Round Optimal Secure Multiparty Computation from Minimal Assumptions

Arka Rai Choudhuri* Michele Ciampi† Vipul Goyal‡ Abhishek Jain§

Rafail Ostrovsky¶

Abstract

We construct a four round secure multiparty computation (MPC) protocol in the plain model that achieves security against any dishonest majority. The security of our protocol only relies on the existence of four round oblivious transfer. This *fully resolves* the round complexity of MPC (w.r.t. black-box simulation) based on minimal assumptions.

All previous results required either a larger number of rounds or stronger assumptions.

*Johns Hopkins University. achoud@cs.jhu.edu

†The University of Edinburgh. mciampi@ed.ac.uk

‡Carnegie Mellon University. goyal@cs.cmu.edu

§Johns Hopkins University. abhishek@cs.jhu.edu

¶University of California, Los Angeles. rafail@cs.ucla.edu

1 Introduction

The ability to securely compute on private datasets of individuals has wide applications of tremendous benefits to society. Secure multiparty computation (MPC) [Yao86, GMW87] provides a solution to the problem of computing on private data by allowing a group of parties to jointly evaluate any function over their private inputs in such a manner that no one learns anything beyond the output of the function.

Since its introduction nearly three decades ago, MPC has been extensively studied along two fundamental lines: necessary *assumptions* [GMW87, Kil88, IPS08], and *round complexity* [GMW87, BMR90, KOS03, KO04, Pas04, PW10, Wee10, Goy11, GMPP16a, ACJ17, BHP17, COSV17b, COSV17a]. For the case of malicious adversaries who may corrupt any number of parties, both of these topics, individually, are by now pretty well understood:

- It is well known that oblivious transfer (OT) is both necessary and sufficient [Kil88, IPS08] for MPC.
- A recent sequence of works have established that *four rounds* are both necessary [GMPP16a] and sufficient [ACJ17, BHP17, BGJ$^+$18, HHPV18] for MPC (with respect to black-box simulation). However, the assumptions required by these works are far from optimal, ranging from sub-exponential hardness assumptions [ACJ17, BHP17] to polynomial hardness of specific forms of encryption schemes [HHPV18] or specific number-theoretic assumptions [BGJ$^+$18].

In this work, we consider the goal of simultaneously minimizing the round complexity and the necessary assumptions for MPC. Namely, we consider the following question:

Can we construct round optimal MPC from minimal assumptions?

That is, we ask whether it is possible to construct four round MPC from four round (malicious-secure) OT. This question was left open in the elegant work of Benhamouda and Lin [BL18] who constructed k-round MPC from k-round OT for $k \geq 5$.

1.1 Our Results

In this work, we resolve the above question in the affirmative.

Theorem 1. *Assuming the existence of four round OT, there exists a four round MPC protocol for any efficiently computable functionality in the plain model.*

Our protocol admits black-box simulation and achieves security against malicious adversaries in the dishonest majority setting.

On the Minimal Assumptions. We study MPC in the standard broadcast communication model, where in each round, every party broadcasts a message to the other parties. In this model, k-round MPC implies k-round *bidirectional* OT, where each round consists of messages from both the OT sender and the receiver. However, it does not necessarily imply k-round OT in the standard, *alternating-message* model for two-party protocols where each round consists of a message from only one of the two parties. In other words, the minimal assumption for k-round MPC is, in fact, k-round bidirectional OT as opposed to alternating-message OT.

Towards establishing the optimality of Theorem 1, we observe that k-round bidirectional OT implies k-round alternating-message OT.

Theorem 2. *k-round bidirectional OT implies k-round alternating-message OT.*

Our transformation is unconditional and generalizes a message rescheduling strategy previously considered by Garg et al. [GMPP16b] for the specific case of three round coin-tossing protocols. In fact, this transformation is even more general and applies to any two-party functionality, with the restriction that only one party learns the output in the alternating-message protocol.

An important corollary of Theorem 2 is that it establishes the *optimality* of the result of Benhamouda and Lin [BL18] who constructed k-round MPC from any k-round alternating-message OT for $k \geq 5$. This result, put together with our main result in Theorem 1 provides a *full resolution* of the fundamental question of basing MPC on minimal assumptions.

In the sequel, for simplicity of exposition, we refer to alternating-message OT as simply OT.

1.2 Technical Overview

In this section, we provide an overview of the key ideas underlying our main result.

How to Enforce Honest Behavior? We start by highlighting the main challenge in the design of four round MPC against malicious adversaries in the plain model. In any candidate four round protocol, a rushing adversary may always choose to *abort* after receiving the messages of honest parties in the last round. At this point, the adversary has already received enough information to obtain the output of the function being computed. This suggests that we must enforce "honest behavior" on the protocol participants within the first three rounds in order to achieve security against malicious adversaries. Indeed, without any such safeguard, a malicious adversary may be able to learn the inputs of the honest parties, e.g., by acting maliciously so as to change the functionality being computed to the identity function.

Zero knowledge (ZK) proofs [GMR89] are a standard tool for enforcing honest behavior on the participants of a protocol. However, ZK proofs with black-box simulation are known to be impossible in three rounds [GK96b]. Indeed, for this reason, all recent works on four round MPC devise non-trivial strategies that only utilize weaker notions of ZK (that are achievable in three or less rounds) to enforce honest behavior within the first three rounds. However, all these approaches end up relying on assumptions that are far from optimal: [ACJ17] and [BHP17] use super-polynomial-time hardness assumptions, [HHPV18] use Zaps [DN00] and affine-homomorphic encryption schemes, and [BGJ$^+$18] use a new notion of promise ZK together with three round strong WI [JKKR17], both of which require specific number-theoretic assumptions.

A Deferred Verification Approach. We use a different approach to address the above challenge. We do not require the parties to explicitly prove honest behavior within the first three rounds. Of course, this immediately opens up the possibility for an adversary to cheat in the first three rounds in such a manner that by observing the messages of the honest parties in the fourth round, it can completely break privacy. To prevent such an attack, we require the parties to "encrypt" their last round message in such a manner that it can only be decrypted by using a "witness" that establishes honest behavior in the first three rounds. In other words, the verification check for honest behavior is deferred to the fourth round.

This raises two immediate questions: what constitutes a valid witness, and how can we implement such a conditional decryption mechanism? Let us start by addressing the first question. A natural idea is to set the input and randomness of a party i used in the first three rounds of the protocol as its witness for establishing honest behavior. However, consider the case where the number of parties is $n > 2$, and the number of corrupted parties is at least $t = 2$. In this case, it is not sufficient for a "decryptor" i to establish its own honest behavior in the first three rounds. Indeed, in this case, a corrupted party who behaved honestly during the first three rounds would be able to decrypt the honest party messages in the fourth round *even when another corrupted party* *behaved maliciously*. Therefore, a valid witness must certify honest behavior by *all* the parties as opposed to a single party. One such witness is simply the input and randomness of all the parties. However, it is not clear how an individual decryptor can obtain such a witness without trivially violating privacy. Indeed, we need a "public" witness that can be obtained by all the parties.

We look towards ZK proof systems to address this issue. Suppose that we require each party to give a *four round* ZK proof of honest behavior. If the ZK proof is delayed-input, it can be parallelized with the rest of the protocol such that the last round of ZK proof occurs in the last round of the MPC protocol. Now, let us set the witness to be the last round messages of all the ZK proofs. This witness can be obtained by any party in the last round, who can then use it for decryption. Indeed, this idea can be made to work if we implement the conditional decryption mechanism using witness encryption [GGSW13]. However, presently witness encryption is only known from non-standard assumptions (let alone OT and injective OWFs).

We, instead, use garbled circuits [Yao86] and four round OT to implement the conditional decryption mechanism. Namely, each party i garbles a circuit that contains hardwired the entire transcript of the first three rounds as well the fourth message of party i. Upon receiving as input a witness $w = w_1, \ldots, w_n$, where w_j is a witness for honest behavior of party j, it outputs the fourth round message. Each party j can encode its witness w_j in the OT receiver messages, and then release its randomness used inside OT in the fourth round so that any other party j' can use it to compute the output of the OT, thereby learning the necessary wire labels for evaluating the garbled circuit sent by party i.

A problem with the above approach is that in a four round OT, the receiver's input must be fixed in the third round. This means that we can no longer use four round ZK proofs, and instead must use *three round* proofs to create public witnesses of honest behavior. But which three round proofs must we use? Towards this, we look to the weaker notion of *promise* ZK introduced by [BGJ+18]. Roughly, promise ZK relaxes the standard notion of ZK by guaranteeing security only against malicious verifiers who do *not* abort. Importantly, unlike standard ZK, distributional[1] promise ZK can be achieved in only three rounds with black-box simulation in the bidirectional message model. This raises two questions – is promise ZK sufficient for our purposes, and what assumptions are required for three round promise ZK?

Promise ZK Under the Hood. Let us start with the first question. An immediate challenge with using promise ZK is that it provides no security in the case where the verifier always aborts. In application to four round MPC, this corresponds to the case where the (rushing) adversary always aborts in the third round. Since the partial transcript at the end of third round (necessarily) contains inputs of honest parties, we still need to argue security in this case. The work of [BGJ+18] addressed this problem by using a "hybrid" ZK protocol that achieves the promise ZK property when the adversary is non-aborting, and the strong witness-indistinguishability (WI) property against aborting adversaries. The idea is that by relying on strong WI property (only in the case where adversary aborts in the third round), we can switch from using real inputs of honest parties to input 0. However, three round strong WI is only known based on specific number-theoretic assumptions [JKKR17].

To minimize our use of assumptions, we do *not* use strong WI, and instead "mimic" its effect by using promise ZK *under the hood*. Specifically, since we use the third round prover message of promise ZK as a witness for conditional decryption, it is not given in the clear, but is instead "encrypted" inside the OT receiver messages in the third round. This has the positive effect of "shielding" promise ZK from the case where the adversary always aborts in the third round. In particular, we can use the following strategy for arguing security against aborting adversaries: we

[1]That is, where the instances are sampled from a public distribution.

first switch from using promise ZK third round prover message to simply using 0's as the OT receiver's inputs. Now, we can replace the honest parties' inputs with 0 inputs by relying on the security of the sub-protocols used within the first three rounds. Next, we can switch back to using honestly computed promise ZK third round prover message as the OT receiver's inputs.

Let us now consider the second question, namely, the assumptions required for three round promise ZK. The work of [BGJ+18] used specific number-theoretic assumptions to construct three round (distributional) promise ZK. However, we only wish to rely on the use of four round OT. Towards this, we note that the only ingredient in the construction of promise ZK by [BGJ+18] that relies on the use of specific number-theoretic assumptions is a three round *rewind-secure* WI proof system. Roughly, this is a proof system where the WI property holds even against verifiers who can rewind the prover an a priori bounded number of times. A very recent work of [GR19] provides a construction of such a rewind-secure WI only based on non-interactive commitments. By using their result, we can obtain three round promise ZK based on non-interactive commitments, which in turn can be obtained from four round OT using the recent observation of Lombardi and Schaeffer [LS19]. For completeness, we describe the construction of [GR19] in Appendix A.

Implementing the Strategy. While the above ideas form the basis of our approach, we run into several challenges during implementation. In order to explain these challenges and our solution ideas, we first describe the high-level template of our four round MPC protocol based on the ideas discussed so far. To narrow the focus of the discussion on the challenges unique to the present work, we ignore several important details for now and discuss them later.

We devise a compiler from four round delayed semi-malicious [BL18, ACJ17] MPC protocols of a special form to a four round malicious-secure MPC protocol. Roughly speaking, a k-round MPC protocol is delayed semi-malicious if in the second to last round, a corrupted party is required to output (on a special tape) a witness (namely, its input and randomness) that establishes its honest behavior in all the rounds so far. We use the four round delayed semi-malicious protocol obtained by plugging in a four-round malicious-secure (which implies delayed semi-malicious security) OT in the k-round semi-malicious MPC protocol of [GS18, BL18] based on k-round semi-malicious OT. An important property of this protocol that we rely upon is that it consists only of OT messages in the first $k-2$ rounds. Further, we also rely upon the random self-reducibility of OT, which implies that the first two rounds do not depend on the OT receiver's input, and the first three rounds do not depend on the sender's input.[2] To achieve malicious security, our compiler uses several building blocks, e.g., a three-round extractable commitment scheme that is executed in *parallel* with the first three rounds of the delayed semi-malicious MPC. The extractable commitment scheme is used by the parties to commit to their inputs and randomness. This allows the simulator for our protocol to extract the adversary's inputs (and randomness) by *rewinding* the second and third rounds, and then use it to simulate the delayed semi-malicious MPC.

A pictorial depiction of our overall protocol as well as a high-level explanation of the various sub-protocols used inside it can be found in Section 4.

Bounded-Rewind-Secure OT. The above template poses an immediate challenge in proving security of the protocol. Since the simulator rewinds the second and third rounds in order to extract the adversary's inputs, this means that the second and third round messages of the delayed semi-malicious MPC also get rewound. For this reason, we cannot simply rely upon delayed semi-malicious security of the MPC. Instead, we need the MPC protocol to remain security *even when it is being rewound*. More specifically, since we are using an MPC protocol where the first two rounds only consist of OT messages, we need a *four round rewind-secure OT protocol*. Since the third

[2] We note that this property was also used by [BL18] in their construction of k-round malicious-secure MPC.

round of a four round OT only contains a message from the OT receiver, we need the following form of rewind security property: an adversarial sender cannot determine the input bit used by the receiver even if it can rewind the receiver during the second and third round.

Clearly, an OT protocol with black-box simulation cannot be secure against an arbitrary number of rewinds. In particular, the best we can hope for is security against an a priori *bounded* number of rewinds. Following observations from [BGJ+18], we note that bounded-rewind security of OT is, in fact, *sufficient* for our purposes. Roughly, the main idea is that the rewind-security of OT is invoked to argue indistinguishability of two consecutive hybrids inside our security proof. In order to establish indistinguishability by contradiction, it suffices to build an adversary that breaks OT security with some non-negligible probability (as opposed to overwhelming probability). This, in turn means that the reduction only needs to extract the adversary's input required for generating its view with non-negligible probability. By using a specific extractable commitment scheme, we can ensure that the number of rewinds necessary for this task are a priori bounded.

Standard OT protocols, however, do not guarantee any form of bounded-rewind security. Towards this, we provide a generic construction of a four round bounded-rewind secure OT starting from any four round OT, which may be of independent interest. Our transformation is in fact more general and works for any $k \geq 4$ round OT, when rewinding is restricted to rounds $k-2$ and $k-1$. For simplicity, we describe our ideas for the case where we need security against *one* rewind; our transformation easily extends to handle more rewinds.

A natural idea to achieve one-rewind security for receivers, previously considered in [BL18], is the following: run two copies of an OT protocol in parallel for the first $k-2$ rounds. In round $k-1$, the receiver randomly chooses one of the two copies and only continues that OT execution, while the sender continues both the OT executions. In the last round, the parties only complete the OT execution that was selected by the receiver in round $k-1$. Now, suppose that an adversarial sender rewinds the receiver in rounds $k-2$ and $k-1$. Then, if the receiver selects different OT copies on the "main" execution thread and the "rewound" execution thread, we can easily reduce one-rewind security of this protocol to stand-alone security of the underlying OT.

The above idea suffers from a subtle issue. Note that the above strategy for dealing with rewinds is inherently *biased*, namely, the choice made by the receiver on the rewound thread is *not* random, and is instead correlated with its choice on the main thread. If we use this protocol in the design of our MPC protocol, it leads to the following issue during simulation: consider an adversary who chooses a random z and then always aborts if the receiver selects the z-th OT copy. Clearly, this adversary only aborts with probability $1/2$ in an honest execution. Now, consider the high-level simulation strategy for our MPC protocol discussed earlier, where the simulator rewinds the second and third rounds to extract the adversary's inputs. In order to ensure rewind security of the OT, this simulator, with overall probability $1/2$, will select the z-th OT copy on *all* the rewound execution threads. However, in this case, the simulator will always *fail* in extracting the adversary's inputs no matter how many times it rewinds.

We address the above problem via a secret-sharing approach to eliminate the bias. Instead of simply running two copies of OT, we run $\ell \cdot n$ copies in parallel during the first $k-2$ rounds. These $\ell \cdot n$ copies can be divided into n tuples, each consisting of ℓ copies. In round $k-1$, the receiver selects a single copy from each of the n tuples at random. It then uses n-out-of-n secret sharing to divide its input bit b into n shares $b_1, \ldots, b_n$, and then uses share b_i in the OT copy selected from the i-th tuple. In the last round, sender now additionally sends a garbled circuit (GC) that contains its input (x_0, x_1) hardwired. The GC takes as input all the bits $b_1, \ldots, b_n$, reconstructs b and then outputs x_b. The sender uses the labels of the GC as its inputs in the OT executions. Intuitively, by setting ℓ appropriately, we can ensure that for at least one tuple i, the OT copies randomly selected by the receiver on the main thread and the rewound threads are different, which ensures that b_i (and thereby, b) remains hidden. We refer the reader to the technical section for more details.

Proofs Of Proofs. We now describe another challenge in implementing our template of four round MPC. As discussed earlier, we use a three round extractable commitment scheme to enable extraction of the adversary's inputs and randomness. For technical reasons, we use an extractable commitment scheme where the third round message of the committer is not "verifiable", namely, the committer may be able to send a malformed message without being detected by the receiver.[3] Further, we require each party to prove the "well-formedness" of its commitment via promise ZK. This, however, poses the following challenge during simulation: since the third round prover message of promise ZK is encrypted inside OT receiver message, the simulator doesn't know whether the adversary's commitment is well-formed or not. In particular, if the adversary's commitment is not well-formed, the simulator may end up running forever, in its attempt to extract the adversary's input via rewinding.

One natural idea to deal with this issue is to first extract adversary's promise ZK message from the OT executions via rewinding, and then decide whether or not to attempt extracting the adversary's input. However, since we are using an *arbitrary* (malicious-secure) OT, we do *not* know in advance the number of rewinds required for extracting the receiver's input. This in turn means that we cannot correctly set the rewind security of the sub-protocols used in our final MPC protocol appropriately in advance.

We address this issue via the following strategy. We use another three round (delayed-input) extractable commitment scheme [PRS02] as well as another copy of promise ZK. This copy of promise ZK proves honest behavior in the first three rounds, and its third message is committed inside the extractable commitment. Further, the third round message of the extractable commitment is such that it allows for polynomial-time extraction (with the possibility of "over-extraction"). This, however, comes at the cost that this extractable commitment does not achieve any rewind security. Interestingly, stand-alone security of this scheme suffices for our purposes since we only use it in the case where the adversary always aborts in the third round (and therefore, no rewinds are performed).

The main idea is that by using such a special-purpose extractable commitment scheme, we can ensure that an a priori fixed constant number of rewinds are sufficient for extracting the committed value, namely, the promise ZK third round prover message, with noticeable probability. This, in turn, allows us to set the rewind security of other sub-protocols used in our MPC protocol in advance to specific constants.

Of course, the adversary may always choose to commit to malformed promise ZK messages within the extractable commitment scheme. In this case, our simulator may always decide not to extract adversary's input, *even if the adversary was behaving honestly otherwise*. This obviously would lead to a view that is distinguishable from the real world. To address this issue, we use a *proofs of proofs* strategy. Namely, we require the first copy of promise ZK, which is encrypted inside OT, to prove that the second copy of promise ZK is "accepting". In this case, if the adversary commits malformed promise ZK messages within the extractable commitment, the promise ZK message inside OT will not be accepting. This, in turn, means that due to the security of garbled circuits, the fourth round messages of the parties will become "opaque".

Finally, we remark that for technical reasons, we do extract the promise ZK encrypted inside the OT receiver message in our *final* hybrid. However, in this particular hybrid, the number of rewinds required for extraction do not matter since we only make change inside a *non-interactive*

[3] This property is crucially used to achieve rewind-security, which in turn is required in the security proof of our MPC protocol for similar reasons as discussed for the case of OT.

primitive (specifically, garbled circuit) that is trivially secure against an unbounded polynomial number of rewinds.

Establishing Non-Malleability. So far, we have largely ignored malleability related issues in our discussion. As noted in many prior works, standard soundness guarantees of ZK proofs do not suffice in the design of constant-round MPC protocols. In particular, since the proofs given by various parties are executed in parallel, we need to ensure that the proofs given by adversarial parties remain sound even when the honest party proofs are simulated [Sah99].

Many prior works use the following template to ensure the above property: the parties are required to send a non-malleable commitment (NMCOM) to 0, and then prove that either they are behaving "honestly" or the NMCOM commits to a "trapdoor" string, which is determined via a separate "trapdoor generation" sub-protocol. The main idea is that now, in order to ensure "simulation soundness" across the hybrids, it suffices to prove an *invariant* that the adversary never commits to the trapdoor in its NMCOM. If the NMCOM scheme supports extraction, then it is indeed possible to prove that the invariant holds: first, the invariant is established in the real world, i.e., the first hybrid, by simply extracting the value inside adversary's NMCOM and invoking the security of the trapdoor generation protocol. In subsequent hybrids, we either continue using the extractor to argue that the value inside adversary's NMCOM does not change, or simply rely on the non-malleability property of NMCOM to argue that the value committed by the adversary did not change. Note that the latter property is only used in one hybrid where we switch the value inside the honest party NMCOM. In the sequel, we refer to this hybrid as the "NMCOM-hybrid".

In the design of four-round MPC, due to aborting adversaries, it is imperative to use a *three* round NMCOM to implement the above strategy. Towards this end, we rely upon the three round NMCOM scheme of Goyal et al. [GPR16] in order to minimize the use of assumptions in our protocol. An important weakeness, however, of their NMCOM scheme is that it suffers from "over-extraction", namely, the extractor can output a valid (non-$\perp$) value even if the adversary's committed to $\perp$ (i.e., its commitment was not valid). This, unfortunately, leads to a failure in the implementation of the above strategy. Roughly, the main issue arises due to adversaries who simply commit to $\perp$ in the NMCOM. In this case, regular non-malleability of NMCOM does not suffice since while it ensures that the committed value does not change, it does *not* ensure that the value output by the extractor does not change. (Indeed, this is true for the extractor of Goyal et al's NMCOM scheme.) Note that in this case, it is possible that the extractor when applied on the adversary's NMCOM in the "NMCOM-hybrid", in fact, outputs the trapdoor even when the adversary was committing to $\perp$. This means that while arguing indistinguishability of subsequent hybrids, we can no longer derive a contradiction by inspecting the value extracted from NMCOM.

