistinguishable from the real view of the protocol execution:

$$\hat{\mathbf{View}}^{Z,\perp}_{\mathsf{NxtMsg}_i^{j}} \leftarrow \langle \mathsf{Sim}^{\mathsf{NxtMsg}_i^j}_{\mathsf{C},\perp}(1^\kappa), \mathcal{A}\rangle$$

$$\left\{\hat{\mathbf{View}}^{Z,\perp}_{\mathsf{NxtMsg}_i^{j}}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n} \stackrel{\mathrm{c}}{\equiv} \left\{\mathbf{View}^{Z,\perp}_{\mathsf{NxtMsg}_i^{j}}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n}$$

Given the above building blocks and notation, we are now ready to argue security for our compiled protocol.

GOD FOR UNBOUNDED ADVERSARIES. Let $\mathcal{A}$ be a malicious unbounded adversary corrupting a set of parties $Z \in \mathcal{Z}_{\mathsf{S}} \cap \mathcal{Z}_{\mathsf{GOD}}$. Consider the following simulator $\mathsf{Sim}_{\mathsf{S,GOD}}$ that makes black-box calls to simulators in $\{\mathsf{Sim}^{\mathsf{NxtMsg}_i^j}_{\mathsf{S,GOD}}\}_{i\in[n],j\in[r]}$ and simulates the view of $\mathcal{A}$ in the ideal world for guaranteed output delivery:

1. $\mathsf{Sim}_{\mathsf{S,GOD}}$ has as input the security parameter 1^s.
2. In the 'Initialize Shared State' phase, on behalf of all the honest parties $P_k \in \mathcal{P} - Z$, it samples inputs x_k and randomness r_k. It then generates a $\mathcal{Z}^*_{\mathsf{C}}$-secret-sharing of $\hat{\mathbf{View}}^{0}_{\mathcal{V},k} = (x_k, r_k)$ and sends all the adversary's shares $\{[\hat{\mathbf{View}}^{0}_{\mathcal{V},k}]_{k'}\}_{V_{k'}\in Z}$ to $\mathcal{A}$.
3. $\mathsf{Sim}_{\mathsf{S,GOD}}$ also accepts the secret shares $\{[\hat{\mathbf{View}}^{0}_{\mathcal{V},k}]_{k'}\}_{V_{k'}\in\mathcal{V}-Z,V_k\in Z}$ of the initial view in the virtual protocol of all corrupt parties.
4. For the simulator of the virtual protocol $\mathsf{Sim}^{\mathcal{V}}_{\mathsf{S}}$, as stated before, parse this as a set of next-message functions: one for each party, for each round.

$$\mathsf{Sim}^{\mathcal{V}}_{\mathsf{S}} = \{\mathsf{NxtMsg}_i^{j*}\}_{i\in[n],j\in[r]}$$

5. In the 'Compute Virtual Protocol' phase, for each next-message-functionality in $\{\mathcal{F}_{\mathsf{NxtMsg}_i^{0*}}\}_{V_i\in\mathcal{V}}$ in the first round of the virtual protocol, $\mathsf{Sim}_{\mathsf{S,GOD}}$ samples randomness and creates, on behalf of each honest party, the commitments that need to be input to these functionalities. It gives the adversary the corresponding randomness that it would need to input, and receives from $\mathcal{A}$ the randomness of all the commitments of the parties that it is corrupting.
6. For each virtual party $V_i \in \mathcal{V}$ and each round $j \in [r]$ of the virtual protocol, for the next-message protocol $\Pi_{\mathsf{NxtMsg}_i^j}$ as in Figure 4, $\mathsf{Sim}_{\mathsf{S,GOD}}$ makes a black-box call to the PPT simulator $\mathsf{Sim}^{\mathsf{NxtMsg}_i^j}_{\mathsf{S,GOD}}$ with input 1^s and interacts with $\mathcal{A}$.
   - In this interaction, $\mathsf{Sim}^{\mathsf{NxtMsg}_i^j}_{\mathsf{S,GOD}}$ extracts the adversary's inputs to $\Pi_{\mathsf{NxtMsg}_i^j}$ to give to the ideal functionality computing $\mathcal{F}_{\mathsf{NxtMsg}_i^j}$

– The simulator $\mathsf{Sim}_{\mathsf{S,GOD}}$, emulating this ideal functionality, accepts these extracted inputs from $\mathsf{Sim}_{\mathsf{S,GOD}}^{\mathsf{NxtMsg}_i^j}$.
– $\mathsf{Sim}_{\mathsf{S,GOD}}$ computes the outputs of the simulated next-message functionality NxtMsg_i^{j*} using the extracted inputs, the view generated from the inputs and randomness that it has sampled on behalf of the honest parties in the 'Initialize Shared State' phase.
  • If this next-message protocol corresponds to the last round of the virtual protocol ($j = r$), then $\mathsf{Sim}_{\mathsf{S,GOD}}$ sends the adversary's inputs, that it extracted from the views received, to the ideal functionality $\mathcal{F}$ and receives $f(x)$.
  • $\mathsf{Sim}_{\mathsf{S,GOD}}$ uses this $f(x)$ to compute the outputs of this simulated next-message-functionality.
– If $\Pi_{\mathsf{NxtMsg}_i^j}$ executes to completion, then $\mathsf{Sim}_{\mathsf{S,GOD}}$, emulating $\mathcal{F}_{\mathsf{NxtMsg}_i^j}$, passes these computed outputs to $\mathsf{Sim}_{\mathsf{S,GOD}}^{\mathsf{NxtMsg}_i^j}$ for it to complete the emulation.
– This simulates the view of the corrupt parties in a way that is statistically close the real view. Let this view be,

$$\hat{\mathbf{View}}_{\mathsf{NxtMsg}_i^{j*}}^{Z,\mathsf{GOD}} \leftarrow \langle \mathsf{Sim}_{\mathsf{S,GOD}}^{\mathsf{NxtMsg}_i^{j*}}(1^s), \mathcal{A} \rangle$$

7. Finally, $\mathsf{Sim}_{\mathsf{S,GOD}}$ gives the adversary shares of the output such that when combined with the shares that it derives from its own views, they reconstruct to $\{f(\vec{x}) = f_i(\vec{x})\}_{P_i \in Z}$

This completes the simulation. Note that the above ideal-world execution does indeed run to completion because for all next-message protocols for which an unbounded adversary does not corrupt the virtual (designated) party, the protocol is secure with GOD. For all other next-message protocols, they run to completion with correctness and the view of the adversary in this simulation is identical to that in the real execution. The complete view of the adversary in the above execution of $\mathsf{Sim}_{\mathsf{S,GOD}}$ is distributed as,

$$\left\{ \{x_i\}_{P_i \in Z}, f(\vec{x}), \{r_i\}_{P_i \in Z}, \left\{ \hat{\mathbf{View}}_{\mathsf{NxtMsg}_i^{j*}}^{Z,\mathsf{GOD}} \right\}_{j \in [r], P_i \notin Z}, \left\{ \mathbf{View}_{\mathsf{NxtMsg}_i^{j}}^{Z,\mathsf{GOD}} \right\}_{j \in [r], P_i \in Z} \right\}_{\kappa \in \mathbb{N}, \vec{r} \in \mathcal{R}^n}$$

It remains to argue that the above view is statistically close to the real view of the adversary. For this, consider the following hybrids:

– Hybrid H_0. This is the distribution of the output of $\mathsf{Sim}_{\mathsf{S,GOD}}$.

$$\mathsf{H}_0 = \left\{ \{x_i\}_{P_i \in Z}, f(\vec{x}), \{r_i\}_{P_i \in Z}, \left\{ \hat{\mathbf{View}}_{\mathsf{NxtMsg}_i^{j*}}^{Z,\mathsf{GOD}} \right\}_{j \in [r], P_i \notin Z}, \left\{ \mathbf{View}_{\mathsf{NxtMsg}_i^{j}}^{Z,\mathsf{GOD}} \right\}_{j \in [r], P_i \in Z} \right\}_{\kappa \in \mathbb{N}, \vec{r} \in \mathcal{R}^n}$$

– Hybrid H_1. An intermediate hybrid distribution where, in the 'Initialize Shared State' phase, on behalf of all honest parties, the inputs and randomness for the real execution are used to create the initial view. The rest of the adversary's view beyond this is derived as in the simulation $\mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}$ but from this initial view:

$$\mathsf{H}_1 = \left\{ \{x_i\}_{P_i \in Z}, f(\vec{x}), \{r_i\}_{P_i \in Z}, \left\{ \hat{\mathbf{View}}^{Z,\mathsf{GOD}'}_{\mathsf{NxtMsg}_i^{j*}} \right\}_{j \in [r], P_i \notin Z}, \left\{ \mathbf{View}^{Z,\mathsf{GOD}}_{\mathsf{NxtMsg}_i^{j}} \right\}_{j \in [r], P_i \in Z} \right\}_{\kappa \in \mathbb{N}, \vec{r} \in \mathcal{R}^n}$$

The difference between this distribution and H_0 stems from the fact that in the adversary's view, all the statistically hiding commitments within the VSS are commitments to shares of the real (masked) input of the honest parties, instead of values picked arbitrarily by $\mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}$.

– Hybrid H_2. An intermediate hybrid distribution with the simulated virtual protocol $\mathsf{Sim}^{\mathcal{V}}_{\mathsf{S}}$ for the view of Z_V but with each protocol execution among the real parties $\mathcal{P}$ being a real execution:

$$\mathsf{H}_2 = \left\{ \{x_i\}_{P_i \in Z}, f(\vec{x}), \{r_i\}_{P_i \in Z}, \left\{ \mathbf{View}^{Z,\mathsf{GOD}'}_{\mathsf{NxtMsg}_i^{j*}} \right\}_{j \in [r], P_i \notin Z}, \left\{ \mathbf{View}^{Z,\mathsf{GOD}}_{\mathsf{NxtMsg}_i^{j}} \right\}_{j \in [r], P_i \in Z} \right\}_{\kappa \in \mathbb{N}, \vec{r} \in \mathcal{R}^n}$$

This differs from H_2 in that the view of real protocol corresponding to the next-message functions of the simulator of the virtual protocol $\mathsf{Sim}^{\mathcal{V}}_{\mathsf{S}}$ is generated as in a real execution of the Protocol 4, instead of using the simulation.

– Hybrid H_3. The distribution of the view of the parties in Z in a real execution of Protocol 4.

$$\mathsf{H}_3 = \left\{ \{x_i\}_{P_i \in Z}, f(\vec{x}), \{r_i\}_{P_i \in Z}, \left\{ \mathbf{View}^{Z,\mathsf{GOD}}_{\mathsf{NxtMsg}_i^{j}} \right\}_{j \in [r], i \in [n]} \right\}_{\kappa \in \mathbb{N}, \vec{r} \in \mathcal{R}^n}$$

Claim. Let $s = |Z|$ be the number of corrupt parties and r be the number of round in the virtual semi-honest protocol $\Pi_{\mathcal{V}}$. Assuming that the commitment schemes $\mathsf{Com}_{\mathsf{SH}}$ used in the next-message functionalities $\mathcal{F}_{\mathsf{NxtMsg}_i^j}$ (Figure 6) are statistically hiding commitment schemes where the statistical distance between commitments of two different messages is a negligible value ϵ, then the statistical distance between the two hybrid distributions H_0 and H_1 is $s^2 r \epsilon$. Hence,

$$\mathsf{H}_0 \overset{\mathrm{s}}{\equiv} \mathsf{H}_1$$

Proof. Let $\overline{Z} \subseteq \mathcal{P}$ denote the set of honest parties in this protocol execution. Let $s = |\overline{Z}|$ be the number of these parties and r be the number of round in the virtual semi-honest protocol $\Pi_{\mathcal{V}}$. The difference between the hybrid distributions H_0 and H_1 stems from the fact that, in the adversary's view, in the 'Initialize Shared State' phase, the shares are created differently.

In the simulated view H_0, on behalf of all honest parties, the randomness to be used in the virtual protocol is sampled uniformly at random and the inputs are arbitrarily picked by the simulator $\mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}$. The $\mathcal{Z}_{\mathsf{C}}^*$-secret-shares are generated on these values and these shared initial states are used as the basis of all views (of the virtual simulated next-message-functions) generated throughout the protocol. This view also contains the honest parties' statistically hiding commitments that need to be input to all next-message-functions of the corrupted virtual parties. In hybrid H_1, all of this is generated in the same way, but from the input and randomness of the honest parties as in the real execution.

Firstly, note that the randomness of the honest parties are sampled in the same way (uniformly at random from the domain) in both hybrid distributions. Therefore any difference between these hybrids needs to stem from the fact that the inputs used are different. Secondly, note that all intermediate views in the virtual protocol are secret-shared among all n parties using a $\mathcal{Z}_{\mathsf{C}}^*$-secret-sharing scheme. So the shares in the adversary's view of the view of *any* virtual party reveals no information about the actual view that is shared.

All that remains is to consider the information revealed from the statistically hiding commitments that the adversary receives for every next-message-functionality $\mathcal{F}_{\mathsf{NxtMsg}_i^j}$ for all $j \in [r]$ and honest party $P_i \in \mathcal{P} - Z$. For each such function, the adversary receives s commitments, one corresponding to each honest party's input view share. These are commitments on different values depending on which initial inputs were used for the honest parties and so it constitutes the only difference between H_0 and H_1.

As such if two commitments on different messages made using $\mathsf{Com}_{\mathsf{SH}}$ have statistical distance ϵ, which is negligible in the security parameter, and there are $s^2 r$ such commitments in the hybrid distributions H_0 and H_1, it follows that these distributions have statistical distance $s^2 r\epsilon$. Hence,

$$\mathsf{H}_0 \stackrel{\mathrm{s}}{\equiv} \mathsf{H}_1$$

Claim. For all $P_i \notin Z, j \in [r]$, assuming that the protocol $\Pi_{\mathsf{NxtMsg}_i^j}$ is secure with guaranteed output delivery in the presence of an unbounded adversary corrupting any subset $Z \in \mathcal{Z}_{\mathsf{S}}^{i,\mathsf{GOD}}$ of the parties, the view in the hybrid distributions H_1 and H_2 are statistically close.

Proof. Letting $s = |\mathcal{P} - Z|$ be the number of honest parties, the proof for the above claim follows from considering the following set of $sr + 1$ hybrids:

- Hybrid $\mathsf{H}'_{0,0}$: this distribution is the same as that of hybrid H_1 where the view in the 'Compute Virtual Protocol' phase is generated as in the simulation $\mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}$.
- $\forall i \in \mathcal{P} - Z, j \in [r] \mathsf{H}'_{i,j}$: in this distribution, up to the i, j^{th} next-message protocol $\Pi_{\mathsf{NxtMsg}_i^j}$ of an honest P_i, the view is generated exactly as in the real execution. Beyond this, each k^{th} protocol emulation view is generated as in the simulation:

$$\mathbf{View}^{Z,\mathsf{GOD}'}_{\mathsf{NxtMsg}_i^{j*}} \leftarrow \langle \mathsf{Sim}^{\mathsf{NxtMsg}_i^j}_{\mathsf{S},\mathsf{GOD}}(1^s), \mathcal{A} \rangle$$

– The last of these hybrids $\mathsf{H}'_{s,r}$ is distributed the same as H_2.

