On the Exact Round Complexity of Best-of-both-Worlds Multi-party Computation

Arpita Patra[1]*, Divya Ravi[1], Swati Singla[2]

[1] Indian Institute of Science, India. {arpita,divyar}@iisc.ac.in
[2] Google India, Bangalore. swatis@iisc.ac.in

Abstract. The two traditional streams of multiparty computation (MPC) protocols consist of– (a) protocols achieving guaranteed output delivery (god) or fairness (fn) in the honest-majority setting and (b) protocols achieving unanimous or selective abort (ua, sa) in the dishonest-majority setting. The favorable presence of honest majority amongst the participants is necessary to achieve the stronger notions of god or fn. While the constructions of each type are abound in the literature, one class of protocols does not seem to withstand the threat model of the other. For instance, the honest-majority protocols do not guarantee privacy of the inputs of the honest parties in the face of dishonest majority and likewise the dishonest-majority protocols cannot achieve god and fn, tolerating even a single corruption, let alone dishonest minority. The promise of the unconventional yet much sought-after species of MPC, termed as 'Best-of-Both-Worlds' (BoBW), is to offer the best possible security depending on the actual corruption scenario.

This work nearly settles the exact round complexity of two classes of BoBW protocols differing on the security achieved in the honest-majority setting, namely god and fn respectively, under the assumption of no setup (plain model), public setup (CRS) and private setup (CRS + PKI or simply PKI). The former class necessarily requires the number of parties to be strictly more than the sum of the bounds of corruptions in the honest-majority and dishonest-majority setting, for a feasible solution to exist. Demoting the goal to the second-best attainable security in the honest-majority setting, the latter class needs no such restriction.

Assuming a network with pair-wise private channels and a broadcast channel, we show that 5 and 3 rounds are necessary and sufficient for the class of BoBW MPC with fn under the assumption of 'no setup' and 'public and private setup' respectively. For the class of BoBW MPC with god, we show necessity and sufficiency of 3 rounds for the public setup case and 2 rounds for the private setup case. In the no setup setting, we show the sufficiency of 5 rounds, while the known lower bound is 4. All our upper bounds are based on polynomial-time assumptions and assume black-box simulation. With distinct feasibility conditions, the classes differ in terms of the round requirement. The bounds are in some cases different and on a positive note at most one more, compared to the maximum of the needs of the honest-majority and dishonest-majority setting.

* Arpita Patra would like to acknowledge financial support from SERB MATRICS (Theoretical Sciences) Grant 2020 and Google India AI/ML Research Award 2020.

Our results remain unaffected when security with abort and fairness are upgraded to their identifiable counterparts.

1 Introduction

In secure multi-party computation (MPC) [1,2,3], n parties wish to jointly perform a computation on their private inputs in a way that no adversary $\mathcal{A}$ actively corrupting a coalition of t parties can learn more information than their outputs (*privacy*), nor can they affect the outputs of the computation other than by choosing their own inputs (*correctness*). MPC protocol comes in distinct flavours with varying degree of robustness– guaranteed output delivery (god), fairness (fn), unanimous abort (ua) and selective abort (sa). The strongest security, god, implies that all parties are guaranteed to obtain the output, regardless of the adversarial strategy. In the weaker notion of fn, the corrupted parties receive their output if and only if all honest parties do. In the further weaker guarantee of ua, fairness may be compromised, yet the adversary cannot break unanimity of honest parties. That is, either all or none of the honest parties receive the output. Lastly, sa security, the weakest in the lot, allows the adversary to selectively deprive some honest parties of the output.

While highly sought-after, the former two properties can only be realised, when majority of the involved population is honest [4]. In the absence of this favorable condition, only the latter two notions can be attained. With these distinct affordable goals, MPC with honest majority [5,6,7,8,9,10,11] and dishonest majority [1,12,13,14,15,16,17] mark one of the earlier demarcations in the world of MPC. With complementary challenges and techniques, each setting independently stands tall with spectacular body of work. Yet, the most worrisome shortcoming of these generic protocols is that: a protocol in *one* setting completely breaks down in the *other* setting i.e. the security promises are very rigid and specific to the setting. For example, a protocol for honest majority might no longer even be "private" or "correct" if half (or more) of the parties are corrupted. A protocol that guarantees security with ua for arbitrary corruptions cannot pull off the stronger security of god or fn even if only a "single" party is corrupt. In many real-life scenarios, it is highly unlikely for anyone to guess upfront how many parties the adversary is likely to corrupt. In such a scenario, the best a practitioner can do, is to employ the 'best' protocol from her favorite class and hope that the adversary will be within assumed corruption limit of the employed protocol. If the guess fails, the employed protocol, depending on whether it is an honest or dishonest majority protocol, will suffer from the above mentioned issues. The quest for attaining the best feasible security guarantee in the respective settings of honest and dishonest majority in a *single* protocol sets the beginning of a brand new class of MPC protocols, termed as 'Best of Both Worlds (BoBW)' [18,19,20]. In critical applications like voting [21,22], secure auctions [23], secure aggregation [24], federated learning and prediction [25,26], financial data analysis [27] and others, where privacy of the inputs of an honest

party needs protection at any cost and yet a robust completion is called for (as much as theoretically feasible), BoBW protocols are arguably the best fit.

Denoting the threshold of corruption in honest and dishonest majority case by t and s respectively, an ideal BoBW MPC should promise the best possible security in each corruption scenario for any population of size n, as long as $t < n/2$ and $s < n$. Quite contrary to the expectation, the grand beginning of BoBW MPC with the works of [18,19,20] is mostly marred with pessimistic results showing the above goal is impossible for many scenarios. For reactive functionalities that receive inputs and provide outputs in multiple rounds maintaining a state information between subsequent invocations, it is impossible to achieve BoBW security [18]. While theoretical feasibility is not declined, non-reactive or standard functionalities are shown to be impossible to realise as long as $t + s \geq n$ in expected polynomial time (in the security parameter), making any positive result practically irrelevant [19,20]. A number of meaningful relaxations were proposed in the literature to get around the impossibility of BoBW security when $t + s \geq n$ [19,20]. The most relevant to our work is the relaxation proposed in [28] where the best possible security of god is compromised to the second-best notion of fn in the honest-majority setting. Other attempts to circumvent the impossibility result appear in [18] and [19,29] where the security in dishonest-majority setting is weakened to allowing the adversary to learn s evaluations of the function (each time with distinct inputs *exclusively* corresponding to the corrupt parties) in the former and achieving a weaker notion of $O(1/p)$-security with abort (actions of any polynomial-time adversary in the real world can be simulated by a polynomial-time adversary in the ideal world such that the distributions of the resulting outcomes cannot be distinguished with probability better than $O(1/p)$) in the latter. [18] shows yet another circumvention by weakening the adversary in dishonest-majority case from active to passive. On the contrary, constructions are known when $t + s < n$ is assumed [18], tolerating active corruptions and giving best possible security in both the honest and dishonest majority case.

In this work, we consider two types of BoBW MPC protocols and study their exact round complexity: (a) MPC achieving the best security of god and ua in the honest and dishonest majority setting respectively assuming $s + t < n$, referred as ($\mathsf{god}|\mathsf{ua}$)-BoBW; (b) MPC achieving second-best security notion of fn in the honest majority and the best possible security of ua in the dishonest majority for any n, referred as ($\mathsf{fn}|\mathsf{ua}$)-BoBW. The adversary is considered malicious, rushing and polynomially-bounded in either world. The latter notion (introduced in [28]) is an elegant and meaningful relaxation that brings back the true essence of BoBW protocols with no constraint on n, apart from the natural bounds of $t < n/2$ and $s < n$. Furthermore, fn is almost as good as god for many practical applications where the adversary is rational enough and does not wish to fail the honest parties at the expense of losing its own output. In spite of immense practical relevance of BoBW protocols, the question of their exact round complexity has not been tackled so far. Below, we review relevant literature on BoBW protocols and exact round complexity of MPC.

1.1 On the Round Complexity of BoBW MPC

The phenomenal body of work done on round complexity catering to various adversarial settings and network models emphasises its theoretical importance and practical relevance. For instance, the exact round complexity of MPC independently in honest and dishonest majority has been examined and the recent literature is awash with a bunch of upper bounds that eluded for quite a long time [30,31,16,17]. We review the round complexity of the honest-majority and dishonest-majority MPC in the cryptographic setting which define natural yet possibly loose bounds for the BoBW MPC. To begin with, 2 rounds are known to be necessary to realize any MPC protocol, regardless of the setting, no matter whether a setup is assumed or not as long as the setup (when assumed) is independent of the inputs of the involved parties [32]. In the dishonest-majority setting, when no setup is assumed (plain model) 4 rounds are necessary [33]. Tight upper bounds appear in [14,15,16,17,34], with the latter three presenting constructions under polynomial-time assumptions, yet with sa security. In the presence of a public setup (Common Reference String a.k.a. CRS setting), the lower bound comes down to 2 rounds [32]. A series of work present matching upper bounds under various assumptions [13,35,36], culminating with the works of [30,31] that attain the goal under the minimal assumption of 2-round oblivious transfer (OT). In the honest-majority setting and in plain model, 3 rounds are shown to be necessary for fn (and hence for god) protocols, in the presence of pairwise-private and broadcast channels for $t \geq 2$ active corruptions [37] and for any t as long as $n/3 < t < n/2$ [38]. The results of [37,38] hold in the presence of CRS but does not hold in the presence of correlated randomness setup such as PKI. Circumventing the lower bound of 3 for fn, [39] shows a 2-round 4PC protocol against a single active corruption achieving god even without a broadcast channel. The matching upper bounds appear in [11] for the general case under public-key assumption, and in [38] for the special case of 3PC under the minimal assumption of (injective) OWF. In the CRS model, 3 rounds remains to be the lower bound for fn in a setting where broadcast is the only medium of communication (broadcast-only setting) [40] and additionally with point-to-point channels [38,37,41]. Given PKI, the bound can be improved to 2 [40].

In the BoBW setting, constant-round protocols are presented in (or can be derived from) [18,20] for (god|ua)-BoBW and BoBW where only semi-honest corruptions are tolerated in the dishonest majority. The recent work of [42] settled the exact round complexity of the latter class, as a special case of a strong adversarial model that allows both active (with threshold t_a) and passive (with threshold t_p, which subsumes the active corruptions) corruption for a range of thresholds for (t_a, t_p) starting from $(\lceil n/2 \rceil - 1, \lfloor n/2 \rfloor)$ to $(0, n-1)$. Lastly, the round complexity of BoBW protocols of [29] that achieve $1/p$- security with abort in dishonest-majority (and god in honest majority), depends on the polynomial $p(\kappa)$ (where κ denotes the security parameter).

1.2 Our Results

This work nearly settles the exact round complexity for two classes of BoBW protocols, (god|ua)-BoBW and (fn|ua)-BoBW, under the assumption of no setup (plain model), public setup (CRS) and private setup (CRS + PKI or simply PKI). The adversary is assumed to be rushing, active and static. The parties are connected via pair-wise private channels and an additional broadcast channel. All our upper bounds are based on polynomial-time assumptions and assume black-box simulation. We summarise our results below.

(fn|ua)-BoBW. We settle the exact round complexity of this class of BoBW protocols by establishing the necessity and sufficiency of: (a) 5 rounds in the plain model and (b) 3 rounds in both the public (CRS) and private (CRS+PKI) setup setting. In the CRS model, the necessity of 3 rounds for honest-majority MPC achieving fn (and hence for (fn|ua)-BoBW) has been demonstrated in [40,37,38], the former in a setting where broadcast is the only mode of communication (broadcast-only) and the latter two additionally with pairwise-private channels. However, these results do not hold in the presence of PKI. Our lower bound argument, on the other hand, is resilient to the presence of both CRS and PKI, and further holds in the presence of broadcast and pairwise-private channels.