We circumvent this problem in the following manner. We first strengthen the above invariant to claim that a particular extractor, when applied on the adversary's NMCOM, does not output the trapdoor. Next, we crucially observe that a weak "split-state" extractor used inside the security proof of Goyal et al's NMCOM scheme *does*, in fact, satisfy the stronger property we need. Specifically, it guarantees the following two properties: (1) If we switch the honest party commitment from m_0 to m_1, the value extracted from adversary's NMCOM does not change, (2) If the adversary sends a well-formed commitment to some value m, then with noticeable probability, the output of the extractor is m. Using these properties, we can establish simulation-soundness as follows. Throughout the hybrids, we first use the above extractor to argue that the value extracted from adversary's NMCOM is not the trapdoor. Then, using the second property, we can argue that the adversary must not be committing to the trapdoor.

For issues related to rewindings, we also rely upon some additional properties of Goyal et al's NMCOM scheme. We describe all of the required properties as well as the specific extractor algorithm we use in Section 2.7.

Other Challenges. The above discussion ignores several additional challenges that arise in fully implementing our template for four round MPC. This includes malleability issues *across* different sub-protocols used inside our protocol. We handle some of these issues by adapting ideas from [BGJ+18]. For example, similar to [BGJ+18], we use different "levels" of bounded-rewinding security for our sub-protocols in order to achieve non-malleability across different sub-protocols. We also carefully use the analysis of [GK96a] to ensure that our simulator runs in expected polynomial time. Finally, we note that we use promise ZK in a non-black-box manner. That is, we directly use all of its building blocks inside our MPC protocol, and rely on their security properties separately.

1.3 Related Work

The round complexity of MPC has been extensively studied over the years in a variety of models. Here, we provide a short survey of malicious-secure MPC protocols in the plain model. We refer the reader to [BGJ+18] for a more comprehensive survey.

Beaver et al. [BMR90] initiated the study of constant round MPC in the honest majority setting. Several follow-up works subsequently constructed constant round MPC against dishonest majority (which is the focus of the present work) [KOS03, Pas04, PW10, Wee10, Goy11]. Garg et al. [GMPP16a] established a lower bound of four rounds for MPC. They constructed five and six round MPC protocols using indistinguishability obfuscation and LWE, respectively, together with three-round robust non-malleable commitments.

The first four round MPC protocols were constructed independently by Ananth et al. [ACJ17] and Brakerski et al. [BHP17] based on different sub-exponential-time hardness assumptions. [ACJ17] also constructed a five round MPC protocol based on polynomial-time hardness assumptions. Ciampi et al. constructed four-round protocols for multiparty coin-tossing [COSV17b] and two-party computation [COSV17a] from polynomial-time assumptions. Benhamouda and Lin [BL18] gave a general transformation from any k-round OT with alternating messages to k-round MPC, for $k > 5$. More recently, independent works of Badrinarayanan et al. [BGJ+18] and Halevi et al. [HHPV18] constructed four round MPC protcols for general functionalities based on different polynomial-time assumptions. Specifically, [BGJ+18] rely on DDH (or QR or N-th Residuosity), and [HHPV18] rely on Zaps, affine-homomorphic encryption schemes and injective one-way functions (which can all be instantiated from QR).

This work is the result of a merge of the works [CO19] and [CGJ19], and subsumes both these works.

2 Preliminaries

2.1 Secure Multiparty Computation

We provide the definition of MPC against malicious adversaries as well as (delayed) semi-malicious adversaries. Parts of this section have been taken verbatim from [Gol04].

A multi-party protocol is cast by specifying a random process that maps pairs of inputs to pairs of outputs (one for each party). We refer to such a process as a functionality. The security of a protocol is defined with respect to a functionality f. In particular, let n denote the number of parties. A non-reactive n-party functionality f is a (possibly randomized) mapping of n inputs to n outputs. A multiparty protocol with security parameter λ for computing a non-reactive functionality f is a protocol running in time $\mathsf{poly}(\lambda)(\lambda)$ and satisfying the following correctness requirement: if parties $P_1, \ldots, P_n$ with inputs $(x_1, \ldots, x_n)$ respectively, all run an honest execution of the protocol, then the joint distribution of the outputs $y_1, \ldots, y_n$ of the parties is statistically close to $f(x_1, \ldots, x_n)$.

A reactive functionality f is a sequence of non-reactive functionalities $f = (f_1, \ldots, f_\ell)$ computed in a stateful fashion in a series of phases. Let x_i^j denote the input of P_i in phase j, and let s^j denote the state of the computation after phase j. Computation of f proceeds by setting s^0 equal to the empty string and then computing $(y_1^j, \ldots, y_n^j, s^j) \leftarrow f_j(s^{j-1}, x_1^j, \ldots, x_n^j)$ for $j \in [\ell]$, where y_i^j denotes the output of P_i at the end of phase j. A multi-party protocol computing f also runs in ℓ phases, at the beginning of which each party holds an input and at the end of which each party obtains an output. (Note that parties may wait to decide on their phase-j input until the beginning of that phase.) Parties maintain state throughout the entire execution. The correctness requirement is that, in an honest execution of the protocol, the joint distribution of all the outputs $\{y_1^j, \ldots, y_n^j\}_{j=1}^{\ell}$ of all the phases is statistically close to the joint distribution of all the outputs of all the phases in a computation of f on the same inputs used by the parties.

Defining Security. We assume that readers are familiar with standard simulation-based definitions of secure multi-party computation in the standalone setting. We provide a self-contained definition for completeness and refer to [Gol04] for a more complete description. The security of a protocol (with respect to a functionality f) is defined by comparing the real-world execution of the protocol with an ideal-world evaluation of f by a trusted party. More concretely, it is required that for every adversary $\mathcal{A}$, which attacks the real execution of the protocol, there exist an adversary Sim, also referred to as a simulator, which can *achieve the same effect* in the ideal-world. Let's denote $\overrightarrow{x} = (x_1, \ldots, x_n)$.

The real execution In the real execution of the n-party protocol π for computing f is executed in the presence of an adversary $\mathcal{A}$. The honest parties follow the instructions of π. The adversary $\mathcal{A}$ takes as input the security parameter k, the set $I \subset [n]$ of corrupted parties, the inputs of the corrupted parties, and an auxiliary input z. $\mathcal{A}$ sends all messages in place of corrupted parties and may follow an arbitrary polynomial-time strategy.

The interaction of $\mathcal{A}$ with a protocol π defines a random variable $\mathsf{REAL}_{\pi,\mathcal{A}(z),I}(k, \overrightarrow{x})$ whose value is determined by the coin tosses of the adversary and the honest players. This random variable contains the output of the adversary (which may be an arbitrary function of its view) as well as the outputs of the uncorrupted parties. We let $\mathsf{REAL}_{\pi,\mathcal{A}(z),I}$ denote the distribution ensemble $\{\mathsf{REAL}_{\pi,\mathcal{A}(z),I}(k, \overrightarrow{x})\}_{k \in \mathsf{N}, \langle \overrightarrow{x}, z \rangle \in \{0,1\}^*}$.

The ideal execution – security with abort . In this second variant of the ideal model, fairness and output delivery are no longer guaranteed. This is the standard relaxation used when a strict majority of honest parties is not assumed. In this case, an ideal execution for a function f proceeds as follows:

- **Send inputs to the trusted party:** As before, the parties send their inputs to the trusted party, and we let x_i' denote the value sent by P_i. Once again, for a semi-honest adversary we require $x_i' = x_i$ for all $i \in I$.

- **Trusted party sends output to the adversary:** The trusted party computes $f(x_1', \ldots, x_n') = (y_1, \ldots, y_n)$ and sends $\{y_i\}_{i \in I}$ to the adversary.

- **Adversary instructs trust party to abort or continue:** This is formalized by having the adversary send either a continue or abort message to the trusted party. (A semi-honest adversary never aborts.) In the latter case, the trusted party sends to each uncorrupted party P_i its output value y_i. In the former case, the trusted party sends the special symbol $\perp$ to each uncorrupted party.

– **Outputs:** Sim outputs an arbitrary function of its view, and the honest parties output the values obtained from the trusted party.

The interaction of Sim with the trusted party defines a random variable $\mathsf{IDEAL}_{f_\perp,\mathcal{A}(z)}(k, \overrightarrow{x})$ as above,and we let $\{\mathsf{IDEAL}_{f_\perp,\mathcal{A}(z),I}(k, \overrightarrow{x})\}_{k\in\mathsf{N},\langle\overrightarrow{x},z\rangle\in\{0,1\}^*}$ where the subscript "$\perp$" indicates that the adversary can abort computation of f.

Having defined the real and the ideal worlds, we now proceed to define our notion of security.

Definition 1. *Let k be the security parameter. Let f be an n-party randomized functionality, and π be an n-party protocol for $n \in \mathsf{N}$.*

1. *We say that π t-securely computes f in the presence of malicious (resp., semi-honest) adversaries if for every PPT adversary (resp., semi-honest adversary) $\mathcal{A}$ there exists a PPT adversary (resp., semi-honest adversary)* Sim *such that for any $I \subset [n]$ with $|I| \leq t$ the following quantity is negligible:*

$$|Pr[\mathsf{REAL}_{\pi,\mathcal{A}(z),I}(k, \overrightarrow{x}) = 1] - Pr[\mathsf{IDEAL}_{f,\mathcal{A}(z),I}(k, \overrightarrow{x}) = 1]|$$

where $\overrightarrow{x} = \{x_i\}_{i\in[n]} \in \{0,1\}^*$ and $z \in \{0,1\}^*$.

2. *Similarly, π t-securely computes f* with abort *in the presence of malicious adversaries if for every PPT adversary $\mathcal{A}$ there exists a polynomial time adversary* Sim *such that for any $I \subset [n]$ with $|I| \leq t$ the following quantity is negligible:*

$$|Pr[\mathsf{REAL}_{\pi,\mathcal{A}(z),I}(k, \overrightarrow{x}) = 1] - Pr[\mathsf{IDEAL}_{f_\perp,\mathcal{A}(z),I}(k, \overrightarrow{x}) = 1]|.$$

Security Against (Delayed) Semi-Malicious Adversaries We also define security against semi-malicious adversaries that are stronger than semi-honest adversaries. A semi-malicious adversary is modeled as an interactive Turing machine (ITM) which, in addition to the standard tapes, has a special witness tape. In each round of the protocol, whenever the adversary produces a new protocol message msg on behalf of some party $\mathbb{P}_k$, it must also write to its special witness tape some pair (x, r) of input x and randomness r that explains its behavior. More specifically, all of the protocol messages sent by the adversary on behalf of $\mathbb{P}_k$ up to that point, including the new message m, must exactly match the honest protocol specification for $\mathbb{P}_k$ when executed with input x and randomness r. Note that the witnesses given in different rounds need not be consistent. Also, we assume that the attacker is rushing and hence may choose the message m and the witness (x, r) in each round adaptively, after seeing the protocol messages of the honest parties in that round (and all prior rounds). Lastly, the adversary may also choose to abort the execution on behalf of $\mathbb{P}_k$ in any step of the interaction.

A delayed semi-malicious adversary [BL18] is similar to semi-malicious adversary, except that it only needs to output the witness (i.e., a defense of honest behavior) in the second last round of the protocol. We refer the reader to [BL18] for a more detailed discussion.

Definition 2. *We say that a protocol π securely realizes f for (delayed) semi-malicious adversaries if it satisfies Definition 1 when we only quantify over all (delayed) semi-malicious adversaries A.*

2.2 Garbled Circuits

Definition 3 (Garbling Scheme)**.** *A garbling scheme for circuits is a tuple of PPT algorithms* $\mathsf{GC} := (\mathsf{Gen}, \mathsf{Garble}, \mathsf{Eval})$ *such that"*

– $(\{\mathsf{lab}^{w,b}\}_{w\in\mathsf{inp},b\in\{0,1\}}) \leftarrow \mathsf{Gen}(1^\lambda,\mathsf{inp})$: Garble *takes the security parameter* 1^λ *and length of input for the circuit as input and outputs a set of input labels* $\{\mathsf{lab}^{w,b}\}_{w\in\mathsf{inp},b\in\{0,1\}}$.

– $\widetilde{C} \leftarrow \mathsf{Garble}(C, \{\mathsf{lab}^{w,b}\}_{w\in\mathsf{inp},b\in\{0,1\}})$: Garble *takes as input a circuit* $C : \{0,1\}^{\mathsf{inp}} \to \{0,1\}^{\mathsf{out}}$ *and a set of input labels* $\{\mathsf{lab}^{w,b}\}_{w\in\mathsf{inp},b\in\{0,1\}}$ *and outputs the garbled circuit* $\widetilde{C}$.

– $y \leftarrow \mathsf{Eval}(\widetilde{C}, \mathsf{lab}^x)$: Eval *takes as input the garbled circuit* $\widetilde{C}$, *input labels* lab^x *corresponding to the input* $x \in \{0,1\}^{\mathsf{inp}}$ *and outputs* $y \in \{0,1\}^{\mathsf{out}}$.

This garbling scheme satisfies the following properties:

1. ***Correctness:*** *For any circuit* C *and input* $x \in \{0,1\}^{\mathsf{inp}}$,

$$\Pr[C(x) = \mathsf{Eval}(\widetilde{C}, \mathsf{lab}^x)] = 1$$

where $(\{\mathsf{lab}^{w,b}\}_{w\in\mathsf{inp},b\in\{0,1\}}) \leftarrow \mathsf{Gen}(1^\lambda,\mathsf{inp})$ *and* $\widetilde{C} \leftarrow \mathsf{Garble}(C, \{\mathsf{lab}^{w,b}\}_{w\in\mathsf{inp},b\in\{0,1\}})$.

2. ***Selective Security:*** *There exists a PPT simulator* $\mathsf{Sim}_{\mathsf{GC}}$ *such that, for any PPT adversary* $\mathcal{A}$, *there exists a negligible function* $\mu(.)$ *such that,*

$$|\Pr[\mathsf{Experiment}_{\mathcal{A},\mathsf{Sim}_{\mathsf{GC}}}(1^\lambda, 0) = 1] - \Pr[\mathsf{Experiment}_{\mathcal{A},\mathsf{Sim}_{\mathsf{GC}}}(1^\lambda, 1) = 1]| \leq \mu(1^\lambda)$$

where the experiment $\mathsf{Experiment}_{\mathcal{A},\mathsf{Sim}_{\mathsf{GC}}}(1^{1^\lambda}, b)$ *is defined as follows:*

(a) *The adversary* $\mathcal{A}$ *specifies the circuit* C *and an input* $x \in \{0,1\}^{\mathsf{inp}}$ *and gets* $\widetilde{C}$ *and* lab^x, *which are computed as follows:*
   - ***If*** $b = 0$:
     - $(\{\mathsf{lab}^{w,b}\}_{w\in\mathsf{inp},b\in\{0,1\}}) \leftarrow \mathsf{Gen}(1^\lambda,\mathsf{inp})$
     - $\widetilde{C} \leftarrow \mathsf{Garble}(C, \{\mathsf{lab}^{w,b}\}_{w\in\mathsf{inp},b\in\{0,1\}})$
   - ***If*** $b = 1$:
     - $(\widetilde{C}, \mathsf{lab}^x) \leftarrow \mathsf{Sim}_{\mathsf{GC}}(1^{1^\lambda}, C(x))$

(b) *The adversary outputs a bit* b', *which is the output of the experiment.*

2.3 Extractable Commitment Scheme

We will use a variant of a simple challenge-response based extractable statistically-binding string commitment scheme $\langle C, R\rangle$ that has been used in several prior works, most notably [PRS02, Ros04]. We note that in contrast to [PRS02] where a multi-slot protocol was used, here (similar to [Ros04]), we only need a one-slot protocol.

Protocol $\langle C, R\rangle$. Let $\mathsf{com}(\cdot)$ denote the commitment function of a non-interactive perfectly binding string commitment scheme which requires the assumption of injective one-way functions for its construction. Let n denote the security parameter. The commitment scheme $\langle C, R\rangle$ is described as follows.

COMMIT PHASE:

1. To commit to a string $\mathtt{str}$, C chooses $k = \omega(\log(n))$ independent random pairs $\{\alpha_i^0, \alpha_i^1\}_{i=1}^k$ of strings such that $\forall i \in [k]$, $\alpha_i^0 \oplus \alpha_i^1 = \mathtt{str}$; and commits to all of them to R using com. Let $B \leftarrow \mathsf{com}(\mathtt{str})$, and $A_i^0 \leftarrow \mathsf{com}(\alpha_i^0)$, $A_i^1 \leftarrow \mathsf{com}(\alpha_i^1)$ for every $i \in [k]$.

2. R sends k uniformly random bits $v_1, \dots, v_n$.

3. For every $i \in [k]$, if $v_i = 0$, C opens A_i^0, otherwise it opens A_i^1 to R by sending the appropriate decommitment information.

Open Phase: C opens all the commitments by sending the decommitment information for each one of them.

For our construction, we require a modified extractor for the extractable commitment scheme. The standard extractor returns the value $\mathtt{str}$ that was committed to in the scheme. Instead, we require that the extractor return i, and the openings of A_i^0 and A_i^1. This extractor can be constructed easily, akin to the standard extractor for the extractable commitment scheme.

This completes the description of $\langle C, R \rangle$.

We say that commit phase between C' and R' is *well formed* with respect to a value $\hat{\mathtt{str}}$ if there exist values $\{\hat{\alpha}_i^0, \hat{\alpha}_i^1\}_{i=1}^k$ such that:

1. For all $i \in [k]$, $\hat{\alpha}_i^0 \oplus \hat{\alpha}_i^1 = \hat{\mathtt{str}}$, and

2. Commitments B, $\{A_i^0, A_i^1\}_{i=1}^k$ can be decommitted to $\hat{\mathtt{str}}$, $\{\hat{\alpha}_i^0, \hat{\alpha}_i^1\}_{i=1}^k$ respectively.

3. Let $\bar{\alpha}_1^{v_1}, \dots, \bar{\alpha}_k^{v_k}$ denote the secret shares revealed by C' in the commit phase. Then, for all $i \in [k]$, $\bar{\alpha}_i^{v_i} = \hat{\alpha}_i^{v_i}$.

We now state the following simple lemma about extraction from commitments when they are well formed.

Lemma 1. *There exists a PPT extractor algorithm* Ext *such that, given a set of* 2 *"well-formed" execution transcripts of* com *where each transcript consists of the same first round sender message, the extractor successfully extracts the value committed in each transcript, except with negligible probability.*

It is easy to see that two random challenge strings will differ in at least a single position other than with negligible probability. From the description of the protocol, if the commit phase was well formed, both commitments at a single position suffice to extract the value.

The above notion is referred to as the "2-extractable" property of the extractable commitment scheme.

2.4 Extractable Commitments with Bounded Rewinding Security

In this section, we describe a three round extractable commitment protocol $\mathsf{RECom} = (S, R)$. The construction is adapted from the construction presented in [BGJ+18], and simplified for our setting since we do not require the stronger notion of "reusability", as defined in their work. Much of the text in this section is taken verbatim from their work.

While several constructions of three round extractable commitment schemes are known in the literature (see, e.g., [PRS02, Ros04]), the commitment scheme satisfies a "bounded-rewinding security" property, which roughly means that the value committed by a sender in an execution of the commitment protocol remains hidden even if a malicious receiver can rewind the sender back to the start of the second round of the protocol an a priori bounded B_{recom} number of times. In our application, we set $B_{\mathsf{recom}} = 4$; however, our construction also supports larger values of B_{recom}. For technical reasons, we don't define or prove B_{recom}-rewinding security property and reusability property for our extractable commitment protocol. Instead, this is done inline in our four round MPC protocol.

Construction. Let Com denote a non-interactive perfectly binding commitment scheme based on injective one-way functions. Let N and B_{recom} be positive integers such that $N - B_{\mathsf{recom}} - 1 \geq \frac{N}{2} + 1$. For $B_{\mathsf{recom}} = 4$, it suffices to set $N = 12$.

The three round extractable commitment protocol RECom is described in Figure 1.

Sender S has input x.

Commitment Phase:

1. **Round 1:**
   S does the following:
   - Pick N random degree B_{recom} polynomials $\mathsf{p}_1, \ldots, \mathsf{p}_N$ over $\mathbb{Z}_q$, where q is a prime larger than 2^λ.
   - Compute $\mathsf{recom}_{1,\ell}^{S \to R} \leftarrow \mathsf{Com}(\mathsf{p}_\ell; r_\ell)$ using a random string r_ℓ, for every $\ell \in [N]$.
   - Send $\mathsf{recom}_1^{S \to R} = (\mathsf{recom}_{1,1}^{S \to R}, \ldots, \mathsf{recom}_{1,N}^{S \to R})$ to R.
2. **Round 2:**
   R does the following:
   - Pick random values $\mathsf{z}_\ell \leftarrow_{\$} \mathbb{Z}_q$ for every $\ell \in [N]$.
   - Send $\mathsf{recom}_2^{R \to S} = (\mathsf{z}_1, \ldots, \mathsf{z}_N)$ to S.
3. **Round 3:**
   S does the following:
   - Compute $\mathsf{recom}_{3,\ell}^{S \to R} \leftarrow (x \oplus \mathsf{p}_\ell(0), \mathsf{p}_\ell(\mathsf{z}_\ell))$ for all $\ell \in [N]$.
   - Send $\mathsf{recom}_3^{S \to R} = (\mathsf{recom}_{3,1}^{S \to R}, \ldots, \mathsf{recom}_{3,N}^{S \to R})$ to R.

Decommitment Phase:

1. S outputs $\mathsf{p}_1, \ldots, \mathsf{p}_N$ together with the randomness $r_1, \ldots, r_N$ used in the first round commitments.
2. R first verifies the following:
   - For each $\ell \in [N]$, $\mathsf{recom}_{1,\ell}^{S \to R} = \mathsf{Com}(\mathsf{p}_\ell; r_\ell)$.
   - Parse $\mathsf{recom}_{3,\ell}^{S \to R} = (\alpha_\ell, \beta_\ell)$. Verify that $\beta_\ell = \mathsf{p}_\ell(\mathsf{z}_\ell)$.
   - For each $\ell \in [N]$, compute $x_\ell = \mathsf{p}_\ell(0) \oplus \alpha_\ell$. Verify that all the x_ℓ values are equal.

   If any of the above verifications fail, R outputs $\perp$. Otherwise, R outputs x.

Figure 1: Extractable Commitment Scheme recom.