It remains to show that for each $i \in \mathcal{P} - Z, j \in [r]$, the adjacent hybrid distributions $\mathsf{H}'_{i,j}$ and $\mathsf{H}'_{i,j-1}$ are statistically close. Note that each such set of adjacent hybrids only differ in the view of the $(i,j)^{th}$ execution $\Pi_{\mathsf{NxtMsg}_i^j}$. In hybrid $\mathsf{H}'_{i,j}$, this is as in the real protocol execution, while in $\mathsf{H}'_{i,j-1}$ this is a simulated view generated using an interaction with $\mathsf{Sim}_{\mathsf{S,GOD}}^{\mathsf{NxtMsg}_i^j}$. For the protocol $\Pi_{\mathsf{NxtMsg}_i^j}$, the real view for the adversary and that generated using $\mathsf{Sim}_{\mathsf{S,GOD}}^{\mathsf{NxtMsg}_i^j}$ are statistically close with distance ϵ. Since this is the only difference between these adjacent hybrids, it follows that the complete hybrids are also statistically close. Hence, the hybrid distributions H_1 and H_2 are also at most $sr\epsilon$ apart.

Claim. Assuming that the virtual protocol $\Pi_{\mathcal{V}}$ is semi-honestly secure in the presence of an unbounded adversary corrupting any subset of parties $Z \in \mathcal{Z}_{\mathsf{S}}$, the view in the hybrid distributions H_2 and H_3 are statistically close.

Proof. The difference between the hybrid distributions H_2 and H_3 stems only from the fact that, in the adversary's view, in the 'Compute Virtual Protocol' phase, the views of the parties from the virtual protocol (and the shares thereof) are created differently. In hybrid H_2, these are views generated from the semi-honest virtual protocol simulator $\mathsf{Sim}_{\mathsf{S}}^{\mathcal{V}}$, while in H_3, these are distributed as views of the real execution of $\Pi_{\mathcal{V}}$. It therefore follows that if the distribution of the simulated and real view of $\Pi_{\mathcal{V}}$ are statistically close, so are the hybrids H_2 and H_3.

GOD FOR PPT ADVERSARIES. Let $\mathcal{A}$ be a malicious PPT adversary corrupting a set of parties $Z \in \mathcal{Z}_{\mathsf{GOD}}$, in particular when $Z \notin \mathcal{Z}_{\mathsf{S}}$. Consider the following simulator $\mathsf{Sim}_{\mathsf{C,GOD}}$ that makes black-box calls to simulators in $\{\mathsf{Sim}_{\mathsf{C,GOD}}^{\mathsf{NxtMsg}_i^j}\}_{i\in[n],j\in[r]}$ and simulates the view of $\mathcal{A}$ in the ideal world for guaranteed output delivery:

1. $\mathsf{Sim}_{\mathsf{C,GOD}}$ has as input the security parameter 1^κ.
2. In the 'Initialize Shared State' phase, on behalf of all the honest parties $P_k \in \mathcal{P} - Z$, it samples inputs x_k and randomness r_k. It then generates a $\mathcal{Z}_{\mathsf{C}}^*$-secret-sharing of $\hat{\mathbf{View}}_{\mathcal{V},k}^0 = (x_k, r_k)$ and sends all the adversary's shares $\{[\hat{\mathbf{View}}_{\mathcal{V},k}^0]_{k'}\}_{V_{k'}\in Z}$ to $\mathcal{A}$.
3. $\mathsf{Sim}_{\mathsf{S,GOD}}$ also accepts the secret shares $\{[\hat{\mathbf{View}}_{\mathcal{V},k}^0]_{k'}\}_{V_{k'}\in\mathcal{V}-Z,V_k\in Z}$ of the initial view in the virtual protocol of all corrupt parties.
4. In the 'Compute Virtual Protocol' phase, for each virtual party $V_i \in \mathcal{V}$ and each round $j \in [r]$ of the virtual protocol $\Pi_{\mathcal{V}}$, for the protocol $\Pi_{\mathsf{NxtMsg}_i^j}$ as in Figure 4, $\mathsf{Sim}_{\mathsf{C,GOD}}$ makes a black-box call to the PPT simulator $\mathsf{Sim}_{\mathsf{C,GOD}}^{\mathsf{NxtMsg}_i^j}$ with input 1^κ and interacts with $\mathcal{A}$.
   – In this interaction, $\mathsf{Sim}_{\mathsf{C,GOD}}^{\mathsf{NxtMsg}_i^j}$ extracts the adversary's inputs to $\Pi_{\mathsf{NxtMsg}_i^j}$ to give to the ideal functionality computing $\mathcal{F}_{\mathsf{NxtMsg}_i^j}$

– The simulator $\mathsf{Sim}_{\mathsf{C,GOD}}$, emulating this ideal functionality, accepts these extracted inputs from $\mathsf{Sim}_{\mathsf{C,GOD}}^{\mathsf{NxtMsg}_i^j}$.
– $\mathsf{Sim}_{\mathsf{C,GOD}}$ computes the outputs of this next-message-functionality using the extracted inputs and the view generated from the inputs and randomness that it has sampled on behalf of the honest parties in the 'Initialize Shared State' phase.
  • If this next-message protocol corresponds to the last round of the virtual protocol ($j = r$), then $\mathsf{Sim}_{\mathsf{C,GOD}}$ sends the adversary's inputs, that it extracted from the views received, to the ideal functionality $\mathcal{F}$ and receives $f(x)$.
  • $\mathsf{Sim}_{\mathsf{C,GOD}}$ uses this $f(x)$ to compute the outputs of this next-message-functionality.
– If $\Pi_{\mathsf{NxtMsg}_i^j}$ executes to completion, then $\mathsf{Sim}_{\mathsf{C,GOD}}$, emulating $\mathcal{F}_{\mathsf{NxtMsg}_i^j}$, passes these computed outputs to $\mathsf{Sim}_{\mathsf{C,GOD}}^{\mathsf{NxtMsg}_i^j}$ for it to complete the emulation.
– This simulates the view of the corrupt parties in a way that is computationally indistinguishable from the real view. Let this view be,

$$\hat{\mathbf{View}}_{\mathsf{NxtMsg}_i^j}^{Z,\mathsf{GOD}} \leftarrow \langle \mathsf{Sim}_{\mathsf{C,GOD}}^{\mathsf{NxtMsg}_i^j}(1^\kappa), \mathcal{A} \rangle$$

5. Finally, $\mathsf{Sim}_{\mathsf{C,GOD}}$ gives the adversary shares of the output such that they reconstruct to $\{f(\vec{x}) = f_i(\vec{x})\}_{P_i \in Z}$

This completes the simulation. Note that this above ideal-world execution does indeed run to completion without aborting since all the building block protocols also provide GOD. The joint view of the adversary in the above execution of $\mathsf{Sim}_{\mathsf{C,GOD}}$ is distributed as,

$$\left\{ \{x_i\}_{P_i \in Z}, f(\vec{x}), \{r_i\}_{P_i \in Z}, \{\hat{\mathbf{View}}_{\mathsf{NxtMsg}_i^j}^{Z,\mathsf{GOD}}\}_{i \in [n], j \in [r]} \right\}_{\kappa \in \mathbb{N}, \vec{r} \in \mathcal{R}^n}$$

It remains to argue that the above view is computationally indistinguishable from the real view of the adversary in Figure 9. Note that this view differs from the real view in the protocol on two fronts. Firstly, in the simulation, on behalf of all the honest parties, the initial shares and commitments in the 'Initialize Shared State' phase are generated from arbitrary values sampled by $\mathsf{Sim}_{\mathsf{C,GOD}}$. Next, the views of the next-message protocols are composed of calls to $\mathsf{Sim}_{\mathsf{C,GOD}}^{\mathsf{NxtMsg}_i^j}$ in the simulation. In order to show that the output of $\mathsf{Sim}_{\mathsf{C,GOD}}$ is indistinguishable from the real view, consider the following set of hybrids:

– Hybrid H_0. This is the distribution of the output of $\mathsf{Sim}_{\mathsf{C,GOD}}$.

$$\mathsf{H}_0 = \left\{ \{x_i\}_{P_i \in Z}, f(\vec{x}), \{r_i\}_{P_i \in Z}, \{\hat{\mathbf{View}}_{\mathsf{NxtMsg}_i^j}^{Z,\mathsf{GOD}}\}_{i \in [n], j \in [r]} \right\}_{\kappa \in \mathbb{N}, \vec{r} \in \mathcal{R}^n}$$

– Hybrid H_1. This is a hybrid experiment where, in the 'Initialize Shared State' phase, the elements in the view of the adversary are generated using the input and randomness of the honest parties as used in the real execution of the protocol. The rest of the view is generated according to the simulation but with these initial states as the basis, instead of deriving views from the state sampled in the simulation. This hybrid is distributed as,

$$\mathsf{H}_1 = \left\{ \{x_i\}_{P_i \in Z}, f(\vec{x}), \{r_i\}_{P_i \in Z}, \{\hat{\mathbf{View}}^{*Z,\mathsf{GOD}}_{\mathsf{NxtMsg}_i^j}\}_{i\in[n],j\in[r]} \right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n}$$

– Hybrid H_2. The last hybrid is distributed exactly as the real execution of Protocol 9. It differs from the previous in that all the protocol executions in the 'Compute Virtual Protocol' phase are carried out as in the real protocol, and not as calls to $\mathsf{Sim}^{\mathsf{NxtMsg}_i^j}_{\mathsf{C},\mathsf{GOD}}$.

$$\mathsf{H}_2 = \left\{ \{x_i\}_{P_i \in Z}, f(\vec{x}), \{r_i\}_{P_i \in Z}, \{\mathbf{View}^{Z,\mathsf{GOD}}_{\mathsf{NxtMsg}_i^j}\}_{i\in[n],j\in[r]} \right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n}$$

Claim. Assuming that the commitment schemes $\mathsf{Com}_{\mathsf{SB}}$ used in the next-message-functionalities $\mathcal{F}_{\mathsf{NxtMsg}_i^j}$ (Figure 6) are at least computationally hiding commitment schemes, then the hybrid distributions H_0 and H_1 are computationally indistinguishable.

Proof. Let $\overline{Z} \subseteq \mathcal{P}$ be the set of honest parties where $s = |\overline{Z}|$. The difference between the hybrid distributions H_0 and H_1 stem from how the inputs and randomness corresponding to each honest party $P_k \in \overline{Z}$ are used to create the view of the adversary. In H_0, these are sampled by the simulator $\mathsf{Sim}_{\mathsf{C},\mathsf{GOD}}$ while in H_1, these are the same as the honest parties' inputs and randomness in the real execution. In both cases, in the 'Initialize Shared State' phase, this input x_k and randomnessr_k is first secret-shared using a $\mathcal{Z}^*_{\mathsf{C}}$-secret-sharing scheme where the view of the adversary contains only shares corresponding to $Z \in \mathcal{Z}_{\mathsf{GOD}} \subset \mathcal{Z}^*_{\mathsf{C}}$. This means that given all the shares that the adversary receives, it can infer no information on the message that was shared.

Further, for the first round of the virtual protocol, for each $i \in [n]$ next-message-functionality $\mathcal{F}_{\mathsf{NxtMsg}_i^1}$, party P_i receives a statistically binding (computationally hiding) commitment made using $\mathsf{Com}_{\mathsf{SB}}$, of the input of all other parties to this functionality. This input is a secret-share of the initial view of the virtual party V_i. For all corrupt parties, this view is distributed identically in both hybrids since the are derived in the exact same way.

For each successive round j in the virtual protocol, however, again, for each $i \in [n]$ next-message-functionality $\mathcal{F}_{\mathsf{NxtMsg}_i^j}$, party P_i receives a statistically binding (computationally hiding) commitment of the input of all other parties to this functionality. The message that is committed to here is a secret-share of the intermediate state of the virtual party V_i generated when the ideal functionality $\mathcal{F}_{\mathsf{NxtMsg}_i^j}$ outputs depend on the view of the honest parties as well. Specifically, in hybrid H_0 these functionality outputs depend on the input and randomness

chosen by the simulator and in hybrid H_1 these are as in the real execution of the honest parties. In order to show that these distributions are computationally indistinguishable consider the following set of $s(n-s)(r-1)+1$ intermediate hybrid distributions:

- Hybrid H'_0 : this distribution is the same as hybrid H_0, the simulated distribution and contains all the inputs and randomness of the honest parties as sampled by the simulator.
- $\forall j \in [s], k \in [n-s], l \in [r-1], \mathsf{H}'_{j,k,l}$: in this distribution, on behalf of the honest parties, in the next-message-functionalities up to before $\mathcal{F}_{\mathsf{NxtMsg}_l^k}$, all the inputs and outputs are derived using the honest parties inputs as in the real execution. For $\mathcal{F}_{\mathsf{NxtMsg}_l^k}$, for up to the j^{th} honest party, the view contains statistically binding commitments to output shares derived from the real honest party inputs. Beyond this, it receives commitments to the simulated input shares. In the next-message-functionalities to come, these simulated shares are also input by the hybrid experiment (the consistency check is skipped) and the rest of the view is generated exactly as in the simulation from this point.
- The last of these hybrids $\mathsf{H}'_{s,n-s,r-1}$ is distributed the same as H_1 as described above with all the inputs of the honest parties being that in the real protocol execution and all outputs of $\mathcal{F}_{\mathsf{NxtMsg}_j^i}$ being derived from this.

It now remains to argue that adjacent hybrid distributions as described above are computationally indistinguishable. For each $j \in [s], k \in [n-s], l \in [r-1]$, note that the difference between the hybrid distributions $\mathsf{H}'_{j,k,l}$ and $\mathsf{H}'_{j-1,k,l}$ stems from the fact that in the 'Compute Virtual Protocol' phase, the input for the j^{th} computationally hiding commitment is different. In both cases this is a commitment of a share in a $\mathcal{Z}_{\mathsf{C}}^*$-secret-sharing of a message and this value committed to does not exist in the view of the adversary.

If there exists a PPT distinguisher D that can distinguish the distributions $\mathsf{H}'_{j,k,l}$ and $\mathsf{H}'_{j-1,k,l}$ with non-negligible advantage ϵ, then we can show that this can be used in a black-box way to create a PPT machine $\mathsf{D}_{\mathsf{Com}_{\mathsf{SB}}}$ that can violate the computational hiding property of this underlying commitment scheme. $\mathsf{D}_{\mathsf{Com}_{\mathsf{SB}}}$ would operate in the computational hiding security game as follows:

- First, $\mathsf{D}_{\mathsf{Com}_{\mathsf{SB}}}$ uses the real inputs and randomness of all the parties and creates the view of the adversary up to the $j-1^{th}$ statistically binding commitment in functionality $\mathcal{F}_{\mathsf{NxtMsg}_l^k}$ exactly as in the real distribution.
- For the j^{th} share to be committed to, it samples a simulated message and secret-shares it under the constraint that the shares that the adversary receives are the same as in the real distribution. Then this j^{th} share of the simulated value s_0 and the real share s_1 are given to the challenger $\mathcal{C}$.
- $\mathcal{C}$ samples a bit b uniformly at random and creates a commitment c to s_b using the computationally hiding commitment scheme $\mathsf{Com}_{\mathsf{SB}}$. This commitment c is returned to $\mathsf{D}_{\mathsf{Com}_{\mathsf{SB}}}$.
- $\mathsf{D}_{\mathsf{Com}_{\mathsf{SB}}}$ uses this and completes the view of the adversary as generated by the simulator. Note that if $b=0$, this view would be distributed as in $\mathsf{H}'_{j,k,l}$ and this would be $\mathsf{H}'_{j-1,k,l}$ otherwise.

– $\mathsf{D}_{\mathsf{Com}_{\mathsf{SB}}}$ passes this view onto D and outputs whatever it outputs.

Note that $\mathsf{D}_{\mathsf{Com}_{\mathsf{SB}}}$ here has the same advantage as D which is non-negligible. However, since the commitment schemes are computationally hiding, no such $\mathsf{D}_{\mathsf{Com}_{\mathsf{SB}}}$ can exist, so no such D can exist. Therefore, all the adjacent hybrids $\mathsf{H}'_{j,k,l}$ and $\mathsf{H}'_{j-1,k,l}$ are computationally indistinguishable.