	No setup (Plain Model)	**Public Setup (CRS)**	**Private Setup (CRS + PKI)**
Honest Majority $t < n/2$ fn / god	Round: 3 Lower Bound: [38,37] Upper Bound: [11,43]	Round: 3 Lower Bound: [38,37] Upper Bound: [40,11,43]	Round: 2 Lower Bound: [32] Upper Bound: [40]
Dishonest Majority $s < n$ sa / ua	Round: 4 Lower Bound: [33] Upper Bound: [16,17,34] (sa only)	Round: 2 Lower Bound: [32] Upper Bound: [13,35] [36,30,31]	Round: 2 Lower Bound: [32] Upper Bound: [13,35] [36,30,31]
(fn|ua)-BoBW $t < n/2, s < n$ fn & ua	Round: 5 Lower Bound: **This paper** Upper Bound: **This paper**	Round: 3 Lower Bound: [37,38] Upper Bound: **This paper**	Round: 3 Lower Bound: **This paper** Upper Bound: **This paper**
(god|ua)-BoBW $t < n/2, t + s < n$ god & ua	Round: – Lower Bound: 4 [33] Upper Bound: 5 **This paper**	Round: 3 Lower Bound: **This paper** Upper Bound: **This paper**	Round: 2 Lower Bound: [32] Upper Bound: **This paper**

Table 1: Summary of results

(god|ua)-BoBW. In this regime, we demonstrate that 4, 3 and 2 are the respective lower bounds in the no-setup, public setup and private setup setting. The first lower bound follows from the fact that BoBW MPC in this class trivially subsumes the dishonest majority MPC when $t = 0$ and the lower bound for dishonest-majority MPC is 4 [33]. The last lower bound follows from the standard 2-round bound for MPC needed to counter "residual function attack" [32]. Regarding the lower bound of 3 for the public setup (CRS) setting, we point that it follows directly from the 2-round impossibility of MPC with fn for honest majority in the CRS model [40,38,37] for *most* values of (t, s, n) satisfying $s+t < n$. However, these existing results do not rule out the possibility of 2-round (god|ua)-BoBW MPC for $(t = 1, s > t, n \geq 4)$. (In fact the protocols of [44,39]

circumvent the 3-round lower bound for fn when $t = 1, n \geq 4$). We address this gap by giving a *unified proof* that works even for $s > t$, for all values of t (including $t = 1$). This is non-trivial and it demonstrably breaks down in the presence of PKI. The bounds are totally different from the ones for previous class, owing to the different feasibility condition of $s + t < n$. While our upper bound falls merely one short of matching the first lower bound in case of no-setup, the upper bounds of the other two settings are tight. We leave the question of designing or alternately proving the impossibility of 4-round (god|ua)-BoBW MPC protocol as open. Our results summarised and put along with the bounds known in the honest and dishonest majority setting appear in Table 1.

Extensions. We can boost the security of all our protocols to offer identifiability (i.e. public identifiability of the parties who misbehaved) when abort happens– (fn|ua)-BoBW protocols with identifiable fairness and abort in honest and dishonest majority setting respectively and (god|ua)-BoBW protocols with identifiable abort in dishonest-majority setting. Our lower bound results hold as is when ua and fn are upgraded to their stronger variants with identifiability. Furthermore, all our upper bounds relying on CRS have instantiations based on a weaker setup, referred as common *random* string, owing to the availability of 2-round OT [45] and Non-Interactive Zero Knowledge (NIZK) [46] under the latter setup assumption. Lastly, we also propose few optimizations to minimize the use of broadcast channels in our compilers upon which our upper bounds are based. Specifically, these optimizations preserve the round complexity of our upper bounds at the cost of relaxing the security notion in dishonest majority setting to sa (as opposed to ua).

1.3 Techniques

(fn|ua)-BoBW. The lower bounds are obtained via a reduction to 3-round OT in plain model and 1-round OT in private setup setting, both of which are known to be impossible [33,32] (albeit under the black-box simulation paradigm which is of concern in this paper). The starting point is a protocol π between 3 parties which provides fn when 1 party is corrupt and ua when 2 parties are corrupt, in 4 rounds when no setup is assumed and 2 rounds when private/public setup is assumed. The heart of the proof lies in devising a function f such that the realization of f via π, barring its last round, leads to an OT.

The upper bounds are settled with a proposed generic compiler that turns an r-round dishonest-majority MPC protocol achieving ua to an $(r + 1)$-round BoBW MPC protocol *information-theoretically*. The compiler churns out a 5-round and a 3-round BoBW protocol in the plain model and in the presence of a CRS respectively, when plugged with appropriate ua-secure dishonest-majority protocol in the respective setting. Since the constructions of the known 4-round dishonest-majority MPC relying on polynomial-time assumptions [16,17,34] provide only sa security, we transform them to achieve ua for our purpose which invokes non-triviality for [16]. With CRS, the known constructions of [30,31] achieve unanimity and readily generate 3-round BoBW protocols.

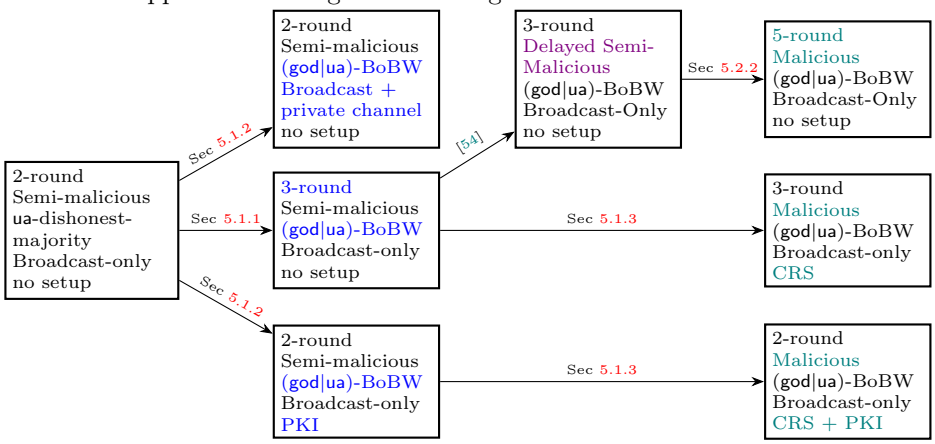
Our compiler motivated by [47] uses the underlying r-round protocol to compute authenticated secret sharing of the output y with a threshold $t(< n/2)$ enabling the output reconstruction to occur in the last round. Fairness is ensured given the unanimity of the underlying protocol and the fact that the adversary (controlling t corrupt parties) has no information about the output y from the t shares he owns. However, using pairwise MACs for authentication defies unanimity in case of arbitrary corruptions because a corrupt party can choose to provide a verified share to a selected set of honest parties enabling their output reconstruction while causing the rest to abort. To address this, a form of authentication used in the Information Checking Protocol (ICP) primitive of [48,49] and unanimously identifiable commitments (UIC) of [50] can be used. This technique maintains unanimity amongst the honest parties during output reconstruction.

(god|ua)-BoBW. The non-trivial lower bound for this class is for the CRS setting. The other bounds imply from the dishonest-majority case. In the CRS setting, we prove a lower bound of 3 rounds. We start with assuming a 2 round BoBW protocol π for a specifically articulated 4-party function f. Next, we consider a sequence of executions of π, with different adversarial strategies in the order of their increasingly malicious behaviour such that the views of a certain party stays the same between the executions. This sequence finally leads us to a strategy where the adversary is able to learn the input of an honest party breaching privacy, hence coming to a contradiction. The crux of the lower bound argument lies in the design of the adversarial strategies that shuffle between the honest and dishonest majority setting encapsulating the challenge in designing BoBW protocols. This is in contrast to existing lower bounds in traditional models that deal with a fixed setting and single security notion at a time.

In the presence of a CRS, we build a 3-round protocol in two steps: a) we provide a generic compiler that transforms a broadcast-only ua-secure 2-round semi-malicious protocol such as [30,31] to a 3-round broadcast-only BoBW protocol of this class against a semi-malicious adversary (that follows the protocol honestly but can choose bad random coins for each round which are available to the simulator) b) then, the round-preserving compiler of [51] (using NIZKs) is applied on the above protocol to attain malicious security. The first compiler, in spirit of [11], ensures god against t non-cooperating corrupt parties in the last round, via secret-sharing the last-round message of the underlying protocol during the penultimate round of the compiled protocol. This is achieved by means of a garbled circuit sent by each party outputting its last-round message of the underlying protocol and the shares of the encoded labels with a threshold of s so that $s+1$ parties (in case of honest majority) can come together in the final round to construct the last-round message of the corrupt parties. This garbled circuit of a party P_i also takes into account the case when some other parties abort in the initial rounds of the protocol by taking the list of aborting parties as input and hard-coding their default input and randomness such that P_i's last round message is computed considering default values for parties who aborted. The compiler is made round-preserving with additional provision of pairwise-

private channels or alternately, PKI. The latter (with PKI) just like its 3-round avatar can be compiled to a malicious protocol via the compiler of [51].

In the plain model, we provide a 5-round construction which is substantially more involved than our other upper bounds. To cope up with the demands of (god|ua)-BoBW security in the plain model, we encountered several roadblocks that were addressed by adapting some existing techniques combined with new tricks. The construction proceeds in two steps: a) we boost the security of our broadcast-only 3-round semi-malicious BoBW protocol to a stronger notion of delayed-semi-malicious security (where the adversary is required to justify his messages by giving a valid witness only in the last but one round) and b) we plug this 3-round BoBW protocol in the compiler of [31] with some additional modifications to obtain a 5-round BoBW protocol secure against a malicious adversary. The compiler of [31] takes as input a $(k-1)$-round protocol secure *with abort* against a delayed-semi-malicious adversary and churns out a k-round protocol secure *with abort* against a malicious adversary for any $k \geq 5$. The major challenges in our construction surface in simulation, where we cannot terminate in the honest-majority case even if the adversary aborts on behalf of a corrupt party (unlike the compiler of [31] that achieves abort security only). Furthermore, we observed that the natural simulation strategy to retain the BoBW guarantee suffered from a subtle flaw, similar to the one pointed in the work of [52], which we resolve with the help of the idea suggested therein. To bound the simulation time by expected polynomial-time, we further needed to introduce two 'dummy' rounds (rounds which do not involve messages of the underlying protocol being compiled) in our compiler as opposed to one as in [31]. This does not inflate the round complexity as our underlying delayed-semi-malicious protocol only consumes 3 rounds (instead of 4 as in the case of [31]). As a step towards resolving the question left open in this work (namely proving the impossibility or alternately constructing a 4-round (god|ua)-BoBW protocol under polynomial-time assumption), we present a sketch of a 4-round (god|ua)-BoBW protocol based on sub-exponentially secure trapdoor permutations and ZAPs. This construction builds upon the work of [53]. The pictorial roadmap to obtain the upper bounds is given in the figure below.



1.4 Related works on BoBW MPC

An orthogonal notion of BoBW security is considered in [55,56,28] where information-theoretic and computational security is the desired goal in honest and dishonest majority setting respectively. Avoiding the relaxation to computational security in dishonest-majority setting, the work of [57] introduces the best possible information-theoretic guarantee achievable in the honest and dishonest majority settings simultaneously; i.e. the one that offers standard information-theoretic security in honest majority and offers residual security (the adversary cannot learn anything more than the residual function of the honest parties' inputs) in dishonest-majority setting. A more fine-grained graceful degradation of security is dealt with in the works of [28,58,59,60,42] considering a mixed adversary that can simultaneously corrupt in both active and semi-honest style. Lastly, [61] studies the communication efficiency in the BoBW setting.

1.5 Our Model

Before moving onto the technical section, we detail our model here. We consider a set of n parties $\mathcal{P} = \{P_1, \dots P_n\}$ connected by pairwise-secure and authentic channels and having access to a broadcast channel. A few protocols in our work that are referred to as being *broadcast-only* do not assume private channels. Each party is modelled as a probabilistic polynomial time (PPT) Turing machine. We assume that there exists a PPT adversary $\mathcal{A}$, who can corrupt a subset of these parties. We denote the set of indices corresponding to parties controlled by $\mathcal{A}$ and the honest parties with $\mathcal{C}$ and $\mathcal{H}$ respectively. We denote the cryptographic security parameter by κ. A negligible function in κ is denoted by $\mathsf{negl}(\kappa)$. A function $\mathsf{negl}(\cdot)$ is negligible if for every polynomial $p(\cdot)$ there exists a value N such that for all $m > N$ it holds that $\mathsf{negl}(m) < \frac{1}{p(m)}$. Lastly, we denote the ideal functionalities for unanimous abort, fairness and guaranteed output delivery with $\mathcal{F}_{\mathsf{ua}}$, $\mathcal{F}_{\mathsf{fair}}$ and $\mathcal{F}_{\mathsf{god}}$ respectively (details appear in full version [54]).

Roadmap. Our lower and upper bounds for ($\mathsf{fn}|\mathsf{ua}$)-BoBW appear in Section 2-3. Our lower and upper bounds for ($\mathsf{god}|\mathsf{ua}$)-BoBW appear in Section 4-5. Our protocols are proven in real-world and ideal-world paradigm. The detailed security definitions, complete security proofs and formal definitions of the primitives used in our upper bounds are described in the full version [54].

2 Lower Bounds for ($\mathsf{fn}|\mathsf{ua}$)-BoBW

In this section, we show two lower bounds concerning ($\mathsf{fn}|\mathsf{ua}$)-BoBW protocols– one with no setup and the other with private setup. In the plain model, we show that it is impossible to design a 4-round ($\mathsf{fn}|\mathsf{ua}$)-BoBW protocol (with black-box simulation). In the CRS setting, the 3-round lower bound for ($\mathsf{fn}|\mathsf{ua}$)-BoBW protocols follows directly from the impossibility of 2-round protocol achieving fn [40,37,38]. However, they do not hold in the presence of PKI. While the argument

of [40] crucially relies on the adversary being able to eavesdrop communication between two honest parties (which does not hold in the presence of PKI), the lower bounds of [37,38] also do not hold if PKI is assumed (as acknowledged / demonstrated in [37,41]). In the setting with CRS and PKI, we show impossibility of a 2-round protocol. The proof of both our lower bounds relies on the following theorem, which we formally state and prove below.