Well-Formedness of recom Transcripts. We now define a "well-formedness" property of an execution transcript of RECom. Roughly, we say that a transcript $(\mathsf{recom}_1^{S \to R}, \mathsf{recom}_2^{R \to S}, \mathsf{recom}_3^{S \to R})$ is well-formed w.r.t. an input x and randomness r if:

- $N - 1$ out of the N tuples $\mathsf{recom}_{3,\ell}^{S \to R} = (\alpha_\ell, \beta_\ell)$ (where $\ell \in [N]$) are "honestly" computed using randomness $r = \left(\{\mathsf{p}_i\}_{i=1}^N, \{r_i\}_{i=1}^N\right)$ in the sense that: each α_ℓ is a one-time pad of x w.r.t. the key $\mathsf{p}_\ell(0)$ where p_ℓ is a polynomial committed (using randomness r_ℓ) in the first round message $\mathsf{recom}_1^{S \to R}$, and each β_ℓ is a correct evaluation of the polynomial p_ℓ over the "challenge" value z_ℓ contained in $\mathsf{recom}_2^{R \to S}$.

We now proceed to formally define the well-formedness property. For any set T, let $T[i]$ denote the i^{th} element of T.

Definition 4 (Well-Formed Transcripts)**.** *An execution transcript* $(\mathsf{recom}_1^{S\to R}, \mathsf{recom}_2^{R\to S}, \mathsf{recom}_3^{S\to R})$ *of* recom *is said to be* well-formed *with respect to an input x and randomness* $r = \left(\{\mathsf{p}_i\}_{i=1}^N, \{r_i\}_{i=1}^N\right)$ *if there exists an index set $\mathcal{I}$ of size $N-1$ such that the following holds:*

– *For every* $j \in |\mathcal{I}|$, $\mathsf{recom}_{1,\mathcal{I}[j]}^{S\to R} = \mathsf{Com}(\mathsf{p}_{\mathcal{I}[j]}; r_{\mathcal{I}[j]})$ *(AND)*

– *For every* $j \in |\mathcal{I}|$, $\mathsf{recom}_{3,\mathcal{I}[j]}^{S\to R} = (x \oplus \mathsf{p}_{\mathcal{I}[j]}(0), \mathsf{p}_{\mathcal{I}[j]}(\mathsf{z}_{\mathcal{I}[j]}))$, *where* $\mathsf{recom}_2^{R\to S} = (\mathsf{z}_1, \ldots, \mathsf{z}_N)$

We remark that the above well-formedness property is "weak" in the sense that we only require $N-1$ out of the N tuples $\mathsf{recom}_{3,\ell}^{S\to R} = (\alpha_\ell, \beta_\ell)$ to be honestly generated (instead of requiring that all N tuples are honestly generated). This relaxation is crucial to establishing the B_{recom}-rewinding-security property for recom.

We now define an "admissibility" property for any input to the extractor.

Definition 5 (Admissible Inputs)**.** *An input set* $(\mathsf{recom}_1, \{\mathsf{recom}_2^i, \mathsf{recom}_3^i\}_{i=1}^{B_{\mathsf{recom}}+1})$ *is said to be* admissible *if for every* $i, j \in [B_{\mathsf{recom}}+1]$ *s.t.* $i \neq j$ *and every* $\ell \in [N]$*, we have that* $\mathsf{z}_\ell^i \neq \mathsf{z}_\ell^j$*, where* $\mathsf{recom}_2^t = (\mathsf{z}_1^t, \ldots, \mathsf{z}_N^t)$*.*

Extractor $\mathsf{Ext}_{\mathsf{recom}}$. The extractor algorithm $\mathsf{Ext}_{\mathsf{recom}}$ is described in Figure 2.[4]

Lemma 2. *There exists a PPT extractor algorithm* $\mathsf{Ext}_{\mathsf{recom}}$ *such that, given a set of* $(B_{\mathsf{recom}}+1)$ *"well-formed" and "admissible" execution transcripts of* RECom *where each transcript consists of the same first round sender message, the extractor successfully extracts the value committed in each transcript, except with negligible probability.*

Proof. We now analyze the extraction algorithm. Recall that for every $i \in [B_{\mathsf{recom}}+1]$, the transcript $(\mathsf{recom}_1, \mathsf{recom}_2^i, \mathsf{recom}_3^i)$ is well-formed w.r.t. some value x_i. By the definition of well-formedness, we have that for every i, there exists at most one $j \in [N]$ such that $\mathsf{recom}_{3,j}^i$ was not computed correctly and consistently with the other $\mathsf{recom}_{3,j'}^i$. This means that overall, across all $i \in [B_{\mathsf{recom}}+1]$ execution transcripts, there exists at most $(B_{\mathsf{recom}}+1)$ values of $\mathsf{recom}_{3,j}^i$ that were not computed correctly. This implies that for at least $(N - B_{\mathsf{recom}} - 1)$ values of j, the values $\mathsf{recom}_{3,j}^i$ were computed correctly in all $B_{\mathsf{recom}}+1$ transcripts. This means that for every $i \in [B_{\mathsf{recom}}+1]$, $(N - B_{\mathsf{recom}} - 1)$ out of N values $\{k_1^i, \ldots, k_N^i\}$ computed by the extractor are the same. Then, since $N - B_{\mathsf{recom}} - 1 \geq \frac{N}{2} + 1$, we have that the extractor computes the correct values k^i and x_i for every $i \in [B_{\mathsf{recom}}]$. $\square$

2.5 Trapdoor Generation Protocol with Bounded Rewind Security

This section, taken verbatim from [BGJ+18], discusses and constructs a Trapdoor Generation Protocol. In such a protocol, a sender S (a.k.a. trapdoor generator) communicates with a receiver R. The protocol satisfies two properties: (i) Sender security, i.e., no cheating PPT receiver can learn a valid trapdoor, and (ii) Extraction, i.e., there exists an expected PPT algorithm (a.k.a. extractor) that can extract a trapdoor from an adversarial sender via rewinding.

[4] An admissible input set consisting of $(B_{\mathsf{recom}}+1)$ "well-formed" execution transcripts of recom that share the same first round sender message can be obtained from a malicious sender via an expected PPT rewinding procedure. The expected PPT simulator in our application performs the necessary rewindings to obtain such transcripts and then feeds them to the extractor $\mathsf{Ext}_{\mathsf{recom}}$.

Input: An admissible set $(\mathsf{recom}_1, \{\mathsf{recom}_2^i, \mathsf{recom}_3^i\}_{i=1}^{B_{\mathsf{recom}}+1})$ where $\forall i$, $(\mathsf{recom}_1, \mathsf{recom}_2^i, \mathsf{recom}_3^i)$ is well-formed w.r.t. some value x_i.

1. For every $i \in [B_{\mathsf{recom}} + 1]$, parse $\mathsf{recom}_2^i = (\mathsf{z}_1^i, \ldots, \mathsf{z}_N^i)$ and $\mathsf{recom}_3^i = (\mathsf{recom}_{3,1}^i, \ldots, \mathsf{recom}_{3,N+2}^i)$.

2. For each $\ell \in [N]$:

   – Parse $\mathsf{recom}_{3,\ell}^i = (\alpha_\ell^i, \beta_\ell^i)$.
   – Using polynomial interpolation, compute a degree B_{recom} polynomial p_ℓ over $\mathbb{Z}_q$ such that on point z_ℓ^i, $\mathsf{p}_\ell(\mathsf{z}_\ell^i) = \beta_\ell^i$.
   – Compute $x_\ell^i = (\alpha_\ell^i \oplus \mathsf{p}_\ell(0))$.

3. For every $i \in [B_{\mathsf{recom}}]$, let x^i be the value that equals a majority of the values in the set $\{x_1^i, \ldots, x_N^i\}$. If no such x^i value exists, set $x_i = \perp$.

4. Output $(x_1, \ldots, x_{B_{\mathsf{recom}}})$.

Figure 2: Strategy of algorithm $\mathsf{Ext}_{\mathsf{recom}}$.

We construct a three-round trapdoor generation protocol where the first message sent by the sender determines the set of valid trapdoors, and in the next two rounds the sender proves that indeed it knows a valid trapdoor. Such schemes are known in the literature based on various assumptions [PRS02, Ros04, COSV17b]. Here, we consider trapdoor generation protocols with a stronger sender security requirement that we refer to as *1-rewinding security*. Below, we formally define this notion and then proceed to give a three-round construction based on one-way functions. Our construction is a minor variant of the trapdoor generation protocol from [COSV17b].

Syntax. A trapdoor generation protocol

$$\mathsf{TDGen} = (\mathsf{TDGen}_1, \mathsf{TDGen}_2, \mathsf{TDGen}_3, \mathsf{TDOut}, \mathsf{TDValid}, \mathsf{TDExt})$$

is a three round protocol between two parties - a sender (trapdoor generator) S and receiver R that proceeds as below.

1. **Round 1 -** $\mathsf{TDGen}_1(\cdot)$**:**
   S computes and sends $\mathsf{td}_1^{S\to R} \leftarrow \mathsf{TDGen}_1(\mathsf{r}_S)$ using a random string r_S.

2. **Round 2 -** $\mathsf{TDGen}_2(\cdot)$**:**
   R computes and sends $\mathsf{td}_2^{R\to S} \leftarrow \mathsf{TDGen}_2(\mathsf{td}_1^{S\to R}; \mathsf{r}_R)$ using randomness r_R.

3. **Round 3 -** $\mathsf{TDGen}_3(\cdot)$**:**
   S computes and sends $\mathsf{td}_3^{S\to R} \leftarrow \mathsf{TDGen}_3(\mathsf{td}_2^{R\to S}; \mathsf{r}_S)$

4. **Output -** $\mathsf{TDOut}(\cdot)$
   The receiver R outputs $\mathsf{TDOut}(\mathsf{td}_1^{S\to R}, \mathsf{td}_2^{R\to S}, \mathsf{td}_3^{S\to R})$.

5. **Trapdoor Validation Algorithm -** $\mathsf{TDValid}(\cdot)$**:**
   Given input $(\mathsf{t}, \mathsf{td}_1^{S\to R})$, output a single bit 0 or 1 that determines whether the value t is a valid trapdoor corresponding to the message td_1 sent in the first round of the trapdoor generation protocol.

In what follows, for brevity, we set td_1 to be $\mathsf{td}_1^{S\to R}$. Similarly we use td_2 and td_3 instead of $\mathsf{td}_2^{R\to S}$ and $\mathsf{td}_3^{S\to R}$, respectively. Note that the algorithm $\mathsf{TDValid}$ does not form a part of the interaction between the trapdoor generator and the receiver. It is, in fact, a public algorithm that enables public verification of whether a value t is a valid trapdoor for a first round message td_1.

Extraction. There exists a PPT extractor algorithm TDExt that, given a set of values[5] $(\mathsf{td}_1, \{\mathsf{td}_2^i, \mathsf{td}_3^i\}_{i=1}^3)$ such that $\mathsf{td}_2^1, \mathsf{td}_2^2, \mathsf{td}_2^3$ are distinct and $\mathsf{TDOut}(\mathsf{td}_1, \mathsf{td}_2^i, \mathsf{td}_3^i) = 1$ for all $i \in [3]$, outputs a trapdoor t such that $\mathsf{TDValid}(\mathsf{t}, \mathsf{td}_1) = 1$.

1-Rewinding Security. We define the notion of *1-rewinding security* for a trapdoor generation protocol TDGen. Consider the following experiment between a sender S and any (possibly cheating) receiver R^*.

Experiment E:

- R^* interacts with S and completes one execution of the protocol TDGen. R^* receives values $(\mathsf{td}_1, \mathsf{td}_3)$ in rounds 1 and 3 respectively.
- Then, R^* rewinds S to the beginning of round 2.
- R^* sends S a new second round message td_2^* and receives a message td_3^* in the third round.
- At the end of the experiment, R^* outputs a value t^*.

Definition 6 (1-Rewinding Security)**.** *A trapdoor generation protocol* $\mathsf{TDGen} = (\mathsf{TDGen}_1, \mathsf{TDGen}_2, \mathsf{TDGen}_3, \mathsf{TDOut}, \mathsf{TDValid})$ *achieves 1-rewinding security if, for every non-uniform PPT receiver* R^* *in the above experiment E,*

$$\Pr\left[\mathsf{TDValid}(\mathsf{t}^*, \mathsf{td}_1) = 1\right] = \mathsf{negl}(\lambda)(\lambda),$$

where the probability is over the random coins of S*, and where* t^* *is the output of* R^* *in the experiment E, and* td_1 *is the message from* S *in round 1.*

2.5.1 Construction

We now describe a three round trapdoor generation protocol based on one way functions.

Let S and R denote the sender and the receiver, respectively. Let λ denote the security parameter. Let $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vf})$ be a signature scheme that is existentially unforgeable against chosen-message attacks. Such schemes are known based on one-way functions [GMR88].

Theorem 3. *Assuming the existence of one way functions, the protocol* Π^{TD} *described in Figure 3 is a 1-rewinding secure trapdoor generation protocol.*

We refer the reader to [BGJ+18] for the proof.

Extractor $\mathsf{TDExt}(\cdot)$**.** The extractor works as follows. It receives a verification key $\mathsf{vk} = \mathsf{td}_1$, and a set of values $\{\mathsf{m}_i, \sigma_i\}_{i=1}^3$ such that m_i are all distinct and $\mathsf{Vf}(\mathsf{vk}, \mathsf{m}_i, \sigma_i) = 1$ for every $i \in [3]$. Then, TDExt outputs $\mathsf{t} = \{\mathsf{m}_i, \sigma_i\}_{i=1}^3$ as a valid trapdoor. Correctness of the extraction is easy to see by inspection.

[5]These values can be obtained from the malicious sender via an expected PPT rewinding procedure. The expected PPT simulator in our applications performs the necessary rewindings and then feeds these values to the extractor TDExt.

1. **Round 1 -** $\mathsf{TDGen}_1(\mathsf{r}_S)$**:**
   S does the following:
   – Generate $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{Gen}(\mathsf{r}_S)$.
   – Send $\mathsf{td}_1^{S \to R} = \mathsf{vk}$ to R.
2. **Round 2 -** $\mathsf{TDGen}_2(\mathsf{td}_1^{S \to R})$**:**
   R sends a random string m as the message $\mathsf{td}_2^{R \to S}$ to S.
3. **Round 3 -** $\mathsf{TDGen3}(\mathsf{td}_1^{S \to R}, \mathsf{td}_2^{R \to S}; \mathsf{r}_S)$**:**
   S computes and sends $\mathsf{td}_3^{S \to R} = \mathsf{Sign}(\mathsf{sk}, \mathsf{m}; \mathsf{r}_\mathsf{m})$ where r_m is randomly chosen.
4. **Output: -** $\mathsf{TDOut}(\mathsf{td}_1^{S \to R}, \mathsf{td}_2^{R \to S}, \mathsf{td}_3^{S \to R})$
   The receiver R outputs 1 if $\mathsf{Vf}(\mathsf{td}_1^{S \to R}, \mathsf{m}, \mathsf{td}_3^{S \to R}) = 1$.
5. **Trapdoor Validation Algorithm -** $\mathsf{TDValid}(\mathsf{t}, \mathsf{td}_1)$**:**
   Given input $(\mathsf{t}, \mathsf{td}_1)$, the algorithm does the following:
   – Let $\mathsf{t} = \{\mathsf{m}_i, \sigma_i\}_{i=1}^3$.
   – Output 1 if $\mathsf{m}_1, \mathsf{m}_2, \mathsf{m}_3$ are distinct and $\mathsf{Vf}(\mathsf{td}_1, \mathsf{m}_i, \sigma_i) = 1$ for all $i \in [3]$.

Figure 3: Trapdoor Generation Protocol Π^{TD}.

Remark: In the application to our MPC protocol, one party is the sender and sends the first round message td_1. Each of the other $(n-1)$ parties send a second round message $\mathsf{td}_{2,i}$ and the sender now sets the concatenation of all of them as the second round message td_2 - that is, $\mathsf{td}_2 = (\mathsf{td}_{2,1}||\dots||\mathsf{td}_{2,n-1})$. The sender then computes td_3 as before.

2.6 Witness Indistinguishable Proofs with Bounded Rewinding Security

We start by defining **Delayed-Input Interactive Arguments**. Without exception, we will require our proof systems to allow for the statement to be chosen after the start of the protocol.

Definition 7 (Delayed-Input Interactive Arguments)**.** *An n-round delayed-input interactive protocol* (P, V) *for deciding a language L is an argument system for L that satisfies the following properties:*

– **Delayed-Input Completeness.** *For every security parameter $\lambda \in \mathbb{N}$, and any $(x, w) \in R_L$ such that $|x| \leq 2^\lambda$,*

$$\Pr[(\mathsf{P}, \mathsf{V})(1^\lambda, x, w) = 1] = 1 - \mathsf{negl}(\lambda)\,.$$

where the probability is over the randomness of P *and* V*. Moreover, the prover's algorithm initially takes as input only 1^λ, and the pair (x, w) is given to* P *only* in the beginning of the n'th round.

– **Delayed-Input Soundness.** *For any PPT cheating prover* P^* *that chooses x^* (adaptively)* after *the first $n-1$ messages, it holds that if $x^* \notin L$ then*

$$\Pr[(\mathsf{P}^*, \mathsf{V})(1^\lambda, x^*) = 1] = \mathsf{negl}(\lambda)\,.$$

where the probability is over the random coins of V.

The primitive we will use extensively in our construction is a witness indistinguishable argument, and its subsequent strengthening.

Definition 8 (Witness Indistinguishability). *A delayed-input interactive argument* (P, V) *for a language* L *is said to be witness indistinguishable if for every PPT algorithm* V^* *and every pair* (w_1, w_2) *such that* $R_L(x, w_1) = 1$ *and* $R_L(x, w_2) = 1$*, the following are computationally indistinguishable.*

$$\mathsf{view}_{\mathsf{V}^*}(\mathsf{P}(w_1), \mathsf{V}^*)(1^\lambda, x) \text{ and } \mathsf{view}_{\mathsf{V}^*}(\mathsf{P}(w_1), \mathsf{V}^*)(1^\lambda, x)$$

where $\mathsf{view}_{\mathsf{V}^*}(\mathsf{P}(w), \mathsf{V}^*)(1^\lambda, x)$ *denotes the view of the verifier during the execution of the protocol.*

Imported Theorem 1 ([LS91]). *Assuming non-interactive commitments there exists 3 round witness indistinguishable proof systems.*

We now strengthen the above definition to define a three round delayed-input witness indistinguishable argument with B_{rwi}-bounded rewinding security, where the same statement is proven across all the rewinds.

Definition 9 (3-Round Delayed-Input WI with Non-Adaptive Fixed Statement Bounded Rewinding Security). *Fix a positive integer* B_{rwi}. *A delayed-input 3-round interactive argument (as defined in Definition 7) for an NP language* L*, with an NP relation* R_L *is said to be WI with Non-Adaptive* Fixed Statement B_{rwi}*-Rewinding Security if for every non-uniform PPT interactive Turing Machine* V^**, it holds that* $\{\mathsf{REAL}_0^{V^*}(1^\lambda)\}_\lambda$ *and* $\{\mathsf{REAL}_1^{V^*}(1^\lambda)\}_\lambda$ *are computationally indistinguishable, where for* $b \in \{0,1\}$ *the random variable* $\mathsf{REAL}_b^{V^*}(1^\lambda)$ *is defined via the following experiment. In what follows we denote by* P_1 *the prover's algorithm in the first round, and similarly we denote by* P_3 *his algorithm in the third round.*

Experiment $\mathsf{REAL}_b^{V^*}(1^\lambda)$***:***

1. *Run* $\mathsf{P}_1(1^\lambda)$ *and denote its output by* (rwi_1, σ)*, where* σ *is its secret state, and* rwi_1 *is the message to be sent to the verifier.*

2. *Run the verifier* $V^*(1^\lambda, \mathsf{rwi}_1)$*, who outputs* (x, w_0, w_1) *and a set of messages* $\{\mathsf{rwi}_2^i\}_{i \in [B_{\mathsf{rwi}}]}$.

3. *For each* $i \in [B_{\mathsf{rwi}} - 1]$*, run* $\mathsf{P}_3(\sigma, \mathsf{rwi}_2^i, x, w_0)$*, where* P_3 *is the (honest) prover's algorithm for generating the third message of the WI protocol, and send its message* P_3 *to* V^**. For* $i = B_{\mathsf{rwi}}$*, run* $\mathsf{P}_3(\sigma, \mathsf{rwi}_2^i, x, w_b)$.

4. *The output of the experiment is the output of* V^*.

The following theorem is proven in [GR19]. For completeness, we provide a full description of their construction in Appendix A.

Theorem 4. *Assuming non-interactive commitments, for every (polynomial) rewinding parameter* B*, there exists a three round delayed-input witness-indistinguishable argument system with* B*-rewinding security.*

2.7 Non-Malleable Commitments

We start with the definition of non-malleable commitments by Pass and Rosen [PR05] and further refined by Lin et al [LPV08] and Goyal [Goy11]. (All of these definitions build upon the original definition of Dwork et al. [DDN91]).

In the real experiment, a man-in-the-middle adversary MIM interacts with a committer C in the left session, and with a receiver R in the right session. Without loss of generality, we assume that each session has identities or tags, and require non-malleability only when the tag for the left session is different from the tag for the right session.

At the start of the experiment, the committer C receives an input val and MIM receives an auxiliary input z, which might contain a priori information about val. Let $\mathsf{MIM}_{\langle C,R\rangle}(\mathsf{val}, z)$ be a random variable that describes the value $\widetilde{\mathsf{val}}$ committed by MIM in the right session, jointly with the view of MIM in the real experiment.

In the ideal experiment, a PPT simulator $\mathcal{S}$ directly interacts with MIM. Let $\mathsf{Sim}_{\langle C,R\rangle}(1^\lambda, z)$ denote the random variable describing the value $\widetilde{\mathsf{val}}$ committed to by $\mathcal{S}$ and the output view of $\mathcal{S}$.

In either of the two experiments, if the tags in the left and right interaction are equal, then the value $\widetilde{\mathsf{val}}$ committed in the right interaction, is defined to be $\perp$.

Definition 10 (Synchronous Non-malleable Commitments)**.** *A 3-round commitment scheme $\langle C, R\rangle$ is said to be synchronous non-malleable if for every* synchronizing[6] *PPT* MIM*, there exists a PPT simulator $\mathcal{S}$ such that the following ensembles are computationally indistinguishable:*

$$\{\mathsf{MIM}_{\langle C,R\rangle}(\mathsf{val}, z)\}_{\lambda\in\mathbb{N},\mathsf{val}\in\{0,1\}^\lambda,z\in\{0,1\}^*} \text{ and } \{\mathsf{Sim}_{\langle C,R\rangle}(1^\lambda, z)\}_{\lambda\in\mathbb{N},\mathsf{val}\in\{0,1\}^\lambda,z\in\{0,1\}^*}$$

Non-malleability with Respect to Extraction. In this section we consider also a different notion of non-malleability that we call *non-malleability with respect to extraction.* Consider the experiment in which the adversary interacts with an honest committer C in the left session, and with an extractor $\mathsf{Ext}_{\mathsf{NMCom}}$ in the right session which guarantees the extraction of the committed value only when the adversary computes a well-formed commitment (if the commitment is ill-formed that it is not guaranteed that the extractor outputs $\perp$). Without loss of generality, we assume that each session has identities or tags, and require our non-malleability property to hold only when the tag for the left session is different from the tag for the right session.