Claim. For all $i \in [n], j \in [r]$, assuming that the protocol $\Pi_{\mathsf{NxtMsg}_i^j}$ is secure with guaranteed output delivery in the presence of a PPT malicious adversary corrupting any subset $Z \in \mathcal{Z}_{\mathsf{GOD}}$ of the parties, the view in the hybrid distribution H_1 is computationally indistinguishable from that in H_2.

Proof. The proof for this claim follows from considering the following set of $rn + 1$ hybrids:

– Hybrid H'_0. This is distributed the same way as H_1.

$$\mathsf{H}'_0 = \left\{ \{x_i\}_{P_i \in Z}, f(\vec{x}), \{r_i\}_{P_i \in Z}, \{\hat{\mathbf{View}}^{*Z,\mathsf{GOD}}_{\mathsf{NxtMsg}_i^j}\}_{i \in [n], j \in [r]} \right\}_{\kappa \in \mathbb{N}, \vec{r} \in \mathcal{R}^n}$$

– Hybrid $\mathsf{H}'_{i,j}$. For each $i \in [n]$ and $j \in [r]$, this hybrid experiment contains the view of parties in Z in the real protocol execution up to the execution of the sub-protocol $\Pi_{\mathsf{NxtMsg}_i^j}$. The rest of the view is generated as in the simulation $\mathsf{Sim}_{\mathsf{C,GOD}}$.

$$\mathsf{H}'_{i,j} = \left\{ \begin{array}{l} \{x_{i'}\}_{P_{i'} \in Z}, f(\vec{x}), \{r_{i'}\}_{P_{i'} \in Z}, \\ \quad \{\mathbf{View}^{Z,\mathsf{GOD}}_{\mathsf{NxtMsg}_{i'}^{j'}}\}_{i' < i, (i'=i, j' \leq j)}, \\ \quad \{\hat{\mathbf{View}}^{*Z,\mathsf{GOD}}_{\mathsf{NxtMsg}_{i'}^{j'}}\}_{i' > i, (i'=i, j' > j)} \end{array} \right\}_{\kappa \in \mathbb{N}, \vec{r} \in \mathcal{R}^n}$$

– Hybrid $\mathsf{H}'_{n,r}$. Note that the last hybrid of the above form is distributed exactly as in H_2, the real execution of Protocol 9.

$$\mathsf{H}'_{n,r} = \left\{ \{x_i\}_{P_i \in Z}, f(\vec{x}), \{r_i\}_{P_i \in Z}, \{\mathbf{View}^{Z,\mathsf{GOD}}_{\mathsf{NxtMsg}_i^j}\}_{i \in [n], j \in [r]} \right\}_{\kappa \in \mathbb{N}, \vec{r} \in \mathcal{R}^n}$$

Note that in the hybrid distributions described above, each pair of adjacent hybrids $\mathsf{H}'_{i,j}$ and $\mathsf{H}'_{i,j-1}$ (similarly, H'_0 and $\mathsf{H}'_{1,1}$; and each pair $\mathsf{H}'_{i,r}$ and $\mathsf{H}'_{i+1,1}$) differ only the view of the execution of the sub-protocol $\Pi_{\mathsf{NxtMsg}_i^j}$. In $\mathsf{H}'_{i,j-1}$, this view is the simulated view $\hat{\mathbf{View}}^{*Z,\mathsf{GOD}}_{\mathsf{NxtMsg}_i^j}$ output by $\mathsf{Sim}^{\mathsf{NxtMsg}_i^j}_{\mathsf{C,GOD}}$. In $\mathsf{H}'_{i,j}$ this is the view of the parties in Z in the real execution of $\Pi_{\mathsf{NxtMsg}_i^j}$.

We show that if there existed a PPT distinguisher D that can distinguish between the hybrid distributions $\mathsf{H}'_{i,j}$ and $\mathsf{H}'_{i,j-1}$ with non-negligible advantage ϵ, then D can be used in a black-box way by a PPT adversary $\mathcal{A}$ that distinguishes between the distributions $\{\hat{\mathbf{View}}^{*Z,\mathsf{GOD}}_{\mathsf{NxtMsg}_i^j}\}_{\kappa \in \mathbb{N}, \vec{r} \in \mathcal{R}^n}$ from the output of $\mathsf{Sim}^{\mathsf{NxtMsg}_i^j}_{\mathsf{C,GOD}}$ and $\{\mathbf{View}^{Z,\mathsf{GOD}}_{\mathsf{NxtMsg}_i^j}\}_{\kappa \in \mathbb{N}, \vec{r} \in \mathcal{R}^n}$ from the real execution of the sub-protocol as in Figure 4. The adversary $\mathcal{A}$ works as follows:

– $\mathcal{A}$ has the indices (i, j), the set of corrupted parties $Z \in \mathcal{Z}_{\mathsf{GOD}}$ and $(\vec{x}, f)$. It samples randomness $\vec{r} \leftarrow \mathcal{R}^n$ for all parties.
– For the next message functions $\mathsf{NxtMsg}_{i'}^{j'}$ of $\Pi_{\mathcal{V}}$ where $i' < i$ or $i' = i$ and $j' < j$, it creates the view as in the real protocol: $\mathbf{View}^{Z,\mathsf{GOD}}_{\mathsf{NxtMsg}_{i'}^{j'}}$. For the instances where $i' > i$ or $i' = i$ and $j' > j$, it creates the view as in the simulation $\hat{\mathbf{View}}^{Z,\mathsf{GOD}}_{\mathsf{NxtMsg}_{i'}^{j'}}$.
– Corresponding to the $(i, j)^{th}$ instance, $\mathcal{A}$ sends all the inputs and outputs to the challenger $\mathcal{C}$ that samples a bit b. If $b = 0$, $\mathcal{C}$ computes,

$$\mathbf{View}_{\mathsf{NxtMsg}} \leftarrow \langle \mathsf{Sim}_{\mathsf{C,GOD}}^{\mathsf{NxtMsg}_i^j}(1^\kappa), \mathcal{A} \rangle$$

Otherwise it computes $\mathbf{View}_{\mathsf{NxtMsg}}$ to be the views of the parties in Z in the real execution of $\Pi_{\mathsf{NxtMsg}_i^j}$.
– $\mathcal{C}$ gives $\mathbf{View}_{\mathsf{NxtMsg}}$ to $\mathcal{A}$ and, with this, $\mathcal{A}$ completes the view $\mathbf{View}$ to be input to D. Note that if $b = 0$, $\mathbf{View}$ is distributed as in $\mathsf{H}'_{i,j-1}$ and if $b = 1$, it is distributed as in $\mathsf{H}'_{i,j}$.
– Finally, $\mathcal{A}$ passes $\mathbf{View}$ to D and outputs whatever D outputs.

In the above strategy, $\mathcal{A}$ has the same distinguishing advantage as D, which is non-negligible. However, since the protocol $\Pi_{\mathsf{NxtMsg}_i^j}$ is secure in the presence of a PPT malicious adversary corrupting any subset $Z \in \mathcal{Z}_{\mathsf{GOD}}$ of the parties, no such $\mathcal{A}$ can exist and therefore no such D can exist. So the distributions $\mathsf{H}'_{i,j}$ and $\mathsf{H}'_{i,j-1}$ are computationally indistinguishable. Since none of the listed set of adjacent hybrids are distinguishable, it follows that the real view of the protocol $\mathsf{H}'_{n,r}$ and the simulated view H'_0 are computationally indistinguishable.

Therefore, it follows that the view produced by $\mathsf{Sim}_{\mathsf{C,GOD}}$ is overall computationally indistinguishable to that of the real execution of Figure 9 for any $Z \in \mathcal{Z}_{\mathsf{GOD}}$.

Fairness for Unbounded Adversaries. Let $\mathcal{A}$ be a malicious unbounded adversary corrupting a set of parties $Z \in \mathcal{Z}_{\mathsf{S}} \cap \mathcal{Z}^*_{\mathsf{fair}}$. Note that since $\mathcal{Z}^*_{\mathsf{fair}} = \mathcal{Z}_{\mathsf{fair}} \cup \mathcal{Z}_{\mathsf{GOD}}$, and we already have from a previous case that GOD, a stronger guarantee than fairness, holds for all corruption within $\mathcal{Z}_{\mathsf{S}} \cap \mathcal{Z}_{\mathsf{GOD}}$, it only remains to argue that fairness holds for any $Z \in \mathcal{Z}_{\mathsf{S}} \cap \mathcal{Z}_{\mathsf{fair}}$. Consider the following PPT simulator $\mathsf{Sim}_{\mathsf{S,fair}}$ that makes black-box calls to simulators in $\{\mathsf{Sim}_{\mathsf{S},\perp}^{\mathsf{NxtMsg}_i^j}\}_{i\in[n],j\in[r]}$ and simulates the view of $\mathcal{A}$ in the ideal world for fairness. This simulator operates in the same way as $\mathsf{Sim}_{\mathsf{S,GOD}}$ would, but with the following differences:

1. During the phase to 'compute virtual protocol', $\mathsf{Sim}_{\mathsf{S,fair}}$ makes black-box calls to $\mathsf{Sim}_{\mathsf{S},\perp}^{\mathsf{NxtMsg}_i^j}$ instead of to $\mathsf{Sim}_{\mathsf{S,GOD}}^{\mathsf{NxtMsg}_i^j}$.
2. Furthermore, if any of these simulation interactions abort, $\mathsf{Sim}_{\mathsf{S,fair}}$ sends $\perp$ to the ideal functionality $\mathcal{F}$ and terminates execution. The view generated up to this point is output by the simulator.

3. If *all* calls to $\mathsf{Sim}_{\mathsf{S},\perp}^{\mathsf{NxtMsg}_i^j}$ execute to completion, $\mathsf{Sim}_{\mathsf{S},\mathsf{fair}}$ sends the inputs of the adversary $\mathcal{A}$ extracted from the views to $\mathcal{F}$ and receives $f(x)$. It creates a $\mathcal{Z}_{\mathsf{fair}}$-verifiable-secret-sharing of $f(x)$ and uses this to complete the protocol execution in the last round of next-message-function calls of the virtual protocol $\Pi_{\mathcal{V}}$.

Note that the adversary causes the protocol to abort in the ideal execution, iff the protocol also aborts in the real execution. The view of the protocol up to that point was generated without the output of the functionality and so it follows that, along with upholding the privacy of honest parties' inputs, both the adversary and the honest parties don't learn the output. If the protocol executes to completion, the honest parties always have enough shares to reconstruct the output of the functionality. In this case, the view of the adversary in the above execution of $\mathsf{Sim}_{\mathsf{S},\mathsf{fair}}$ is distributed as,

$$\Big\{\{x_i\}_{P_i\in Z}, f(\vec{x}), \{r_i\}_{P_i\in Z}, \big\{\hat{\mathbf{View}}^{Z,\perp}_{\mathsf{NxtMsg}_i^{j*}}\big\}_{j\in[r],P_i\notin Z}, \big\{\mathbf{View}^{Z,\perp}_{\mathsf{NxtMsg}_i^{j}}\big\}_{j\in[r],P_i\in Z}\Big\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n}$$

This distribution can be argued as statistically close to the real view of the adversary by considering a similar set of hybrids as in the GOD case, with the only difference being that the argument would now follow from the fact that the views output from $\{\mathsf{Sim}_{\mathsf{S},\perp}^{\mathsf{NxtMsg}_i^j}\}_{i\in[n],j\in[r]}$ (as opposed to $\{\mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}^{\mathsf{NxtMsg}_i^j}\}_{i\in[n],j\in[r]}$) are statistically close to the real views of the protocol execution.

Fairness for PPT Adversaries. Let $\mathcal{A}$ be a malicious PPT adversary corrupting a set of parties $Z \in \mathcal{Z}^*_{\mathsf{fair}}$. Note that since $\mathcal{Z}^*_{\mathsf{fair}} = \mathcal{Z}_{\mathsf{fair}} \cup \mathcal{Z}_{\mathsf{GOD}}$, and we already have from a previous case that GOD, a stronger guarantee than fairness, holds for all corruption within $\mathcal{Z}_{\mathsf{GOD}}$, it only remains to argue that fairness holds for any $Z \in \mathcal{Z}_{\mathsf{fair}}$, in particular, also for sets not included in $\mathcal{Z}_{\mathsf{S}}$. Consider the following PPT simulator $\mathsf{Sim}_{\mathsf{C},\mathsf{fair}}$ that makes black-box calls to simulators in $\{\mathsf{Sim}_{\mathsf{C},\perp}^{\mathsf{NxtMsg}_i^j}\}_{i\in[n],j\in[r]}$ and simulates the view of $\mathcal{A}$ in the ideal world for fairness. This simulator operates in the same way as $\mathsf{Sim}_{\mathsf{C},\mathsf{GOD}}$ would, but with the following differences:

1. During the phase to 'compute virtual protocol', $\mathsf{Sim}_{\mathsf{C},\mathsf{fair}}$ makes black-box calls to $\mathsf{Sim}_{\mathsf{C},\perp}^{\mathsf{NxtMsg}_i^j}$ instead of to $\mathsf{Sim}_{\mathsf{C},\mathsf{GOD}}^{\mathsf{NxtMsg}_i^j}$.
2. Furthermore, if any of these simulation interactions abort, $\mathsf{Sim}_{\mathsf{C},\mathsf{fair}}$ sends $\perp$ to the ideal functionality $\mathcal{F}$ and terminates execution. The view generated up to this point is output by the simulator.
3. If *all* calls to $\mathsf{Sim}_{\mathsf{C},\perp}^{\mathsf{NxtMsg}_i^j}$ execute to completion, $\mathsf{Sim}_{\mathsf{C},\mathsf{fair}}$ sends the inputs of the adversary $\mathcal{A}$ extracted from the views to $\mathcal{F}$ and receives $f(x)$. It creates a $\mathcal{Z}_{\mathsf{fair}}$-verifiable-secret-sharing of $f(x)$ and uses this to complete the protocol execution in the last round of next-message-function calls of the virtual protocol $\Pi_{\mathcal{V}}$.

Note that the adversary causes the protocol to abort in the ideal execution, iff the protocol also aborts in the real execution. The view of the protocol up to that point was generated without the output of the functionality and so it follows that, along with upholding the privacy of honest parties' inputs, both the adversary and the honest parties don't learn the output. If the protocol executes to completion, the honest parties always have enough shares to reconstruct the output of the functionality. In this case, the view of the adversary in the above execution of $\mathsf{Sim}_{\mathsf{C},\mathsf{fair}}$ is distributed as,

$$\left\{\{x_i\}_{P_i\in Z}, f(\vec{x}), \{r_i\}_{P_i\in Z}, \left\{\hat{\mathbf{View}}^{Z,\perp}_{\mathsf{NxtMsg}_i^{j*}}\right\}_{j\in[r],i\in[n]}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n}$$

This distribution can be argued as computationally indistinguishable from the real view of the adversary by considering a similar set of hybrids as in the GOD case, with the only difference being that the argument would now follow from a reduction to the fact that the views output from $\{\mathsf{Sim}^{\mathsf{NxtMsg}_i^j}_{\mathsf{C},\perp}\}_{i\in[n],j\in[r]}$ (as opposed to $\{\mathsf{Sim}^{\mathsf{NxtMsg}_i^j}_{\mathsf{C},\mathsf{GOD}}\}_{i\in[n],j\in[r]}$) are computationally indistinguishable from the real views of the protocol execution.