Theorem 1. *An n-party r-round $(\mathsf{fn}|\mathsf{ua})$-BoBW protocol implies a 2-party $(r-1)$-round maliciously-secure oblivious transfer (OT).*

Proof. We prove the theorem for $n = 3$ parties with $t = 1$ and $s = 2$ which can be extended for higher values of n in a natural manner (elaborated in the full version). Let $\mathcal{P} = \{P_1, P_2, P_3\}$ denote the 3 parties and the adversary $\mathcal{A}$ may corrupt at most two parties. As per the hypothesis, we assume that there exists a r-round $(\mathsf{fn}|\mathsf{ua})$-BoBW protocol protocol π_f that can compute the function f defined as $f((m_0, m_1), (c, R_2), R_3) = ((m_c + R_2 + R_3), m_c, m_c)$ which simultaneously achieves fn when $t = 1$ parties are corrupt and ua when $s = 2$ parties are corrupt. At a high-level, we transform the r-round 3-party protocol π_f among $\{P_1, P_2, P_3\}$ into a $(r-1)$-round 2-party OT protocol between a sender P_S with inputs (m_0, m_1) and a receiver P_R with input c.

Let $\mathsf{q} = 1 - \mathsf{negl}(\kappa)$ denote the overwhelming probability with which security of π_f holds, where the probability is defined over the choice of setup (in case a setup is assumed) and the random coins used by the parties. Before describing the transformation, we present the following lemma:

Lemma 1. *Protocol π_f must be such that the combined view of $\{P_2, P_3\}$ at the end of Round $(r-1)$ suffices to compute their output, with overwhelming probability.*

Proof. Consider an adversary $\mathcal{A}$ who corrupts only a minority of the parties ($t = 1$). $\mathcal{A}$ controls party P_1 with the following strategy: P_1 behaves honestly in the first $(r-1)$ rounds while he simply remains silent in Round r (last round). Since P_1 receives all the desired communication throughout the protocol, it follows directly from correctness of π_f (which holds with overwhelming probability q) that $\mathcal{A}$ must be able to compute the output with probability q. Since π_f is assumed to be fair (with probability q) for the case of $t = 1$, it must hold that when P_1 learns the output, the honest parties P_2 and P_3 must also be able to compute the output with overwhelming probability $\mathsf{q} \times \mathsf{q} = \mathsf{q}^2$; without any communication from P_1 in Round r. This implies that the combined view of $\{P_2, P_3\}$ at the end of Round $(r-1)$ must suffice to compute the output with overwhelming probability q^2. □

Our transformation from π_f to a $(r-1)$-round OT protocol π_{OT} between a sender P_S with inputs (m_0, m_1) and a receiver P_R with input c goes as follows. P_S emulates the role of P_1 during π_f while P_R emulates the role of both parties $\{P_2, P_3\}$ during π_f using random inputs R_2, R_3 respectively. In more detail, let $\mathsf{m}^{\mathsf{r}}_{i \to j}$ denote the communication from P_i to P_j in round r of π_f. Then for $\mathsf{r} \in [r-1]$, the interaction in round r of protocol π_{OT} is the following: P_S sends

$\mathsf{m}^{\mathsf{r}}_{1\to 2}$ and $\mathsf{m}^{\mathsf{r}}_{1\to 3}$ to P_R while P_R sends $\mathsf{m}^{\mathsf{r}}_{2\to 1}$ and $\mathsf{m}^{\mathsf{r}}_{3\to 1}$ to P_S. P_R computes the output m_c using the combined view of $\{P_2, P_3\}$ at the end of Round $(r-1)$. P_S outputs nothing. Recall that the output of the OT between (P_S, P_R) is $(\perp, m_c)$ respectively. We now argue that π_{OT} realizes the OT functionality.

Lemma 2. *Protocol π_{OT} realizes the OT functionality.*

Proof. We first prove that π_{OT} is correct. By Lemma 1, it follows that P_R emulating the role of both $\{P_2, P_3\}$ of π_f must be able to compute the correct output m_c with overwhelming probability by the end of Round $(r-1)$. We now consider the security properties. First, we consider a corrupt P_R (emulating the roles of $\{P_2, P_3\}$ in π_f). Since by assumption, π_f is a protocol that should preserve privacy of P_1's input even in the presence of an adversary corrupting $\{P_2, P_3\}$ ($s = 2$ corruptions), the input m_{1-c} of P_S must remain private against a corrupt P_R. Next, we note that privacy of π_f against a corrupt P_1 ($t = 1$ corruption) guarantees that P_1 does not learn anything beyond the output $(m_c + R_2 + R_3)$ in the protocol π_f which leaks nothing about c. It thus follows that a corrupt P_S in π_{OT} emulating the role of P_1 in π_f will also not be able to learn anything about P_R's input c. More formally, we can construct a simulator for the OT protocol π_{OT} for the cases of corrupt P_R and corrupt P_S by invoking the simulator of π_f for the case of dishonest majority ($s = 2$) and honest majority ($t = 1$) respectively. In each case, it follows from the security of π_f (which holds with overwhelming probability) that the simulator of π_f would return a view indistinguishable from the real-world view with overwhelming probability; directly implying the security of the OT protocol π_{OT}. □
Thus, we can conclude that a $(r-1)$-round 2-party OT protocol π_{OT} can be derived from r-round π_f. This concludes the proof of Theorem 1. □

Theorem 2. *There exists a function f for which there is no 4-round (resp. 2 round) protocol computing f in the plain model (resp. with CRS and PKI) that simultaneously realises– (1) $\mathcal{F}_{\mathsf{fair}}$ when $t < n/2$ parties are corrupted (2) $\mathcal{F}_{\mathsf{ua}}$ when $s < n$ parties are corrupted. In the former setting (plain model), we assume black-box simulation.*

Proof. We start with the proof in the plain model, followed by the proof with CRS and PKI. We assume for contradiction that there exists a 4-round $(\mathsf{fn}|\mathsf{ua})$-BoBW protocol (with black-box simulation) in the plain model. Then, it follows from Theorem 1 that there must exist a 3-round 2-party maliciously-secure OT protocol with black-box simulation in the plain model. We point that this OT derived as per the transformation of Theorem 1 is a bidirectional OT, where each round consists of messages from both the OT sender and the receiver. Using the round-preserving transformation from bidirectional OT to alternating-message OT (where each round consists of a message from only one of the two parties) [34], we contradict the necessity of 4 rounds for alternating OT in the plain model with black-box simulation [33]. This completes the proof for plain model.

Next, we assume for contradiction that there exists a 2-round $(\mathsf{fn}|\mathsf{ua})$-BoBW MPC protocol in the presence of CRS and PKI. Then, it follows from Theorem

1 that there exists 1-round OT protocol in this model. We have arrived at a contradiction since non-interactive OT is impossible to achieve in a model with input-independent setup that includes CRS and PKI (notably 1-round OT constructions which use an *input-dependent* PKI setup such as [62] exist). To be more specific, a 1-round OT protocol would be vulnerable to the following residual attack by a corrupt receiver P_R: P_R can participate in the OT protocol with input c and get the output m_c at the end of the 1-round OT protocol (where (m_0, m_1) denote the inputs of sender P_S). Now, since the Round 1 messages of P_S and P_R are independent of each other, P_R can additionally plug in his input as being $(1-c)$ to locally compute m_{1-c} as well which is a violation of sender's security as per the ideal OT functionality. □

3 Upper Bounds for (fn|ua)-BoBW

In this section, we construct two upper bounds for the $(\mathsf{fn}|\mathsf{ua})$-BoBW class.

Our upper bounds take 5 and 3 rounds in the plain model and in the CRS setting respectively, tightly matching the lower bounds presented in Section 2. We begin with a general compiler that transforms any n-party r-round actively-secure MPC protocol achieving ua in dishonest majority into an $(r+1)$-round $(\mathsf{fn}|\mathsf{ua})$-BoBW protocol.

3.1 The Compiler

At a high-level, our compiler uses the compiler of [47] and a form of authentication used in the Information Checking Protocol (ICP) primitive of [48,49] and unanimously identifiable commitments (UIC) of [50]. Drawing motivation from the compiler of [47] from ua to fn in the honest majority setting, our compiler uses the given r-round protocol achieving ua security to compute an "authenticated" secret sharing with a threshold of t of the output y and reconstruct the output y during the $(r+1)^{\text{th}}$ round. The correct reconstruction is guaranteed thanks to unanimity offered by the underlying protocol and the authentication mechanism that makes equivocation of a share hard. Alternatively termed as error-correcting secret sharing (ECSS) [47], the authenticated secret sharing was instantiated with pairwise information-theoretic or one-time MAC as a form of authentication. This, when taken as is in our case, achieves fairness in the honest majority setting as in the original transformation. The sharing threshold t ensures that the shares of the honest set, consisting of at least $t+1$ parties, dictate the reconstruction of the output, no matter whether the corrupted parties cooperate or not. The pairwise MAC, however, makes it challenging to maintain unanimity in the dishonest majority case of the transformed protocol, where a corrupt party may choose to verify its share to *selected* few enabling their output reconstruction. This seems to call for a MAC that cannot be manipulated partwise to keep the verifiers on different pages. A possible approach to achieve the property of public verifiability is by means of digital signatures i.e. each party obtains a signed output share which it broadcasts during reconstruction and can

be verified by remaining parties using a common public verification key (that the parties obtain as part of the output of the r-round protocol achieving ua). Alternately, if the form of authentication used in the ICP of [48,49] and UIC of [50] is used, then digital signatures can be avoided and the compiler (transforming any n-party r-round actively-secure MPC protocol achieving ua in dishonest majority into an $(r+1)$-round $(\mathsf{fn}|\mathsf{ua})$-BoBW protocol) achieves the desirable property of being information-theoretic (i.t).

Achieving i.t security is a worthwhile goal, as substantiated by its extensive study in various settings including those where achieving this desirable security notion demands additional tools. For instance, there are well-known results circumventing the impossibility of achieving i.t security in dishonest majority by relying on additional assistance such as tamper-proof hardware tokens [63,50] and Physically Uncloneable Functions (PUFs) [64,65]. Having an i.t compiler opens up the possibility of achieving i.t BoBW MPC by plugging in an i.t. secure dishonest majority protocol (say, that uses hardware tokens / PUFs or other assistance) in the compiler. The details of the i.t compiler appear in [54].

3.2 The Upper Bounds

Building our round-optimal $(\mathsf{fn}|\mathsf{ua})$-BoBW protocols in the plain and CRS model involves constructing 2 and 4 round protocols that achieve ua security against dishonest majority in the respective models. Such protocols when plugged in our compiler of Section 3.1 would directly yield the round-optimal $(\mathsf{fn}|\mathsf{ua})$-BoBW protocols.

In the CRS setting, the known 2-round protocols of [30,31] achieve ua and thereby lead to a 3-round $(\mathsf{fn}|\mathsf{ua})$-BoBW protocol, matching the lower bound. Unfortunately, the existing 4-round MPC protocols in the plain model relying on polynomial-time assumptions [16,17,34], in spite of convenient use of broadcast, only satisfy the weaker notion of sa. We demonstrate how the protocol of [16] and [17,34] can be tweaked to achieve ua in the full version [54]. With respect to the above mentioned ua protocols, our $(\mathsf{fn}|\mathsf{ua})$-BoBW MPC protocols rely on the assumption of 2-round OT in the common random/reference string model and 4-round OT in the plain model.

Theorem 3. *Assuming the existence of a 4 (resp., 2) round MPC protocol that realizes $\mathcal{F}_{\mathsf{ua}}$ for upto $n-1$ malicious corruptions in the plain (resp., CRS) model, there exists a 5 (resp., 3)-round MPC protocol in the plain (resp., CRS) model that simultaneously realises– (1) $\mathcal{F}_{\mathsf{fair}}$ when $t < n/2$ parties are corrupted (2) $\mathcal{F}_{\mathsf{ua}}$ when $s < n$ parties are corrupted.*

A minor observation regarding the use of broadcast in our compiler is that we can replace it with point-to-point communication at the expense of relaxing ua to sa security in the dishonest majority setting.

Security with Identifiability. Our compiler preserves the property of identifiability. Since the underlying dishonest-majority protocols [30,31] can be boosted to achieve identifiable abort (as shown by [66]), the upper bound in the CRS

model achieves identifiable fairness and abort in the honest and dishonest majority setting respectively. With respect to the plain model, we show how security of [17] can be boosted to achieve identifiable abort with minor tweaks, in the full version. This variant, when compiled using our compiler of Section 3.1 would achieve identifiable fairness and abort in the honest and dishonest majority setting respectively.

4 Lower Bounds for (god|ua)-BoBW

In this section, we prove that it is impossible to design a 2-round (god|ua)-BoBW protocol with $t + s < n$ in the CRS model. Note that the necessity of 3 rounds for (god|ua)-BoBW protocol for most values of (n, s, t) follows from the 2-round impossibility of fair MPC for honest majority in the CRS model [40,38,37]. Accounting for the fact that these existing results do not rule out the possibility of 2-round (god|ua)-BoBW MPC for $(t = 1, s > t, n \geq 4)$, we present a *unified proof* that works even for $s > t$, for all values of t (including $t = 1$). Our proof approach deals with adversarial strategies that shuffle between the honest and dishonest majority setting, highlighting the challenge of designing protocols that simultaneously provide different guarantees for different settings. This is in contrast to the existing lower bounds of [40,38,37] which deal only with honest majority setting and single security notion of fn. Lastly, we demonstrate why our proof breaks down in the presence of PKI. Indeed, we construct a 2-round (god|ua)-BoBW protocol assuming CRS and PKI in this work.