At the start of the experiment, the committer C receives an input m and $\mathcal{A}^{\mathsf{NMCom}}$ receives an auxiliary input z, which might contain a priori information about m. Let $\mathsf{MIM}^{\mathtt{Ext}}_{\langle C,\mathsf{Ext}_{\mathsf{NMCom}}\rangle}(\ m\ , z)$ be a random variable that describes the value $\widetilde{\mathsf{val}}$ output by $\mathsf{Ext}_{\mathsf{NMCom}}$ jointly with the view of $\mathcal{A}^{\mathsf{NMCom}}$ in the real experiment.

In either of the two experiments, if the tags in the left and right interaction are equal, then the value $\widetilde{\mathsf{val}}$ committed in the right interaction, is defined to be $\perp$.

Definition 11. *A 3-round commitment scheme $\langle C, R\rangle$ is said to be* non-malleable with respect to extraction *if for every* synchronizing *PPT $\mathcal{A}^{\mathsf{NMCom}}$ there exists an extractor $\mathsf{Ext}_{\mathsf{NMCom}}$ such that the following ensembles are computationally indistinguishable:*

$$\{\mathsf{MIM}^{\mathtt{Ext}}_{\langle C,\mathsf{Ext}_{\mathsf{NMCom}}\rangle}(\ m_0\ , z)\}_{\lambda\in\mathbb{N},m_0\in\{0,1\}^\lambda,z\in\{0,1\}^*} \text{ and } \{\mathsf{MIM}^{\mathtt{Ext}}_{\langle C,\mathsf{Ext}_{\mathsf{NMCom}}\rangle}(\ m_1\ , z)\}_{\lambda\in\mathbb{N},m_1\in\{0,1\}^\lambda,z\in\{0,1\}^*}$$

In this work we make use of a commitment scheme that is non-malleable, non-malleable with respect to extraction and enjoys some additional properties. We refer to such a commitment scheme as a *special non-malleable commitment scheme.*

Definition 12 (Special Non-malleable Commitments)**.** *A three round commitment scheme $\langle C, R\rangle$ is said to be special non-malleable if:*

[6] A synchronizing adversary is one that sends its message for every round before obtaining the honest party's message for the next round.

– $\langle C, R\rangle$ *is synchronous non-malleable and non-malleable with respect to extraction.*

– $\langle C, R\rangle$ *is delayed-input, that is, correctness holds even when the committer obtains his input only in the last round.*

– $\langle C, R\rangle$ *satisfies* last-message pseudorandomness*, that is, for every non-uniform PPT receiver* R^**, it holds that* $\{\mathsf{REAL}_0^{R^*}(1^\lambda)\}_\lambda$ *and* $\{\mathsf{REAL}_1^{R^*}(1^\lambda)\}_\lambda$ *are computationally indistinguishable, where for* $b \in \{0,1\}$*, the random variable* $\mathsf{REAL}_b^{R^*}(1^\lambda)$ *is defined via the following experiment.*

1. *Run* $C(1^\lambda)$ *and denote its output by* (Com_1, σ)*, where* σ *is its secret state, and* Com_1 *is the message to be sent to the receiver.*
2. *Run the receiver* $R^*(1^\lambda, \mathit{Com}_1)$*, who outputs a message* Com_2*.*
3. *If* $b = 0$*, run* $C(\sigma, \mathit{Com}_2)$ *and send its message* Com_3 *to* R^**. Otherwise, if* $b = 1$*, compute* $\mathit{Com}_3 \leftarrow_{\$} \{0,1\}^m$ *and send it to* R^**. Here* $m = m(\lambda)$ *denotes* $|\mathit{Com}_3|$*.*
4. *The output of the experiment is the output of* R^**.*

Goyal et al. [GPR16] construct three-round special non-malleable commitments satisfying Definition 12 based on non-interactive commitments.

Imported Theorem 2 ([GPR16])**.** *Assuming non-interactive commitments, there exists a three round non-malleable commitment satisfying Definition 12.*

We briefly detail the non-malleable commitment scheme of [GPR16] and show that it is a special non-malleable commitment scheme.

Let $(\mathsf{Com}, \mathrm{Dec})$ be a non-interactive statistically binding commitment scheme, and (E, D) be a split-state non-malleable code that splits the input into two codewords L and R. The scheme $\mathsf{NMCom} = (\mathsf{Sen}, \mathsf{Rec})$ proposed in [GPR16] can be described as follows.

Commitment phase. Let m be the message to be committed.

$\mathsf{Sen} \to \mathsf{Rec}$: Sen chooses $(L, R) \leftarrow \mathsf{Enc}(m)$ where L is viewed as a field element in $\mathbb{Z}_q$; Sen also draws $r \leftarrow \mathbb{Z}_q$ at random, compute $\mathsf{com}, \mathsf{dec} \leftarrow \mathsf{Com}(L||r)$ and sends com to Rec.

$\mathsf{Rec} \to \mathsf{Sen}$: Rec chooses a random $\alpha \leftarrow \mathbb{Z}_q^\star$ and sends it to Sen.

$\mathsf{Sen} \to \mathsf{Rec}$: Sen sends $a = r\alpha + L$ and R to Rec.

Decommitment phase To decommit, Sen sends dec to Rec.

Intuitively, Sen commits to a polynomial-based 2-out-of-2 secret sharing of L in the first round, and in the third round sends R along with one share. We now give an intuition about why this commitment scheme is special non-malleable. We refer the reader to [GPR16] for the formal proof.

Non-malleability against synchronizing PPT adversary. In [GPR16] the authors show how to use an adversary $\mathcal{A}^{\mathsf{NMCom}}$ that breaks the non-malleability of $(\mathsf{Sen}, \mathsf{Rec})$ to construct two tampering functions (f, g) that break the security of the underling split-state non-malleable code. The functions f and g share a partial transcript consisting of the first two messages of an interaction of $(\mathsf{Sen}, \mathsf{Rec})$ with $\mathcal{A}^{\mathsf{NMCom}}$ and the value a. Note that g contains a non-interactive commitment of L and this could be an issue given that the non-malleable code is split-state (and therefore g should not contain information about L). However, this does not represent a problem since L is committed and from the hiding of the non-interactive commitment L can be replaced with another value without the adversary noticing that. The output of g simply consists of the value $\tilde{R}$ that is sent from $\mathcal{A}^{\mathsf{NMCom}}$ to the receiver in the last round (more details on how g works are given later in this section).

The function f does not contain any information about R, but in this case the challenging part is to compute its output since the left part $(\tilde{L})$ of the non-malleable code is committed. However, f can extract $\tilde{L}$ by rewinding $\mathcal{A}^{\mathsf{NMCom}}$. In more details, f on input L chooses a random value $R_\$$ and sends $(a, R_\$)$ to $\mathcal{A}^{\mathsf{NMCom}}$. Upon receiving $(\tilde{a}_\$, \tilde{R}_\$)$ from $\mathcal{A}^{\mathsf{NMCom}}$, f rewinds the adversary and sends a freshly generated second round $\tilde{\beta}$ on the right and upon receiving β on the left f computes and sends (b, R) where $b = (a - L)(\beta/\alpha) + L$. At this point f receives $(\tilde{b}, \cdot)$ on the right from the adversary and computes its output, which consists of the constant term on the line spanned by $\{(\tilde{a}_\$, \tilde{\alpha}), (\tilde{b}, \tilde{\beta})\}$.

We are now ready to complete the description of the function g. This function also shares the random value $R_\$$ and therefore it can compute $\tilde{a}_\$$. At this point $g(R)$ rewinds $\mathcal{A}^{\mathsf{NMCom}}$ and sends (a, R) on the left and receives $(\tilde{a}, \tilde{R})$ on the right. If $\tilde{a} = \tilde{a}_\$$ then $g(R)$ outputs $\tilde{R}$, otherwise it outputs $\perp$.

Note that for (f, g) to succeed in extracting $(\tilde{L}, \tilde{R})$, it must be that the answer $\tilde{a}_\$$ $\mathcal{A}^{\mathsf{NMCom}}$ provides when given the random $R_\$$ is equal to the $\tilde{a}$ he provides given R. This will follow from and additional property that the authors of [GPR16] require on the underling non-malleable code. Given this property the authors show that the chance that $\mathcal{A}^{\mathsf{NMCom}}$ answers correctly (i.e., consistently with the linear map he committed to in the first round) given $R_\$$ is about the same as the chance he answers correctly given R. So either both are incorrect with high probability, in which case $\mathcal{A}^{\mathsf{NMCom}}$ is always committing to $\perp$ and so cannot be mauling; or is it possible to show that f and g extract the correct value.

Delayed-input property. The delayed-input property comes immediately form the fact the non-malleable code (E, D) is such that (L, R) represent a 2-out-of-2 secret sharing of m (where $(L, R) \leftarrow \mathsf{E}(m)$) and the fact that R is sent only in the last round of the protocol. We recall that a delayed-input non-malleable commitment retains its security properties (hiding, binding and non-malleability) against an adversary that adaptively decides what is the message to commit to in the last round (see [COSV16] for more details on delayed-input commitments).

Last-message pseudorandomness. This property comes immediately from the hiding of the non-interactive commitment Com and from the fact that R is the right state of a split-state non-malleable code, which is also a 2-out-of-2 secret sharing (like any split-state non-malleable code).

Non-malleability with respect to extraction. To show that this property holds we first need to construct an extractor $\mathsf{Ext}_{\mathsf{NMCom}}$. $\mathsf{Ext}_{\mathsf{NMCom}}$ interacts with the the adversarial sender using random coins α acting as the honest receiver in the right session. Let $\tau = (\mathsf{com}, \alpha, a, R, \tilde{\mathsf{com}}, \tilde{\alpha}, \tilde{a}, \tilde{R})$ be the transcript of $\mathcal{A}^{\mathsf{NMCom}}$'s view, $\mathsf{Ext}_{\mathsf{NMCom}}$ extracts $\tilde{L}$ and $\tilde{R}$ in two steps.

- To extract $\tilde{L}$ $\mathsf{Ext}_{\mathsf{NMCom}}$ chooses a random value $R_\$$ and sends $(a, R_\$)$ to $\mathcal{A}^{\mathsf{NMCom}}$. Upon receiving $\tilde{a}_\$, \tilde{R}_\$$ from $\mathcal{A}^{\mathsf{NMCom}}$, f rewinds the adversary and sends a freshly generated second round $\tilde{\beta}$ on the right and upon receiving β computes and sends (b, R) where $b = (a-L)(\beta/\alpha)+L$. Upon receiving $(\tilde{b}, \cdot)$ on the right by the adversary, $\mathsf{Ext}_{\mathsf{NMCom}}$ computes $\tilde{L}$, which consists of the constant term on the line spanned by $\{(\tilde{a}_\$, \tilde{\alpha}), (\tilde{b}, \tilde{\beta})\}$.
- To extract $\tilde{R}$ $\mathsf{Ext}_{\mathsf{NMCom}}$ checks if $\tilde{a} = \tilde{a}_\$$. If it is the case then $\mathsf{Ext}_{\mathsf{NMCom}}$ outputs $\mathsf{D}(\tilde{L}, \tilde{R})$, otherwise he outputs $\perp$.

In summary, $\mathsf{Ext}_{\mathsf{NMCom}}$ simply runs the extraction procedures described by the function f and g defined in the non-malleability proof of [GPR16] (that we have also sketched above). We now note that an adversary attacking the property of non-malleability with respect to extraction yields to an adversary for the non-malleable code. The only difference with the non-malleability proof of [GPR16] is that we do not need to check whether the extracted values actually corresponds to the committed value. That is, the adversary could compute an ill-formed commitment that yields to the extraction of a message $m \neq \bot$. We note, however, that if the commitment is well formed then $\mathsf{Ext}_{\mathsf{NMCom}}$ outputs the actual committed value (we refer the reader to [GPR16, Claim 8] for the formal proof).

3 Oblivious Transfer with Bounded Rewind Security

In this section we define, and then construct, a strengthening of regular oblivious transfer. We construct a rewinding secure Oblivious Transfer (OT) assuming the existence of four round OT protocol. For an OT protocol to be rewind secure, we require security against an adversary who is allowed to re-execute the second and third round of the protocol multiple times. But the first and fourth round are executed only once.

3.1 Definition

We start by formalizing the notion of a rewind secure oblivious transfer protocol.

Definition 13. *A four round bounded rewind secure oblivious transfer (OT) is a tuple of polynomial time interactive Turing machines $\mathsf{OT} = (\mathsf{OT}_S, \mathsf{OT}_R)$ where $(t, x) = (\mathsf{OT}_S(s_0, s_1; \rho), \mathsf{OT}_R(b; \rho'))$ is the pair composed of the transcript t and the output of x after the interaction between the sender OT_S with inputs $s_0, s_1 \in \{0,1\}$ and randomness ρ while receiver OT_R has input b and randomness ρ' satisfying the following properties:*

- ***Correctness.*** *For any selection bit b, for any messages $s_0, s_1 \in \{0,1\}$, for any $\rho, \rho' \in \{0,1\}^\tau$ it holds that*

$$\mathsf{Pr}\left[s_b = s : \rho, \rho' \leftarrow_\$ \{0,1\}^\tau; (t, x) = \big(\mathsf{OT}_S(s_0, s_1; \rho), \mathsf{OT}_R(b; \rho')\big)\right] = 1$$

- ***Security against Malicious Sender with B rewinds.*** *Here, we require indistinguishability security against a malicious receiver where the receiver uses input $b[k]$ in the k-th rewound execution of the second and third round. Specifically, consider the experiment described below. $\forall \left\{b^0[k], b^1[k]\right\}_{k \in [B]} \in \{0,1\}$ where **Experiment** E^σ:*

  1. *Run OT_R to obtain ot_1 which is independent of its input. Send to $\mathcal{A}$.*
  2. *$\mathcal{A}$ then returns $\{\mathsf{ot}_2[j]\}_{j \in [B]}$ messages.*
  3. *For each $j \in [B]$, run OT_R on $(\mathsf{ot}_1, \mathsf{ot}_2[j], b^\sigma[j])$ and send the response to $\mathcal{A}$.*
  4. *The output of the experiment is the entire transcript.*

  We say that the scheme is secure against malicious senders with B rewinds if the experiments E^0 and E^1 are indistinguishable.

$\mathsf{C}_{\mathsf{ot}}[s_0, s_1]$
Input: $b_1, \cdots, b_n$ **Output**: s_b where $b \coloneqq \bigoplus_{i=1}^{n} b_i$

Figure 4: Circuit C_{ot}

3.2 Construction

We describe below the protocol Π^{R} which achieves rewind security against malicious senders. The Sender S's input is $s_0, s_1 \in \{0,1\}$ while the receiver R's input is $b \in \{0,1\}$.

Components. We require the following two components:

- $n \cdot \mathsf{B}_{\mathsf{OT}}$ instances of a 4 round OT protocol which achieves indistinguishability security against malicious senders.
- $\mathsf{GC} = (\mathsf{Garble}, \mathsf{Eval})$ is a secure garbling scheme (see Section 2.2).

Protocol. The basic idea is to split the receiver input across multiple different OT executions such that during any rewind, a different set of OTs will be selected to proceed with the execution thereby preserving the security of the receiver's input. The sender constructs a garbled circuit which is used to internally recombine the various inputs shares and only return the appropriate output. The protocol is described below.

Round 1. (Π_1^{R}) : The receiver R computes the first round message of all the OTs. $\forall i \in [n], k \in [\mathsf{B}_{\mathsf{OT}}]$

$$\mathsf{ot}_1^{i,k} \coloneqq \mathsf{OT}_1\left(1^\lambda; \mathsf{r}_R\right)$$

and send $\left\{\mathsf{ot}_1^{i,k}\right\}_{i\in[n],k\in[\mathsf{B}_{\mathsf{OT}}]}$ to S. We refer to index i as the outer index, and k as the inner index.

Round 2. (Π_2^{R}) : The sender S responds to all of the OT messages. $\forall i \in [n], k \in [\mathsf{B}_{\mathsf{OT}}]$, compute

$$\mathsf{ot}_2^{i,k} \coloneqq \mathsf{OT}_2\left(\mathsf{ot}_1^{i,k}; \mathsf{r}_S\right)$$

and sends $\left\{\mathsf{ot}_2^{i,k}\right\}_{i\in[n],k\in[\mathsf{B}_{\mathsf{OT}}]}$ to R.

Round 3. (Π_3^{R}) : The receiver now selects only a single OT to continue for i. It then encodes its input b by computing n additive shares and using each share as an input to a separate OT. Specifically, receiver R does the following:

- Compute n additive shares of b. Specifically, sample the first $n-1$ shares at random $\forall \ell \in [n-1]$

$$b_\ell \leftarrow_{\$} \{0,1\}$$

and set the last share

$$b_n \coloneqq b \bigoplus_{\ell=1}^{n-1} b_j.$$

– Sample within each tuple, the index for which to continue the OT. $\forall i \in [n]$,

$$\sigma_i \leftarrow\!\!\$\; [\mathsf{B_{OT}}] \,.$$

– Use input b_i to compute the receiver message for $\mathsf{ot}_3^{i,\sigma_i}$. The other OTs are discontinued. Specifically, $\forall i \in [n]$, compute

$$\mathsf{ot}_3^{i,\sigma_i} \leftarrow \mathsf{OT}_3\left(b_i, \mathsf{ot}_1^{i,\sigma_i}, \mathsf{ot}_2^{i,\sigma_i};\ \mathsf{r}_R\right)$$

and send $\left\{\mathsf{ot}_3^{i,\sigma_i},\ \sigma_i\right\}_{i\in[n]}$ to S.

Round 4. (Π_4^{R}) : The sender encodes its inputs (s_0, s_1) in a garbled circuit and uses the corresponding labels to complete the OT protocol.

– Compute the garbled circuits containing s_0, s_1. Specifically,

$$\left(\widetilde{\mathsf{C}}_{\mathsf{ot}}, \overline{\mathsf{lab}}\right) := \mathsf{Garble}\left(\mathsf{C}_{\mathsf{ot}}\left[s_0, s_1\right]; \mathsf{r}_{\mathsf{gc},i}\right)$$

where C_{ot} is described in Figure 4.

– For $i \in [n]$, compute

$$\mathsf{ot}_4^{i,\sigma_i} := \mathsf{OT}_4\left(\mathsf{lab}_{i,0}, \mathsf{lab}_{i,1}, \mathsf{ot}_1^{i,\sigma_i}, \mathsf{ot}_2^{i,\sigma_i}, \mathsf{ot}_3^{i,\sigma_i}; \mathsf{r}_S\right)$$

and send $\left\{\mathsf{ot}_4^{i,\sigma_i}\right\}_{i\in[n]}$ to R.

Evaluation. (OTEval') : The receiver R now evaluates the OT protocol to obtain labels needed to evaluate the output of the garbled circuit.

– For $i \in [n]$, compute

$$\widetilde{\mathsf{lab}}_i := \mathsf{OTEval}\left(b_i, \mathsf{ot}_1^{i,\sigma_i}, \mathsf{ot}_2^{i,\sigma_i}, \mathsf{ot}_3^{i,\sigma_i}, \mathsf{ot}_4^{i,\sigma_i};\ \mathsf{r}_R\right)$$

– Output $s' := \mathsf{Eval}\left(\widetilde{\mathsf{C}}_{\mathsf{ot}}, \left\{\widetilde{\mathsf{lab}}_i\right\}_{i\in[n]}\right)$

Security. We prove security of our constructed protocol below.

Lemma 3. *Assuming receiver indistinguishability of* OT *against malicious senders, the receiver input in* Π^{R} *remains indistinguishable under* $\mathsf{B_{OT}}$*-rewinds.*

Proof. Suppose the $\mathsf{B_{OT}}$ inputs used by the receiver are

$$b^0[1], \cdots, b^0[\mathsf{B_{OT}}] \text{ and } b^1[1], \cdots, b^1[\mathsf{B_{OT}}]$$

in experiment 0 and 1 respectively, where $b[j]$ is the receiver input in the j-th rewind. We want to show that an adversarial rewinding sender's view is indistinguishable in both experiments.

We do this by via a sequence of hybrids, where in hybrid ℓ we change the input of the ℓ-th rewind. Consider two adjacent hybrids, $\mathsf{Hyb}_{\ell-1}$ and Hyb_{ℓ} which use inputs

$$b^1[1], \cdots, b^1[\ell-1], b^0[\ell], \cdots, b^0[\mathsf{B_{OT}}] \text{ and } b^1[1], \cdots, b^1[\ell-1], b^1[\ell], \cdots, b^0[\mathsf{B_{OT}}]$$

respectively.

Suppose there is an adversarial sender $\mathcal{A}$ that can distinguish $\mathsf{Hyb}_{\ell-1}$ and Hyb_{ℓ}, then we construct an adversary $\mathcal{A}_{\mathsf{OT}}$ that breaks the indistinguishability security of OT. We now describe the working of $\mathcal{A}_{\mathsf{OT}}$.

To rely on the security of OT, we need to find an instance of OT that is not rewound during the experiment. Since the OT indices are sampled independently and uniformly, with non-negligible probability, any given outer index i will have inner indices in each of the $\mathsf{B_{OT}}$ rewinds to be distinct. The probability being non-negligible follows from the fact that $\mathsf{B_{OT}}$ is a constant.

We sample an outer index $\widetilde{i}$ randomly from $[n]$. We will expose one of the OTs from this tuple to an external OT receiver. To determine the index of the exposed OT, $\forall i \in [n], \ell \in [\mathsf{B_{OT}}]$, sample

$$\sigma_i[\ell] \leftarrow_{\$} [\mathsf{B_{OT}}].$$

Here we denote by $\sigma_i[\ell]$, the inner index picked for the ℓ-th rewind. If for $\widetilde{i}$, $\left\{\sigma_{\widetilde{i}}[\ell]\right\}_{\ell \in [\mathsf{B_{OT}}]}$ are not distinct, we sample again. Thus, the OT we will expose externally is the one with outer index $\widetilde{i}$, and inner index $\sigma_{\widetilde{i}}[\ell]$.