Security with Abort for Unbounded Adversaries. Let $\mathcal{A}$ be a malicious unbounded adversary corrupting a set of parties $Z \in \mathcal{Z}_{\mathsf{S}}$, in particular a set not in $\mathcal{Z}^*_{\mathsf{fair}}$. Consider the following PPT simulator $\mathsf{Sim}_{\mathsf{S},\perp}$ that makes black-box calls to simulators in $\{\mathsf{Sim}^{\mathsf{NxtMsg}_i^j}_{\mathsf{S},\perp}\}_{i\in[n],j\in[r]}$ and simulates the view of $\mathcal{A}$ in the ideal world for security with abort. This simulator operates in the same way as $\mathsf{Sim}_{\mathsf{S},\mathsf{fair}}$ would, but with the following differences:

1. If *all* calls to $\mathsf{Sim}^{\mathsf{NxtMsg}_i^j}_{\mathsf{S},\perp}$ execute to completion, $\mathsf{Sim}_{\mathsf{S},\perp}$, like $\mathsf{Sim}_{\mathsf{S},\mathsf{fair}}$, sends the inputs of the adversary $\mathcal{A}$ extracted from the views to $\mathcal{F}$ and receives $f(x)$. It creates a $\mathcal{Z}_{\mathsf{fair}}$-verifiable-secret-sharing of $f(x)$ and uses this to complete the protocol execution in the last round of next-message-function calls of the virtual protocol $\Pi_{\mathcal{V}}$.
2. If the set of parties corrupted by the adversary $Z \notin \mathcal{Z}^*_{\mathsf{fair}}$, send $\mathsf{Sim}_{\mathsf{S},\perp}$ sends $\perp$ to the ideal functionality $\mathcal{F}$ and $\mathcal{F}$ sends $\perp$ to all honest parties. Otherwise, $\mathsf{Sim}_{\mathsf{S},\perp}$ sends $\mathsf{continue}$ to $\mathcal{F}$ and the honest parties receive the outputs as well.

Note that the adversary causes the protocol to abort in the ideal execution, iff the protocol also aborts in the real execution. If the protocol executes to completion, and there are not enough honest parties to reconstruct the function output, the protocol would abort with only the adversary receiving the function output. In this case, the view of the adversary in the above execution of $\mathsf{Sim}_{\mathsf{S},\perp}$ is distributed as,

$$\left\{\{x_i\}_{P_i\in Z}, f(\vec{x}), \{r_i\}_{P_i\in Z}, \left\{\hat{\mathbf{View}}^{Z,\perp}_{\mathsf{NxtMsg}_i^{j*}}\right\}_{j\in[r],P_i\notin Z},\right.$$

$$\left\{\left\{\mathbf{View}^{Z,\perp}_{\mathsf{NxtMsg}^j_i}\right\}_{j\in[r],P_i\in Z}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n}$$

This distribution can be argued as statistically close to the real view of the adversary by considering the exact same set of hybrids as in the fairness case.

Security with Abort for PPT adversaries. Let $\mathcal{A}$ be a malicious PPT adversary corrupting a set of parties $Z \in \mathcal{Z}^*_{\mathsf{C}}$, in particular a set not in $\mathcal{Z}^*_{\mathsf{fair}}$. Consider the following PPT simulator $\mathsf{Sim}_{\mathsf{C},\perp}$ that makes black-box calls to simulators in $\{\mathsf{Sim}^{\mathsf{NxtMsg}^j_i}_{\mathsf{C},\perp}\}_{i\in[n],j\in[r]}$ and simulates the view of $\mathcal{A}$ in the ideal world for security with abort. This simulator operates in the same way as $\mathsf{Sim}_{\mathsf{C},\mathsf{fair}}$ would, but with the following differences:

1. If *all* calls to $\mathsf{Sim}^{\mathsf{NxtMsg}^j_i}_{\mathsf{C},\perp}$ execute to completion, $\mathsf{Sim}_{\mathsf{C},\perp}$, like $\mathsf{Sim}_{\mathsf{C},\mathsf{fair}}$, sends the inputs of the adversary $\mathcal{A}$ extracted from the views to $\mathcal{F}$ and receives $f(x)$. It creates a $\mathcal{Z}_{\mathsf{fair}}$-verifiable-secret-sharing of $f(x)$ and uses this to complete the protocol execution in the last round of next-message-function calls of the virtual protocol $\Pi_{\mathcal{V}}$.
2. If the set of parties corrupted by the adversary $Z \notin \mathcal{Z}^*_{\mathsf{fair}}$, send $\mathsf{Sim}_{\mathsf{C},\perp}$ sends $\perp$ to the ideal functionality $\mathcal{F}$ and $\mathcal{F}$ sends $\perp$ to all honest parties. Otherwise, $\mathsf{Sim}_{\mathsf{C},\perp}$ sends $\mathsf{continue}$ to $\mathcal{F}$ and the honest parties receive the outputs as well.

Note that the adversary causes the protocol to abort in the ideal execution, iff the protocol also aborts in the real execution. If the protocol executes to completion, and there are not enough honest parties to reconstruct the function output, the protocol would abort with only the adversary receiving the function output. In this case, the view of the adversary in the above execution of $\mathsf{Sim}_{\mathsf{C},\perp}$ is distributed as,

$$\left\{\{x_i\}_{P_i\in Z}, f(\vec{x}), \{r_i\}_{P_i\in Z}, \left\{\hat{\mathbf{View}}^{Z,\perp}_{\mathsf{NxtMsg}^{j^*}_i}\right\}_{j\in[r],i\in[n]}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n}$$

This distribution can be argued as computationally indistinguishable from the real view of the adversary by considering the exact same set of hybrids as in the fairness case.

E Proof of Theorem 5

Proof. Let $\mathcal{Z}_{\mathsf{GOD}} \in \mathcal{Q}^2$ and where $\mathcal{Z}_{\mathsf{GOD}} = \mathcal{Z}^*_{\mathsf{GOD}} \cup \mathcal{Z}'_{\mathsf{GOD}}$. Let $\mathcal{Z}^*_{\mathsf{C}}$ be the maximal adversary structure such it satisfies Equation 2 with $\mathcal{Z}'_{\mathsf{GOD}}$ and Equation 3 with $\mathcal{Z}^*_{\mathsf{GOD}}$. Let $\mathcal{Z}_{\mathsf{fair}} \in \mathcal{Q}^2$ be the adversary structure for which fairness is required and it holds that $\mathcal{Z}^*_{\mathsf{fair}} = \mathcal{Z}_{\mathsf{GOD}} \cup \mathcal{Z}_{\mathsf{fair}} \in \mathcal{Q}^2$. Let $\mathcal{Z}_{\mathsf{S}} \in \mathcal{Q}^2$ be the adversary structure for which security, at least with abort, in the presence of an unbounded adversary is required. Additionally, let it hold that $\mathcal{Z}^*_{\mathsf{fair}}, \mathcal{Z}_{\mathsf{S}} \subset \mathcal{Z}^*_{\mathsf{C}}$. Let Π_1 be

an n-party protocol satisfying $\mathcal{Z}_{\mathsf{GOD}}$. Let Π_2 and Π_3 be n-party best-in-the-multiverse protocols satisfying Equation 2 and 3 respectively. Let Π be a protocol compiled as indicated in Section 5.4. For this protocol, the proof for Theorem 4 follows from analysing the following cases separately:

1. We need to show that security with Guaranteed Output Delivery (GOD) holds in the presence of an unbounded adversary when it corrupts any set of parties in the adversary structure $\mathcal{Z}_{\mathsf{S}} \cap \mathcal{Z}_{\mathsf{GOD}}$.
2. We need to show that GOD is also satisfied for a PPT adversary corrupting any set of parties in $\mathcal{Z}_{\mathsf{GOD}}$, even for the sets not included above case.
3. We need to show that security with fairness holds in the presence of an unbounded adversary when it corrupts any set of parties in the adversary structure $\mathcal{Z}_{\mathsf{S}} \cap \mathcal{Z}_{\mathsf{fair}}$.
4. We need to show that fairness is also satisfied for a PPT adversary corrupting any set of parties in $\mathcal{Z}_{\mathsf{fair}}$, even for the sets not included above case.
5. We show that security with abort holds for an unbounded adversary when it corrupts any set of parties in $\mathcal{Z}_{\mathsf{S}}$, in particular, for those not in $\mathcal{Z}^*_{\mathsf{fair}}$.
6. We also show that security with abort holds for a PPT adversary when it corrupts any set of parties in $\mathcal{Z}^*_{\mathsf{C}}$, even for those not included in $\mathcal{Z}^*_{\mathsf{fair}}$.

In order to examine the security of Π in the plain model, it becomes necessary to describe the following properties of its building blocks:

– For the fail-stop secure virtual 3-party protocol $\Pi_{\mathcal{V}}$, let $\mathsf{Sim}_{\mathcal{V}}$ be the simulator for an unbounded adversary $\mathcal{A}$ corrupting any one of the 3 parties $\mathcal{V}_j \in \{\mathcal{V}_1, \mathcal{V}_2, \mathcal{V}_3\}$.

$$\hat{\mathbf{View}}^{j}_{\mathcal{V}} \leftarrow \mathsf{Sim}_{\mathcal{V}}(1^s, x_j, f(\vec{x}))$$

$$\left\{\hat{\mathbf{View}}^{j}_{\mathcal{V}}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n} \overset{\mathrm{s}}{\equiv} \left\{\mathbf{View}^{j}_{\mathcal{V}}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n}$$

Letting r be the number of rounds in this protocol, recall that the final protocol Π consists of $3r$ executions of protocols computing next-message-functions of this virtual protocol. Much like how $\Pi_{\mathcal{V}}$ can be written as the set of functions $\{\mathsf{NxtMsg}^k_j\}_{j\in[3],k\in[r]}$, the simulator $\mathsf{Sim}_{\mathcal{V}}$ can be written as,

$$\mathsf{Sim}_{\mathcal{V}} = \{\mathsf{NxtMsg}^{k*}_j\}_{j\in[3],k\in[r]}$$

Here, for the corrupted party $\mathcal{V}_j$, the functions $\{\mathsf{NxtMsg}^{k*}_j\}_{k\in[r]} = \{\mathsf{NxtMsg}^{k}_j\}_{k\in[r]}$ are as in the real protocol $\Pi_{\mathcal{V}}$. However, for each honest party $\mathcal{V}_{j'} \neq \mathcal{V}_j$, for each round $k \in [r]$, the next-message function has the same structure as the real function but produces simulated messages that do not depend on the party's input:

$$\{m^{k'}_{j'\to j''}\}_{j''\neq j'\in[3]} \leftarrow \mathsf{NxtMsg}^{k*}_{j'}(\hat{\mathbf{View}}^{k-1}_{\mathcal{V},j'})$$

Here $\hat{\mathbf{View}}^{k-1}_{\mathcal{V},j'}$ is the simulated view with all the messages that the corrupt party sends to $\mathcal{V}_{j'}$ up to round $k-1$. The set $\{m^{k'}_{j'\to j''}\}_{j''\neq j\in[3]}$ is the set of simulated messages output on behalf of $\mathcal{V}_{j'}$ to the other parties in round k.

– The real protocol Π is composed of the set of protocols $\{\Pi_{\mathsf{NxtMsg}_j^k}\}_{j\in[3],k\in[r]}$ corresponding to the $\Pi_{\mathcal{V}}$ in question. For each $j \in [3]$, protocol Π_j as described before is used to execute all next-message-function sub-protocols $\Pi_{\mathsf{NxtMsg}_j^k}$ of virtual party $\mathcal{V}_j$. Due to Theorem 4, we know that the following exist:
  - Let $\mathsf{Sim}_{\mathsf{S,GOD}}^{\mathsf{NxtMsg}_1^k}$ be the PPT simulator for the view of an unbounded malicious adversary $\mathcal{A}$ corrupting any set $Z \in \mathcal{Z}_{\mathsf{GOD}}$ in the protocols $\Pi_{\mathsf{NxtMsg}_1^k}$. This interacts with the adversary and produces a view that is statistically close to the real view of the protocol execution with guaranteed output delivery:

$$\hat{\mathbf{View}}_{\mathsf{NxtMsg}_1^k}^{Z,\mathsf{GOD}} \leftarrow \langle \mathsf{Sim}_{\mathsf{S,GOD}}^{\mathsf{NxtMsg}_1^k}(1^s), \mathcal{A}\rangle$$

$$\left\{\hat{\mathbf{View}}_{\mathsf{NxtMsg}_1^k}^{Z,\mathsf{GOD}}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n} \overset{\mathrm{s}}{\equiv} \left\{\mathbf{View}_{\mathsf{NxtMsg}_1^k}^{Z,\mathsf{GOD}}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n}$$

    If these simulators are given the outputs based on $\{[m_{1\to j'}^{k'}]_i\}_{j'\neq 1\in[3]}\}_{i\in[n]}$, as output by the simulator of the virtual protocol $\mathsf{NxtMsg}_1^{k*} \in \mathsf{Sim}_{\mathsf{S}}^{\mathcal{V}}$, the view simulated is a view that is still statistically close to that in the real execution of $\Pi_{\mathsf{NxtMsg}_1^k}$:

$$\hat{\mathbf{View}}_{\mathsf{NxtMsg}_1^{k*}}^{Z,\mathsf{GOD}} \leftarrow \langle \mathsf{Sim}_{\mathsf{GOD}}^{\mathsf{NxtMsg}_1^k}(1^s), \mathcal{A}\rangle$$

$$\left\{\hat{\mathbf{View}}_{\mathsf{NxtMsg}_1^{k*}}^{Z,\mathsf{GOD}}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n} \overset{\mathrm{s}}{\equiv} \left\{\mathbf{View}_{\mathsf{NxtMsg}_1^k}^{Z,\mathsf{GOD}}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n}$$

  - For protocols $\Pi_{\mathsf{NxtMsg}_2^k}$, let $\mathsf{Sim}_{\mathsf{S,GOD}}^{\mathsf{NxtMsg}_2^k}$ be the PPT simulator for the view of an unbounded malicious adversary $\mathcal{A}$ corrupting any set $Z \in \mathcal{Z}'_{\mathsf{GOD}} \cap \mathcal{Z}_{\mathsf{S}}$. This interacts with the adversary and produces a view that is statistically close to the real view of the protocol execution:

$$\hat{\mathbf{View}}_{\mathsf{NxtMsg}_2^k}^{Z,\mathsf{GOD}} \leftarrow \langle \mathsf{Sim}_{\mathsf{S,GOD}}^{\mathsf{NxtMsg}_2^k}(1^s), \mathcal{A}\rangle$$

$$\left\{\hat{\mathbf{View}}_{\mathsf{NxtMsg}_2^k}^{Z,\mathsf{GOD}}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n} \overset{\mathrm{s}}{\equiv} \left\{\mathbf{View}_{\mathsf{NxtMsg}_2^k}^{Z,\mathsf{GOD}}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n}$$

    Let $\mathsf{Sim}_{\mathsf{S,fair}}^{\mathsf{NxtMsg}_2^k}$ be the PPT simulator for the view of an unbounded malicious adversary $\mathcal{A}$ corrupting any set $Z \in \mathcal{Z}_{\mathsf{fair}} \cap \mathcal{Z}_{\mathsf{S}}$. This interacts with the adversary and produces a view that is statistically close to the real view of the protocol execution:

$$\hat{\mathbf{View}}_{\mathsf{NxtMsg}_2^k}^{Z,\mathsf{fair}} \leftarrow \langle \mathsf{Sim}_{\mathsf{S,fair}}^{\mathsf{NxtMsg}_2^k}(1^s), \mathcal{A}\rangle$$

$$\left\{\hat{\mathbf{View}}_{\mathsf{NxtMsg}_2^k}^{Z,\mathsf{fair}}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n} \overset{\mathrm{s}}{\equiv} \left\{\mathbf{View}_{\mathsf{NxtMsg}_2^k}^{Z,\mathsf{fair}}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n}$$