Theorem 4. *Let n, t, s be such that $t + s < n$ and $t < n/2$. There exist functions f for which there is no two-round protocol in the CRS model computing f that simultaneously realizes– (1) $\mathcal{F}_{\mathsf{god}}$ when $t < n/2$ parties are corrupted (2) $\mathcal{F}_{\mathsf{ua}}$ when $s < n$ parties are corrupted.*

Proof. We prove the theorem for $n = 4$ parties with $t = 1$ and $s = 2$. The result then can be extended for higher values of n in a natural manner (elaborated in the full version). Let $\mathcal{P} = \{P_1, P_2, P_3, P_4\}$ denote the set of 4 parties and $\mathcal{A}$ may corrupt at most two among them. We prove the theorem by contradiction. We assume that there exists a 2-round (god|ua) BoBW protocol π in the CRS model that can compute the function $f(x_1, x_2, x_3, x_4)$ defined below for P_i's input x_i: $f(x_1, x_2, x_3, x_4) = 1$ if $x_1 = x_2 = 1; 0$ otherwise. By assumption, π achieves god when $t = 1$ parties are corrupt and ua security when $s = 2$ parties are corrupt (satisfying feasibility criteria $t + s < n$).

At a high level, we discuss three adversarial strategies $\mathcal{A}_1, \mathcal{A}_2$ and $\mathcal{A}_3$ of $\mathcal{A}$. While both $\mathcal{A}_1$ and $\mathcal{A}_3$ deal with $t = 1$ corruption with the adversary corrupting P_1, $\mathcal{A}_2$ involves $s = 2$ corruptions where the adversary corrupts $\{P_3, P_4\}$. We consider $\mathcal{A}_i$ strategy as being launched in execution Σ_i $(i \in [3])$ of π. The executions are assumed to be run for the same input tuple $(x_1, x_2, \perp, \perp)$ and the same random inputs (r_1, r_2, r_3, r_4) of the parties. (Same random inputs are considered for simplicity and without loss of generality. The same arguments

hold for distribution ensembles as well.) Our executions and adversarial strategies are sequenced in the order of increasingly more non-cooperating malicious adversaries. Yet, keeping the views of a certain party between two consecutive executions same, we are able to conclude the party would output the correct value even in the face of stronger malicious behaviour. Finally, we reach to the final execution Σ_3 where we show that a party can deduce the output in the end of Round 1 itself. Lastly, we show a strategy for the party to explicitly breach the input privacy of one of the input-contributing parties.

We assume that the communication done in the second round of π is via broadcast alone. This holds without loss of generality since the parties can perform point-to-point communication by exchanging random pads in the first round and then use these random pads to unmask later broadcasts. We use the following notation: Let $\mathsf{p}^1_{i\to j}$ denote the pairwise communication from P_i to P_j in round 1 and b^r_i denote the broadcast by P_i in round r, where $r \in [2], \{i,j\} \in [4]$. These values may be function of CRS as per the working of the protocol. V^ℓ_i denotes the view of party P_i at the end of execution Σ_ℓ ($\ell \in [3]$) of π. Below we describe the strategies $\mathcal{A}_1, \mathcal{A}_2$ and $\mathcal{A}_3$.

- $\mathcal{A}_1$**:** $\mathcal{A}$ corrupts P_1 here. P_1 behaves honestly towards P_2 in Round 1, i.e. sends the messages $\mathsf{p}^1_{1\to 2}, \mathsf{b}^1_1$ as per the protocol. However P_1 does not communicate privately to $\{P_3, P_4\}$ in Round 1. In Round 2, P_1 behaves honestly as per the protocol.
- $\mathcal{A}_2$**:** $\mathcal{A}$ corrupts $\{P_3, P_4\}$ here. $\{P_3, P_4\}$ behave honestly in Round 1 of the protocol. In Round 2, P_k ($k \in \{3,4\}$) acts as per the protocol specification when no *private* message from P_1 is received in Round 1. Specifically, suppose P_k did not receive $\mathsf{p}^1_{1\to k}$ in Round 1. Let $\overline{\mathsf{b}^2_k}$ denote the message that should be sent by P_k as per the protocol in Round 2 in such a scenario. Then as per $\mathcal{A}_2$, corrupt P_k sends $\overline{\mathsf{b}^2_k}$ in Round 2.
- $\mathcal{A}_3$**:** Same as in $\mathcal{A}_1$ and in addition– during Round 2, P_1 simply remains silent i.e. waits to receive the messages from other parties, but does not communicate at all.

Next we present the views of the parties in Σ_1, Σ_2 and Σ_3 in Table 2. Here, $\overline{\mathsf{b}^2_k}$ ($k \in \{3,4\}$) denotes the message that should be sent by P_k according to the protocol in Round 2 in case P_k did not receive any private communication from P_1 in Round 1.

	Σ_1				Σ_2				Σ_3			
	V^1_1	V^1_2	V^1_3	V^1_4	V^2_1	V^2_2	V^2_3	V^2_4	V^3_1	V^3_2	V^3_3	V^3_4
Input	(x_1, r_1)	(x_2, r_2)	r_3	r_4	(x_1, r_1)	(x_2, r_2)	r_3	r_4	(x_1, r_1)	(x_2, r_2)	r_3	r_4
R1	$\mathsf{p}^1_{2\to 1}, \mathsf{p}^1_{3\to 1}$ $\mathsf{p}^1_{4\to 1}$; $\mathsf{b}^1_2, \mathsf{b}^1_3, \mathsf{b}^1_4$	$\mathsf{p}^1_{1\to 2}, \mathsf{p}^1_{3\to 2},$ $\mathsf{p}^1_{4\to 2}$; $\mathsf{b}^1_1, \mathsf{b}^1_3, \mathsf{b}^1_4$	$-, \mathsf{p}^1_{2\to 3},$ $\mathsf{p}^1_{4\to 3}$; $\mathsf{b}^1_1, \mathsf{b}^1_2, \mathsf{b}^1_4$	$-, \mathsf{p}^1_{2\to 4},$ $\mathsf{p}^1_{3\to 4}$; $\mathsf{b}^1_1, \mathsf{b}^1_2, \mathsf{b}^1_3$	$\mathsf{p}^1_{2\to 1}, \mathsf{p}^1_{3\to 1}$ $\mathsf{p}^1_{4\to 1}$; $\mathsf{b}^1_2, \mathsf{b}^1_3, \mathsf{b}^1_4$	$\mathsf{p}^1_{1\to 2}, \mathsf{p}^1_{3\to 2},$ $\mathsf{p}^1_{4\to 2}$; $\mathsf{b}^1_1, \mathsf{b}^1_3, \mathsf{b}^1_4$	$\mathsf{p}^1_{1\to 3}, \mathsf{p}^1_{2\to 3},$ $\mathsf{p}^1_{4\to 3}$; $\mathsf{b}^1_1, \mathsf{b}^1_2, \mathsf{b}^1_4$	$\mathsf{p}^1_{1\to 4}, \mathsf{p}^1_{2\to 4},$ $\mathsf{p}^1_{3\to 4}$; $\mathsf{b}^1_1, \mathsf{b}^1_2, \mathsf{b}^1_3$	$\mathsf{p}^1_{2\to 1}, \mathsf{p}^1_{3\to 1}$ $\mathsf{p}^1_{4\to 1}$; $\mathsf{b}^1_2, \mathsf{b}^1_3, \mathsf{b}^1_4$	$\mathsf{p}^1_{1\to 2}, \mathsf{p}^1_{3\to 2},$ $\mathsf{p}^1_{4\to 2}$; $\mathsf{b}^1_1, \mathsf{b}^1_3, \mathsf{b}^1_4$	$-, \mathsf{p}^1_{2\to 3},$ $\mathsf{p}^1_{4\to 3}$; $\mathsf{b}^1_1, \mathsf{b}^1_2, \mathsf{b}^1_4$	$-, \mathsf{p}^1_{2\to 4},$ $\mathsf{p}^1_{3\to 4}$; $\mathsf{b}^1_1, \mathsf{b}^1_2, \mathsf{b}^1_3$
R2	$\mathsf{b}^2_2, \overline{\mathsf{b}^2_3}, \overline{\mathsf{b}^2_4}$	$\mathsf{b}^2_1, \overline{\mathsf{b}^2_3}, \overline{\mathsf{b}^2_4}$	$\mathsf{b}^2_1, \mathsf{b}^2_2, \overline{\mathsf{b}^2_4}$	$\mathsf{b}^2_1, \mathsf{b}^2_2, \overline{\mathsf{b}^2_3}$	$\mathsf{b}^2_2, \overline{\mathsf{b}^2_3}, \overline{\mathsf{b}^2_4}$	$\mathsf{b}^2_1, \overline{\mathsf{b}^2_3}, \overline{\mathsf{b}^2_4}$	$\mathsf{b}^2_1, \mathsf{b}^2_2, \overline{\mathsf{b}^2_4}$	$\mathsf{b}^2_1, \mathsf{b}^2_2, \overline{\mathsf{b}^2_3}$	$\mathsf{b}^2_2, \overline{\mathsf{b}^2_3}, \overline{\mathsf{b}^2_4}$	$-, \overline{\mathsf{b}^2_3}, \overline{\mathsf{b}^2_4}$	$-, \mathsf{b}^2_2, \overline{\mathsf{b}^2_4}$	$-, \mathsf{b}^2_2 \overline{\mathsf{b}^2_3}$

Table 2: Views of P_1, P_2, P_3, P_4 in Σ_1, Σ_2, Σ_3

We now prove a sequence of lemmas to complete our proof. Let y denote the output computed as per the inputs (x_1, x_2) provided by the honest P_1 and P_2.

Let $\mathsf{q} = 1 - \mathsf{negl}(\kappa)$ denote the overwhelming probability with which security of π holds, where the probability is defined over choice of setup and the random coins used by the parties.

Lemma 3. *The view of P_2 is the same in Σ_1 and Σ_2 and it outputs y in both with overwhelming probability.*

Proof. We observe that as per both strategies $\mathcal{A}_1$ and $\mathcal{A}_2$, P_2 receives communication from P_1, P_3, P_4 as per honest execution in Round 1. In Round 2, according to $\mathcal{A}_1$, corrupt P_1 did not send private messages to P_3 and P_4 who therefore broadcast $\overline{\mathsf{b}_3^2}$ and $\overline{\mathsf{b}_4^2}$ respectively as per protocol specification. On the other hand, according to $\mathcal{A}_2$, corrupt P_3 and corrupt P_4 send the same messages respectively as per protocol specification for case when P_3, P_4 receive no private message from P_1 in Round 1. It is now easy to check (refer Table 2) that $\mathsf{V}_2^1 = \mathsf{V}_2^2$. Now, since Σ_1 involves $t = 1$ corruption, by assumption, π must be robust (with overwhelming probability q) and V_2^1 must lead to output computation, say of output y'. Due to view equality, P_2 in Σ_2 must also output y' with probability q. In Σ_2, P_1 and P_2 are honest and their inputs are x_1 and x_2 respectively. Due to correctness of π (which holds with overwhelming probability q) during Σ_2, it must then hold that $y' = y$ i.e. the output computed based on V_2^2 is according to honest P_1's input x_1 during Σ_2, with overwhelming probability $\mathsf{q} \times \mathsf{q} = \mathsf{q}^2$. □

Lemma 4. *The view of P_1 is the same in Σ_2 and Σ_3 and it outputs y in both, with overwhelming probability.*

Proof. An honest P_2 has the same view in both Σ_1 and Σ_2 and outputs y with overwhelming probability as per Lemma 3. As π achieves ua (with probability q) in the presence of $s = 2$ corruptions, when P_2 learns the output in Σ_2, P_1 must learn y in Σ_2 with overwhelming probability $\mathsf{q}^2 \times \mathsf{q} = \mathsf{q}^3$. We now show that P_1's view in Σ_2 and Σ_3 are the same and so it outputs y in Σ_3 with overwhelming probability q^3. First, it is easy to see that the Round 1 communication towards P_1 is as per honest execution in both Σ_2, Σ_3. Next, recall that as per $\mathcal{A}_2$, both corrupt $\{P_3, P_4\}$ send messages in Round 2 according to the scenario when they didn't receive any private communication from P_1 in Round 1. A similar message would be sent by honest $\{P_3, P_4\}$ in Σ_3 who did not receive private message from corrupt P_1 as per $\mathcal{A}_3$. Finally, since corrupt P_1 behaved honestly to P_2 in Round 1 of Σ_3 as per $\mathcal{A}_3$, the Round 2 communication from P_2 is similar to that in execution Σ_2. It is now easy to verify (refer Table 2) that $\mathsf{V}_1^2 = \mathsf{V}_1^3$ from which output y can be computed. □

Lemma 5. *P_2 in Σ_3 should learn the output y by the end of Round 1, with overwhelming probability.*

Proof. Firstly, it follows directly from Lemma 4 and the assumption that protocol π is robust against $t = 1$ corruption that all parties including P_2 must learn output y at the end of Σ_3 with overwhelming probability $\mathsf{q}^3 \times \mathsf{q} = \mathsf{q}^4$. Next, we note that as per strategy $\mathcal{A}_3$, P_1 only communicates to P_2 in Round 1. We argue that the second round communication from P_3, P_4 does not impact P_2's

output computation as follows: we observe that the output y depends only on (x_1, x_2). Clearly, Round 1 messages of P_3, P_4 does not depend on x_1. Next, since there is no private communication to P_3, P_4 from P_1 as per strategy $\mathcal{A}_3$, the only communication that can possibly hold information on x_1 and can impact the round 2 messages of P_3, P_4 is b_1^1. However, since this is a broadcast message, P_2 also holds this by the end of Round 1 itself. Thus, P_2 must be able to compute the output y at the end of Round 1.