Specifically, on receiving ot_1 message from the external challenger set

$$\mathsf{ot}_1^{\widetilde{i}, \sigma_{\widetilde{i}}[\ell]} = \mathsf{ot}_1.$$

All other $\mathsf{ot}_1^{i,k}$ messages are computed honestly, using fresh randomness, by $\mathcal{A}_{\mathsf{OT}}$. All first round messages are sent to $\mathcal{A}$.

$\mathcal{A}$ responds with $\left\{\mathsf{ot}_2^{i,k}[j]\right\}_{i \in [n], k \in [\mathsf{B_{OT}}], j \in [\mathsf{B_{OT}}]}$ where $\mathsf{ot}_2^{i,k}[j]$ corresponds to the sender message to be used in the j-the thread.

From our assumption of distinct indices for outer index $\widetilde{i}$, $\forall \ell \neq \ell'$, $\sigma_{\widetilde{i}}[\ell] \neq \sigma_{\widetilde{i}}[\ell']$. This means that $\mathsf{ot}_2^{\widetilde{i}, \sigma_{\widetilde{i}}[\ell]}$ is only going to be picked once across rewinds. Thus $\mathsf{ot}_2^{\widetilde{i}, \sigma_{\widetilde{i}}[\ell]}$ can be forwarded to the external challenger without any fear of rewinding. But we also need to send it two challenge receiver bits, which we compute below.

For receiver inputs to the OT, we need to generate additive shares: $\forall i \in [n-1], \ell' \in [\mathsf{B_{OT}}] \setminus \{\ell\}$

$$b_i[\ell'] \leftarrow_{\$} \{0, 1\}$$

Now to complete the sharing, we need to set the last share bit appropriately. This is done as follows: $\forall \ell'$,

– if $\ell' < \ell$,

$$b_n[\ell'] := b^1[\ell'] \bigoplus_{i=1}^{n-1} b_i[\ell']$$

– if $\ell' < \ell$,

$$b_n[\ell'] := b^0[\ell'] \bigoplus_{i=1}^{n-1} b_i[\ell']$$

Now for ℓ, we want the $\widetilde{i}$-th share to differ, but all others to be the same. With this in mind, sample $\forall i \in [n] \setminus \left\{\widetilde{i}\right\}$

$$b_i[\ell] \leftarrow_{\$} \{0, 1\}$$

We set two special shares below:

$$b^{*,0} := b^0[\ell] \bigoplus_{\substack{i=1 \\ i \neq \widetilde{i}}}^{n} b_i[\ell]$$

$$b^{*,1} := b^1[\ell] \bigoplus_{\substack{i=1 \\ i \neq \widetilde{i}}}^{n} b_i[\ell]$$

Now if we set the challenge to be $(b^{*,0}, b^{*,1})$ then depending on the receiver bit chosen by the external challenger, we are either in hybrid $\mathsf{Hyb}_{\ell-1}$ and Hyb_{ℓ}.

Once we send the challenge, we get as response the 3round OT message corresponding to the choice bit sampled by the challenger. The remaining OT messages can be answered internally using the shares computed. The collected third round messages are now sent to $\mathcal{A}$. Thus, if $\mathcal{A}$ can distinguish the two hybrids with non-negligible probability, then $\mathcal{A}_{\mathsf{OT}}$ wins the challenge game with non-negligible probability. The only loss in advantage comes from the probability of sampling B_{OT} inner indices from the set $[\mathsf{B}_{\mathsf{OT}}]$ such that the indices are all distinct. Since B_{OT} is a constant, this still leaves the advantage to be non-negligible.

□

Remark 1. *We note that while our construction is proved against malicious senders, for our application it suffices to have the following two properties:*

- *bounded rewind security against semi malicious senders.*
- *standalone security against receivers.*

Remark 2. *While not relevant to the bounded rewind security of the scheme, we note that in our applications, a malicious sender might compute the garbled circuit incorrectly. This stems from the fact that there will be multiple participants evaluating the garbled circuit to compute the OT output. We will therefore have to prove that the messages of the protocol were in fact computed correctly.*

3.3 Four Round Delayed Input Multiparty Computation with Bounded Rewind Security

In this section we define a new notion of bounded rewind security for delayed semi-malicious MPC. To this end, we consider a four round delayed input semi-malicious protocols satisfying the following additional properties, where we denote by msg_k the messages of all parties output in the k-th round by Π.

1. **Property 1:** msg_1 and msg_2 of Π contain only messages of OT instances.
2. **Property 2:** msg_1 and msg_2 of Π do not depend on the input. The input is used only in the computation of msg_3 and msg_4.
3. **Property 3:** The simulator $\mathcal{S}$ simulates the honest parties' messages msg_1 and msg_2 via $\mathcal{S}_1$ and $\mathcal{S}_2$ by simply running the honest OT sender and receiver algorithms.

4. **Property 4:** msg_3 can be divided into two parts: (i) components independent of the OT messages; and (ii) OT messages.

Here we clarify what it means for a component of a message to be independent of OT messages. We say a component of msg_3 is independent of OT messages if its computation in the third round is independent of the both the private and public state of OT.

The recent works of [GS18, BL18] construct two round semi-malicious protocols. Both protocols when instantiated with a four round OT protocol, satisfy the above structure. This follows from the fact that when their protocols are instantiated with a four round OT protocol, the non-OT components of their protocol are executed only in round 3.

We consider the bounded rewind security of protocols satisfying the above structure, where only the second and third rounds of the protocol can be rewound. For clarity of exposition, we will refer to protocols satisfying the above properties to be *special four round delayed input semi-malicious MPC* protocols.

Definition 14 (Bounded rewind secure special four round delayed input semi-malicious MPC)**.** *A special four round delayed input semi-malicious MPC protocol is said to be secure against* B *rewinds against a semi malicious adversary if the outputs of the experiments* E^0 *and* E^1 *are indistinguishable. The experiments are parameterized by the total number of parties* n *and the total number of corrupted parties* t*. We denote the set of honest parties as* $\mathcal{H}$*, and correspondingly the set of adversarial parties as* $\overline{\mathcal{H}}$*.* Trans_k *denotes the transcript of the first* k *rounds, and by extension* $\mathsf{Trans}_{k,\ell}$ *is the transcript of the first* k *rounds on rewind* ℓ*. The experiment* E^σ *with* $\sigma \in \{0,1\}$ *is defined as follows.*

1. *Compute* $\forall i \in \mathcal{H},\ \mathsf{msg}_{1,i} := \Pi(\mathsf{r}_i)$ *and send to* $\mathcal{A}$*.*

2. *Receive* $\left\{\mathsf{msg}_{1,i}\right\}_{i\in\overline{\mathcal{H}}}$ *from* $\mathcal{A}$*.*

3. *Compute* $\forall i \in \mathcal{H},\ \mathsf{msg}_{2,i} := \Pi(\mathsf{Trans}_1, \mathsf{r}_i)$ *and send to* $\mathcal{A}$*.*

4. *Receive* $\left\{\mathsf{msg}_{2,i,\ell}\right\}_{i\in\overline{\mathcal{H}},\ell\in[\mathsf{B}-1]}$ *from* $\mathcal{A}$

5. *Compute responses to the queries as follows.* $\forall \ell \in [\mathsf{B}]$*, compute third round messages as:* $\forall i \in \mathcal{H},\ \mathsf{msg}_{3,i,\ell} \leftarrow \Pi(0, \mathsf{Trans}_{2,\ell}, \mathsf{r}_i)$*. Send* $\left\{\mathsf{msg}_{3,i,\ell}\right\}_{i\in\mathcal{H},\ell\in[\mathsf{B}]}$ *to* $\mathcal{A}$*.*

6. *Receive* $\left(\{x_i\}_{i\in[n]}, \{\mathsf{r}_i\}_{i\in\overline{\mathcal{H}}}\right)$ *and* $\left\{\mathsf{msg}_{2,i}\right\}_{i\in\overline{\mathcal{H}}}$ *from the* $\mathcal{A}$*.*

7. *Based on the value of* σ*, the the messages are computed as follows:*

   – *if* $\sigma = 0$*, compute the third and fourth round messages of the last query using the inputs provided. Specifically, compute* $\forall i \in \mathcal{H},\ \mathsf{msg}_{3,i} \leftarrow \Pi(x_i, \mathsf{Trans}_2, \mathsf{r}_i)$*, and send* $\left\{\mathsf{msg}_{3,i}\right\}_{i\in\mathcal{H}}$ *to* $\mathcal{A}$*. On receiving,* $\left\{\mathsf{msg}_{3,i}\right\}_{i\in\overline{\mathcal{H}}}$*, compute* $\forall i \in \mathcal{H},\ \mathsf{msg}_{4,i} \leftarrow \Pi(x_i, \mathsf{Trans}_3, \mathsf{r}_i)$*, and send to* $\mathcal{A}$*.*

   – *if* $\sigma = 1$*, simulate the third and fourth round messages of the last query. Specifically, compute* $\left\{\mathsf{msg}_{3,i}\right\}_{i\in\mathcal{H}} \leftarrow \mathcal{S}_3\left(\mathsf{Trans}_2, \{\mathsf{r}_i\}_{i\in\mathcal{H}}\right)$*, and send* $\left\{\mathsf{msg}_{3,i}\right\}_{i\in\mathcal{H}}$ *to* $\mathcal{A}$*. On receiving,* $\left\{\mathsf{msg}_{3,i}\right\}_{i\in\overline{\mathcal{H}}}$*, compute* $\left\{\mathsf{msg}_{4,i}\right\}_{i\in\mathcal{H}} \leftarrow \mathcal{S}_4\left(\mathsf{Trans}_3, \{x_i\}_{i\in\overline{\mathcal{H}}}, \{\mathsf{r}_i\}_{i\in[n]}\right)$*, and send to* $\mathcal{A}$*.*

8. *The output of the experiment is the view of the adversary* $\mathcal{A}$*.*

Lemma 4. *The semi malicious protocols of [GS18, BL18], when instantiated with our constructed 4 round OT with bounded rewind security, satisfies the above definition. The rewind security parameter of the resultant protocol is identical to that of the rewind secure parameter of the OT with bounded rewind security.*

We refer the reader to Remark 1 for the sufficient properties from the underlying oblivious transfer (OT) with bounded rewind security.

Proof sketch. We briefly describe why the resultant protocol is rewind secure. This primarily follows from the structure of the protocols and the bounded rewind security of the OT scheme.

To argue security, consider augmenting the protocol to allow additional threads that execute only the second and third round of the protocol multiple times. The adversary has control over what messages to send in each of the threads. On these threads, the honest inputs used are always going to be 0, with fresh randomness sampled for each thread. From the structure of the protocol, other than the OT, all components of the protocol are oblivious to rewinds in the second and third round. This follows from the fact that the components have messages no earlier than the third round.

Note that since fresh randomness is sampled to compute the third round of the protocol, this is akin to restarting the components (other than OT) with fresh randomness. Thus, when we have to rely on the bounded rewind security of OT, the other components of the third round can be computed without knowledge of the private state of the OT challenger.

4 Four Round MPC

Components. We list below the components of our protocol.

- $\mathsf{TDGen} = (\mathsf{TDGen}_1, \mathsf{TDGen}_2, \mathsf{TDGen}_3, \mathsf{TDOut}, \mathsf{TDValid}, \mathsf{TDExt})$ is a three round bounded rewind secure trapdoor generation protocol based on one-way functions (see Section 2.5).
  - TDOut computes the receiver's output.
  - $\mathsf{TDValid}$ determines whether an input trapdoor value is valid with respect to the first round of the protocol transcript.
  - TDExt computes a valid trapdoor given B_{td} distinct protocol transcripts that share the same first message

  Here B_{td} is set to be 2.

- $\mathsf{WI} = (\mathsf{WI}_1, \mathsf{WI}_2, \mathsf{WI}_3, \mathsf{WI}_4)$ is a three round delayed-input witness indistinguishable proof system (see Section 2.6), where WI_4 is used to compute the decision of the verifier.

- $\mathsf{RWI} = (\mathsf{RWI}_1, \mathsf{RWI}_2, \mathsf{RWI}_3, \mathsf{RWI}_4)$ is a three round delayed-input witness-indistinguishable proof with B-bounded rewind security (see Section 2.6). RWI_4 is used to compute the decision of the verifier. Their construction can be parameterized by multiple values of B, but we set B_{rwi} to be some polynomial.

- $\mathsf{NMCom} = (\mathsf{NMCom}_1, \mathsf{NMCom}_2, \mathsf{NMCom}_3)$ is a three round special non-malleable commitment scheme (see Section 2.7). Let $\mathsf{Ext}_{\mathsf{NMCom}}$ denote the extractor associated with NMCom.

- $\mathsf{OT} = (\mathsf{OT}_1, \mathsf{OT}_2, \mathsf{OT}_3, \mathsf{OT}_4)$ is a four round oblivious transfer protocol. We abuse notation slightly and use this as implementing parallel OT executions where the receiver's input is a string of length ℓ and the sender now has ℓ pairs of inputs. We require regular indistinguishability security against a malicious sender. In addition, we require extraction of the receiver's input bit.

– $\mathsf{RECom} = (\mathsf{RECom}_1, \mathsf{RECom}_2, \mathsf{RECom}_3, \mathsf{Ext}_{\mathsf{RECom}})$ is the three round bounded rewind secure delayed-input extractable commitment based on non-interactive commitments (see Section 2.4). We set rewinding security parameter B_{recom} to be 4. $\mathsf{Ext}_{\mathsf{RECom}}$ is the extractor associated with RECom.

– $\mathsf{Ecom} = (\mathsf{Ecom}_1, \mathsf{Ecom}_2, \mathsf{Ecom}_3, \mathsf{Ext}_{\mathsf{Ecom}})$ is the three round delayed-input extractable commitment scheme based on statistically binding commitment schemes (see Section 2.3). These have been used is several prior works, most notably in [PRS02]. They satisfy the 2-extraction property.

– A four round B_Π-bounded rewind secure delayed input MPC protocol Π based on oblivious transfer (see Section 3.3). We denote by Trans_k, the transcript of the first k rounds of the protocol Π. We set B_Π to be 9.

– $\mathsf{GC} = (\mathsf{Garble}, \mathsf{Eval})$ is a secure garbling scheme (see Section 2.2). We denote the labels $\{\mathsf{lab}_{i,0}, \mathsf{lab}_{i,1}\}_{i\in[L]}$ by $\overline{\mathsf{lab}}$.

For primitives with bounded rewind security, we require

$$B_{\mathsf{rwi}_a}, B_{\mathsf{rwi}_b}, B_\Pi > B_{\mathsf{recom}} > B_{\mathsf{td}}$$

where they denote the total number of rewinds (including the main thread) that they are secure against. In addition, we require all of them to be larger than the number of threads required to extract from NMCom and Ecom. For the primitives picked, we have, $B_{\mathsf{rwi}_a} = B_{\mathsf{rwi}_b} = \mathsf{poly}(\lambda)$ (for some fixed polynomial), $B_\Pi = 9$, $B_{\mathsf{recom}} = 4$ and $B_{\mathsf{td}} = 2$ thus satisfying our requirements.

NP languages. The proofs are associated with the following languages.

– Language L_a is characterized by the following relation R_a:

Statement: $\mathsf{st} := \left(\left\{\mathsf{recom}_i^j, \mathsf{nmcom}_i\right\}_{i\in[3], j\in[n]}, \mathsf{Trans}_2, \{\mathsf{msg}_i\}_{i\in[3]}, \mathsf{td}_1\right)$

Witness: $\mathsf{w} := \left(\mathsf{inp}, \mathsf{r}, \left\{\mathsf{r}_{\mathsf{recom}}^j\right\}_{j\in[n]}, \mathsf{t}, \mathsf{r}_{\mathsf{nmcom}}\right)$

$R_a(\mathsf{st}, \mathsf{w}) = 1$ if and only if

1. for every j, $\left(\mathsf{recom}_1^j, \mathsf{recom}_2^j, \mathsf{recom}_3^j\right)$ is a well-formed transcript of RECom with respect to the input $(\mathsf{inp}, \mathsf{r})$ and randomness $\mathsf{r}_{\mathsf{recom}}^j$, and for every $i \le 3$, msg_i is an honestly computed third round message in the protocol Π with respect to input inp, randomness r and the first $i-1$ round protocol transcript Trans_{i-1} (OR)
2. $(\mathsf{nmcom}_1, \mathsf{nmcom}_2, \mathsf{nmcom}_3)$ is a transcript of a non-malleable commitment of NMCom with respect to the input t and randomness $\mathsf{r}_{\mathsf{nmcom}}$ and t is a valid trapdoor with respect to td_1

Formally, $R_a(\mathsf{st}, \mathsf{w}) = 1$ if and only if:

– $\forall j \in [n]$, $\mathsf{recom}_1^j = \mathsf{RECom}_1(\mathsf{r}_{\mathsf{recom}}^j)$ AND
– $\forall j \in [n]$, $\mathsf{recom}_3^j = \mathsf{RECom}_3((\mathsf{inp}, \mathsf{r}), \mathsf{recom}_1^j, \mathsf{recom}_2^j; \mathsf{r}_{\mathsf{recom}}^j)$ AND
– $\forall j \in [n]$, $\left(\mathsf{recom}_1^j, \mathsf{recom}_2^j, \mathsf{recom}_3^j\right)$ is well-formed with respect to input $(\mathsf{inp}, \mathsf{r})$ and randomness $\mathsf{r}_{\mathsf{recom}}^j$ AND

- $\mathsf{msg}_1 = \Pi_1(\mathsf{r})$ AND
- $\mathsf{msg}_2 = \Pi_2(\mathsf{Trans}_1; \mathsf{r})$ AND
- $\mathsf{msg}_3 = \Pi_3(\mathsf{inp}, \mathsf{Trans}_2; \mathsf{r})$

(OR)

- $\mathsf{TDValid}(\mathsf{td}_1, \mathsf{t}) = 1$ AND
- $\mathsf{nmcom}_1 = \mathsf{NMCom}_1(\mathsf{r}_{\mathsf{nmcom}})$ AND
- $\mathsf{nmcom}_3 = \mathsf{NMCom}_3(\mathsf{t}, \mathsf{nmcom}_1, \mathsf{nmcom}_2, \mathsf{r}_{\mathsf{nmcom}})$

In our protocol, the language L_a will be used for the first instance of the bounded-rewinding secure delayed-input RWI proofs. When we consider proofs between prover P_i and verifier P_j, we denote the language as $L_a^{i\to j}$.

– Language L_b is characterized by the following relation R_b:

Statement:

$$\mathsf{st}^\ell := \left(\left\{\mathsf{recom}_i^j, \mathsf{ecom}_i^j, \mathsf{nmcom}_i^j\right\}_{i\in[3], j\in[n]}, \mathsf{Trans}_2, \{\mathsf{msg}_i\}_{i\in[3]}, \left\{\mathsf{rwi}_i^j\right\}_{i\in[2], j\in[n]}, \mathsf{td}_1\right)$$

Witness: $\mathsf{w} := \left(\left\{\mathsf{r}_{\mathsf{ecom}}^j, \mathsf{r}_{\mathsf{rwi}}^j, \widehat{\mathsf{w}}^j\right\}_{j\in[n]}, \mathsf{t}, \mathsf{r}_{\mathsf{nmcom}}\right)$ where $\widehat{\mathsf{w}}^j := \left(\mathsf{inp}, \mathsf{r}, \{\mathsf{r}_{\mathsf{recom}}^k\}_{k\in[n]}, \perp, \perp\right)$.

$R_b(\mathsf{st}, \mathsf{w}) = 1$ if and only if

1. $\forall j \in [n]$, $\left(\mathsf{ecom}_1^j, \mathsf{ecom}_2^j, \mathsf{ecom}_3^j\right)$ is a well-formed transcript of Ecom with respect to the input $\{\mathsf{rwi}_3^k\}_{k\in[n]}$ and randomness $\mathsf{r}_{\mathsf{ecom}}^j$, and $\forall k$, $\{\mathsf{rwi}_i^k\}_{i\in[3]}$ is an honestly computed transcript for L_a (OR)
2. $(\mathsf{nmcom}_1, \mathsf{nmcom}_2, \mathsf{nmcom}_3)$ is a transcript of a non-malleable commitment of NMCom with respect to the input t and randomness $\mathsf{r}_{\mathsf{nmcom}}$ and t is a valid trapdoor with respect to td_1.

Formally, $R_b(\mathsf{st}, \mathsf{w}) = 1$ if and only if

- $\forall j \in [n]$, $\mathsf{ecom}_1^j = \mathsf{Ecom}_1(\mathsf{r}_{\mathsf{ecom}}^j)$ AND
- $\forall j \in [n]$, $\mathsf{ecom}_3^j = \mathsf{Ecom}_3(\{\mathsf{rwi}_3^k\}_{k\in[n]}, \mathsf{ecom}_1^j, \mathsf{ecom}_2^j; \mathsf{r}_{\mathsf{ecom}}^j)$ AND
- $\forall j \in [n]$, $\left(\mathsf{ecom}_1^j, \mathsf{ecom}_2^j, \mathsf{ecom}_3^j\right)$ is well-formed with respect to input $\{\mathsf{rwi}_3^k\}_{k\in[n]}$ and randomness $\mathsf{r}_{\mathsf{ecom}}^j$ AND
- $\forall j \in [n]$, $\mathsf{rwi}_1^j = \mathsf{RWI}_1(1^\lambda; \mathsf{r}_{\mathsf{rwi}}^j)$ AND
- $\forall j \in [n]$, $\mathsf{rwi}_3^j = \mathsf{RWI}_3(\widehat{\mathsf{st}}^j, \widehat{\mathsf{w}}^j, \mathsf{rwi}_1^j, \mathsf{rwi}_2^j; \mathsf{r}_{\mathsf{rwi}}^j)$ where

$$\widehat{\mathsf{st}}^j := \left(\left\{\mathsf{recom}_i^j\right\}_{i\in[3]}, \mathsf{Trans}_2, \{\mathsf{msg}_i\}_{i\in[3]}, \left\{\mathsf{nmcom}_i^j\right\}_{i\in[3]}, \mathsf{td}_1\right)$$
$$\widehat{\mathsf{w}}^j := \left(\mathsf{inp}, \mathsf{r}, \left\{\mathsf{r}_{\mathsf{recom}}^k\right\}_{k\in[n]}, \perp, \perp\right)$$

(OR)

- $\mathsf{TDValid}(\mathsf{td}_1, \mathsf{t}) = 1$ AND
- $\mathsf{nmcom}_1^\ell = \mathsf{NMCom}_1(\mathsf{r}_{\mathsf{nmcom}})$ AND
- $\mathsf{nmcom}_3^\ell = \mathsf{NMCom}_3(\mathsf{t}, \mathsf{nmcom}_1^\ell, \mathsf{nmcom}_2^\ell, \mathsf{r}_{\mathsf{nmcom}})$

In our protocol, the language L_b will be used for the second instance of the bounded-rewinding secure delayed-input RWI proofs. When we consider proofs between prover P_i and verifier P_j, we denote the language as $L_b^{i \to j}$.