    Let $\mathsf{Sim}_{\mathsf{S},\perp}^{\mathsf{NxtMsg}_2^k}$ be the PPT simulator for the view of an unbounded malicious adversary $\mathcal{A}$ corrupting any set $Z \in \mathcal{Z}_{\mathsf{S}}$. This interacts with

the adversary and produces a view that is statistically close to the real view of the protocol execution:

$$\hat{\mathbf{View}}^{Z,\perp}_{\mathsf{NxtMsg}_2^k} \leftarrow \langle \mathsf{Sim}^{\mathsf{NxtMsg}_2^k}_{\mathsf{S},\perp}(1^s), \mathcal{A} \rangle$$

$$\left\{\hat{\mathbf{View}}^{Z,\perp}_{\mathsf{NxtMsg}_2^k}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n} \overset{\mathrm{s}}{\equiv} \left\{\mathbf{View}^{Z,\perp}_{\mathsf{NxtMsg}_2^k}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n}$$

Similar to what is stated above, if these simulators are given the outputs based on $\{[m^{k'}_{2\rightarrow j'}]_i\}_{j'\neq 2\in[3]}\}_{i\in[n]}$, as output by the simulator of the virtual protocol $\mathsf{NxtMsg}_2^{k*} \in \mathsf{Sim}^{\mathcal{V}}_{\mathsf{S}}$, the view simulated is a view that is still statistically close to that in the real execution of $\Pi_{\mathsf{NxtMsg}_2^k}$:

$$\hat{\mathbf{View}}^{Z,\mathsf{GOD}}_{\mathsf{NxtMsg}_2^{k*}} \leftarrow \langle \mathsf{Sim}^{\mathsf{NxtMsg}_2^k}_{\mathsf{S},\mathsf{GOD}}(1^s), \mathcal{A} \rangle$$

$$\left\{\hat{\mathbf{View}}^{Z,\mathsf{GOD}}_{\mathsf{NxtMsg}_2^{k*}}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n} \overset{\mathrm{s}}{\equiv} \left\{\mathbf{View}^{Z,\mathsf{GOD}}_{\mathsf{NxtMsg}_2^k}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n}$$

$$\hat{\mathbf{View}}^{Z,\mathsf{fair}}_{\mathsf{NxtMsg}_2^{k*}} \leftarrow \langle \mathsf{Sim}^{\mathsf{NxtMsg}_2^k}_{\mathsf{S},\mathsf{fair}}(1^s), \mathcal{A} \rangle$$

$$\left\{\hat{\mathbf{View}}^{Z,\mathsf{fair}}_{\mathsf{NxtMsg}_2^{k*}}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n} \overset{\mathrm{s}}{\equiv} \left\{\mathbf{View}^{Z,\mathsf{fair}}_{\mathsf{NxtMsg}_2^k}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n}$$

$$\hat{\mathbf{View}}^{Z,\perp}_{\mathsf{NxtMsg}_2^{k*}} \leftarrow \langle \mathsf{Sim}^{\mathsf{NxtMsg}_2^k}_{\mathsf{S},\perp}(1^s), \mathcal{A} \rangle$$

$$\left\{\hat{\mathbf{View}}^{Z,\perp}_{\mathsf{NxtMsg}_2^{k*}}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n} \overset{\mathrm{s}}{\equiv} \left\{\mathbf{View}^{Z,\perp}_{\mathsf{NxtMsg}_2^k}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n}$$

Let $\mathsf{Sim}^{\mathsf{NxtMsg}_2^k}_{\mathsf{C},\mathsf{GOD}}$ be the PPT simulator for the view of a PPT malicious adversary $\mathcal{A}$ corrupting any set $Z \in \mathcal{Z}'_{\mathsf{GOD}} \cap \mathcal{Z}^*_{\mathsf{C}}$. This interacts with the adversary and produces a view that is computationally indistinguishable from the real view of the protocol execution:

$$\hat{\mathbf{View}}^{Z,\mathsf{GOD}}_{\mathsf{NxtMsg}_2^k} \leftarrow \langle \mathsf{Sim}^{\mathsf{NxtMsg}_2^k}_{\mathsf{C},\mathsf{GOD}}(1^\kappa), \mathcal{A} \rangle$$

$$\left\{\hat{\mathbf{View}}^{Z,\mathsf{GOD}}_{\mathsf{NxtMsg}_2^k}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n} \overset{\mathrm{c}}{\equiv} \left\{\mathbf{View}^{Z,\mathsf{GOD}}_{\mathsf{NxtMsg}_2^k}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n}$$

Let $\mathsf{Sim}^{\mathsf{NxtMsg}_2^k}_{\mathsf{C},\mathsf{fair}}$ be the PPT simulator for the view of a PPT malicious adversary $\mathcal{A}$ corrupting any set $Z \in \mathcal{Z}_{\mathsf{fair}} \cap \mathcal{Z}^*_{\mathsf{C}}$. This interacts with the adversary and produces a view that is computationally indistinguishable from the real view of the protocol execution:

$$\hat{\mathbf{View}}^{Z,\mathsf{fair}}_{\mathsf{NxtMsg}_2^k} \leftarrow \langle \mathsf{Sim}^{\mathsf{NxtMsg}_2^k}_{\mathsf{C},\mathsf{fair}}(1^\kappa), \mathcal{A} \rangle$$

$$\left\{\hat{\mathbf{View}}^{Z,\mathsf{fair}}_{\mathsf{NxtMsg}_2^k}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n} \overset{\mathrm{c}}{\equiv} \left\{\mathbf{View}^{Z,\mathsf{fair}}_{\mathsf{NxtMsg}_2^k}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n}$$

Let $\mathsf{Sim}_{\mathsf{C},\perp}^{\mathsf{NxtMsg}_2^k}$ be the PPT simulator for the view of a PPT malicious adversary $\mathcal{A}$ corrupting any set $Z \in \mathcal{Z}_{\mathsf{C}}^*$. This interacts with the adversary and produces a view that is computationally indistinguishable from the real view of the protocol execution:

$$\hat{\mathbf{View}}_{\mathsf{NxtMsg}_2^k}^{Z,\perp} \leftarrow \langle \mathsf{Sim}_{\mathsf{C},\perp}^{\mathsf{NxtMsg}_2^k}(1^\kappa), \mathcal{A} \rangle$$

$$\left\{\hat{\mathbf{View}}_{\mathsf{NxtMsg}_2^k}^{Z,\perp}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n} \overset{\mathrm{c}}{\equiv} \left\{\mathbf{View}_{\mathsf{NxtMsg}_2^k}^{Z,\perp}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n}$$

- For protocols $\Pi_{\mathsf{NxtMsg}_3^k}$, let $\mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}^{\mathsf{NxtMsg}_3^k}$ be the PPT simulator for the view of an unbounded malicious adversary $\mathcal{A}$ corrupting any set $Z \in \mathcal{Z}_{\mathsf{GOD}}^* \cap \mathcal{Z}_{\mathsf{S}}$. This interacts with the adversary and produces a view that is statistically close to the real view of the protocol execution:

$$\hat{\mathbf{View}}_{\mathsf{NxtMsg}_3^k}^{Z,\mathsf{GOD}} \leftarrow \langle \mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}^{\mathsf{NxtMsg}_3^k}(1^s), \mathcal{A} \rangle$$

$$\left\{\hat{\mathbf{View}}_{\mathsf{NxtMsg}_3^k}^{Z,\mathsf{GOD}}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n} \overset{\mathrm{s}}{\equiv} \left\{\mathbf{View}_{\mathsf{NxtMsg}_3^k}^{Z,\mathsf{GOD}}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n}$$

Let $\mathsf{Sim}_{\mathsf{S},\mathsf{fair}}^{\mathsf{NxtMsg}_3^k}$ be the PPT simulator for the view of an unbounded malicious adversary $\mathcal{A}$ corrupting any set $Z \in \mathcal{Z}_{\mathsf{fair}} \cap \mathcal{Z}_{\mathsf{S}}$. This interacts with the adversary and produces a view that is statistically close to the real view of the protocol execution:

$$\hat{\mathbf{View}}_{\mathsf{NxtMsg}_3^k}^{Z,\mathsf{fair}} \leftarrow \langle \mathsf{Sim}_{\mathsf{S},\mathsf{fair}}^{\mathsf{NxtMsg}_3^k}(1^s), \mathcal{A} \rangle$$

$$\left\{\hat{\mathbf{View}}_{\mathsf{NxtMsg}_3^k}^{Z,\mathsf{fair}}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n} \overset{\mathrm{s}}{\equiv} \left\{\mathbf{View}_{\mathsf{NxtMsg}_3^k}^{Z,\mathsf{fair}}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n}$$

Let $\mathsf{Sim}_{\mathsf{S},\perp}^{\mathsf{NxtMsg}_3^k}$ be the PPT simulator for the view of an unbounded malicious adversary $\mathcal{A}$ corrupting any set $Z \in \mathcal{Z}_{\mathsf{S}}$. This interacts with the adversary and produces a view that is statistically close to the real view of the protocol execution:

$$\hat{\mathbf{View}}_{\mathsf{NxtMsg}_3^k}^{Z,\perp} \leftarrow \langle \mathsf{Sim}_{\mathsf{S},\perp}^{\mathsf{NxtMsg}_3^k}(1^s), \mathcal{A} \rangle$$

$$\left\{\hat{\mathbf{View}}_{\mathsf{NxtMsg}_3^k}^{Z,\perp}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n} \overset{\mathrm{s}}{\equiv} \left\{\mathbf{View}_{\mathsf{NxtMsg}_3^k}^{Z,\perp}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n}$$

If these simulators are given the outputs based on $\{[m_{3\to j'}^{k'}]_i\}_{j'\neq 3\in[3]}\}_{i\in[n]}$, as output by the simulator of the virtual protocol $\mathsf{NxtMsg}_3^{k*} \in \mathsf{Sim}_{\mathsf{S}}^{\mathcal{V}}$, the view simulated is a view that is still statistically close to that in the real execution of $\Pi_{\mathsf{NxtMsg}_3^k}$:

$$\hat{\mathbf{View}}_{\mathsf{NxtMsg}_3^{k*}}^{Z,\mathsf{GOD}} \leftarrow \langle \mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}^{\mathsf{NxtMsg}_3^k}(1^s), \mathcal{A} \rangle$$

$$\left\{\hat{\mathbf{View}}_{\mathsf{NxtMsg}_3^{k*}}^{Z,\mathsf{GOD}}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n} \overset{\mathrm{s}}{\equiv} \left\{\mathbf{View}_{\mathsf{NxtMsg}_3^k}^{Z,\mathsf{GOD}}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n}$$

$$\hat{\mathbf{View}}^{Z,\mathsf{fair}}_{\mathsf{NxtMsg}^{k*}_3} \leftarrow \langle \mathsf{Sim}^{\mathsf{NxtMsg}^k_3}_{\mathsf{S},\mathsf{fair}}(1^s), \mathcal{A}\rangle$$

$$\left\{\hat{\mathbf{View}}^{Z,\mathsf{fair}}_{\mathsf{NxtMsg}^{k*}_3}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n} \stackrel{\mathrm{s}}{\equiv} \left\{\mathbf{View}^{Z,\mathsf{fair}}_{\mathsf{NxtMsg}^k_3}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n}$$

$$\hat{\mathbf{View}}^{Z,\perp}_{\mathsf{NxtMsg}^{k*}_3} \leftarrow \langle \mathsf{Sim}^{\mathsf{NxtMsg}^k_3}_{\mathsf{S},\perp}(1^s), \mathcal{A}\rangle$$

$$\left\{\hat{\mathbf{View}}^{Z,\perp}_{\mathsf{NxtMsg}^{k*}_3}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n} \stackrel{\mathrm{s}}{\equiv} \left\{\mathbf{View}^{Z,\perp}_{\mathsf{NxtMsg}^k_3}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n}$$

Let $\mathsf{Sim}^{\mathsf{NxtMsg}^k_3}_{\mathsf{C},\mathsf{GOD}}$ be the PPT simulator for the view of a PPT malicious adversary $\mathcal{A}$ corrupting any set $Z \in \mathcal{Z}^*_{\mathsf{GOD}} \cap \mathcal{Z}^*_{\mathsf{C}}$. This interacts with the adversary and produces a view that is computationally indistinguishable from the real view of the protocol execution:

$$\hat{\mathbf{View}}^{Z,\mathsf{GOD}}_{\mathsf{NxtMsg}^k_3} \leftarrow \langle \mathsf{Sim}^{\mathsf{NxtMsg}^k_3}_{\mathsf{C},\mathsf{GOD}}(1^\kappa), \mathcal{A}\rangle$$

$$\left\{\hat{\mathbf{View}}^{Z,\mathsf{GOD}}_{\mathsf{NxtMsg}^k_3}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n} \stackrel{\mathrm{c}}{\equiv} \left\{\mathbf{View}^{Z,\mathsf{GOD}}_{\mathsf{NxtMsg}^k_3}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n}$$

Let $\mathsf{Sim}^{\mathsf{NxtMsg}^k_3}_{\mathsf{C},\mathsf{fair}}$ be the PPT simulator for the view of a PPT malicious adversary $\mathcal{A}$ corrupting any set $Z \in \mathcal{Z}_{\mathsf{fair}} \cap \mathcal{Z}^*_{\mathsf{C}}$. This interacts with the adversary and produces a view that is computationally indistinguishable from the real view of the protocol execution:

$$\hat{\mathbf{View}}^{Z,\mathsf{fair}}_{\mathsf{NxtMsg}^k_3} \leftarrow \langle \mathsf{Sim}^{\mathsf{NxtMsg}^k_3}_{\mathsf{C},\mathsf{fair}}(1^\kappa), \mathcal{A}\rangle$$

$$\left\{\hat{\mathbf{View}}^{Z,\mathsf{fair}}_{\mathsf{NxtMsg}^k_3}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n} \stackrel{\mathrm{c}}{\equiv} \left\{\mathbf{View}^{Z,\mathsf{fair}}_{\mathsf{NxtMsg}^k_3}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n}$$

Let $\mathsf{Sim}^{\mathsf{NxtMsg}^k_3}_{\mathsf{C},\perp}$ be the PPT simulator for the view of a PPT malicious adversary $\mathcal{A}$ corrupting any set $Z \in \mathcal{Z}^*_{\mathsf{C}}$. This interacts with the adversary and produces a view that is computationally indistinguishable from the real view of the protocol execution:

$$\hat{\mathbf{View}}^{Z,\perp}_{\mathsf{NxtMsg}^k_3} \leftarrow \langle \mathsf{Sim}^{\mathsf{NxtMsg}^k_3}_{\mathsf{C},\perp}(1^\kappa), \mathcal{A}\rangle$$

$$\left\{\hat{\mathbf{View}}^{Z,\perp}_{\mathsf{NxtMsg}^k_3}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n} \stackrel{\mathrm{c}}{\equiv} \left\{\mathbf{View}^{Z,\perp}_{\mathsf{NxtMsg}^k_3}\right\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n}$$

Given the above building blocks and notation, we are now ready to argue security for our compiled protocol.

GOD FOR UNBOUNDED ADVERSARIES. Let $\mathcal{A}$ be a malicious unbounded adversary corrupting a set of parties $Z \in \mathcal{Z}_{\mathsf{S}} \cap \mathcal{Z}_{\mathsf{GOD}}$. Consider the following simulator $\mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}$ that makes black-box calls to simulators in $\{\mathsf{Sim}^{\mathsf{NxtMsg}^k_j}_{\mathsf{S},\mathsf{GOD}}\}_{j\in[3],k\in[r]}$ and simulates the view of $\mathcal{A}$ in the ideal world for guaranteed output delivery:

1. $\mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}$ has as input the security parameter 1^s.