In more detail, P_2 can choose randomness r_3, r_4 on behalf of P_3, P_4 to locally emulate their following Round 1 messages $\{\mathsf{p}^1_{3\to 2}, \mathsf{p}^1_{4\to 2}, \mathsf{p}^1_{3\to 4}, \mathsf{p}^1_{4\to 3}, \mathsf{b}^1_3, \mathsf{b}^1_4\}$. Next, P_2 can now simulate P_3's Round 2 message $\overline{\mathsf{b}^2_3}$ which is a function of its view comprising of $\{\mathsf{p}^1_{2\to 3}, \mathsf{p}^1_{4\to 3}, \mathsf{b}^1_1, \mathsf{b}^1_2, \mathsf{b}^1_4\}$ (all of which are available to P_2, where b^1_1 was broadcast by P_1 in Round 1). Similarly, P_2 can locally compute P_4's Round 2 message $\overline{\mathsf{b}^2_4}$. We can thus conclude that P_2's view at the end of Σ_3 comprising of $\{\mathsf{p}^1_{1\to 2}, \mathsf{p}^1_{3\to 2}, \mathsf{p}^1_{4\to 2}, \mathsf{b}^1_1, \mathsf{b}^1_3, \mathsf{b}^1_4, \overline{\mathsf{b}^2_3}, \overline{\mathsf{b}^2_4}\}$ can be locally simulated by him at the end of Round 1 itself from which the output y can be computed. □

Lemma 6. *A corrupt P_2 violates the privacy property of π.*

Proof. The adversary corrupting P_2 participates in the protocol honestly by fixing input $x_2 = 0$. Since P_2 can get the output at the end of Round 1 with overwhelming probability (Lemma 5), it must be true that P_2 can evaluate f locally by plugging in any value of x_2. Now a corrupt P_2 can plug in $x_2 = 1$ locally and learn x_1 (via the output $x_1 \wedge x_2$) with overwhelming probability. In the ideal world, corrupt P_2 must learn nothing beyond the output 0 as it has participated in the protocol with input 0. But in the execution of π (in which P_2 participated honestly with input $x_2 = 0$), P_2 has learnt x_1 with overwhelming probability. This is a breach of privacy as P_2 learns x_1 regardless of his input. □

Hence, we have arrived at a contradiction, completing proof of Theorem 4. □

We draw attention to the fact that Lemma 5 would not hold in the presence of any additional setup such as PKI. With additional setup, P_3, P_4 may possibly hold some private information (such as their secret key in case of PKI used to decode P_1's broadcast message in Round 1) that is not available to P_2. Due to this reason, we cannot claim that P_2 can emulate Round 2 messages of $\{P_3, P_4\}$ locally at the end of Round 1. However, this holds in case of CRS as the knowledge of CRS is available to all parties at the beginning of the protocol.

5 Upper Bounds for (god|ua)-BoBW

In this section, we present three (god|ua)-BoBW MPC protocols, assuming $t+s < n$ which is the feasibility condition for such protocols ([20]) consuming– *a*) 3-rounds with CRS *b*) 2-rounds with an additional PKI setup *c*) 5-rounds in plain model. The first two are round-optimal in light of the lower bound of Section 4 and [32] respectively. The third construction is nearly round-optimal (falls just one short of the 4-round lower bound of [33]). Among our upper bounds, the

construction in the plain model is considerably more involved and uses several new tricks in conjugation with existing techniques.

5.1 (god|ua)-BoBW MPC with Public and Private Setup

To arrive at the final destination, the roadmap followed is: (i) A 2-round MPC achieving ua security is compiled to a 3-round (god|ua)-BoBW MPC protocol, both against a weaker semi-malicious adversary. With the additional provision of PKI, this compiler can be turned to a round-preserving one. (ii) The semi-malicious (god|ua)-BoBW MPC protocols are compiled to malicious ones in CRS setting via the known round-preserving compiler of [51] (using NIZKs). All the involved and resultant constructions are in *broadcast-only* setting. The protocol just with CRS tightly upper bounds the 3-round lower bound presented in Section 4, which accounts for both pair-wise and broadcast channels. The protocol with additional PKI setup works in 2 rounds, displaying the power of PKI and that our lower bound of 3-rounds in Theorem 4 breaks down in the presence of PKI. Yet, this construction is round optimal, in light of the known impossibility of 1-round MPC [32].

5.1.1 3-round (god|ua)-BoBW MPC in semi-malicious setting. Here, we present a generic compiler that transforms any 2-round MPC protocol $\pi_{\mathsf{ua.sm}}$ achieving ua security into a 3-round broadcast-only (god|ua)-BoBW MPC protocol $\pi_{\mathsf{bw.god.sm}}$ assuming $t + s < n$. Our compiler borrows techniques from the compiler of [11] which is designed for the honest majority setting and makes suitable modifications to obtain BoBW guarantees. Recall that a semi-malicious adversary needs to follow the protocol specification, but has the liberty to decide the input and random coins in each round. Additionally, the parties controlled by the semi-malicious adversary may choose to abort at any step. The underlying and the resultant protocol use broadcast as the *only* medium of communication.

To transform $\pi_{\mathsf{ua.sm}}$ to guarantee BoBW security, the compiler banks on the idea of giving out the Round 2 message of $\pi_{\mathsf{ua.sm}}$ in a way that ensures god in case of honest majority. The dishonest majority protocols usually do not provide this feature even against a single corruption, let alone a minority. Mimicking the Round 1 of $\pi_{\mathsf{ua.sm}}$ as is, $\pi_{\mathsf{bw.god.sm}}$ achieves this property by essentially giving out a secret sharing of the Round 2 messages of $\pi_{\mathsf{ua.sm}}$ with a threshold of s. When at most t parties are corrupt, the set of $s+1$ honest parties pool their shares to reconstruct Round 2 messages of $\pi_{\mathsf{ua.sm}}$ and compute the output robustly as in $\pi_{\mathsf{ua.sm}}$. This idea is enabled by encoding (i.e. garbling) the next message functions of the second round of $\pi_{\mathsf{ua.sm}}$ and secret-sharing their encoding information using a threshold of s in Round 2 and reconstructing the appropriate input labels in the subsequent round. The next-message circuit of a party P_i hard-codes Round 1 broadcasts of $\pi_{\mathsf{ua.sm}}$, P_i's input and randomness and the default input and randomness of all the other parties. It takes n flags as input, the j^{th} one indicating the alive/non-alive status of P_j. P_j turning non-alive (aborting) translates to the j^{th} flag becoming 0 in which case the circuit makes sure P_j's default input

is taken for consideration by internally recomputing P_j's first round broadcast and subsequently using that to compute the Round 2 message of P_i. Since the flag bits become public by the end of Round 2 (apparent as broadcast is the only mode of communication), the parties help each other by reconstructing the correct label, enabling all to compute the garbled next-message functions of all the parties and subsequently run the output computation of $\pi_{\mathsf{ua.sm}}$. The agreement of the flag bits further ensures output computation is done on a unique set of inputs. The transfer of the shares in broadcast-only setting is enabled via setting up a (public key, secret key) pair in the first round by every party. Broadcasting the encrypted shares emulates sending the share privately. This technique of garbled circuits computing the augmented next-message function (taking the list of alive (non-aborting) parties as input) followed by reconstruction of the appropriate input label was used in the work of [11] for the honest majority setting. The primary difference in our compiler is with respect to the threshold of the secret-sharing of the labels, to ensure BoBW guarantees. The formal description of protocol $\pi_{\mathsf{bw.god.sm}}$, its security and correctness proofs appear in the full version. We only state the theorems for correctness and security below.

Theorem 5. *Protocol $\pi_{\mathsf{bw.god.sm}}$ is correct, except with negligible probability.*

Theorem 6. *Let (n, s, t) be such that $s + t < n$. Let $\pi_{\mathsf{ua.sm}}$ realises $\mathcal{F}_{\mathsf{ua}}$ for upto $n-1$ semi-malicious corruptions. Then protocol $\pi_{\mathsf{bw.god.sm}}$ realises– (i) $\mathcal{F}_{\mathsf{god}}$ when at most $t < n/2$ parties are corrupt and (ii) $\mathcal{F}_{\mathsf{ua}}$ when at most $s < n$ parties are corrupt, semi-maliciously in both cases. It takes 3 rounds, assuming that $\pi_{\mathsf{ua.sm}}$ takes 2 rounds.*

5.1.2 2-round (god|ua)-BoBW MPC in semi-malicious setting. The compiler of the previous section can be made round preserving by assuming pair-wise channels or alternately, PKI. The main difference lies in preponing the actions of Round 2 of $\pi_{\mathsf{bw.god.sm}}$ to Round 1, by exploiting the presence of private channels or PKI. We describe these extensions that can be used to obtain a 2-round semi-malicious (god|ua)-BoBW MPC assuming pair-wise channels (protocol $\phi_{\mathsf{bw.god.sm}}$) or alternately, PKI (protocol $\psi_{\mathsf{bw.god.sm}}$) in the full version.

5.1.3 The upper bounds with public and private setup The 2-round semi-malicious broadcast-only protocol of [30,31] can be plugged in as $\pi_{\mathsf{ua.sm}}$ in our compilers from previous sections to directly yield a 3-round broadcast-only protocol $\pi_{\mathsf{bw.god.sm}}$, 2-round protocol $\phi_{\mathsf{bw.god.sm}}$ that uses both broadcast and pairwise-private channels and 2-round broadcast-only protocol $\psi_{\mathsf{bw.god.sm}}$ assuming PKI, all in the semi-malicious setting. Next, the compiler of [51] that upgrades any broadcast-only semi-malicious protocol to maliciously-secure by employing NIZKs, can be applied on $\pi_{\mathsf{bw.god.sm}}$ and $\psi_{\mathsf{bw.god.sm}}$ to yield a 3-round (god|ua)-BoBW protocol in the CRS model and a 2-round (god|ua)-BoBW protocol given both CRS and PKI. Note that the compiler of [51] works only for broadcast-only protocols and cannot be used to boost security of $\phi_{\mathsf{bw.god.sm}}$ to malicious setting (details appear in full version). Assumption wise, our upper bound

constructions rely on 2-round semi-malicious oblivious transfer and NIZK in the common random/reference string model upon using the protocols of [30,31] to realize $\pi_{\mathsf{ua.sm}}$. The formal description of the (god|ua)-BoBW upper bounds with public and private setup appear in the full version. We state the theorem below.

Theorem 7. *Let (n, s, t) be such that $s + t < n$. Assuming the existence of a 3-round (resp., 2-round with PKI) broadcast-only semi-malicious* (god|ua)*-BoBW MPC and NIZKs, there exists a 3 (resp., 2)-round MPC protocol in the presence of CRS (resp., CRS and PKI) that simultaneously achieves (i) $\mathcal{F}_{\mathsf{god}}$ when at most $t < n/2$ parties are corrupt and (ii) $\mathcal{F}_{\mathsf{ua}}$ when at most $s < n$ parties are corrupt, maliciously in both cases.*

Security with Identifiability. Since the compiler of [51] uses NIZKs to prove correctness of each round, it offers identifiability. Thus our maliciously-secure (god|ua)-BoBW protocols achieve the stronger notion of identifiable abort in case of dishonest majority, with no extra assumption. A minor observation is that we can replace the last round broadcast with point-to-point communication at the expense of relaxing ua to sa security in the dishonest majority setting.