– Language L_c is characterized by the following relation R_c:

Statement:

$$\mathsf{st} := \left(\{\mathsf{msg}_i, \mathsf{nmcom}_i\}_{i\in[3]}, \left\{\mathsf{recom}_i^j\right\}_{i\in[3], j\in[n]}, \left\{\mathsf{rwi}_i^j\right\}_{i\in[2], j\in[n]}, \mathsf{Trans}_3, \left\{\mathsf{ot}_i^j\right\}_{i\in[4], j\in[n]}, \mathsf{td}_1, \left\{\mathsf{st}^j\right\}_{j\in[n]}, \widetilde{\mathsf{C}} \right)$$

Witness: $\mathsf{w} := \left(\mathsf{inp}, \mathsf{r}, \left\{\mathsf{r}_{\mathsf{recom}}^j\right\}_{j\in[n]}, \mathsf{r}_{\mathsf{gc}}, \left\{\mathsf{r}_{\mathsf{ot}}^j\right\}_{j\in[n]}, \mathsf{t}, \mathsf{r}_{\mathsf{nmcom}} \right)$

$R_c(\mathsf{st}, \mathsf{w}) = 1$ if and only if

1. for every j, $\left(\mathsf{recom}_1^j, \mathsf{recom}_2^j, \mathsf{recom}_3^j\right)$ is a well-formed transcript of RECom with respect to the input $(\mathsf{inp}, \mathsf{r})$ and randomness $\mathsf{r}_{\mathsf{recom}}^j$. The garbled circuit $\widetilde{\mathsf{C}}$ is computed correctly with randomness r_{gc} and embeds msg_4, the honestly computed fourth round message in the protocol Π with respect to input inp, randomness r and first three round protocol transcript Trans_2 (OR)
2. $(\mathsf{nmcom}_1, \mathsf{nmcom}_2, \mathsf{nmcom}_3)$ is a transcript of a non-malleable commitment of NMCom with respect to the input t and randomness $\mathsf{r}_{\mathsf{nmcom}}$ and t is a valid trapdoor with respect to td_1

Formally, $R_c(\mathsf{st}, \mathsf{w}) = 1$ if and only if:

- $\forall j \in [n]$, $\mathsf{recom}_1^j = \mathsf{RECom}_1(\mathsf{r}_{\mathsf{recom}}^j)$ AND
- $\forall j \in [n]$, $\mathsf{recom}_3^j = \mathsf{RECom}_3((\mathsf{inp}, \mathsf{r}), \mathsf{recom}_1^j, \mathsf{recom}_2^j; \mathsf{r}_{\mathsf{recom}}^j)$ AND
- $\forall j \in [n]$, $(\mathsf{recom}_1, \mathsf{recom}_2, \mathsf{recom}_3)$ is well-formed with respect to input $(\mathsf{inp}, \mathsf{r})$ and randomness $\mathsf{r}_{\mathsf{recom}}$ AND
- $\mathsf{msg}_1 = \Pi_1(\mathsf{r})$ AND
- $\mathsf{msg}_2 = \Pi_2(\mathsf{Trans}_1; \mathsf{r})$ AND
- $\mathsf{msg}_3 = \Pi_3(\mathsf{inp}, \mathsf{Trans}_2; \mathsf{r})$ AND
- $\mathsf{msg}_4 = \Pi_4(\mathsf{inp}, \mathsf{Trans}_3; \mathsf{r})$ AND
- $\left(\widetilde{\mathsf{C}}, \overline{\mathsf{lab}}\right) := \mathsf{Garble}\left(\mathsf{C}\left[\mathsf{msg}_4, \left\{\mathsf{rwi}_i^j\right\}_{i\in[2], j\in[n]} \left\{\mathsf{st}^j, \mathsf{r}_{\mathsf{rwi}}^{j\to i}\right\}_{j\in[n]}\right]; \mathsf{r}_{\mathsf{gc}}\right)$ where the circuit C is defined in Figure 5 AND
- for all $j \in [n]$, $\mathsf{ot}_4^j := \mathsf{OT}_4\left(\overline{\mathsf{lab}}_{|_j}, \mathsf{ot}_1^j, \mathsf{ot}_2^j, \mathsf{ot}_3^j; \mathsf{r}_{\mathsf{ot}}^j\right)$.

(OR)

$$\mathsf{C}\left[i, \mathsf{msg}_{4,i}, \left\{\mathsf{rwi}_{\ell}^{j\to i}\right\}_{\ell\in[2], j\in[n]\setminus\{i\}} \left\{\mathsf{st}^{j\to i}, \mathsf{r}_{\mathsf{rwi}}^{j\to i}\right\}_{j\in[n]\setminus\{i\}}\right]$$

Input: $\left\{\mathsf{rwi}_3^{j\to i}\right\}_{j\in[n]\setminus\{i\}}$

– If $\forall j \in [n] \setminus \{i\}$,

$$\mathsf{RWI}_4\left(\mathsf{rwi}_1^{j\to i}, \mathsf{rwi}_2^{j\to i}, \mathsf{rwi}_3^{j\to i}, \mathsf{st}^{j\to i}; \mathsf{r}_{\mathsf{rwi}}^{j\to i}\right) = 1$$

then **output** $\mathsf{msg}_{4,i}$;

– Else, **output** $\perp$.

Figure 5: Circuit C

– $\mathsf{TDValid}(\mathsf{td}_1, \mathsf{t}) = 1$ AND
– $\mathsf{nmcom}_1 = \mathsf{NMCom}_1(\mathsf{r}_{\mathsf{nmcom}})$ AND
– $\mathsf{nmcom}_3 = \mathsf{NMCom}_3(\mathsf{t}, \mathsf{nmcom}_1, \mathsf{nmcom}_2\mathsf{r}_{\mathsf{nmcom}})$

In our protocol, the language L_c will be used for the delayed-input WI proofs. When we consider proofs between prover P_i and verifier P_j, we denote the language as $L_c^{i\to j}$.

Given the above primitives, our main result is stated in the following theorem.

Theorem 5. *Assuming the hiding property of oblivious transfer, the hiding property of extractable commitment, the hiding property of extractable commitment with bounded rewind security, delayed semi malicious protocol with bounded rewind security computing any function $\mathcal{F}$, special non-malleable commitments, witness indistinguishable proofs with bounded rewind security, security of garbled circuits, trapdoor generation protocol with bounded rewind security, the presented protocol is a four round protocol for $\mathcal{F}$ secure against a dishonest majority.*

Remark 3. *All the above primitives can be based on one-way functions, non-interactive commitments and oblivious trasnfer (OT).*

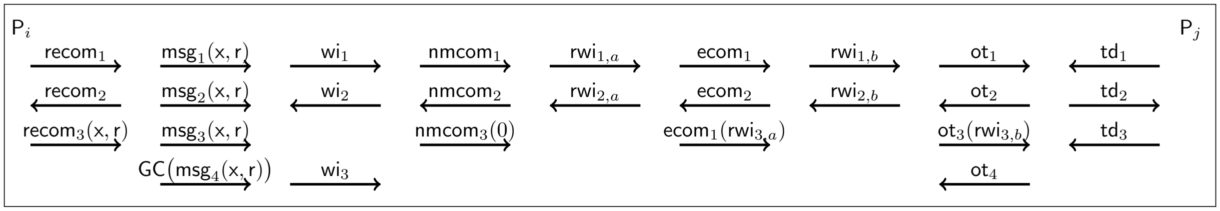
In a recent note by Lombardi and Schaeffer [LS19], they give a construction of a perfectly binding non-interactive commitment based on perfectly correct key agreement. As they point out, such key agreement schemes can be based on perfectly correct oblivious transfer [GKM+00].

This gives us both a non-interactive commitment schemes, and one-way functions, based on perfectly correct oblivious transfer. Thus it suffices to instantiate all our primitives using just oblivious transfer.

We thus have the following corollary.

Corollary 1. *Assuming polynomially secure oblivious transfer, our constructed protocol is a four round multiparty computation protocol for any function $\mathcal{F}$.*

4.1 The Protocol



Due to the complex nature of the protocol, and the presence of multiple primitives, we describe the overall structure of the protocol at a high level to demonstrate the purpose of its various components in the context of the protocol. For simplicity we consider the messages sent from P_i to P_j. Note that even though P_j is the intended recipient for the messages in a two party sub-protocol, the messages are broadcast to all parties.

– P_i uses input x and randomness r to compute the messages msg_k for Π. Since we are not guaranteed of the honest behavior of adversarial parties, the last message of Π is not sent in the clear but instead sent inside a garbled circuit GC. The garbled circuit is used to verify the honest behavior in the first three rounds of the protocol. That is, if P_j misbehaved in the first three rounds that the garbled circuit becomes "opaque", and P_j cannot retrieve the last message of Π.

– The same input and randomness used to compute messages for Π is committed via a bounded-rewinding-secure extractable commitment recom_k. This is done to enable the simulator to extract the inputs and randomness of the adversary for simulation. The bounded-rewinding security property is used in a critical way in our security proof to establish that the honest party inputs remain "hidden" even when we extract the adversary's input (and randomness).

– The trapdoor messages td_k are used to establish a trapdoor condition for the witness indistinguishable proofs, which is not relevant for the honest execution. Trapdoor extraction requires the simulator to rewind the transcript of the trapdoor generation protocol.

– The nmcom_k messages are likewise only used for the purpose of simulation, where we commit to the trapdoor during simulation. In the honest case this will just be a commitment to 0.

– In order to decide whether or not to extract the adversary's input, the simulator needs to know if the extractable commitments to the input are "well formed" (since otherwise, polynomial-time extraction is not guaranteed). The well-formedness of the extractable commitment is proven via a bounded-rewinding-secure witness indistinguishable proof rwi_a. For technical reasons that will become apparent in the proof, we do not send them in the clear but put them inside another (non-rewinding-secure) extractable commitment ecom. This will enable the simulator to extract the proofs, check if they are accepting before proceeding to extract adversary's input.

– To be able to evaluate the garbled circuit and obtain the last round message of a party, the circuit must take in as inputs proofs of honest behavior, to verify. We need an OT to facilitate this, but our proofs of honest behavior are already within an extractable commitment. Towards that end, we require the parties to commit to another bounded-rewinding-secure WI proof rwi_b inside the OT receiver message that proves that the proof inside extractable commitment ecom is accepting. These "proofs of proofs" constitute the input that is verified by the garbled circuits.

– The last witness indistinguishable proof wi is used to establish honest behavior of the last round of the protocol which constitutes computing the garbled circuit and the corresponding labels. We do not require any bounded-rewinding security from this WI; instead, stand-alone security suffices.

In all three proofs, the trapdoor condition is the same. Namely, that the non-malleable commitment commits to the relevant trapdoor.

We point out that the OT messages in the figure above do not "pair" with the garbled circuit GC that is depicted in the figure. They are shown here to illustrate the various computations done by a single party.

We now describe our four round protocol between n players $\mathsf{P}_1, \cdots, \mathsf{P}_n$. The input of party P_i is denoted as x_i. We will denote by r_π the randomness used in primitive π, with appropriate superscript to identify the specific instantiation of the primtive.

Round 1: P_i does the following:

1. Compute the first round message of the underlying protocol Π,

$$\mathsf{msg}_{1,i} := \Pi_1\left(\mathsf{r}_i\right)$$

   using randomness r_i Recall that the first two messages of Π are independent of the party's input.

2. Compute the first round of the trapdoor generation phase TDGen,

$$\mathsf{td}_{1,i} := \mathsf{TDGen}_1\left(\mathsf{r}_{\mathsf{td},i}\right)$$

   using randomness $\mathsf{r}_{\mathsf{td},i}$

3. Compute first round of the three input delayed witness indistinguishable proof systems. Specifically, $\forall j \in [n] \setminus \{i\}$, compute

   – first round of the input-delayed witness indistinguishable proof system for L_c

$$\mathsf{wi}_1^{i\to j} := \mathsf{WI}_1\left(\mathsf{r}_{\mathsf{wi}}^{i\to j}\right)$$

   – first round of the input-delayed rewinding secure witness indistinguishable proof system for L_a

$$\mathsf{rwi}_{a,1}^{i\to j} := \mathsf{RWI}_1\left(\mathsf{r}_{\mathsf{rwi},a}^{i\to j}\right)$$

   – first round of the input-delayed rewinding secure witness indistinguishable proof system for L_b

$$\mathsf{rwi}_{b,1}^{i\to j} := \mathsf{RWI}_1\left(\mathsf{r}_{\mathsf{rwi},b}^{i\to j}\right)$$

4. Compute first round of the three commitment schemes. Specifically, $\forall j \in [n] \setminus \{i\}$, compute

   – first round of the extractable commitment scheme:

   $$\mathsf{ecom}_1^{i \to j} := \mathsf{Ecom}_1\left(\mathsf{r}_{\mathsf{ecom}}^{i \to j}\right)$$

   using the randomness $\mathsf{r}_{\mathsf{ecom}}^{i \to j}$.

   – first round of the rewinding-secure extractable commitment scheme:

   $$\mathsf{recom}_1^{i \to j} := \mathsf{RECom}_1\left(\mathsf{r}_{\mathsf{recom}}^{i \to j}\right)$$

   using the randomness $\mathsf{r}_{\mathsf{recom}}^{i \to j}$.

   – first round of non-malleable commitment scheme:

   $$\mathsf{nmcom}_1^{i \to j} := \mathsf{NMCom}_1\left(\mathsf{r}_{\mathsf{nmcom}}^{i \to j}\right)$$

   using the randomness $\mathsf{r}_{\mathsf{nmcom}}^{i \to j}$.

5. The first round of the OT scheme, where P_i is the receiver. Specifically, $\forall j \in [n] \setminus \{i\}$, compute

   $$\mathsf{ot}_1^{j \to i} := \mathsf{OT}_1\left(\mathsf{r}_{\mathsf{ot}}^{j \to i}\right)$$

   using the randomness $\mathsf{r}_{\mathsf{ot}}^{j \to i}$. Here the superscript $j \to i$ indicates that the OT message is for the instances where P_i is the receiver.

6. Broadcast

   $$\left(\mathsf{msg}_{1,i}, \mathsf{td}_{1,i}, \left\{\mathsf{wi}_1^{i \to j}, \mathsf{rwi}_{a,1}^{i \to j}, \mathsf{rwi}_{b,1}^{i \to j}, \mathsf{ecom}_1^{i \to j}, \mathsf{recom}_1^{i \to j}, \mathsf{nmcom}_1^{i \to j}, \mathsf{ot}_1^{j \to i}\right\}_{j \in [n] \setminus \{i\}}\right)$$

   to all other parties.

Round 2: P_i does the following:

1. Compute the second round message of the underlying protocol Π,

   $$\mathsf{msg}_{2,i} := \Pi_2\left(\mathsf{Trans}_1; \mathsf{r}_i\right)$$

   using randomness r_i and the transcript obtained so far.

2. Compute the second round of the trapdoor generation phase TDGen, $\forall j \in [n] \setminus \{i\}$

   $$\mathsf{td}_2^{i \to j} \leftarrow \mathsf{TDGen}_2\left(\mathsf{td}_{1,j}\right)$$

   using randomness $\mathsf{r}_{\mathsf{td},i}$

3. Compute second round of the three input delayed witness indistinguishable proof systems, where P_i takes the role of the verifier. Specifically, $\forall j \in [n] \setminus \{i\}$, compute

   – second round of the input-delayed witness indistinguishable proof system

   $$\mathsf{wi}_2^{j \to i} := \mathsf{WI}_2\left(\mathsf{wi}_1^{j \to i}; \mathsf{r}_{\mathsf{wi}}^{j \to i}\right)$$

– second round of the input-delayed rewinding secure witness indistinguishable proof system for L_a

$$\mathsf{rwi}_{a,2}^{j\to i} := \mathsf{RWI}_2\left(\mathsf{rwi}_{a,1}^{j\to i}; \mathsf{r}_{\mathsf{rwi},a}^{j\to i}\right)$$

– second round of the input-delayed rewinding secure witness indistinguishable proof system for L_b

$$\mathsf{rwi}_{b,2}^{j\to i} := \mathsf{RWI}_2\left(\mathsf{rwi}_{b,1}^{j\to i}; \mathsf{r}_{\mathsf{rwi},b}^{j\to i}\right)$$

Note that the superscript $j \to i$ denotes that P_i is computing the second round message of the proof where P_j is the prover and P_i is the verifier.

4. Compute second round of the three commitment schemes. Specifically, $\forall j \in [n] \setminus \{i\}$, compute

– second round of the extractable commitment scheme:

$$\mathsf{ecom}_2^{j\to i} \leftarrow \mathsf{Ecom}_2\left(\mathsf{ecom}_1^{j\to i}\right)$$

– second round of the rewinding-secure extractable commitment scheme:

$$\mathsf{recom}_2^{j\to i} \leftarrow \mathsf{RECom}_2\left(\mathsf{recom}_1^{j\to i}\right)$$

– second round of non-malleable commitment scheme:

$$\mathsf{nmcom}_2^{j\to i} \leftarrow \mathsf{NMCom}_2\left(\mathsf{nmcom}_1^{j\to i}\right)$$

As in the case of the proofs, the superscript $j \to i$ denotes that P_i is the receiver in the commitment from P_j.

5. The second round of the OT scheme, where P_i is the sender. Specifically, $\forall j \in [n] \setminus \{i\}$, compute

$$\mathsf{ot}_2^{i\to j} := \mathsf{OT}_2\left(\mathsf{ot}_1^{i\to j}; \mathsf{r}_{i,\mathsf{ot}}^{i\to j}\right).$$

Here the superscript $i \to j$ indicates that the OT message is for the instances where P_i is the sender.

6. Broadcast

$$\left(\mathsf{msg}_{2,i}, \left\{\mathsf{td}_2^{i\to j}, \mathsf{wi}_2^{j\to i}, \mathsf{rwi}_{a,2}^{j\to i}, \mathsf{rwi}_{b,1}^{j\to i}, \mathsf{ecom}_2^{j\to i}, \mathsf{recom}_2^{j\to i}, \mathsf{nmcom}_2^{j\to i}, \mathsf{ot}_2^{i\to j}\right\}_{j\in[n]\setminus\{i\}}\right)$$

to all other parties.

Round 3: P_i does the following:

1. Compute the third round message of the underlying protocol Π,

$$\mathsf{msg}_{3,i} := \Pi_3\left(\mathsf{x}_i, \mathsf{Trans}_2; \mathsf{r}_i\right)$$

using input x_i, randomness r_i and the transcript obtained so far. Note that this is the first step in the protocol that P_i is using its input x_i.

2. Compute the third round of the trapdoor generation phase TDGen. Let $\mathsf{td}_{2,i} := \left(\mathsf{td}_2^{1\to i}||\cdots||\mathsf{td}_2^{1\to i}\right)$, compute

$$\mathsf{td}_{3,i} \leftarrow \mathsf{TDGen}_2\left(\mathsf{td}_{1,i}, \mathsf{td}_{2,i}; \mathsf{r}_{\mathsf{td},i}\right).$$

3. Compute the third round of the non-malleable commitment scheme to commit to $\perp$. Specifically, $\forall j \in [n] \setminus \{i\}$, compute

$$\mathsf{nmcom}_3^{i\to j} \leftarrow \mathsf{NMCom}_3\left(0, \mathsf{nmcom}_1^{i\to j}, \mathsf{nmcom}_3^{i\to j}; \mathsf{r}_{\mathsf{nmcom}}^{i\to j}\right)$$

using the randomness $\mathsf{r}_{\mathsf{nmcom}}^{i\to j}$.

4. Compute the third round of the rewinding-secure extractable commitment scheme to commit to $(\mathsf{x}_i, \mathsf{r}_i)$. Specifically, $\forall j \in [n] \setminus \{i\}$, compute

$$\mathsf{recom}_3^{i\to j} \leftarrow \mathsf{RECom}_3\left((\mathsf{x}_i, \mathsf{r}_i), \mathsf{recom}_1^{i\to j}, \mathsf{recom}_2^{i\to j}; \mathsf{r}_{\mathsf{recom}}^{i\to j}\right)$$

using the randomness $\mathsf{r}_{\mathsf{recom}}^{i\to j}$.

5. Compute the third round of the input-delayed rewinding secure witness indistinguishable proof system for L_a i.e. prove that *all* rewind secure extractable commitments contain the input and randomness used in the underlying protocol Π. Specifically, $\forall j \in [n] \setminus \{i\}$, set

$$\mathsf{st}_a^{i\to j} := \left(\left\{\mathsf{recom}_\ell^{i\to k}, \mathsf{nmcom}_\ell^{i\to k}\right\}_{\ell\in[3],k\in[n]\setminus\{i\}}, \left\{\mathsf{msg}_{\ell,i}\ \right\}_{\ell\in[3]} \mathsf{Trans}_2, \mathsf{td}_{1,j}\right)$$
$$\mathsf{w}_a^{i\to j} := \left(\mathsf{x}_i, \mathsf{r}_i, \left\{\mathsf{r}_{\mathsf{recom}}^{i\to k}\right\}_{k\in[n]\setminus\{i\}}, \perp, \perp\right)$$

and compute

$$\mathsf{rwi}_{a,3}^{i\to j} := \mathsf{RWI}_3\left(\mathsf{st}_a^{i\to j}, \mathsf{w}_a^{i\to j}, \mathsf{rwi}_{a,1}^{i\to j}, \mathsf{rwi}_{a,2}^{i\to j}; \mathsf{r}_{\mathsf{rwi},a}^{i\to j}\right).$$

6. Compute the third round of the extractable commitment scheme to commit to the third round proof $\mathsf{rwi}_{a,3}^{i\to j}$. Specifically, $\forall j \in [n] \setminus \{i\}$, compute

$$\mathsf{ecom}_3^{i\to j} \leftarrow \mathsf{Ecom}_3\left(\left\{\mathsf{rwi}_{a,3}^{i\to k}\right\}_{k\in[n]\setminus\{i\}}, \mathsf{ecom}_1^{i\to j}, \mathsf{ecom}_3^{i\to j}; \mathsf{r}_{\mathsf{ecom}}^{i\to j}\right)$$

using the randomness $\mathsf{r}_{\mathsf{ecom}}^{i\to j}$.