2. In the 'Initialize Shared State' phase, on behalf of all the honest parties $P_k \in \mathcal{P} - Z$, it samples inputs x_k and randomness r_k. It then generates a 2-out-of-3-secret-sharing of (x_k, r_k). Then, letting $\mathcal{Z} = \mathcal{Z}^*_{\mathsf{C}} \cup \mathcal{Z}_{\mathsf{GOD}}$, it computes $\mathcal{Z}$-secret-sharings of each of these shares.
3. It sends $\{[\{[x_k]_j, [r_k]_j\}_{k\in\mathcal{P}-Z}]_i\}_{i\in Z, j\in[3]}$ to the adversary and receives from it $\{[\{[x_k]_j, [r_k]_j\}_{k\in Z}]_i\}_{i\in\mathcal{P}-Z, j\in[3]}$ with which it completes $\{[\hat{\mathbf{View}}^0_{\Pi_{\mathcal{V}},j}]_i\}_{i\in\mathcal{P}-Z,j\in[3]}$.
4. If $Z \in \mathcal{Z}'_{\mathsf{GOD}}$, set $j = 2$ and otherwise set $j = 3$. Then for the virtual protocol, the simulator corresponding to the case where $\mathcal{V}_j$ is fail-stop corrupted is executed to derive the view. For this simulator $\mathsf{Sim}_{\mathcal{V}}$, as stated before, parse this as a set of next-message functions: one for each party, for each round.

$$\mathsf{Sim}_{\mathcal{V}} = \{\mathsf{NxtMsg}^{k*}_{j'}\}_{j'\in[3],k\in[r]}$$

5. In the 'Compute Virtual Protocol' phase, for each virtual party $V_{j'} \in \mathcal{V}$ and each round $k \in [r]$ of the virtual protocol, for the next-message protocol $\Pi_{\mathsf{NxtMsg}^k_{j'}}$, $\mathsf{Sim}_{\mathsf{S,GOD}}$ makes a black-box call to the PPT simulator $\mathsf{Sim}^{\mathsf{NxtMsg}^k_{j'}}_{\mathsf{S,GOD}}$ with input 1^s and interacts with $\mathcal{A}$.
   - In this interaction, $\mathsf{Sim}^{\mathsf{NxtMsg}^k_{j'}}_{\mathsf{S,GOD}}$ extracts the adversary's inputs to $\Pi_{\mathsf{NxtMsg}^k_{j'}}$ to give to the ideal functionality computing $\mathcal{F}_{\mathsf{NxtMsg}^k_{j'}}$
   - The simulator $\mathsf{Sim}_{\mathsf{S,GOD}}$, emulating this ideal functionality, accepts these extracted inputs from $\mathsf{Sim}^{\mathsf{NxtMsg}^k_{j'}}_{\mathsf{S,GOD}}$.
   - $\mathsf{Sim}_{\mathsf{S,GOD}}$ computes the outputs of the simulated next-message functionality $\mathsf{NxtMsg}^{k*}_{j'}$ using the extracted inputs, the view generated from the inputs and randomness that it has sampled on behalf of the honest parties in the 'Initialize Shared State' phase.
     - If this next-message protocol corresponds to the last round of the virtual protocol ($k = r$), then $\mathsf{Sim}_{\mathsf{S,GOD}}$ sends the adversary's inputs, that it extracted from the views received, to the ideal functionality $\mathcal{F}$ and receives $f(x)$.
     - $\mathsf{Sim}_{\mathsf{S,GOD}}$ uses this $f(x)$ to compute the outputs of this simulated next-message-functionality.
   - When $\Pi_{\mathsf{NxtMsg}^k_{j'}}$ executes to completion, $\mathsf{Sim}_{\mathsf{S,GOD}}$, emulating $\mathcal{F}_{\mathsf{NxtMsg}^k_{j'}}$, passes the outputs to $\mathsf{Sim}^{\mathsf{NxtMsg}^k_{j'}}_{\mathsf{S,GOD}}$ for it to complete the emulation.
   - This simulates the view of the corrupt parties in a way that is statistically close the real view. Let this view be,

$$\hat{\mathbf{View}}^{Z,\mathsf{GOD}}_{\mathsf{NxtMsg}^{k*}_{j'}} \leftarrow \langle \mathsf{Sim}^{\mathsf{NxtMsg}^{k*}_{j'}}_{\mathsf{S,GOD}}(1^s), \mathcal{A}\rangle$$

6. Finally, $\mathsf{Sim}_{\mathsf{S,GOD}}$ gives the adversary shares of the output such that when combined with the shares that it derives from its own views, they reconstruct to $\{f(\vec{x}) = f_i(\vec{x})\}_{P_i\in Z}$

This completes the simulation. Note that the above ideal-world execution does indeed run to completion because all next-message protocols except those corresponding to $\mathcal{V}_j$ are secure with GOD. The rest of the next-message protocols run

to completion with correctness and the view of the adversary in this simulation is identical to that in the real execution. The complete view of the adversary in the above execution of $\mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}$ is distributed as,

$$\left\{ \{x_i\}_{P_i \in Z}, f(\vec{x}), \{r_i\}_{P_i \in Z}, \left\{ \hat{\mathbf{View}}^{Z,\mathsf{GOD}}_{\mathsf{NxtMsg}^{k*}_{j'}} \right\}_{j' \in [3], k \in [r], P_i \notin Z}, \left\{ \mathbf{View}^{Z,\mathsf{GOD}}_{\mathsf{NxtMsg}^{k}_{j'}} \right\}_{j' \in [3], k \in [r], P_i \in Z} \right\}_{\kappa \in \mathbb{N}, \vec{r} \in \mathcal{R}^n}$$

It remains to argue that the above view is statistically close to the real view of the adversary. For this, consider the following hybrids:

- Hybrid H_0. This is the distribution of the output of $\mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}$.

$$\mathsf{H}_0 = \left\{ \{x_i\}_{P_i \in Z}, f(\vec{x}), \{r_i\}_{P_i \in Z}, \left\{ \hat{\mathbf{View}}^{Z,\mathsf{GOD}}_{\mathsf{NxtMsg}^{k*}_{j'}} \right\}_{j' \in [3], k \in [r], P_i \notin Z}, \left\{ \mathbf{View}^{Z,\mathsf{GOD}}_{\mathsf{NxtMsg}^{k}_{j'}} \right\}_{j' \in [3], k \in [r], P_i \in Z} \right\}_{\kappa \in \mathbb{N}, \vec{r} \in \mathcal{R}^n}$$

- Hybrid H_1. An intermediate hybrid distribution where, in the 'Initialize Shared State' phase, on behalf of all honest parties, the inputs and randomness for the real execution are used to create the initial view. The rest of the adversary's view beyond this is derived as in the simulation $\mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}$ but from this initial view:

$$\mathsf{H}_1 = \left\{ \{x_i\}_{P_i \in Z}, f(\vec{x}), \{r_i\}_{P_i \in Z}, \left\{ \hat{\mathbf{View}}^{Z,\mathsf{GOD}'}_{\mathsf{NxtMsg}^{k*}_{j'}} \right\}_{j \in [3], k \in [r], P_i \notin Z}, \left\{ \mathbf{View}^{Z,\mathsf{GOD}}_{\mathsf{NxtMsg}^{k}_{j'}} \right\}_{j \in [3], k \in [r], P_i \in Z} \right\}_{\kappa \in \mathbb{N}, \vec{r} \in \mathcal{R}^n}$$

  The difference between this distribution and H_0 stems from the fact that in the adversary's view, all the secret-shares are of the real input of the honest parties, instead of values picked arbitrarily by $\mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}$.
- Hybrid H_2. An intermediate hybrid distribution with the simulated virtual protocol $\mathsf{Sim}_{\mathcal{V}}$ for the view of $\mathcal{V}_j$ but with each protocol execution among the real parties $\mathcal{P}$ being a real execution:

$$\mathsf{H}_2 = \left\{ \{x_i\}_{P_i \in Z}, f(\vec{x}), \{r_i\}_{P_i \in Z}, \left\{ \mathbf{View}^{Z,\mathsf{GOD}'}_{\mathsf{NxtMsg}^{k*}_{j'}} \right\}_{j \in [3], k \in [r], P_i \notin Z}, \left\{ \mathbf{View}^{Z,\mathsf{GOD}}_{\mathsf{NxtMsg}^{k}_{j'}} \right\}_{j \in [3], k \in [r], P_i \in Z} \right\}_{\kappa \in \mathbb{N}, \vec{r} \in \mathcal{R}^n}$$

  This differs from H_2 in that the view of real protocol corresponding to the next-message functions of the simulator of the virtual protocol $\mathsf{Sim}_{\mathcal{V}}$ is generated as in a real execution of the protocol instead of using the simulation.

– Hybrid H_3. The distribution of the view of the parties in Z in a real execution of the protocol in Figure 9.

$$\mathsf{H}_3 = \left\{ \{x_i\}_{P_i \in Z}, f(\vec{x}), \{r_i\}_{P_i \in Z}, \left\{ \mathbf{View}^{Z,\mathsf{GOD}}_{\mathsf{NxtMsg}^k_{j'}} \right\}_{j \in [3], k \in [r], i \in [n]} \right\}_{\kappa \in \mathbb{N}, \vec{r} \in \mathcal{R}^n}$$

Claim. The two hybrid distributions H_0 and H_1 are identically distributed.

$$\mathsf{H}_0 \equiv \mathsf{H}_1$$

Proof. Let $\overline{Z} \subseteq \mathcal{P}$ denote the set of honest parties in this protocol execution. The difference between the hybrid distributions H_0 and H_1 stems from the fact that, in the adversary's view, in the 'Initialize Shared State' phase, the shares are created differently.

In the simulated view H_0, on behalf of all honest parties, the randomness to be used in the virtual protocol is sampled uniformly at random and the inputs are arbitrarily picked by the simulator $\mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}$. The secret-shares are generated on these values and these shared initial states are used as the basis of all views (of the virtual simulated next-message-functions) generated throughout the protocol. In hybrid H_1, all of this is generated in the same way, but from the input and randomness of the honest parties as in the real execution.

Firstly, note that the randomness of the honest parties are sampled in the same way (uniformly at random from the domain) in both hybrid distributions. Therefore any difference between these hybrids needs to stem from the fact that the inputs used are different. Secondly, note that all intermediate views in the virtual protocol are secret-shared among all n parties using a $\mathcal{Z}$-secret-sharing scheme. So the shares in the adversary's view of the view of *any* virtual party reveals no information about the actual view that is shared. Hence,

$$\mathsf{H}_0 \equiv \mathsf{H}_1$$

Claim. For all $j' \neq j \in [3], k \in [r]$, assuming that the protocol $\Pi_{\mathsf{NxtMsg}^k_{j'}}$ is secure with guaranteed output delivery in the presence of an unbounded adversary corrupting any subset $Z \in \mathcal{Z}_{\mathsf{GOD}} \cap \mathcal{Z}_{\mathsf{S}}$ of the parties, the view in the hybrid distributions H_1 and H_2 are statistically close.

Proof. The proof for the above claim follows from considering the following set of $2r + 1$ hybrids:

– Hybrid $\mathsf{H}'_{0,0}$: this distribution is the same as that of hybrid H_1 where the view in the 'Compute Virtual Protocol' phase is generated as in the simulation $\mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}$.
– $\forall j' \neq j \in [3], k \in [r], \mathsf{H}'_{j',k}$: in this distribution, up to the j', k^{th} next-message protocol $\Pi_{\mathsf{NxtMsg}^k_{j'}}$, the view is generated exactly as in the real execution. Beyond this, each k^{th} protocol emulation view is generated as in the simulation:

$$\mathbf{View}^{Z,\mathsf{GOD}'}_{\mathsf{NxtMsg}^{k*}_{j'}} \leftarrow \langle \mathsf{Sim}^{\mathsf{NxtMsg}^k_{j'}}_{\mathsf{S},\mathsf{GOD}}(1^s), \mathcal{A} \rangle$$

– The last of these hybrids $\mathsf{H}'_{2,r}$ is distributed the same as H_2.

It remains to show that for each $j' \neq j \in [3], k \in [r]$, the adjacent hybrid distributions $\mathsf{H}'_{j',k}$ and $\mathsf{H}'_{j',k-1}$ are statistically close. Note that each such set of adjacent hybrids only differ in the view of the $(j',k)^{th}$ execution $\Pi_{\mathsf{NxtMsg}^k_{j'}}$. In hybrid $\mathsf{H}'_{j',k}$, this is as in the real protocol execution, while in $\mathsf{H}'_{j',k-1}$ this is a simulated view generated using an interaction with $\mathsf{Sim}^{\mathsf{NxtMsg}^k_{j'}}_{\mathsf{S},\mathsf{GOD}}$. For the protocol $\Pi_{\mathsf{NxtMsg}^k_{j'}}$, the real view for the adversary and that generated using $\mathsf{Sim}^{\mathsf{NxtMsg}^k_{j'}}_{\mathsf{S},\mathsf{GOD}}$ are statistically close with distance ϵ. Since this is the only difference between these adjacent hybrids, it follows that the complete hybrids are also statistically close. Hence, the hybrid distributions H_1 and H_2 are also at most $2r\epsilon$ apart.

Claim. Assuming that the virtual protocol $\Pi_{\mathcal{V}}$ is fail-stop secure in the presence of an unbounded adversary corrupting either $\mathcal{V}_2$ or $\mathcal{V}_3$, the view in the hybrid distributions H_2 and H_3 are statistically close.

Proof. The difference between the hybrid distributions H_2 and H_3 stems only from the fact that, in the adversary's view, in the 'Compute Virtual Protocol' phase, the views of the parties from the virtual protocol (and the shares thereof) are created differently. In hybrid H_2, these are views generated from the virtual protocol simulator $\mathsf{Sim}_{\mathcal{V}}$, while in H_3, these are distributed as views of the real execution of $\Pi_{\mathcal{V}}$. It therefore follows that if the distribution of the simulated and real view of $\Pi_{\mathcal{V}}$ are statistically close, so are the hybrids H_2 and H_3.

Fairness for Unbounded Adversaries. Let $\mathcal{A}$ be a malicious unbounded adversary corrupting a set of parties $Z \in \mathcal{Z}_{\mathsf{S}} \cap \mathcal{Z}^*_{\mathsf{fair}}$. Note that since $\mathcal{Z}^*_{\mathsf{fair}} = \mathcal{Z}_{\mathsf{fair}} \cup \mathcal{Z}_{\mathsf{GOD}}$, and we already have from a previous case that GOD, a stronger guarantee than fairness, holds for all corruption within $\mathcal{Z}_{\mathsf{S}} \cap \mathcal{Z}_{\mathsf{GOD}}$, it only remains to argue that fairness holds for any $Z \in \mathcal{Z}_{\mathsf{S}} \cap \mathcal{Z}_{\mathsf{fair}}$. Consider the following PPT simulator $\mathsf{Sim}_{\mathsf{S},\mathsf{fair}}$ that makes black-box calls to simulators in $\{\mathsf{Sim}^{\mathsf{NxtMsg}^k_{j'}}_{\mathsf{S},\mathsf{fair}}\}_{j' \in \{2,3\}, k \in [r]}$ and simulates the view of $\mathcal{A}$ in the ideal world for fairness. This simulator operates in the same way as $\mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}$ would, but with the following differences:

1. During the phase to 'compute virtual protocol', $\mathsf{Sim}_{\mathsf{S},\mathsf{fair}}$ makes black-box calls to $\mathsf{Sim}^{\mathsf{NxtMsg}^k_{j'}}_{\mathsf{S},\mathsf{fair}}$ instead of to $\mathsf{Sim}^{\mathsf{NxtMsg}^k_{j'}}_{\mathsf{S},\mathsf{GOD}}$.
2. Furthermore, if any of these simulation interactions abort, $\mathsf{Sim}_{\mathsf{S},\mathsf{fair}}$ sends $\perp$ to the ideal functionality $\mathcal{F}$ and terminates execution. The view generated up to this point is output by the simulator.
3. If *all* calls to $\mathsf{Sim}^{\mathsf{NxtMsg}^k_{j'}}_{\mathsf{S},\mathsf{fair}}$ execute to completion, $\mathsf{Sim}_{\mathsf{S},\mathsf{fair}}$ sends the inputs of the adversary $\mathcal{A}$ extracted from the views to $\mathcal{F}$ and receives $f(x)$. It uses this to complete the protocol execution in the last round of next-message-function calls of the virtual protocol $\Pi_{\mathcal{V}}$.