5.2 Upper Bound for (god|ua)-BoBW MPC in Plain Model

In this section, we present a 5-round (god|ua)-BoBW protocol in the plain model. For our construction, we resort to the compiler of [31] that transforms any generic $(k-1)$-round delayed-semi-malicious MPC protocol to a k-round malicious MPC protocol for any $k \geq 5$. Our 5-round construction comes in two steps: a) first, we show that our 3-round semi-malicious protocol $\pi_{\mathsf{bw.god.sm}}$ (described in Section 5.1.1) is delayed-semi-maliciously secure (refer full version for proof) and then b) we plug in this 3-round BoBW protocol in a modified compiler of [31] that carries over the BoBW guarantees, while the original compiler works for security with abort. Our final 5-round compiled protocol faces several technical difficulties in the proof, brought forth mainly by the need to continue the simulation in case the protocol must result in god, which needs deep and non-trivial redressals. The techniques we use to tackle the challenges in simulation are also useful in constructing a 4-round (god|ua)-BoBW protocol based on sub-exponentially secure trapdoor permutations and ZAPs. We give a sketch of this construction in the full version (built upon the protocol of [53]) as a step towards resolving the open question of proving the impossibility or alternately constructing a 4-round (god|ua)-BoBW protocol under polynomial-time assumptions.

5.2.1 The compiler of [31].

Substituting $k = 5$, we recall the relevant details of the compiler of [31] that transforms a 4-round delayed-semi-malicious protocol ϕ_{dsm} to a 5-round maliciously-secure protocol π achieving security with abort. The tools used in this compiler appears in Fig 1. Each party commits to her input and randomness using a 2-round statistically binding commitment scheme Com in the first two rounds. The four rounds of the delayed-semi-malicious protocol

ϕ_{dsm} are run as it is in Round $1, 2, 4$ and 5 respectively (Round 3 is skipped) with two additional sets of public-coin delayed-input witness indistinguishable proofs (WI). The first set of proofs (WI^1) which is completed by Round 4, is associated with the first 3 rounds of ϕ_{dsm}. In addition to proving honest behaviour in these rounds, this set of proofs enables the simulator of the malicious protocol to extract the inputs of the corrupt parties, in order to appropriately emulate the adversary for the delayed-semi-malicious simulator in the last but one round. The second set of proofs (WI^2) which is completed by Round 5, is associated with proving honest behaviour in *all* rounds of ϕ_{dsm}. To enable the simulator to pass the WI proofs without the knowledge of the inputs of the honest parties, it is endowed with a *cheat* route (facilitated by the *cheating* statement of the WI proof, while the *honest* statement involves proving honest behaviour wrt inputs committed via Com) which requires the knowledge of the trapdoor of the corrupt parties; which the simulator can obtain by rewinding the last 2 rounds of a trapdoor-generation protocol (Trap) run in the first 3 rounds of the final construction. To enable this cheat route of the simulator, the compiler has an additional component, namely 4-round non-malleable commitment NMCom run in Rounds 1 - 4. We refer to the full version for further details of the compiler.

Tools used in the compiler [31]

- A $(k-1)$-round delayed-semi-malicious protocol ϕ_{dsm} for computing a function f.
- A 2-message statistically binding commitment scheme Com from one-way functions.
- A 3-round protocol Trap to set up a trapdoor between a sender (S) and a receiver (R) as the following sequence of rounds:
  - **R1:** S samples a signing and verification key pair $(\mathsf{sk}, \mathsf{vk})$ of a signature scheme and sends vk to R.
  - **R2:** R sends a random message $m \leftarrow \{0,1\}^\lambda$ to S.
  - **R3:** S computes a signature σ on m using sk and sends σ to R who accepts if (m, σ) is valid w.r.t. vk.

  A valid trapdoor td w.r.t. a verification key vk constitutes of (m, σ, m', σ') such that $m' \neq m$ and σ and σ' are valid signatures of messages m and m' respectively corresponding to vk.
- A 4-round non-malleable commitment scheme NMCom.
- A 4-round public-coin delayed-input witness indistinguishable proof WI.

Fig. 1: Tools used in the compiler of [31]

Next, we give an overview of the simulator $\mathcal{S}$ (details appear in [31]) for the 5-round compiled protocol π that uses the simulator $\mathcal{S}_\phi$ of the underlying 4-round protocol ϕ_{dsm}. To emulate the ideal-world adversary corrupting parties in set $\mathcal{C}$, $\mathcal{S}$ invokes the malicious adversary $\mathcal{A}_\pi$ and simulates a real execution of π for $\mathcal{A}_\pi$ by acting on behalf of the honest parties in set $\mathcal{H}$. Recall that the delayed-semi-malicious security of ϕ_{dsm} guarantees that it is secure against an adversary $\mathcal{A}_\phi$ who can choose to behave arbitrarily in the protocol as long as it writes a valid witness (which consists of an input randomness pair $(\{x_i, r_i\}_{i \in \mathcal{C}})$ on behalf of

all corrupt parties) on the witness tape of the simulator $\mathcal{S}_\phi$ in the penultimate round such that the witness (x, r) can justify *all* the messages sent by him. In order to avail the services of $\mathcal{S}_\phi$, $\mathcal{S}$ needs to transform the malicious adversary $\mathcal{A}_\pi$ to a delayed-semi-malicious adversary $\mathcal{A}_\phi$ i.e. it needs a mechanism to write (x, r) on the witness tape of $\mathcal{S}_\phi$. This is enabled via extraction of witness i.e. $\{x_i, r_i\}_{i\in\mathcal{C}}$ from the WI^1 proofs sent by $\mathcal{A}_\pi$ as the prover via rewinding its last two rounds (Round $3, 4$ of π).

Apart from the above set of rewinds for extraction of corrupt parties' inputs, another set of rewinds is required for the following reason: Consider messages of honest parties simulated by $\mathcal{S}_\phi$ that are used by $\mathcal{S}$ to interact with $\mathcal{A}_\pi$ during the execution of π. Here, $\mathcal{S}$ cannot convince $\mathcal{A}_\pi$ in the two sets of WI proofs that these messages are honestly generated. Hence, he opts for the route of the *cheating* statement of the WI proofs which requires the knowledge of the trapdoor of the corrupt parties. The trapdoor of a party, say P_i consists of *two* valid message-signature pairs with respect to the verification key of P_i (described in Fig 1). The simulator extracts the trapdoor of parties in $\mathcal{C}$ by rewinding the adversary $\mathcal{A}_\pi$ in Rounds 2 and 3 till he gets an additional valid message-signature pair. The trapdoor has been established this way to ensure that only the simulator (and not the adversary) is capable of passing the proofs via the *cheating* statement.

Finally, we point that the two sets of rewinds (Round 2-3 and Round 3-4 of π) can be executed by $\mathcal{S}$ while maintaining that the interaction with $\mathcal{S}_\phi$ is *straight-line* since Round 3 of the compiled protocol is 'dummy' i.e. does not involve messages of ϕ_{dsm}. This 'dummy' round is crucial to avoid rewinding of messages in ϕ_{dsm}. Since there are no messages of ϕ_{dsm} being sent in Round 3, $\mathcal{S}$ can simply replay the messages of ϕ_{dsm} (obtained via $\mathcal{S}_\phi$) to simulate Round 2 and Round 4 of π during the rewinds.

5.2.2 Our 5-round BoBW construction. Our final goal of a $(\mathsf{god}|\mathsf{ua})$-BoBW protocol $\pi_{\mathsf{bw.god.plain}}$ is obtained by applying the compiler of [31] to our delayed-semi-malicious-secure $(\mathsf{god}|\mathsf{ua})$-BoBW protocol $\pi_{\mathsf{bw.god.sm}}$ (described in Section 5.1.1) with slight modifications. Broadly speaking, to preserve the BoBW guarantees from semi-malicious to malicious setting upon applying the compiler, the malicious behaviour of corrupt P_i in the compiled protocol is translated to an analogous scenario when semi-malicious P_i aborts (stops communicating) in the underlying protocol $\pi_{\mathsf{bw.god.sm}}$. Towards this, we make the following modification: Recall from the construction of $\pi_{\mathsf{bw.god.sm}}$ that each party P_i is unanimously assigned a boolean indicator i.e. flag_i by the remaining parties which is initialized to 1 and is later set to 0 if P_i aborts (stops) in the first two rounds. Accounting for malicious behavior, we now require the value of flag_i to be decided based on not just P_i's decision to abort in a particular round but also on whether he misbehaves in the publicly-verifiable Trap protocol or WI proofs. Specifically, if P_i misbehaves in Trap or the first set of proofs WI^1 with P_i as prover fails, flag_i is set to 0 (analogous to P_i aborting in Round 1 or 2 of $\pi_{\mathsf{bw.god.sm}}$). Further, if the second set of proofs WI^2 with P_i as prover fails, then the last round message of P_i is discarded (analogous to P_i aborting in last round of $\pi_{\mathsf{bw.god.sm}}$).

Next, we point that in our compiled protocol, the 3 rounds of the underlying semi-malicious protocol $\pi_{\mathsf{bw.god.sm}}$ are run in Rounds 1, 4 and 5 respectively. As opposed to compiler of [31] which needed a single 'dummy' round on top of the delayed-semi-malicious protocol, we face an additional simulation technicality (elaborated in the next section) that demands two 'dummy' rounds. This could be enabled while maintaining the round complexity of 5, owing to our 3 (and not 4) round delayed semi-malicious protocol. Furthermore, as described earlier, in order to simulate the WI proofs on behalf of an honest prover towards some corrupt verifier P_i, the simulator requires the knowledge of the trapdoor of P_i which would be possible only if P_i is alive (has not aborted) during the rounds in which trapdoor extraction occurs i.e. Round 2 and Round 3. While the simulator of [31] simply aborts incase any party aborts, the simulator of our BoBW protocol cannot afford to do so as god must be achieved even if upto $t < n/2$ parties abort. We handle this by adding a supplementary condition in our construction, namely, a prover needs to prove the WI proofs only to verifiers who have been alive until the round in consideration. This completes the description of the modifications of our compiler over [31]. The round-by-round interplay of the different components is given in Table 3. We present our 5-round (god|ua)-BoBW MPC protocol $\pi_{\mathsf{bw.god.plain}}$ (incorporating the above modifications) in the plain model in Fig 2-3.

	$\pi_{\mathsf{bw.god.sm}}$	Com	Trap	NMCom	WI^1	WI^2
Round 1	R1	R1	R1	R1	R1	
Round 2		R2	R2	R2	R2	R1
Round 3			R3	R3	R3	R2
Round 4	R2			R4	R4	R3
Round 5	R3					R4

Table 3: $\pi_{\mathsf{bw.god.plain}}$

5.2.3 Proof-sketch for 5-round (god|ua)-BoBW protocol. The simulator for the compiler of [31] runs in different stages. Plugging it for our 5-round (god|ua)-BoBW construction with appropriate modifications, we present a high-level overview of the simulation. Let $\mathcal{S}_{\mathsf{bw.god.plain}}$ and $\mathcal{S}_{\mathsf{bw.god.sm}}$ denote the simulators corresponding to $\pi_{\mathsf{bw.god.plain}}$ and the underlying delayed semi-malicious protocol $\pi_{\mathsf{bw.god.sm}}$ respectively. Stage 1 involves running the first three rounds with the following changes compared to the real-execution of the protocol: a) Commit to 0 in Com instances (run in Round 1, 2) involving honest party as committer. b) Invoke the simulator for the semi-malicious protocol, $\mathcal{S}_{\mathsf{bw.god.sm}}$ to generate the first message of $\pi_{\mathsf{bw.god.sm}}$ in Round 1 on behalf of honest parties. The rest of the actions in Round 1 - 3 on behalf of honest parties are emulated by $\mathcal{S}_{\mathsf{bw.god.plain}}$ as per protocol specifications. Note that the simulator wrt compiler in [31] proceeds beyond the first stage only when the adversary did not cause an *abort* on behalf of any corrupt party in Stage 1. Else, it aborts. This works out because their protocol promises security with abort and hence, simply terminates if a party aborts. However our protocol, in case of honest majority, promises god with the output being computed on the actual input of the parties who have been alive till last but one round. To accommodate this, $\mathcal{S}_{\mathsf{bw.god.plain}}$ cannot simply afford to terminate in case a corrupt party aborts. It needs to continue the simulation with respect to corrupt parties who are alive, which de-

5-round Malicious (god|ua)-BoBW MPC Protocol $\pi_{\mathsf{bw.god.plain}}$ from 3-round delayed-semi-malicious BoBW protocol ϕ_{dsm}

Primitives: Tools mentioned in Fig 1 with ϕ_{dsm} instantiated with $\pi_{\mathsf{bw.god.sm}}$ (described in Section 5.1.1).

Round 1. Each party $P_i, i \in [n]$ does the following with $P_j, j \in [n] \setminus \{i\}$:
- Execute Round 1 of ϕ_{dsm}. Initialize $\mathsf{flag}_k = 1$ for all $k \in [n]$ as per ϕ_{dsm}.
- Run Round 1 of $\mathsf{Com}_{i\to j}$ to commit to his input and randomness (x_i, r_i) to P_j. Let the commitment be denoted by $c_{i\to j}$. Run Round 1 of $\mathsf{Com}_{j\to i}$ (where P_j acts as committer) as receiver.
- Run Round 1 of $\mathsf{Trap}_{i\to j}$ as sender, with $\mathsf{vk}_{i\to j}$ denoting the verification key.
- Run Round 1 of $\mathsf{NMCom}_{i\to j}$ as committer and Round 1 of $\mathsf{NMCom}_{j\to i}$ as receiver (with P_j as committer).
- Run Round 1 of $\mathsf{WI}^1_{i\to j}$ as prover and Round 1 of $\mathsf{WI}^1_{j\to i}$ as verifier (with P_j as prover).