7. Compute the third round of the input-delayed rewinding secure witness indistinguishable proof system for L_b. Specifically, $\forall j \in [n] \setminus \{i\}$, set

$$\mathsf{st}_b^{i\to j} := \Big(\left\{\mathsf{recom}_\ell^{i\to k}, \mathsf{ecom}_\ell^{i\to k}, \mathsf{nmcom}_\ell^{i\to k}\right\}_{\ell\in[3],k\in[n]\setminus\{i\}},$$
$$\left\{\mathsf{msg}_{\ell,i}\right\}_{\ell\in[3]}, \left\{\mathsf{rwi}_{a,\ell}^{i\to j}\right\}_{\ell\in[2]}, \mathsf{Trans}_2, \mathsf{td}_{1,j}\Big)$$
$$\mathsf{w}_b^{i\to j} := \left(\left\{\mathsf{r}_{\mathsf{ecom}}^{i\to k}, \mathsf{r}_{\mathsf{rwi},a}^{i\to j}, \mathsf{w}_a^{i\to k}\right\}_{k\in[n]\setminus\{i\}}, \perp, \perp\right)$$

where $\mathsf{w}_a^{i\to j}$ is as defined above. Compute

$$\mathsf{rwi}_{b,3}^{i\to j} := \mathsf{RWI}_3\left(\mathsf{st}_b^{i\to j}, \mathsf{w}_b^{i\to j}, \mathsf{rwi}_{b,1}^{i\to j}, \mathsf{rwi}_{b,2}^{i\to j}; \mathsf{r}_{\mathsf{rwi},b}^{i\to j}\right).$$

8. Compute the third round of the OT scheme, where P_i is the receiver. Specifically, $\forall j \in [n]\setminus\{i\}$, compute
$$\mathsf{ot}_3^{j\to i} := \mathsf{OT}_3\left(\mathsf{rwi}_{b,3}^{i\to j}, \mathsf{ot}_1^{j\to i}, \mathsf{ot}_2^{j\to i}; \mathsf{r}_{i,\mathsf{ot}}^{j\to i}\right).$$
Here the superscript $j \to i$ indicates that the OT message is for the instances where P_i is the receiver.

9. Broadcast
$$\left(\mathsf{msg}_{3,i}, \mathsf{td}_{3,i}, \left\{\mathsf{ecom}_3^{i\to j}, \mathsf{recom}_3^{i\to j}, \mathsf{nmcom}_3^{i\to j}, \mathsf{ot}_3^{j\to i}\right\}_{j\in[n]\setminus\{i\}}\right)$$
to all other parties.

Round 4: P_i does the following:

1. If $\exists j \in [n] \setminus \{i\}$, such that
$$\mathsf{TDValid}(\mathsf{td}_{1,j}, \mathsf{td}_{2,j}, \mathsf{td}_{3,j}) \neq 1$$
where $\mathsf{td}_{2,j}$ is computed as earlier, **abort**.

2. Compute the fourth round message of the underlying protocol Π,
$$\mathsf{msg}_{4,i} := \Pi_4\left(\mathsf{x}_i, \mathsf{Trans}_3; \mathsf{r}_i\right)$$
using input x_i, randomness r_i and the transcript obtained so far.

3. Compute the garbled circuits containing $\mathsf{msg}_{4,i}$. Specifically,
$$\left(\widetilde{\mathsf{C}}_i, \overline{\mathsf{lab}}_i\right) := \mathsf{Garble}\left(\mathsf{C}\left[i, \mathsf{msg}_{4,i}, \left\{\mathsf{rwi}_{b,\ell}^{j\to i}\right\}_{\ell\in[2], j\in[n]\setminus\{i\}} \left\{\mathsf{st}_b^{j\to i}, \mathsf{r}_{\mathsf{rwi},b}^{j\to i}\right\}_{j\in[n]\setminus\{i\}}\right]; \mathsf{r}_{\mathsf{gc},i}\right)$$
where $\mathsf{st}_b^{j\to i}$ is computed as above.

4. Compute the fourth round of the OT scheme, where P_i is the sender. Specifically, $\forall j \in [n]\setminus\{i\}$, compute
$$\mathsf{ot}_4^{i\to j} := \mathsf{OT}_4\left(\overline{\mathsf{lab}}_{i|_j}, \mathsf{ot}_1^{i\to j}, \mathsf{ot}_2^{i\to j}, \mathsf{ot}_3^{i\to j}; \mathsf{r}_{i,\mathsf{ot}}^{i\to j}\right).$$
Here the superscript $i \to j$ indicates that the OT message is for the instances where P_i is the sender. Where $\overline{\mathsf{lab}}_{i|_j}$ indicates the labels corresponding to the input wire of P_j's input.

5. Reveal the randomness used in OT executions where P_i is the receiver. Specifically, $\forall j \in [n] \setminus \{i\}$ reveal $\mathsf{r}_{i,\mathsf{ot}}^{j\to i}$. This will be used by other parties to obtain the result of the OT.

6. Compute the third round of the input-delayed rewinding secure witness indistinguishable proof system for L_c. Specifically, $\forall j \in [n] \setminus \{i\}$, set
$$\mathsf{st}_c^{i\to j} := \Bigg(\left\{\mathsf{msg}_{\ell,i}, \mathsf{nmcom}_\ell^{i\to j}\right\}_{\ell\in[3]}, \left\{\mathsf{recom}_\ell^{i\to j}\right\}_{\ell\in[3], j\in[n]\setminus\{i\}},$$
$$\left\{\mathsf{rwi}_{b,\ell}^{i\to j}\right\}_{\ell\in[2], j\in[n]\setminus\{i\}}, \left\{\mathsf{st}_b^{j\to i}\right\}_{j\in[n]\setminus\{i\}}, \mathsf{Trans}_3, \left\{\mathsf{ot}_\ell^{i\to j}\right\}_{\ell\in[4], j\in[n]\setminus\{i\}}, \mathsf{td}_{1,j}, \widetilde{\mathsf{C}}_i\Bigg)$$
$$\mathsf{w}_c^{i\to j} := \left(\mathsf{x}_i, \mathsf{r}_i, \left\{\mathsf{r}_{\mathsf{recom}}^{i\to j}\right\}_{j\in[n]\setminus\{i\}}, \mathsf{r}_{\mathsf{gc},i}, \left\{\mathsf{r}_{i,\mathsf{ot}}^{j\to i}\right\}_{j\in[n]\setminus\{i\}}, \perp, \perp\right)$$
and compute
$$\mathsf{wi}_3^{i\to j} := \mathsf{WI}_3\left(\mathsf{st}_c^{i\to j}, \mathsf{w}_c^{i\to j}, \mathsf{wi}_1^{i\to j}, \mathsf{wi}_2^{i\to j}; \mathsf{r}_{\mathsf{wi}}^{i\to j}\right).$$

7. Broadcast
$$\left(\widetilde{\mathsf{C}}_i, \left\{\mathsf{wi}_3^{i\to j}, \mathsf{ot}_4^{i\to j}, \mathsf{r}_{i,\mathsf{ot}}^{j\to i}\right\}_{j\in[n]\setminus\{i\}}\right)$$
to all other parties.

Output Computation: P_i does the following:

1. If $\exists j \in [n] \setminus \{i\}$, such that
$$\mathsf{WI}_4\left(\mathsf{st}_c^{j\to i}, \mathsf{wi}_1^{j\to i}, \mathsf{wi}_2^{j\to i}, \mathsf{wi}_3^{j\to i}\right) \neq 1$$
where $\mathsf{st}_c^{j\to i}$ is computed as earlier, then output $\perp$ and **abort**.

2. Open the OT messages using the randomness broadcast by other parties. Specifically, $\forall j \in [n] \setminus \{i\}, \forall k \in [n] \setminus \{i, j\}$
$$\widetilde{\mathsf{lab}}_{j|_k} := \mathsf{OTEval}\left(\mathsf{ot}_1^{j\to k}, \mathsf{ot}_2^{j\to k}, \mathsf{ot}_3^{j\to k}, \mathsf{ot}_4^{j\to k}; \mathsf{r}_{k,\mathsf{ot}}^{j\to k}\right)$$

3. Evaluate the garbled circuit with the labels obtained above. $\forall j \in [n] \setminus \{i\}$ set
$$\widetilde{\mathsf{lab}}_j := \left(\widetilde{\mathsf{lab}}_{j|_1} || \cdots || \widetilde{\mathsf{lab}}_{j|_n}\right)$$
and evaluate
$$\widetilde{\mathsf{msg}}_{4,j} := \mathsf{Eval}\left(\mathsf{C}_j, \widetilde{\mathsf{lab}}_j\right).$$
If any of the evaluations return $\perp$, then output $\perp$ and **abort**.

4. Compute output
$$y_i := \mathsf{OUT}(\mathsf{x}_i, \mathsf{Trans}_4; \mathsf{r}_i)$$
where Trans_4 is the four round transcript derived by combining all the $\widetilde{\mathsf{msg}}_{4,j}$ obtained above with Trans_3.

4.2 Security

Consider a malicious non-uniform PPT adversary $\mathcal{A}$ who corrupts $t < n$ parties. Let p be a polynomial such that $p(\lambda)$ denotes the total length of the input and randomness of each party P_i in the underlying protocol Π, i.e., $|(\mathsf{x}_i, \mathsf{r}_i)| = p(\lambda)$.

Before providing a complete description of the simulator, we provide a high level overview of the various steps in our simulation strategy:

1. The first thing our simulator does is to determine if the adversary aborts in the first three rounds of the protocol. If so, the adversary can simulate the first three rounds using input 0. But there is a small subtlety here. By the end of the third round, none of the proofs are sent in the clear. It is possible that the adversary is implicitly aborting by sending incorrect messages, and hence the proofs will fail, but the honest parties are unaware of this.

   Since they both constitute as aborts, we want to treat them identically. But in the latter case, we're still required to send the fourth round messages of the honest parties, since as mentioned earlier, they aren't aware of an implicit abort until the fourth round.

- if the adversary aborts in a manner that is identifiable by the honest parties, i.e. by not sending the protocol message or an identifiable incorrect message (such as failed trapdoor validity), the simulator just outputs an aborted transcript.
- if the adversary aborts implicitly, then we set all the garbled circuits to output $\perp$.

This step is performed so as to ensure that if an adversary aborts with a disproportionately high probability, we don't have to bother attempting to simulate all the other components in the protocol. In this case the simulation ends here.

2. If the adversary has not aborted, we need to produce a non aborting transcript. To enable us to do so, we need to first extract relevant information. This is done by creating multiple "look-ahead" threads that share a common first round prefix with the main thread. On the look ahead threads, we're using input 0, as in the previous step, to compute the first three rounds honestly (with respect to input 0). These threads also help us estimate the probability that an adversary does not abort. With sufficiently many look-ahead threads, we can extract all the relevant information.

3. Now that we have extracted the trapdoor information and input, we need to sample the "main thread", which corresponds to the actual view of the adversary. Note we are at this point because the adversary didn't abort, and thus to avoid skewing the distribution of aborting transcripts, we must force a non-aborting transcript on to the adversary. We use the earlier estimate of the non-aborting probability of the adversary to repeatedly try to force the transcript. By a careful analysis, this step will succeed other than with negligible probability.

4. Given that we have managed to force a non-aborting transcript, corresponding to the first three rounds, on the adversary, we need to simulate the last round of the protocol. This is done by first querying the ideal functionality using the extracted inputs, and then simulating the last round of the protocol to put into garbled circuits.

5. It is still possible that the adversary has put in non-accepting proofs as the OT receiver input even though it did not implicitly abort. We want to rely on the "opaqueness" of the garbled circuits in such a situation. To do so, we must extract from the oblivious transfer to determine which circuits to set to output $\perp$. Once this is done, we output the final transcript.

While the above suffices for a high level overview of our strategy, the proof is quite delicate involving the security levels of the various primitives.

4.2.1 Description of the simulator

Simulator Sim.

Step 1 - Check Abort: In this step, Sim checks if the adversary aborts prior to the completion of the third round. This can be via either an implicit or explicit abort.

1. **Round 1:** Compute the first round message of all honest parties of the underlying protocol Π,

$$\left\{\mathsf{msg}_{1,i}\right\}_{\mathsf{P}_i\in\mathcal{H}} := \mathcal{S}_1\left(1^\lambda;\mathsf{r}_\mathcal{S}\right)$$

where $\mathcal{H}$ denotes the set of honest parties. Recall that this is done as just the honest execution of the first round on behalf of the honest players P_i using randomness $\mathsf{r}_\mathcal{S} := \{\mathsf{r}_i\}_{\mathsf{P}_i\in\mathcal{H}}$.

For each honest party P_i, Sim follows the honest party protocol in the first round. Specifically,

(a) Compute the first round of the trapdoor generation phase TDGen,

$$\mathsf{td}_{1,i} := \mathsf{TDGen}_1\left(\mathsf{r}_{\mathsf{td},i}\right)$$

using randomness $\mathsf{r}_{\mathsf{td},i}$

(b) Compute first round of the three input delayed witness indistinguishable proof systems. Specifically, $\forall j \in [n] \setminus \{i\}$, compute

- first round of the input-delayed witness indistinguishable proof system

$$\mathsf{wi}_1^{i\to j} := \mathsf{WI}_1\left(\mathsf{r}_{\mathsf{wi}}^{i\to j}\right)$$

- first round of the input-delayed rewinding secure witness indistinguishable proof system for L_a

$$\mathsf{rwi}_{a,1}^{i\to j} := \mathsf{RWI}_1\left(\mathsf{r}_{\mathsf{rwi},a}^{i\to j}\right)$$

- first round of the input-delayed rewinding secure witness indistinguishable proof system for L_b

$$\mathsf{rwi}_{b,1}^{i\to j} := \mathsf{RWI}_1\left(\mathsf{r}_{\mathsf{rwi},b}^{i\to j}\right)$$

(c) Compute first round of the three commitment schemes. Specifically, $\forall j \in [n] \setminus \{i\}$, compute

- first round of the extractable commitment scheme:

$$\mathsf{ecom}_1^{i\to j} := \mathsf{Ecom}_1\left(\mathsf{r}_{\mathsf{ecom}}^{i\to j}\right)$$

using the randomness $\mathsf{r}_{\mathsf{ecom}}^{i\to j}$.

- first round of the rewinding-secure extractable commitment scheme:

$$\mathsf{recom}_1^{i\to j} := \mathsf{RECom}_1\left(\mathsf{r}_{\mathsf{recom}}^{i\to j}\right)$$

using the randomness $\mathsf{r}_{\mathsf{recom}}^{i\to j}$.

- first round of non-malleable commitment scheme:

$$\mathsf{nmcom}_1^{i\to j} := \mathsf{NMCom}_1\left(\mathsf{r}_{\mathsf{nmcom}}^{i\to j}\right)$$

using the randomness $\mathsf{r}_{\mathsf{nmcom}}^{i\to j}$.

(d) The first round of the OT scheme, where P_i is the receiver. Specifically, $\forall j \in [n] \setminus \{i\}$, compute

$$\mathsf{ot}_1^{j\to i} := \mathsf{OT}_1\left(\mathsf{r}_{\mathsf{ot}}^{j\to i}\right)$$

using the randomness $\mathsf{r}_{\mathsf{ot}}^{j\to i}$. Here the superscript $j \to i$ indicates that the OT message is for the instances where P_i is the receiver.

For all honest parties P_i, send

$$\left(\mathsf{msg}_{1,i}, \mathsf{td}_{1,i}, \left\{\mathsf{wi}_1^{i\to j}, \mathsf{rwi}_{a,1}^{i\to j}, \mathsf{rwi}_{b,1}^{i\to j}, \mathsf{ecom}_1^{i\to j}, \mathsf{recom}_1^{i\to j}, \mathsf{nmcom}_1^{i\to j}, \mathsf{ot}_1^{j\to i}\right\}_{j\in[n]\setminus\{i\}}\right)$$

to $\mathcal{A}$.

2. **Round 2:**

For the second round too, Sim follows the honest strategy since the inputs of the honest parties are not required up until the third round of the protocol. Compute the second round message of all honest parties of the underlying protocol Π,

$$\left\{\mathsf{msg}_{2,i}\right\}_{\mathsf{P}_i \in \mathcal{H}} := \mathcal{S}_2\left(\mathsf{Trans}_1; \mathsf{r}_{\mathcal{S}}\right)$$

using the transcript obtained so far. Where $\mathcal{H}$ denotes the set of honest parties. Recall that this is done as just the honest execution of the second round on behalf of the honest players P_i using randomness $\mathsf{r}_{\mathcal{S}} := \{\mathsf{r}_i\}_{\mathsf{P}_i \in \mathcal{H}}$.

(a) Compute the second round of the trapdoor generation phase TDGen, $\forall j \in [n] \setminus \{i\}$

$$\mathsf{td}_2^{i \to j} \leftarrow \mathsf{TDGen}_2\left(\mathsf{td}_{1,j}\right)$$

using randomness $\mathsf{r}_{\mathsf{td},i}$

(b) Compute second round of the three input delayed witness indistinguishable proof systems, where P_i takes the role of the verifier. Specifically, $\forall j \in [n] \setminus \{i\}$, compute

- second round of the input-delayed witness indistinguishable proof system

$$\mathsf{wi}_2^{j \to i} := \mathsf{WI}_2\left(\mathsf{wi}_1^{j \to i}; \mathsf{r}_{\mathsf{wi}}^{j \to i}\right)$$

- second round of the input-delayed rewinding secure witness indistinguishable proof system for L_a

$$\mathsf{rwi}_{a,2}^{j \to i} := \mathsf{RWI}_2\left(\mathsf{rwi}_{a,1}^{j \to i}; \mathsf{r}_{\mathsf{rwi},a}^{j \to i}\right)$$

- second round of the input-delayed rewinding secure witness indistinguishable proof system for L_b

$$\mathsf{rwi}_{b,2}^{j \to i} := \mathsf{RWI}_2\left(\mathsf{rwi}_{b,1}^{j \to i}; \mathsf{r}_{\mathsf{rwi},b}^{j \to i}\right)$$

Note that the superscript $j \to i$ denotes that P_i is computing the second round message of the proof where P_j is the prover and P_i is the verifier.

(c) Compute second round of the three commitment schemes. Specifically, $\forall j \in [n] \setminus \{i\}$, compute

- second round of the extractable commitment scheme:

$$\mathsf{ecom}_2^{j \to i} \leftarrow \mathsf{Ecom}_2\left(\mathsf{ecom}_1^{j \to i}\right)$$

- second round of the rewinding-secure extractable commitment scheme:

$$\mathsf{recom}_2^{j \to i} \leftarrow \mathsf{RECom}_2\left(\mathsf{recom}_1^{j \to i}\right)$$

- second round of non-malleable commitment scheme:

$$\mathsf{nmcom}_2^{j \to i} \leftarrow \mathsf{NMCom}_2\left(\mathsf{nmcom}_1^{j \to i}\right)$$

As in the case of the proofs, the superscript $j \to i$ denotes that P_i is the receiver in the commitment from P_i.

(d) The second round of the OT scheme, where P_i is the sender. Specifically, $\forall j \in [n] \setminus \{i\}$, compute

$$\mathsf{ot}_2^{i \to j} := \mathsf{OT}_2\left(\mathsf{ot}_1^{i \to j}; \mathsf{r}_{i,\mathsf{ot}}^{i \to j}\right).$$

Here the superscript $i \to j$ indicates that the OT message is for the instances where P_i is the sender.

For every honest party P_i, send

$$\left(\mathsf{msg}_{2,i}, \left\{\mathsf{td}_2^{i \to j}, \mathsf{wi}_2^{j \to i}, \mathsf{rwi}_{a,2}^{j \to i}, \mathsf{rwi}_{b,1}^{j \to i}, \mathsf{ecom}_2^{j \to i}, \mathsf{recom}_2^{j \to i}, \mathsf{nmcom}_2^{j \to i}, \mathsf{ot}_2^{i \to j}\right\}_{j \in [n] \setminus \{i\}}\right)$$

to $\mathcal{A}$.

3. **Round 3:**

For round 3, Sim generates the third round messages honestly using input 0.

(a) Compute the third round message of the underlying protocol Π,

$$\mathsf{msg}_{3,i} := \Pi_3\left(0, \mathsf{Trans}_2; \mathsf{r}_i\right)$$

using input 0, randomness r_i and the transcript obtained so far. Recall that we are able to do this since the "simulation" of the first two rounds of the underlying protocol were done honestly.

(b) Compute the second round of the trapdoor generation phase TDGen. Let $\mathsf{td}_{2,i} := \left(\mathsf{td}_2^{1 \to i} || \cdots || \mathsf{td}_2^{1 \to i}\right)$, compute

$$\mathsf{td}_{3,i} \leftarrow \mathsf{TDGen}_3\left(\mathsf{td}_{1,i}, \mathsf{td}_{2,i}; \mathsf{r}_{\mathsf{td},i}\right).$$

(c) Compute the third round of the non-malleable commitment scheme to commit to $\perp$. Specifically, $\forall j \in [n] \setminus \{i\}$, compute

$$\mathsf{nmcom}_3^{i \to j} \leftarrow \mathsf{NMCom}_3\left(0, \mathsf{nmcom}_1^{i \to j}, \mathsf{nmcom}_3^{i \to j}; \mathsf{r}_{\mathsf{nmcom}}^{i \to j}\right)$$

using the randomness $\mathsf{r}_{\mathsf{nmcom}}^{i \to j}$.

(d) Compute the third round of the rewinding-secure extractable commitment scheme to commit to $(0, \mathsf{r}_i)$. Specifically, $\forall j \in [n] \setminus \{i\}$, compute

$$\mathsf{recom}_3^{i \to j} \leftarrow \mathsf{RECom}_3\left((0, \mathsf{r}_i), \mathsf{recom}_1^{i \to j}, \mathsf{recom}_2^{i \to j}; \mathsf{r}_{\mathsf{recom}}^{i \to j}\right)$$

using the randomness $\mathsf{r}_{\mathsf{recom}}^{i \to j}$.