Note that the adversary causes the protocol to abort in the ideal execution, iff the protocol also aborts in the real execution. The view of the protocol up to that point was generated without the output of the functionality and so it follows that, along with upholding the privacy of honest parties' inputs, both the adversary and the honest parties don't learn the output. If the protocol executes to completion, the honest parties always have enough shares to reconstruct the output of the functionality. In this case, the view of the adversary in the above execution of $\mathsf{Sim}_{\mathsf{S},\mathsf{fair}}$ is distributed as,

$$\left\{ \{x_i\}_{P_i \in Z}, f(\vec{x}), \{r_i\}_{P_i \in Z}, \left\{ \hat{\mathbf{View}}^{Z,\mathsf{fair}}_{\mathsf{NxtMsg}^{k*}_{j'}} \right\}_{j' \in [3], k \in [r], P_i \notin Z}, \left\{ \mathbf{View}^{Z,\mathsf{fair}}_{\mathsf{NxtMsg}^{k}_{j'}} \right\}_{j' \in [3], k \in [r], P_i \in Z} \right\}_{\kappa \in \mathbb{N}, \vec{r} \in \mathcal{R}^n}$$

This distribution can be argued as statistically close to the real view of the adversary by considering a similar set of hybrids as in the GOD case, with the only difference being that the argument would now follow from the fact that the views output from $\{\mathsf{Sim}^{\mathsf{NxtMsg}^{k}_{j'}}_{\mathsf{S},\mathsf{fair}}\}_{j' \in [3], k \in [r]}$ (as opposed to $\{\mathsf{Sim}^{\mathsf{NxtMsg}^{k}_{j'}}_{\mathsf{S},\mathsf{GOD}}\}_{j' \in [3], k \in [r]}$) are statistically close to the real views of the protocol execution.

Security with Abort for Unbounded Adversaries. Let $\mathcal{A}$ be a malicious unbounded adversary corrupting a set of parties $Z \in \mathcal{Z}_{\mathsf{S}}$, in particular a set not in $\mathcal{Z}^*_{\mathsf{fair}}$. Consider the following PPT simulator $\mathsf{Sim}_{\mathsf{S},\perp}$ that makes black-box calls to simulators in $\{\mathsf{Sim}^{\mathsf{NxtMsg}^{k}_{j'}}_{\mathsf{S},\perp}\}_{j' \in \{2,3\}, k \in [r]}$ and simulates the view of $\mathcal{A}$ in the ideal world for security with abort. This simulator operates in the same way as $\mathsf{Sim}_{\mathsf{S},\mathsf{fair}}$ would, but with the following differences:

1. If *all* calls to $\mathsf{Sim}^{\mathsf{NxtMsg}^{k}_{j'}}_{\mathsf{S},\perp}$ execute to completion, $\mathsf{Sim}_{\mathsf{S},\perp}$, like $\mathsf{Sim}_{\mathsf{S},\mathsf{fair}}$, sends the inputs of the adversary $\mathcal{A}$ extracted from the views to $\mathcal{F}$ and receives $f(x)$. It uses this to complete the protocol execution in the last round of next-message-function calls of the virtual protocol $\Pi_{\mathcal{V}}$.
2. If the set of parties corrupted by the adversary $Z \notin \mathcal{Z}^*_{\mathsf{fair}}$, $\mathsf{Sim}_{\mathsf{S},\perp}$ sends $\perp$ to the ideal functionality $\mathcal{F}$ and $\mathcal{F}$ sends $\perp$ to all honest parties. Otherwise, $\mathsf{Sim}_{\mathsf{S},\perp}$ sends $\mathsf{continue}$ to $\mathcal{F}$ and the honest parties receive the outputs.

Note that the adversary causes the protocol to abort in the ideal execution, iff the protocol also aborts in the real execution. If the protocol executes to completion, and there are not enough honest parties to reconstruct the function output, the protocol would abort with only the adversary receiving the function output. In this case, the view of the adversary in the above execution of $\mathsf{Sim}_{\mathsf{S},\perp}$ is distributed as,

$$\left\{ \{x_i\}_{P_i \in Z}, f(\vec{x}), \{r_i\}_{P_i \in Z}, \left\{ \hat{\mathbf{View}}^{Z,\perp}_{\mathsf{NxtMsg}^{k*}_{j'}} \right\}_{j' \in [3], k \in [r], P_i \notin Z}, \left\{ \mathbf{View}^{Z,\perp}_{\mathsf{NxtMsg}^{k}_{j'}} \right\}_{j' \in [3], k \in [r], P_i \in Z} \right\}_{\kappa \in \mathbb{N}, \vec{r} \in \mathcal{R}^n}$$

This distribution can be argued as statistically close to the real view of the adversary by considering the exact same set of hybrids as in the fairness case.

GOD FOR PPT ADVERSARIES. Let $\mathcal{A}$ be a malicious unbounded adversary corrupting a set of parties $Z \in \mathcal{Z}_{\mathsf{GOD}}$, in particular, a set not contained in $\mathcal{Z}_{\mathsf{S}}$. Consider the following simulator $\mathsf{Sim}_{\mathsf{C,GOD}}$ that makes black-box calls to simulators in $\{\mathsf{Sim}_{\mathsf{C,GOD}}^{\mathsf{NxtMsg}_j^k}\}_{j\in[3],k\in[r]}$ and simulates the view of $\mathcal{A}$ in the ideal world for guaranteed output delivery:

1. $\mathsf{Sim}_{\mathsf{C,GOD}}$ has as input the security parameter 1^κ.
2. In the 'Initialize Shared State' phase, on behalf of all the honest parties $P_k \in \mathcal{P} - Z$, it samples inputs x_k and randomness r_k. It then generates a 2-out-of-3-secret-sharing of (x_k, r_k). Then, letting $\mathcal{Z} = \mathcal{Z}^*\mathsf{C} \cup \mathcal{Z}_{\mathsf{GOD}}$, it computes $\mathcal{Z}$-secret-sharings of each of these shares.
3. It sends $\{[\{[x_k]_j, [r_k]_j\}_{k\in\mathcal{P}-Z}]_i\}_{i\in Z, j\in[3]}$ to the adversary and receives from it $\{[\{[x_k]_j, [r_k]_j\}_{k\in Z}]_i\}_{i\in\mathcal{P}-Z,j\in[3]}$ with which it completes $\{[\hat{\mathbf{View}}^0_{\Pi_\mathcal{V},j}]_i\}_{i\in\mathcal{P}-Z,j\in[3]}$.
4. If $Z \in \mathcal{Z}'_{\mathsf{GOD}}$, set $j = 2$ and otherwise set $j = 3$. Then for the virtual protocol, the simulator corresponding to the case where $\mathcal{V}_j$ is fail-stop corrupted is executed to derive the view. For this simulator $\mathsf{Sim}_\mathcal{V}$, as stated before, parse this as a set of next-message functions: one for each party, for each round.

$$\mathsf{Sim}_\mathcal{V} = \{\mathsf{NxtMsg}_{j'}^{k*}\}_{j'\in[3],k\in[r]}$$

5. In the 'Compute Virtual Protocol' phase, for each virtual party $V_{j'} \in \mathcal{V}$ and each round $k \in [r]$ of the virtual protocol, for the next-message protocol $\Pi_{\mathsf{NxtMsg}_{j'}^k}$, $\mathsf{Sim}_{\mathsf{C,GOD}}$ makes a black-box call to the PPT simulator $\mathsf{Sim}_{\mathsf{C,GOD}}^{\mathsf{NxtMsg}_{j'}^k}$ with input 1^κ (or for $j' = 1$, to $\mathsf{Sim}_{\mathsf{S,GOD}}^{\mathsf{NxtMsg}_1^k}$ with input 1^s) and interacts with $\mathcal{A}$.
   - In this interaction, $\mathsf{Sim}_{\mathsf{C,GOD}}^{\mathsf{NxtMsg}_{j'}^k}$ extracts the adversary's inputs to $\Pi_{\mathsf{NxtMsg}_{j'}^k}$ to give to the ideal functionality computing $\mathcal{F}_{\mathsf{NxtMsg}_{j'}^k}$
   - The simulator $\mathsf{Sim}_{\mathsf{C,GOD}}$, emulating this ideal functionality, accepts these extracted inputs from $\mathsf{Sim}_{\mathsf{C,GOD}}^{\mathsf{NxtMsg}_{j'}^k}$.
   - $\mathsf{Sim}_{\mathsf{C,GOD}}$ computes the outputs of the simulated next-message functionality $\mathsf{NxtMsg}_{j'}^{k*}$ using the extracted inputs, the view generated from the inputs and randomness that it has sampled on behalf of the honest parties in the 'Initialize Shared State' phase.
     - If this next-message protocol corresponds to the last round of the virtual protocol ($k = r$), then $\mathsf{Sim}_{\mathsf{C,GOD}}$ sends the adversary's inputs, that it extracted from the views received, to the ideal functionality $\mathcal{F}$ and receives $f(x)$.
     - $\mathsf{Sim}_{\mathsf{C,GOD}}$ uses this $f(x)$ to compute the outputs of this simulated next-message-functionality.

– When $\Pi_{\mathsf{NxtMsg}^k_{j'}}$ executes to completion, $\mathsf{Sim}_{\mathsf{C,GOD}}$, emulating $\mathcal{F}_{\mathsf{NxtMsg}^k_{j'}}$, passes these computed outputs to $\mathsf{Sim}^{\mathsf{NxtMsg}^k_{j'}}_{\mathsf{C,GOD}}$ for it to complete the emulation.
– This simulates the view of the corrupt parties in a way that is computationally indistinguishable from the real view. Let this view be,

$$\hat{\mathbf{View}}^{Z,\mathsf{GOD}}_{\mathsf{NxtMsg}^{k*}_{j'}} \leftarrow \langle \mathsf{Sim}^{\mathsf{NxtMsg}^{k*}_{j'}}_{\mathsf{C,GOD}}(1^\kappa), \mathcal{A} \rangle$$

6. Finally, $\mathsf{Sim}_{\mathsf{C,GOD}}$ gives the adversary shares of the output such that when combined with the shares that it derives from its own views, they reconstruct to $\{f(\vec{x}) = f_i(\vec{x})\}_{P_i \in Z}$

This completes the simulation. Note that the above ideal-world execution does indeed run to completion because all next-message protocols except those corresponding to $\mathcal{V}_j$ are secure with GOD. The rest of the next-message protocols run to completion with correctness and the view of the adversary in this simulation is identical to that in the real execution. The complete view of the adversary in the above execution of $\mathsf{Sim}_{\mathsf{C,GOD}}$ is distributed as,

$$\left\{ \{x_i\}_{P_i \in Z}, f(\vec{x}), \{r_i\}_{P_i \in Z}, \left\{ \hat{\mathbf{View}}^{Z,\mathsf{GOD}}_{\mathsf{NxtMsg}^{k*}_{j'}} \right\}_{j' \in [3], k \in [r], P_i \notin Z}, \left\{ \mathbf{View}^{Z,\mathsf{GOD}}_{\mathsf{NxtMsg}^{k}_{j'}} \right\}_{j' \in [3], k \in [r], P_i \in Z} \right\}_{\kappa \in \mathbb{N}, \vec{r} \in \mathcal{R}^n}$$

It remains to argue that the above view is computationally indistinguishable from the real view of the adversary. For this, consider the following hybrids:

– Hybrid H_0. This is the distribution of the output of $\mathsf{Sim}_{\mathsf{C,GOD}}$.

$$\mathsf{H}_0 = \left\{ \{x_i\}_{P_i \in Z}, f(\vec{x}), \{r_i\}_{P_i \in Z}, \left\{ \hat{\mathbf{View}}^{Z,\mathsf{GOD}}_{\mathsf{NxtMsg}^{k*}_{j'}} \right\}_{j' \in [3], k \in [r], P_i \notin Z}, \left\{ \mathbf{View}^{Z,\mathsf{GOD}}_{\mathsf{NxtMsg}^{k}_{j'}} \right\}_{j' \in [3], k \in [r], P_i \in Z} \right\}_{\kappa \in \mathbb{N}, \vec{r} \in \mathcal{R}^n}$$

– Hybrid H_1. An intermediate hybrid distribution where, in the 'Initialize Shared State' phase, on behalf of all honest parties, the inputs and randomness for the real execution are used to create the initial view. The rest of the adversary's view beyond this is derived as in the simulation $\mathsf{Sim}_{\mathsf{C,GOD}}$ but from this initial view:

$$\mathsf{H}_1 = \left\{ \{x_i\}_{P_i \in Z}, f(\vec{x}), \{r_i\}_{P_i \in Z}, \left\{ \hat{\mathbf{View}}^{Z,\mathsf{GOD}'}_{\mathsf{NxtMsg}^{k*}_{j'}} \right\}_{j \in [3], k \in [r], P_i \notin Z}, \left\{ \mathbf{View}^{Z,\mathsf{GOD}}_{\mathsf{NxtMsg}^{k}_{j'}} \right\}_{j \in [3], k \in [r], P_i \in Z} \right\}_{\kappa \in \mathbb{N}, \vec{r} \in \mathcal{R}^n}$$

The difference between this distribution and H_0 stems from the fact that in the adversary's view, all the secret-shares are of the real input of the honest parties, instead of values picked arbitrarily by $\mathsf{Sim}_{\mathsf{C,GOD}}$.

– Hybrid H_2. An intermediate hybrid distribution with the simulated virtual protocol $\mathsf{Sim}_{\mathcal{V}}$ for the view of $\mathcal{V}_j$ but with each protocol execution among the real parties $\mathcal{P}$ being a real execution:

$$\mathsf{H}_2 = \left\{ \{x_i\}_{P_i \in Z}, f(\vec{x}), \{r_i\}_{P_i \in Z}, \left\{ \mathbf{View}^{Z,\mathsf{GOD}'}_{\mathsf{NxtMsg}^{k*}_{j'}} \right\}_{j \in [3], k \in [r], P_i \notin Z}, \left\{ \mathbf{View}^{Z,\mathsf{GOD}}_{\mathsf{NxtMsg}^{k}_{j'}} \right\}_{j \in [3], k \in [r], P_i \in Z} \right\}_{\kappa \in \mathbb{N}, \vec{r} \in \mathcal{R}^n}$$

This differs from H_2 in that the view of real protocol corresponding to the next-message functions of the simulator of the virtual protocol $\mathsf{Sim}_{\mathcal{V}}$ is generated as in a real execution of the protocol instead of using the simulation.