Round 2. Each party $P_i, i \in [n]$ does the following with $P_j, j \in [n] \setminus \{i\}$:
- Run Round 2 of $\mathsf{Com}_{i\to j}$ and $\mathsf{Com}_{j\to i}$.
- Run Round 2 of $\mathsf{Trap}_{j\to i}$ (as receiver).
- Run Round 2 of $\mathsf{NMCom}_{i\to j}$ and $\mathsf{NMCom}_{j\to i}$.
- Run Round 2 of $\mathsf{WI}^1_{i\to j}$ and $\mathsf{WI}^1_{j\to i}$. Also, run Round 1 of $\mathsf{WI}^2_{i\to j}$ as prover and Round 1 of $\mathsf{WI}^2_{j\to i}$ as verifier (with P_j as prover).
- Set $\mathsf{flag}_j = 0$ if P_j aborts in Round 1 or Round 2.

Round 3. Each party $P_i, i \in [n]$ does the following with $P_j, j \in [n] \setminus \{i\}$:
- Run Round 3 of $\mathsf{Trap}_{i\to j}$ (as sender).
- Run Round 3 of $\mathsf{NMCom}_{i\to j}$ and $\mathsf{NMCom}_{j\to i}$.
- Run Round 3 of $\mathsf{WI}^1_{i\to j}$ and $\mathsf{WI}^1_{j\to i}$. Also, run Round 2 of $\mathsf{WI}^2_{i\to j}$ and $\mathsf{WI}^2_{j\to i}$.
- Set $\mathsf{flag}_j = 0$ if either P_j aborts in Round 3 or if there exists a $k \in [n], k \neq j$ such that the message-signature pair (m, σ) in $\mathsf{Trap}_{j\to k}$ is not valid w.r.t. $\mathsf{vk}_{j\to k}$. Broadcast enables everyone to agree on this.

Fig. 2: The Modified Compiler for (god|ua)-BoBW MPC (Part 1)

mands rewinding. It can thus be inferred that $\mathcal{S}_{\mathsf{bw.god.plain}}$ must always proceed to rewinds unless all the corrupt parties are exposed by adversary in Stage 1.

The second and the fourth stage, in particular, are concerned with rewinding of the adversary to enable $\mathcal{S}_{\mathsf{bw.god.plain}}$ to extract some information. In Stage 2, the adversary is reset to the end of Round 1 and Rounds 2, 3 are rewound in order to enable $\mathcal{S}_{\mathsf{bw.god.plain}}$ to extract trapdoor of corrupt parties. In more detail, consider $\mathsf{Trap}_{j\to i}$ executed between corrupt sender P_j and honest P_i wrt verification key $\mathsf{vk}_{j\to i}$. Now, $\mathcal{S}_{\mathsf{bw.god.plain}}$ acting on behalf of P_i computes the trapdoor of P_j wrt $\mathsf{vk}_{j\to i}$ to be *two* message-signature pairs constituted by one obtained in Stage 1 and the other as a result of rewinding in Stage 2 (note that both signatures are wrt $\mathsf{vk}_{j\to i}$ sent in Round 1 of $\mathsf{Trap}_{j\to i}$; rewinds involve only Round 2, 3). To enable continuation of the simulation after Stage 2, which requires the knowledge of the trapdoors of corrupt parties who are alive, the logical *halt* condition for the rewinds is: *stop when you have enough!* This translates to- stop at the ℓ^{th} rewind if a valid trapdoor has been obtained for the set of corrupt parties alive across the ℓ^{th} rewind. Since the ℓ^{th} (last) rewind is expected to provide one valid (m, σ) pair (i.e. message, signature pair) out of two required for the trapdoor,

5-round Malicious (god|ua)-BoBW MPC Protocol $\pi_{\mathsf{bw.god.plain}}$ from 3-round delayed-semi-malicious BoBW protocol ϕ_{dsm}

Round 4. Each party $P_i, i \in [n]$ does the following with $P_j, j \in [n] \setminus \{i\}$:

- Execute Round 2 of ϕ_{dsm}.
- Run Round 4 of $\mathsf{NMCom}_{i\to j}$ in order to commit to a random string $s^0_{i\to j}$. Run Round 4 of $\mathsf{NMCom}_{j\to i}$ as receiver. Additionally, send another random string $s^1_{i\to j}$ on clear to P_j.
- Run Round 4 of $\mathsf{WI}^1_{j\to i}$ as verifier. If $\mathsf{flag}_j = 1$, run Round 4 of $\mathsf{WI}^1_{i\to j}$ to prove to P_j the correctness of the first 2 messages of ϕ_{dsm}. In detail, $\mathsf{WI}^1_{i\to j}$ proves correctness of one of the following statements: (1) *Honest Statement:* P_i has correctly generated the first 2 messages of ϕ_{dsm} using the input and randomness committed in $c_{i\to j}$. (2) *Cheating Statement:* XOR of the share $s^0_{i\to j}$ committed to in $\mathsf{NMCom}_{i\to j}$ and the share $s^1_{i\to j}$ is a valid trapdoor $\mathsf{td}_{j\to i}$ w.r.t. verification key $\mathsf{vk}_{j\to i}$.
- Run Round 3 of $\mathsf{WI}^2_{i\to j}$ and $\mathsf{WI}^2_{j\to i}$.
- Set $\mathsf{flag}_j = 0$ if either P_j aborts in Round 4 or if there exists a $k \in [n], k \neq j$ such that $\mathsf{WI}^1_{j\to k}$ leads to *reject*. Public verifiability of WI proofs enables this.

Round 5. Each party $P_i, i \in [n]$ does the following $P_j, j \in [n] \setminus \{i\}$:

- Execute Round 3 of ϕ_{dsm}.
- Run Round 4 of $\mathsf{WI}^2_{j\to i}$ as verifier. If $\mathsf{flag}_j = 1$, run Round 4 of $\mathsf{WI}^2_{i\to j}$ to prove to P_j the correctness of *all* messages of ϕ_{dsm} that he broadcasted. In detail, $\mathsf{WI}^2_{i\to j}$ proves correctness of one of the following statements: (1) *Honest Statement:* P_i has correctly generated *all* messages of ϕ_{dsm} using the input and randomness committed in $c_{i\to j}$ (2) *Cheating Statement:* XOR of the share $s^0_{i\to j}$ committed to in $\mathsf{NMCom}_{i\to j}$ and the share $s^1_{i\to j}$ is a valid trapdoor $\mathsf{td}_{j\to i}$ w.r.t. verification key $\mathsf{vk}_{j\to i}$.
- **Output Computation:** If any proof $\mathsf{WI}^2_{j\to k}$ is not accepting for any $k \in [n], k \neq j$, discard the message from P_j. Compute the output as per ϕ_{dsm}.

Fig. 3: The Modified Compiler for (god|ua)-BoBW MPC (Part 2)

all that is required is for the corrupt party to have been alive across at least one previous rewind. Let the set of parties alive across i^{th} rewind be denoted by $\mathbb{A}_{i+1}$ ($\mathbb{A}_1$ represents the set of parties that were alive in the execution preceeding the rewinds i.e. after Stage 1), then the condition formalizes to: halt at rewind ℓ if $\mathbb{A}_{\ell+1} \subseteq \mathbb{A}_1 \cup \cdots \cup \mathbb{A}_\ell$.

While this condition seems appropriate, it leads to the following subtle issue. The malicious adversary can exploit this stopping condition by coming up with a strategy to choose the set of aborting and the alive parties (say, according to some unknown distribution D pre-determined by the adversary) such that the final set of alive parties $\mathbb{A}$ in the transcript output by the simulator (when the rewinds halt) will be biased towards the set of parties that were alive in the earlier rewinds. (Ideally the distribution of the set of alive parties when simulator halts should be identical to D). This would lead to the view output by the simulator being distinguishable from the real view. A very similar subtle issue appears in zero-knowledge (ZK) protocol of [52] - While the details of this issue of [52] appear in the full version, we give a glimpse into how their scenario is

analogous to ours below. Consider a basic 4-round ZK protocol with the following skeleton: the verifier commits to a challenge in Round 1 which is subsequently decommitted in Round 3. The prover responds to the challenge in Round 4. At a very high-level, the protocol of [52] follows a cut-and-choose paradigm involving N instances of the above basic protocol. Here, the verifier chooses a random subset $S \subset [N]$ of indices and decommits to the challenges made in those indices in Round 3. Subsequently, the prover completes the ZK protocol for instances with indices in S. The simulator for the zero-knowledge acting on behalf of the honest prover involves rewinds to obtain 'trapdoors' corresponding to the indices in S. However, note that the verifier can choose different S in different rewinds. Therefore, the simulator is in a position to produce an accepting transcript and stop at the ℓ^{th} rewind only when it has trapdoors corresponding to all indices in S chosen by the adversary during the ℓ^{th} rewind. However, if the simulation is stopped at the execution where the above scenario happens for the 'first' time, their protocol suffers an identical drawback as ours. In particular, the malicious verifier can choose the set of indices S in a manner that the distribution of the views output by the simulator is not indistinguishable from the real view. Drawing analogy in a nutshell, the set of indices chosen by the malicious verifier is analogous to the set of alive corrupt parties in our context (details in full version). We thereby adopt the solution of [52] and modify our halting condition as: halt at rewind ℓ if $\mathbb{A}_{\ell+1} \subseteq \mathbb{A}_1 \cup \cdots \cup \mathbb{A}_\ell$ **and** $\mathbb{A}_{\ell+1} \nsubseteq \mathbb{A}_1 \cup \cdots \cup \mathbb{A}_{\ell-1}$. [52] gives an elaborate analysis showing why this simulation strategy results in the right distribution. With this change in simulation of Stage 2, the simulation of Stage 3 can proceed identical to [31] which involves simulating the WI^1 proofs via the *fake* statement using the knowledge of trapdoor.

Proceeding to simulation of Stage 4, we recall that the simulator of [31] involves another set of rewinds in Stage 4 which requires to rewind Round 3 and 4 to extract the witness i.e. the inputs and randomness of the corrupt parties from WI^1. Similar to Stage 2, two successful transcripts are sufficient for extraction. Thus, the simulator is in a position to halt at ℓ^{th} rewind if all the corrupt parties that are alive in Stage 4 have been alive across at least one previous rewind. Next, following the same argument as Stage 2, it seems like the *halting* condition for Stage 2 should work, as is, for Stage 4 too.

With this conclusion, we stumbled upon another hurdle elaborated in this specific scenario: Recall that the trapdoors extracted for corrupt parties in Stage 2 are used here to simulate the WI^1 proofs (as described in Stage 3). It is thereby required that $\mathcal{S}_{\mathsf{bw.god.plain}}$ already has the trapdoors for the corrupt parties that are alive in Stage 4. Let $\mathbb{T}$ be the set of trapdoors accumulated at the end of Stage 2. Consider a party, say P_i, which stopped participating in Round 3 of the last rewind ℓ of Stage 2 (P_i was alive till Round 2 of ℓ^{th} rewind). $\mathcal{S}_{\mathsf{bw.god.plain}}$ still proceeds to Stage 4 without being bothered about the trapdoor of P_i (as the halting condition is satisfied). However in Stage 4, when the adversary is reset to the end of Round 2 of ℓ^{th} rewind, P_i came back to life again in Round 3. The simulation of WI^1 proofs with P_i as a verifier will be stuck if $\mathbb{T}$ does not contain the trapdoor for P_i. Hence, it is required to accommodate the knowledge of set

$\mathbb{T}$ during Stage 4. Accordingly $\mathcal{S}_{\mathsf{bw.god.plain}}$ does the following in Stage 4: During each rewind, if a party (say P_i) whose trapdoor is not known becomes alive during Round 3, store the signature sent by P_i in Round 3 (as part of Trap) and go back to Stage 2 rewinds (if P_i's trapdoor is still unknown). Looking ahead, storing the signature of P_i ensures that the missing trapdoor of P_i in $\mathbb{T}$ can cause $\mathcal{S}_{\mathsf{bw.god.plain}}$ to revert to Stage 2 rewinds at most once (if the same scenario happens again i.e. P_i becomes alive in Round 3 during Stage 4 rewinds, then another (message, signature) pair wrt verification key of P_i is obtained in this rewind by $\mathcal{S}_{\mathsf{bw.god.plain}}$; totaling upto 2 pairs which suffices to constitute valid trapdoor of P_i which can now be added to $\mathbb{T}$). Else, if $\mathbb{T}$ comprises of the trapdoor of all the corrupt parties that are alive during the rewind of Stage 4, then adhere to the same halting condition as Stage 2. This trick tackles the above described problematic scenario, while ensuring that the simulation terminates in polynomial time and maintains indistinguishability of views.