– Hybrid H_3. The distribution of the view of the parties in Z in a real execution of the protocol in Figure 9.

$$\mathsf{H}_3 = \left\{ \{x_i\}_{P_i \in Z}, f(\vec{x}), \{r_i\}_{P_i \in Z}, \left\{ \mathbf{View}^{Z,\mathsf{GOD}}_{\mathsf{NxtMsg}^{k}_{j'}} \right\}_{j \in [3], k \in [r], i \in [n]} \right\}_{\kappa \in \mathbb{N}, \vec{r} \in \mathcal{R}^n}$$

Claim. The two hybrid distributions H_0 and H_1 are identically distributed.

$$\mathsf{H}_0 \equiv \mathsf{H}_1$$

Proof. Let $\overline{Z} \subseteq \mathcal{P}$ denote the set of honest parties in this protocol execution. The difference between the hybrid distributions H_0 and H_1 stems from the fact that, in the adversary's view, in the 'Initialize Shared State' phase, the shares are created differently.

In the simulated view H_0, on behalf of all honest parties, the randomness to be used in the virtual protocol is sampled uniformly at random and the inputs are arbitrarily picked by the simulator $\mathsf{Sim}_{\mathsf{S},\mathsf{GOD}}$. The secret-shares are generated on these values and these shared initial states are used as the basis of all views (of the virtual simulated next-message-functions) generated throughout the protocol. In hybrid H_1, all of this is generated in the same way, but from the input and randomness of the honest parties as in the real execution.

Firstly, note that the randomness of the honest parties are sampled in the same way (uniformly at random from the domain) in both hybrid distributions. Therefore any difference between these hybrids needs to stem from the fact that the inputs used are different. Secondly, note that all intermediate views in the virtual protocol are secret-shared among all n parties using a $\mathcal{Z}$-secret-sharing scheme. So the shares in the adversary's view of the view of *any* virtual party reveals no information about the actual view that is shared. Hence,

$$\mathsf{H}_0 \equiv \mathsf{H}_1$$

Claim. For all $j' \neq j \in [3], k \in [r]$, assuming that the protocol $\Pi_{\mathsf{NxtMsg}^{k}_{j'}}$ is secure with guaranteed output delivery in the presence of a PPT adversary corrupting any subset $Z \in \mathcal{Z}_{\mathsf{GOD}}$ of the parties, the view in the hybrid distributions H_1 and H_2 are computationally indistinguishable.

Proof. The proof for the above claim follows from considering the following set of $2r + 1$ hybrids:

- Hybrid $\mathsf{H}'_{0,0}$: this distribution is the same as that of hybrid H_1 where the view in the 'Compute Virtual Protocol' phase is generated as in the simulation $\mathsf{Sim}_{\mathsf{C,GOD}}$.
- $\forall j' \neq j \in [3], k \in [r], \mathsf{H}'_{j',k}$: in this distribution, up to the j', k^{th} next-message protocol $\Pi_{\mathsf{NxtMsg}^k_{j'}}$, the view is generated exactly as in the real execution. Beyond this, each k^{th} protocol emulation view is generated as in the simulation:
$$\mathbf{View}^{Z,\mathsf{GOD}'}_{\mathsf{NxtMsg}^{k*}_{j'}} \leftarrow \langle \mathsf{Sim}^{\mathsf{NxtMsg}^k_{j'}}_{\mathsf{C,GOD}}(1^\kappa), \mathcal{A} \rangle$$
- The last of these hybrids $\mathsf{H}'_{2,r}$ is distributed the same as H_2.

Note that in the hybrid distributions described above, each pair of adjacent hybrids $\mathsf{H}'_{j',k}$ and $\mathsf{H}'_{j',k-1}$ (similarly, each pair $\mathsf{H}'_{j',r}$ and $\mathsf{H}'_{j'+1,1}$) differ only the view of the execution of the sub-protocol $\Pi_{\mathsf{NxtMsg}^k_{j'}}$. In $\mathsf{H}'_{j',k-1}$, this view is the simulated view $\hat{\mathbf{View}}^{*Z,\mathsf{GOD}}_{\mathsf{NxtMsg}^k_{j'}}$ output by $\mathsf{Sim}^{\mathsf{NxtMsg}^k_{j'}}_{\mathsf{C,GOD}}$. In $\mathsf{H}'_{j',k}$ this is the view of the parties in Z in the real execution of $\Pi_{\mathsf{NxtMsg}^k_{j'}}$.

We show that if there existed a PPT distinguisher D that can distinguish between the hybrid distributions $\mathsf{H}'_{j',k}$ and $\mathsf{H}'_{j',k-1}$ with non-negligible advantage ϵ, then D can be used in a black-box way by a PPT adversary $\mathcal{A}$ that distinguishes between the distributions $\{\hat{\mathbf{View}}^{*Z,\mathsf{GOD}}_{\mathsf{NxtMsg}^k_{j'}}\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n}$ from the output of $\mathsf{Sim}^{\mathsf{NxtMsg}^k_{j'}}_{\mathsf{C,GOD}}$ and $\{\mathbf{View}^{Z,\mathsf{GOD}}_{\mathsf{NxtMsg}^k_{j'}}\}_{\kappa\in\mathbb{N},\vec{r}\in\mathcal{R}^n}$ from the real execution of the sub-protocol. The adversary $\mathcal{A}$ works as follows:

- $\mathcal{A}$ has the indices (j', k), the set of corrupted parties $Z \in \mathcal{Z}_{\mathsf{GOD}}$ and $(\vec{x}, f)$. It samples randomness $\vec{r} \leftarrow \mathcal{R}^n$ for all parties.
- For the next message functions $\mathsf{NxtMsg}^{k'}_{j''}$ of $\Pi_{\mathcal{V}}$ where $j'' < j'$ or $j'' = j'$ and $k' < k$, it creates the view as in the real protocol: $\mathbf{View}^{Z,\mathsf{GOD}}_{\mathsf{NxtMsg}^{k'}_{j''}}$. For the instances where $j'' > j'$ or $k' = k$ and $k' > k$, it creates the view as in the simulation $\hat{\mathbf{View}}^{Z,\mathsf{GOD}}_{\mathsf{NxtMsg}^{k'}_{j''}}$.
- Corresponding to the $(j', k)^{th}$ instance, $\mathcal{A}$ sends all the inputs and outputs to the challenger $\mathcal{C}$ that samples a bit b. If $b = 0$, $\mathcal{C}$ computes,
$$\mathbf{View}_{\mathsf{NxtMsg}} \leftarrow \langle \mathsf{Sim}^{\mathsf{NxtMsg}^k_{j'}}_{\mathsf{C,GOD}}(1^\kappa), \mathcal{A} \rangle$$
Otherwise it computes $\mathbf{View}_{\mathsf{NxtMsg}}$ to be the views of the parties in Z in the real execution of $\Pi_{\mathsf{NxtMsg}^k_{j'}}$.
- $\mathcal{C}$ gives $\mathbf{View}_{\mathsf{NxtMsg}}$ to $\mathcal{A}$ and, with this, $\mathcal{A}$ completes the view $\mathbf{View}$ to be input to D. Note that if $b = 0$, $\mathbf{View}$ is distributed as in $\mathsf{H}'_{j',k-1}$ and if $b = 1$, it is distributed as in $\mathsf{H}'_{j',k}$.

– Finally, $\mathcal{A}$ passes **View** to D and outputs whatever D outputs.

In the above strategy, $\mathcal{A}$ has the same distinguishing advantage as D, which is non-negligible. However, since the protocol $\Pi_{\mathsf{NxtMsg}^k_{j'}}$ is secure in the presence of a PPT malicious adversary corrupting any subset $Z \in \mathcal{Z}_{\mathsf{GOD}}$ of the parties, no such $\mathcal{A}$ can exist and therefore no such D can exist. So the distributions $\mathsf{H}'_{j',k}$ and $\mathsf{H}'_{j',k-1}$ are computationally indistinguishable. Since none of the listed set of adjacent hybrids are distinguishable, it follows that the real view of the protocol $\mathsf{H}'_{0,0}$ and the simulated view $\mathsf{H}'_{2,r}$ are computationally indistinguishable.

Claim. Assuming that the virtual protocol $\Pi_{\mathcal{V}}$ is fail-stop secure in the presence of an unbounded adversary corrupting either $\mathcal{V}_2$ or $\mathcal{V}_3$, the view in the hybrid distributions H_2 and H_3 are statistically close.

Proof. The difference between the hybrid distributions H_2 and H_3 stems only from the fact that, in the adversary's view, in the 'Compute Virtual Protocol' phase, the views of the parties from the virtual protocol (and the shares thereof) are created differently. In hybrid H_2, these are views generated from the virtual protocol simulator $\mathsf{Sim}_{\mathcal{V}}$, while in H_3, these are distributed as views of the real execution of $\Pi_{\mathcal{V}}$. It therefore follows that if the distribution of the simulated and real view of $\Pi_{\mathcal{V}}$ are statistically close, so are the hybrids H_2 and H_3.

Fairness for PPT Adversaries. Let $\mathcal{A}$ be a malicious PPT adversary corrupting a set of parties $Z \in \mathcal{Z}^*_{\mathsf{fair}}$. Note that since $\mathcal{Z}^*_{\mathsf{fair}} = \mathcal{Z}_{\mathsf{fair}} \cup \mathcal{Z}_{\mathsf{GOD}}$, and we already have from a previous case that GOD, a stronger guarantee than fairness, holds for all corruption within $\mathcal{Z}_{\mathsf{GOD}}$, it only remains to argue that fairness holds for any $Z \in \mathcal{Z}_{\mathsf{fair}}$, in particular, when it is not included in $\mathcal{Z}_{\mathsf{S}}$. Consider the following PPT simulator $\mathsf{Sim}_{\mathsf{C,fair}}$ that makes black-box calls to simulators in $\{\mathsf{Sim}^{\mathsf{NxtMsg}^k_{j'}}_{\mathsf{C,fair}}\}_{j' \in \{2,3\}, k \in [r]}$ and simulates the view of $\mathcal{A}$ in the ideal world for fairness. This simulator operates in the same way as $\mathsf{Sim}_{\mathsf{C,GOD}}$ would, but with the following differences:

1. During the phase to 'compute virtual protocol', $\mathsf{Sim}_{\mathsf{C,fair}}$ makes black-box calls to $\mathsf{Sim}^{\mathsf{NxtMsg}^k_{j'}}_{\mathsf{C,fair}}$ instead of to $\mathsf{Sim}^{\mathsf{NxtMsg}^k_{j'}}_{\mathsf{C,GOD}}$.
2. Furthermore, if any of these simulation interactions abort, $\mathsf{Sim}_{\mathsf{C,fair}}$ sends $\perp$ to the ideal functionality $\mathcal{F}$ and terminates execution. The view generated up to this point is output by the simulator.
3. If *all* calls to $\mathsf{Sim}^{\mathsf{NxtMsg}^k_{j'}}_{\mathsf{C,fair}}$ execute to completion, $\mathsf{Sim}_{\mathsf{C,fair}}$ sends the inputs of the adversary $\mathcal{A}$ extracted from the views to $\mathcal{F}$ and receives $f(x)$. It uses this to complete the protocol execution in the last round of next-message-function calls of the virtual protocol $\Pi_{\mathcal{V}}$.

Note that the adversary causes the protocol to abort in the ideal execution, iff the protocol also aborts in the real execution. The view of the protocol up to that point was generated without the output of the functionality and so it

follows that, along with upholding the privacy of honest parties' inputs, both the adversary and the honest parties don't learn the output. If the protocol executes to completion, the honest parties always have enough shares to reconstruct the output of the functionality. In this case, the view of the adversary in the above execution of $\mathsf{Sim}_{\mathsf{C},\mathsf{fair}}$ is distributed as,

$$\left\{ \{x_i\}_{P_i \in Z}, f(\vec{x}), \{r_i\}_{P_i \in Z}, \left\{ \hat{\mathbf{View}}^{Z,\mathsf{fair}}_{\mathsf{NxtMsg}^{k_*}_{j'}} \right\}_{j' \in [3], k \in [r], P_i \notin Z}, \left\{ \mathbf{View}^{Z,\mathsf{fair}}_{\mathsf{NxtMsg}^{k}_{j'}} \right\}_{j' \in [3], k \in [r], P_i \in Z} \right\}_{\kappa \in \mathbb{N}, \vec{r} \in \mathcal{R}^n}$$

This distribution can be argued as computationally indistinguishable from the real view of the adversary by considering a similar set of hybrids as in the GOD case, with the only difference being that the argument would now follow from the fact that the views output from $\{\mathsf{Sim}^{\mathsf{NxtMsg}^{k}_{j'}}_{\mathsf{C},\mathsf{fair}}\}_{j' \in [3], k \in [r]}$ (as opposed to $\{\mathsf{Sim}^{\mathsf{NxtMsg}^{k}_{j'}}_{\mathsf{C},\mathsf{GOD}}\}_{j' \in [3], k \in [r]}$) are computationally indistinguishable from the real views of the protocol execution.

Security with Abort for PPT Adversaries. Let $\mathcal{A}$ be a malicious PPT adversary corrupting a set of parties $Z \in \mathcal{Z}^*_{\mathsf{C}}$, in particular a set not in $\mathcal{Z}^*_{\mathsf{fair}}$ or $\mathcal{Z}_{\mathsf{S}}$. Consider the following PPT simulator $\mathsf{Sim}_{\mathsf{C},\perp}$ that makes black-box calls to simulators in $\{\mathsf{Sim}^{\mathsf{NxtMsg}^{k}_{j'}}_{\mathsf{C},\perp}\}_{j' \in \{2,3\}, k \in [r]}$ and simulates the view of $\mathcal{A}$ in the ideal world for security with abort. This simulator operates in the same way as $\mathsf{Sim}_{\mathsf{C},\mathsf{fair}}$ would, but with the following differences:

1. If *all* calls to $\mathsf{Sim}^{\mathsf{NxtMsg}^{k}_{j'}}_{\mathsf{C},\perp}$ execute to completion, $\mathsf{Sim}_{\mathsf{C},\perp}$, like $\mathsf{Sim}_{\mathsf{C},\mathsf{fair}}$, sends the inputs of the adversary $\mathcal{A}$ extracted from the views to $\mathcal{F}$ and receives $f(x)$. It uses this to complete the protocol execution in the last round of next-message-function calls of the virtual protocol $\Pi_{\mathcal{V}}$.
2. If the set of parties corrupted by the adversary $Z \notin \mathcal{Z}^*_{\mathsf{fair}}$, $\mathsf{Sim}_{\mathsf{C},\perp}$ sends $\perp$ to the ideal functionality $\mathcal{F}$ and $\mathcal{F}$ sends $\perp$ to all honest parties. Otherwise, $\mathsf{Sim}_{\mathsf{C},\perp}$ sends $\mathsf{continue}$ to $\mathcal{F}$ and the honest parties receive the outputs.

Note that the adversary causes the protocol to abort in the ideal execution, iff the protocol also aborts in the real execution. If the protocol executes to completion, and there are not enough honest parties to reconstruct the function output, the protocol would abort with only the adversary receiving the function output. The view of the adversary in the above execution of $\mathsf{Sim}_{\mathsf{C},\perp}$ is distributed as,

$$\left\{ \{x_i\}_{P_i \in Z}, f(\vec{x}), \{r_i\}_{P_i \in Z}, \left\{ \hat{\mathbf{View}}^{Z,\perp}_{\mathsf{NxtMsg}^{k_*}_{j'}} \right\}_{j' \in [3], k \in [r], P_i \notin Z}, \left\{ \mathbf{View}^{Z,\perp}_{\mathsf{NxtMsg}^{k}_{j'}} \right\}_{j' \in [3], k \in [r], P_i \in Z} \right\}_{\kappa \in \mathbb{N}, \vec{r} \in \mathcal{R}^n}$$

This distribution can be argued as computationally indistinguishable from the real view of the adversary by considering the exact same set of hybrids as in the fairness case.