Before concluding the section, we highlight two important features regarding the simulation of $\pi_{\mathsf{bw.god.plain}}$: Despite the simulator $\mathcal{S}_{\mathsf{bw.god.plain}}$ reverting to Stage 2 rewinds in some cases (unlike the simulation of [31]), the simulation terminates in polynomial-time since this can occur at most once per corrupt party (as argued above). Lastly, since there is a possibility of reverting back to simulation of Round 2 after simulation of Round 4, we keep an additional 'dummy' Round 2 as well (on top of 'dummy' Round 3 as in [31]) in our construction. This allows us to maintain the invariant that $\mathcal{S}_{\mathsf{bw.god.sm}}$ is never rewound. To be more specific, as there are no messages of underlying semi-malicious protocol being sent in Round 2, 3; even if $\mathcal{S}_{\mathsf{bw.god.plain}}$ needs to return to Stage 2 from Stage 4 (after Round 4 has been simulated by obtaining the relevant message from $\mathcal{S}_{\mathsf{bw.god.sm}}$) and resume the simulation from Stage 2 onwards, the message of $\pi_{\mathsf{bw.god.sm}}$ sent in Round 4 can simply be replayed. We are able to accommodate two dummy rounds while maintaining the round complexity of 5 owing to the privilege that our delayed-semi-malicious protocol is just 3 rounds. This completes the simulation sketch. Assumption wise, our construction relies on 2-round semi-malicious oblivious transfer (a building block of our 3-round delayed-semi-malicious BoBW MPC $\pi_{\mathsf{bw.god.sm}}$). We state the formal theorem below.

Theorem 8. *Let (n, s, t) be such that $s + t < n$. Let $\pi_{\mathsf{bw.god.sm}}$ realises– (i) $\mathcal{F}_{\mathsf{god}}$ when at most $t < n/2$ parties are corrupt and (ii) $\mathcal{F}_{\mathsf{ua}}$ when at most $s < n$ parties are corrupt, delayed-semi-maliciously in both cases. Then $\pi_{\mathsf{bw.god.plain}}$ in the plain model realises– (i) $\mathcal{F}_{\mathsf{god}}$ when at most $t < n/2$ parties are corrupt and (ii) $\mathcal{F}_{\mathsf{ua}}$ when at most $s < n$ parties are corrupt, maliciously in both cases. It takes 5 rounds, assuming that $\pi_{\mathsf{bw.god.sm}}$ takes 3 rounds.*

Proof. The proof which includes the complete description of the simulator, a discussion about its indistinguishability to the real view and its running time appears in the full version [54]. □

Extension to Identifiability. We additionally point that the publicly-verifiable WI proofs render identifiability to our construction. Thus our maliciously-secure

(god|ua)-BoBW protocol achieves the stronger notion of identifiable abort in case of dishonest majority, with no extra assumption. A minor observation is that we can replace the last round broadcast with point-to-point communication in our (god|ua)-BoBW protocol $\pi_{\mathsf{bw.god.plain}}$ at the expense of relaxing ua to sa security in the dishonest-majority setting.

References

1. O. Goldreich, S. Micali, and A. Wigderson, "How to play any mental game or A completeness theorem for protocols with honest majority," in *ACM STOC*, 1987.
2. D. Chaum, I. Damgård, and J. Graaf, "Multiparty computations ensuring privacy of each party's input and correctness of the result," in *CRYPTO*, 1987.
3. A. C. Yao, "Protocols for secure computations (extended abstract)," in *FOCS*, 1982.
4. R. Cleve, "Limits on the security of coin flips when half the processors are faulty (extended abstract)," in *ACM STOC*, 1986.
5. M. Ben-Or, S. Goldwasser, and A. Wigderson, "Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract)," in *ACM STOC*, 1988.
6. D. Chaum, C. Crépeau, and I. Damgård, "Multiparty unconditionally secure protocols (extended abstract)," in *ACM STOC*, 1988.
7. T. Rabin and M. Ben-Or, "Verifiable secret sharing and multiparty protocols with honest majority (extended abstract)," in *ACM STOC*, 1989.
8. D. Beaver, S. Micali, and P. Rogaway, "The round complexity of secure protocols (extended abstract)," in *ACM STOC*, 1990.
9. D. Beaver, "Efficient multiparty protocols using circuit randomization," in *CRYPTO*, 1991.
10. I. Damgård and J. B. Nielsen, "Scalable and unconditionally secure multiparty computation," in *CRYPTO*, 2007.
11. P. Ananth, A. R. Choudhuri, A. Goel, and A. Jain, "Round-optimal secure multiparty computation with honest majority," in *CRYPTO*, 2018.
12. I. Damgård and C. Orlandi, "Multiparty computation for dishonest majority: From passive to active security at low cost," in *CRYPTO*, 2010.
13. S. Garg, C. Gentry, S. Halevi, and M. Raykova, "Two-round secure MPC from indistinguishability obfuscation," in *TCC*, 2014.
14. Z. Brakerski, S. Halevi, and A. Polychroniadou, "Four round secure computation without setup," in *TCC*, 2017.
15. P. Ananth, A. R. Choudhuri, and A. Jain, "A new approach to round-optimal secure multiparty computation," in *CRYPTO*, 2017.
16. S. Halevi, C. Hazay, A. Polychroniadou, and M. Venkitasubramaniam, "Round-optimal secure multi-party computation," in *CRYPTO*, 2018.
17. S. Badrinarayanan, V. Goyal, A. Jain, Y. T. Kalai, D. Khurana, and A. Sahai, "Promise zero knowledge and its applications to round optimal MPC," in *CRYPTO*, 2018.
18. Y. Ishai, E. Kushilevitz, Y. Lindell, and E. Petrank, "On combining privacy with guaranteed output delivery in secure multiparty computation," in *CRYPTO*, 2006.
19. J. Katz, "On achieving the "best of both worlds" in secure multiparty computation," in *ACM STOC*, 2007.

20. Y. Ishai, J. Katz, E. Kushilevitz, Y. Lindell, and E. Petrank, "On achieving the "best of both worlds" in secure multiparty computation," *SIAM J. Comput.*, 2011.
21. J. Katz, S. Myers, and R. Ostrovsky, "Cryptographic counters and applications to electronic voting," in *EUROCRYPT*, 2001.
22. D. G. Nair, V. P. Binu, and G. S. Kumar, "An improved e-voting scheme using secret sharing based secure multi-party computation," *CoRR*, 2015.
23. I. Damgård, M. Geisler, and M. Krøigaard, "Efficient and secure comparison for on-line auctions," in *ACISP*, 2007.
24. K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for privacy-preserving machine learning," in *ACM CCS*, 2017.
25. P. Mohassel and P. Rindal, "Aby^3: A mixed protocol framework for machine learning," in *ACM CCS*, 2018.
26. P. Mohassel and Y. Zhang, "Secureml: A system for scalable privacy-preserving machine learning," in *IEEESP*, 2017.
27. D. Bogdanov, R. Talviste, and J. Willemson, "Deploying secure multi-party computation for financial data analysis - (short paper)," in *FC*, 2012.
28. C. Lucas, D. Raub, and U. M. Maurer, "Hybrid-secure MPC: trading information-theoretic robustness for computational privacy," in *PODC*, 2010.
29. A. Beimel, Y. Lindell, E. Omri, and I. Orlov, "$1/p$-secure multiparty computation without honest majority and the best of both worlds," in *CRYPTO*, 2011.
30. S. Garg and A. Srinivasan, "Two-round multiparty secure computation from minimal assumptions," in *EUROCRYPT*, 2018.
31. F. Benhamouda and H. Lin, "k-round multiparty computation from k-round oblivious transfer via garbled interactive circuits," in *EUROCRYPT*, 2018.
32. S. Halevi, Y. Lindell, and B. Pinkas, "Secure computation on the web: Computing without simultaneous interaction," in *CRYPTO*, 2011.
33. S. Garg, P. Mukherjee, O. Pandey, and A. Polychroniadou, "The exact round complexity of secure computation," in *EUROCRYPT*, 2016.
34. A. R. Choudhuri, M. Ciampi, V. Goyal, A. Jain, and R. Ostrovsky, "Round optimal secure multiparty computation from minimal assumptions." Cryptology ePrint Archive, Report 2019/216, 2019.
35. P. Mukherjee and D. Wichs, "Two round multiparty computation via multi-key FHE," in *EUROCRYPT*, 2016.
36. S. Garg and A. Srinivasan, "Garbled protocols and two-round MPC from bilinear maps," in *FOCS*, 2017.
37. R. Gennaro, Y. Ishai, E. Kushilevitz, and T. Rabin, "On 2-round secure multiparty computation," in *CRYPTO*, 2002.
38. A. Patra and D. Ravi, "On the exact round complexity of secure three-party computation," in *CRYPTO*, 2018.
39. Y. Ishai, R. Kumaresan, E. Kushilevitz, and A. Paskin-Cherniavsky, "Secure computation with minimal interaction, revisited," in *CRYPTO*, 2015.
40. S. D. Gordon, F. Liu, and E. Shi, "Constant-round MPC with fairness and guarantee of output delivery," in *CRYPTO*, 2015.
41. A. Patra and D. Ravi, "On the exact round complexity of secure three-party computation." Cryptology ePrint Archive, Report 2018/481, 2018.
42. A. Patra and D. Ravi, "Beyond honest majority: The round complexity of fair and robust multi-party computation," in *ASIACRYPT*, 2019.
43. S. Badrinarayanan, A. Jain, N. Manohar, and A. Sahai, "Threshold multi-key fhe and applications to round-optimal mpc." Cryptology ePrint Archive, Report 2018/580, 2018.

44. Y. Ishai, E. Kushilevitz, and A. Paskin, "Secure multiparty computation with minimal interaction," in *CRYPTO*, 2010.
45. C. Peikert, V. Vaikuntanathan, and B. Waters, "A framework for efficient and composable oblivious transfer," in *CRYPTO*, 2008.
46. A. D. Santis, G. D. Crescenzo, R. Ostrovsky, G. Persiano, and A. Sahai, "Robust non-interactive zero knowledge," in *CRYPTO*, 2001.
47. Y. Ishai, E. Kushilevitz, M. Prabhakaran, A. Sahai, and C. Yu, "Secure protocol transformations," in *CRYPTO*, 2016.
48. A. Patra, A. Choudhary, and C. P. Rangan, "Simple and efficient asynchronous byzantine agreement with optimal resilience," in *PODC*, 2009.
49. A. Patra and C. P. Rangan, "Communication and round efficient information checking protocol," *CoRR*, 2010.
50. Y. Ishai, R. Ostrovsky, and H. Seyalioglu, "Identifying cheaters without an honest majority," in *TCC*, 2012.
51. G. Asharov, A. Jain, A. López-Alt, E. Tromer, V. Vaikuntanathan, and D. Wichs, "Multiparty Computation with Low Communication, Computation and Interaction via Threshold FHE," in *EUROCRYPT*, 2012.
52. C. Hazay and M. Venkitasubramaniam, "Round-optimal fully black-box zero-knowledge arguments from one-way permutations," in *TCC*, 2018.
53. M. Ciampi and R. Ostrovsky, "Four-round secure multiparty computation from general assumptions." Cryptology ePrint Archive, Report 2019/214, 2019.
54. A. Patra, D. Ravi, and S. Singla, "On the exact round complexity of best-of-both-worlds multi-party computation." Cryptology ePrint Archive, Report 2020/1050, 2020. https://eprint.iacr.org/2020/1050.
55. D. Chaum, "The spymasters double-agent problem: Multiparty computations secure unconditionally from minorities and cryptographically from majorities," in *CRYPTO*, 1989.
56. M. Hirt, U. M. Maurer, and V. Zikas, "MPC vs. SFE : Unconditional and computational security," in *ASIACRYPT*, 2008.
57. S. Halevi, Y. Ishai, E. Kushilevitz, and T. Rabin, "Best possible information-theoretic MPC," in *TCC*, 2018.
58. M. Hirt, C. Lucas, U. Maurer, and D. Raub, "Graceful degradation in multi-party computation (extended abstract)," in *ICITS*, 2011.
59. M. Hirt, C. Lucas, U. Maurer, and D. Raub, "Passive corruption in statistical multi-party computation - (extended abstract)," in *ICITS*, 2012.
60. M. Hirt, C. Lucas, and U. Maurer, "A dynamic tradeoff between active and passive corruptions in secure multi-party computation," in *CRYPTO*, 2013.
61. D. Genkin, S. D. Gordon, and S. Ranellucci, "Best of both worlds in secure computation, with low communication overhead," in *ACNS*, 2018.
62. M. Bellare and S. Micali, "Non-interactive oblivious transfer and spplications," in *CRYPTO*, 1989.
63. V. Goyal, Y. Ishai, A. Sahai, R. Venkatesan, and A. Wadia, "Founding cryptography on tamper-proof hardware tokens," in *TCC*, 2010.
64. R. Ostrovsky, A. Scafuro, I. Visconti, and A. Wadia, "Universally composable secure computation with (malicious) physically uncloneable functions," in *EUROCRYPT*, 2013.
65. C. Brzuska, M. Fischlin, H. Schröder, and S. Katzenbeisser, "Physically uncloneable functions in the universal composition framework," in *CRYPTO*, 2011.
66. R. Cohen, J. A. Garay, and V. Zikas, "Broadcast-optimal two-round MPC," in *EUROCRYPT*, 2020.