(e) Compute the third round of the input-delayed rewinding secure witness indistinguishable proof system for L_a. Specifically, $\forall j \in [n] \setminus \{i\}$, set

$$\mathsf{st}_a^{i \to j} := \left(\left\{\mathsf{recom}_\ell^{i \to k}, \mathsf{nmcom}_\ell^{i \to k}\right\}_{\ell \in [3], k \in [n] \setminus \{i\}}, \left\{\mathsf{msg}_{\ell,i}\ \right\}_{\ell \in [3]} \mathsf{Trans}_2, \mathsf{td}_{1,j}\right)$$
$$\mathsf{w}_a^{i \to j} := \left(0, \mathsf{r}_i, \left\{\mathsf{r}_{\mathsf{recom}}^{i \to k}\right\}_{k \in [n] \setminus \{i\}}, \perp, \perp\right)$$

and compute

$$\mathsf{rwi}_{a,3}^{i \to j} := \mathsf{RWI}_3\left(\mathsf{st}_a^{i \to j}, \mathsf{w}_a^{i \to j}, \mathsf{rwi}_{a,1}^{i \to j}, \mathsf{rwi}_{a,2}^{i \to j}; \mathsf{r}_{\mathsf{rwi},a}^{i \to j}\right).$$

(f) Compute the third round of the extractable commitment scheme to commit to the third round proof $\mathsf{rwi}_{a,3}^{i\to j}$. Specifically, $\forall j \in [n] \setminus \{i\}$, compute

$$\mathsf{ecom}_3^{i\to j} := \mathsf{Ecom}_3\left(\mathsf{rwi}_{a,3}^{i\to j}, \mathsf{ecom}_1^{i\to j}, \mathsf{ecom}_3^{i\to j}; \mathsf{r}_{\mathsf{ecom}}^{i\to j}\right)$$

using the randomness $\mathsf{r}_{\mathsf{ecom}}^{i\to j}$.

(g) Compute the third round of the input-delayed rewinding secure witness indistinguishable proof system for L_b. Specifically, $\forall j \in [n] \setminus \{i\}$, set

$$\mathsf{st}_b^{i\to j} := \left(\left\{\mathsf{recom}_\ell^{i\to k}, \mathsf{ecom}_\ell^{i\to k}, \mathsf{nmcom}_\ell^{i\to k}\right\}_{\ell\in[3],k\in[n]\setminus\{i\}}, \right.$$
$$\left. \left\{\mathsf{msg}_{\ell,i}\right\}_{\ell\in[3]}, \left\{\mathsf{rwi}_{a,\ell}^{i\to j}\right\}_{\ell\in[2]}, \mathsf{Trans}_2, \mathsf{td}_{1,j} \right)$$
$$\mathsf{w}_b^{i\to j} := \left(\left\{\mathsf{r}_{\mathsf{ecom}}^{i\to k}, \mathsf{r}_{\mathsf{rwi},a}^{i\to j}, \mathsf{w}_a^{i\to k}\right\}_{k\in[n]\setminus\{i\}}, \perp, \perp \right)$$

and compute

$$\mathsf{rwi}_{b,3}^{i\to j} := \mathsf{RWI}_3\left(\mathsf{st}_b^{i\to j}, \mathsf{w}_b^{i\to j}, \mathsf{rwi}_{b,1}^{i\to j}, \mathsf{rwi}_{b,2}^{i\to j}; \mathsf{r}_{\mathsf{rwi},b}^{j\to i}\right).$$

(h) Compute the third round of the OT scheme, where P_i is the receiver. Specifically, $\forall j \in [n] \setminus \{i\}$, compute

$$\mathsf{ot}_3^{j\to i} := \mathsf{OT}_3\left(\mathsf{rwi}_{b,3}^{i\to j}, \mathsf{ot}_1^{j\to i}, \mathsf{ot}_2^{j\to i}; \mathsf{r}_{i,\mathsf{ot}}^{j\to i}\right).$$

Here the superscript $j \to i$ indicates that the OT message is for the instances where P_i is the receiver.

For every honest party P_i, send

$$\left(\mathsf{msg}_{3,i}, \mathsf{td}_{3,i}, \left\{\mathsf{ecom}_3^{i\to j}, \mathsf{recom}_3^{i\to j}, \mathsf{nmcom}_3^{i\to j}, \mathsf{ot}_3^{j\to i}\right\}_{j\in[n]\setminus\{i\}}\right)$$

to $\mathcal{A}$.

4. **Check Abort Condition:**

   Sim now checks whether $\mathcal{A}$ aborted in the third round. This happens if $\mathcal{A}$ doesn't send its third round messages, or if every honest party aborts if the trapdoor condition does not verify. Let $\mathcal{H}$ denote the set of honest parties. Then check $\forall \mathsf{P}_i \in \mathcal{H}, \exists \mathsf{P}_j \in \mathcal{A}$,

   $$\text{if } \mathsf{TDOut}\left(\mathsf{td}_{1,j}, \mathsf{td}_{2,j}, \mathsf{td}_{3,j}\right) \neq 1$$

   then Sim outputs the partial view generated so far and stops. Otherwise, we say that "Check Abort" succeeded and we proceed.

5. **Check Implicit Abort:** We run a look ahead threads to extract the RWI proofs for L_a from each malicious party P_j. These are extracted from Ecom. We check if all the extracted RWI proofs verify. This ensures that on the given thread, the malicious parties exhibit honest behavior. If for even a single malicious party P_j the proofs don't verify, then we take evasive action as mentioned in Step 1.5. We denote the "Check Abort" thread as GOOD if the adversary doesn't abort explicitly or implicitly.

Remark 4. *We use a specific property of* $\mathsf{Ext}_{\mathsf{ecom}}$*, namely that since it's input delayed, the commitment in the first round is to a mask* mask *and the input delayed property is achieved by masking the input with* mask*. In fact,* mask *is statistically determined by the first round of* Ecom*. Thus, to extract from multiple instances of the input-delayed extractable commitment with a single shared first message that potentially commit to different inputs, it suffices to extract* mask *in a single instance and using* mask *to unmask, and thus retrieve, other inputs. Since the* mask *is extracted via decommittment information, it's easy to verify that the extracted value* mask *is indeed correct.*

Step 1.5 - Evasive Action for Implicit Abort: We run this step only if there is an implicit abort. Since we cannot do an explicit abort on behalf of the honest parties, we want to continue the main thread from Step 1 but garble the $\mathsf{C}_\perp$ circuit in the fourth round since we are sure that adversary will not be able to evaluate the garbled circuit to produce any other output. But in order to do this, we will need to extract the trapdoor to prove the WI statement for L_c claiming that the garbled circuit was computed honestly. We can do this because the adversary did not cause an explicit abort, and the extracted trapdoor can be publicly checked. Specifically,

1. Create look-ahead threads running rounds 2 and 3 as before, and extract trapdoor t_j for ever adversarial player P_j by running the TDExt on the look-ahead threads. Since the adversary did not cause an explicit abort, this can be done.
2. Compute the garbled circuit as $\left(\mathsf{C}_i, \overline{\mathsf{lab}}_i\right) := \mathsf{Garble}\left(\mathsf{C}_\perp; \mathsf{r}_{\mathsf{gc},i}\right)$ where $\mathsf{C}_\perp$ is the circuit that always outputs $\perp$, but has the same topology as C_i.
3. The OT messages are computed as before.
4. This completes the simulation of the protocol.

Step 2 - Rewinding: Since the adversary has not aborted, implicitly or explicitly, we will need to start simulation the underlying protocol to produce an appropriate transcript.

1. Sim now rewinds $\mathcal{A}$ to the end of round 1 and freezes the main thread at this point. Then, Sim creates a set of T (to be determined later) look-ahead threads, where on each thread, only rounds 2 and 3 of the protocol are executed in the following manner:

   (a) **Round 2:**
   In every look-ahead thread, for each honest party P_i and for each $j \in [n] \setminus \{i\}$, Sim executes the same strategy as in round 2 of step 1, using fresh randomness each time(for each primitive).

   (b) **Round 3:**
   In every look-ahead thread, for each honest party P_i and for each $j \in [n] \setminus \{i\}$, Sim executes the same strategy as in round 3 of step 1, using fresh randomness each time.

2. For each look-ahead thread, define a thread to be GOOD with respect to P_{i^*} if for all malicious parties P_j:
   - P_j does send its third round messages.
   - if $\mathsf{TDOut}\left(\mathsf{td}_{1,j}, \mathsf{td}_{2,j}, \mathsf{td}_{3,j}\right) = 1$ where $\mathsf{td}_{2,j}$ is as computed in round 3.

– The extracted RWI proofs for L_a where P_j is the prover are all accepting. *We use* mask *obtained in Step 1 to do the extractions by simply unmasking the commitment in* Ecom*.*

3. The number of threads T created is such that at least $(12 \cdot \lambda)$ GOOD threads exists. That is, Sim keeps running till it obtains $(12 \cdot \lambda)$ GOOD threads.

Step 3 - Input and Trapdoor Extraction: Now, Sim extracts all relevant information. Note that all the relevant information can be extracted from sufficient number of GOOD threads with respect to a single honest party.

Sim does the following:

1. Select 5 threads that are GOOD with respect to some honest party P_{i^*}. In each GOOD thread, we know $\exists$ honest party P_i such that for all malicious parties P_j, the adversary does not cause P_i to abort. Since $(12 \cdot \lambda) > (5 \cdot n)$[7], there must exist one honest party P_{i^*} corresponding to a set of 5 GOOD threads.

2. **Trapdoor Extraction:** For every corrupted party P_j, extract a trapdoor t_j by running the trapdoor extractor TDExt on input the transcript of the trapdoor generation protocol with P_j playing the role of the trapdoor generator from any 3 GOOD threads. Specifically, compute

$$\mathsf{t}_j \leftarrow \mathsf{TDExt}\left(\mathsf{td}_1, \left\{\mathsf{td}_2^k, \mathsf{td}_3^k\right\}_{k\in[3]}\right)$$

where $\left(\mathsf{td}_1, \mathsf{td}_2^k, \mathsf{td}_3^k\right)$ denotes the transcript of the trapdoor generation protocol with P_j as the sender of the k-th GOOD thread.

3. **Input Extraction:** For every corrupted party P_j, extract the mask for the input and randomness pair $(\mathsf{x}_j, \mathsf{r}_j)$ by running the extractor $\mathsf{Ext}_{\mathsf{RECom}}$ on input the transcript of the extractable commitment protocol between P_j and P_{i^*} from the 5 GOOD threads picked above. That is, compute

$$\mathsf{mask}^{j\to i^*} \leftarrow \mathsf{Ext}_{\mathsf{RECom}}\left(\mathsf{recom}_1^{j\to i^*}, \left\{\mathsf{recom}_{2,k}^{j\to i^*}, \mathsf{recom}_{3,k}^{j\to i^*}\right\}_{k\in[5]}\right)$$

where $\mathsf{recom}_1^{j\to i^*}, \mathsf{recom}_{2,k}^{j\to i^*}, \mathsf{recom}_{3,k}^{j\to i^*}$ denotes the transcript of the extractable commitment protocol between P_j and P_{i^*} on the k-th GOOD thread.

4. **Proof Extraction:** Since we've already extracted the proofs in Step 1, by Remark 4 we can extract the proofs in each thread without having to rewind, by just unmasking with the extracted mask from Step 1.

5. Output $\perp_{\mathsf{extract}}$ if any of steps 2 or 3 fail.

Step 4 - Abort Probability Estimation:

Set $\varepsilon' = \frac{12\cdot\lambda}{T}$ as the probability that the adversary doesn't abort.

Step 5 - Re-sampling the Main Thread: Using the information extracted, Sim samples the main thread. It also needs to force this transcript, and uses the estimate obtained earlier to upper bound the number of attempts to do try this.

Sim sets a counter to value 0. Now Sim attempts to force the following transcript in the main thread until it accepts, or the counter reaches the cut-off point.

[7] without loss of generality, assume the number of parties $n = \lambda$

1. **Round 2** :

   Run exactly as done in Step 1.

2. **Round 3** :

   There are some key differences from the threads generated in the previous steps:

   – The non-malleable commitment from an honest party P_i to a malicious party P_j now contains the extracted trapdoor t_j.
   – The witness indistinguishable proofs use the "trapdoor witness".
   – The third round of the MPC is generated by the simulator for the underlying protocol.

   In more detail.

   (a) Compute the third round message of all honest parties of the underlying protocol Π,

   $$\left\{\mathsf{msg}_{3,i}\right\}_{\mathsf{P}_i \in \mathcal{H}} := \mathcal{S}_3\left(\mathsf{Trans}_2; \mathsf{r}_{\mathcal{S}}\right)$$

   using the transcript obtained so far. Where $\mathcal{H}$ denotes the set of honest parties.

   (b) Compute the third round of the trapdoor generation phase TDGen. Let $\mathsf{td}_{2,i} := \left(\mathsf{td}_2^{1\to i}||\cdots||\mathsf{td}_2^{1\to i}\right)$, compute

   $$\mathsf{td}_{3,i} := \mathsf{TDGen}_3\left(\mathsf{td}_{1,i}, \mathsf{td}_{2,i}; \mathsf{r}_{\mathsf{td},i}\right).$$

   (c) Compute the third round of the non-malleable commitment scheme to commit to the extracted trapdoor. Specifically, $\forall j \in [n] \setminus \{i\}$, compute

   $$\mathsf{nmcom}_3^{i\to j} := \mathsf{NMCom}_3\left(\mathsf{t}_j, \mathsf{nmcom}_1^{i\to j}, \mathsf{nmcom}_3^{i\to j}; \mathsf{r}_{\mathsf{nmcom}}^{i\to j}\right)$$

   using the randomness $\mathsf{r}_{\mathsf{nmcom}}^{i\to j}$.

   (d) Compute the third round of the rewinding-secure extractable commitment scheme to commit to 0. Specifically, $\forall j \in [n] \setminus \{i\}$, compute

   $$\mathsf{recom}_3^{i\to j} := \mathsf{RECom}_3\left(0, \mathsf{recom}_1^{i\to j}, \mathsf{recom}_2^{i\to j}; \mathsf{r}_{\mathsf{recom}}^{i\to j}\right)$$

   using the randomness $\mathsf{r}_{\mathsf{recom}}^{i\to j}$.

   (e) Compute the third round of the input-delayed rewinding secure witness indistinguishable proof system for L_a. Specifically, $\forall j \in [n] \setminus \{i\}$, set

   $$\mathsf{st}_a^{i\to j} := \left(\left\{\mathsf{recom}_\ell^{i\to j}, \mathsf{nmcom}_\ell^{i\to j}\right\}_{\ell\in[3], j\in[n]\setminus\{i\}}, \left\{\mathsf{msg}_{\ell,i}\right\}_{\ell in[3]} \mathsf{Trans}_2, \mathsf{td}_{1,j}\right)$$
   $$\mathsf{w}_a^{i\to j} := \left(\perp, \perp, \perp, \mathsf{t}_j, \mathsf{r}_{\mathsf{nmcom}}^{i\to j}\right)$$

   and compute

   $$\mathsf{rwi}_{a,3}^{i\to j} := \mathsf{RWI}_3\left(\mathsf{st}_a^{i\to j}, \mathsf{w}_a^{i\to j}, \mathsf{rwi}_{a,1}^{i\to j}, \mathsf{rwi}_{a,2}^{i\to j}; \mathsf{r}_{\mathsf{rwi},a}^{i\to j}\right).$$

   (f) Compute the third round of the extractable commitment scheme to commit to the third round proof $\mathsf{rwi}_{a,3}^{i\to j}$. Specifically, $\forall j \in [n] \setminus \{i\}$, compute

   $$\mathsf{ecom}_3^{i\to j} := \mathsf{Ecom}_3\left(\mathsf{rwi}_{a,3}^{i\to j}, \mathsf{ecom}_1^{i\to j}, \mathsf{ecom}_3^{i\to j}; \mathsf{r}_{\mathsf{ecom}}^{i\to j}\right)$$

   using the randomness $\mathsf{r}_{\mathsf{ecom}}^{i\to j}$.

(g) Compute the third round of the input-delayed rewinding secure witness indistinguishable proof system for L_b. Specifically, $\forall j \in [n] \setminus \{i\}$, set

$$\begin{aligned}\mathsf{st}_b^{i\to j} &:= \Big(\Big\{\mathsf{recom}_\ell^{i\to j}, \mathsf{ecom}_\ell^{i\to j}, \mathsf{nmcom}_\ell^{i\to j}\Big\}_{\ell\in[3], j\in[n]\setminus\{i\}}, \\ &\qquad \Big\{\mathsf{nmcom}_\ell^{i\to j}, \mathsf{msg}_{\ell,i}\Big\}_{\ell\in[3]} \Big\{\mathsf{rwi}_{b,\ell}^{i\to j}\Big\}_{\ell\in[2]} \mathsf{Trans}_2, \mathsf{td}_{1,j}\Big) \\ \mathsf{w}_b^{i\to j} &:= \left(\perp, \mathsf{t}_j, \mathsf{r}_{\mathsf{nmcom}}^{i\to j}\right)\end{aligned}$$

$$\begin{aligned}\mathsf{st}_b^{i\to j} &:= \Big(\Big\{\mathsf{recom}_\ell^{i\to k}, \mathsf{ecom}_\ell^{i\to k}, \mathsf{nmcom}_\ell^{i\to k}\Big\}_{\ell\in[3], k\in[n]\setminus\{i\}}, \\ &\qquad \Big\{\mathsf{msg}_{\ell,i}\Big\}_{\ell\in[3]}, \Big\{\mathsf{rwi}_{a,\ell}^{i\to j}\Big\}_{\ell\in[2]}, \mathsf{Trans}_2, \mathsf{td}_{1,j}\Big) \\ \mathsf{w}_b^{i\to j} &:= \left(\perp, \mathsf{t}_j, \mathsf{r}_{\mathsf{nmcom}}^{i\to j}\right)\end{aligned}$$

and compute

$$\mathsf{rwi}_{b,3}^{i\to j} := \mathsf{RWI}_3\left(\mathsf{st}_b^{i\to j}, \mathsf{w}_b^{i\to j}, \mathsf{rwi}_{b,1}^{i\to j}, \mathsf{rwi}_{b,2}^{i\to j}; \mathsf{r}_{\mathsf{rwi},b}^{i\to j}\right).$$

(h) Compute the third round of the OT scheme, where P_i is the receiver. Specifically, $\forall j \in [n] \setminus \{i\}$, compute

$$\mathsf{ot}_3^{j\to i} := \mathsf{OT}_3\left(\mathsf{rwi}_{b,3}^{i\to j}, \mathsf{ot}_1^{j\to i}, \mathsf{ot}_2^{j\to i}; \mathsf{r}_{i,\mathsf{ot}}^{j\to i}\right).$$

Here the superscript $j \to i$ indicates that the OT message is for the instances where P_i is the receiver.

For every honest party P_i, send

$$\left(\mathsf{msg}_{3,i}, \mathsf{td}_{3,i}, \left\{\mathsf{ecom}_3^{i\to j}, \mathsf{recom}_3^{i\to j}, \mathsf{nmcom}_3^{i\to j}, \mathsf{ot}_3^{j\to i}\right\}_{j\in[n]\setminus\{i\}}\right)$$

to $\mathcal{A}$.

3. **Abort Condition:**

(a) If the adversary doesn't send its third round message or $\forall \mathsf{P}_i \in \mathcal{H}, \exists \mathsf{P}_j \in \mathcal{A}$,

$$\text{if } \mathsf{TDOut}\left(\mathsf{td}_{1,j}, \mathsf{td}_{2,j}, \mathsf{td}_{3,j}\right) = 1$$

or the extracted proofs for L_a from P_j do not accept, increment counter by 1.

(b) If Sim's running time is 2^λ. Abort.

(c) If the counter value was not increased, then the adversary did not abort in the third round. We can proceed to Step 7.

(d) Else, if the counter value is less that $\frac{\lambda^2}{\varepsilon'}$ rewind back to the beginning of round 2 in Step 6 and re-sample the main thread. Otherwise, Abort.

Step 6 - Query the Ideal Functionality:

1. Sim queries the ideal functionality with the set of values $\{x_j\}$ where x_j is the input of adversarial party P_j that was extracted in the previous step using mask obtained through extraction by rewinding. *This is done in this manner since the adversary may use a different input in each thread, and we want to use the input it uses on the main thread.* Since the adversary commits to its input only on completion of the third round on the main thread.

2. Sim receives output y from the ideal functionality.

Step 7 - Extract proofs from OT: In order to determine whether we need to put in simulated messages into garbled circuits in the fourth round, we extract from all OT receiver messages in parallel by running sufficiently many look-ahead threads. Note that this is different from an implicit abort since if there is no implicit abort, it is guaranteed that the adversary behaved honestly in the underlying protocol. It is still possible that it doesn't put the correct proof inside of the OT receiver messages. We just need to ensure that the relevant garbled circuits then become "opaque".

Step 8 - Finishing the Main Thread: Sim now finishes off the main thread by computing the last round of the protocol.

1. **Round 4:**

   – Compute the simulated fourth round message of Π

   $$\mathsf{msg}_{4,i} \leftarrow \mathcal{S}_4\left(y, \{\mathsf{x}_j, \mathsf{r}_j\}_{\mathsf{P}_j \notin \mathcal{H}}, \mathsf{Trans}_3;\ \mathsf{r}_{\mathcal{S}}\right)_i$$

   where i on the right hand side indexes the i-th component of the output. Note that $\mathcal{S}_4$ will not be called if there is an implicit or explicit abort.

   – Compute the garbled circuit using the simulated fourth round message, but taking into account the extract RWI proof for L_b. For each honest P_i,

     – if $\exists \mathsf{P}_j \in \mathcal{A}$ such that RWI proof did not accept. i.e.

     $$\mathsf{RWI}_4\left(\mathsf{rwi}_1^{j\to i}, \mathsf{rwi}_2^{j\to i}, \mathsf{rwi}_3^{j\to i}, \mathsf{st}^{j\to i}; \mathsf{r}_{\mathsf{rwi}}^{j\to i}\right) \neq 1$$

     , then

     $$\left(\mathsf{C}_i, \overline{\mathsf{lab}}_i\right) := \mathsf{Garble}\left(\mathsf{C}_\perp; \mathsf{r}_{\mathsf{gc},i}\right)$$

     – else

     $$\left(\mathsf{C}_i, \overline{\mathsf{lab}}_i\right) := \mathsf{Garble}\left(\mathsf{C}\left[i, \mathsf{msg}_{4,i}, \left\{\mathsf{rwi}_{b,\ell}^{j\to i}\right\}_{\ell\in[2], j\in[n]\setminus\{i\}} \left\{\mathsf{st}_b^{j\to i}, \mathsf{r}_{\mathsf{rwi},b}^{j\to i}\right\}_{j\in[n]\setminus\{i\}}\right]; \mathsf{r}_{\mathsf{gc},i}\right)$$

   – Compute the third round of the input-delayed rewinding secure witness indistinguishable proof system for L_b. Specifically, $\forall j \in [n] \setminus \{i\}$, set

   $$\mathsf{st}_c^{i\to j} := \Bigg(\left\{\mathsf{msg}_{\ell,i}, \mathsf{nmcom}_\ell^{i\to j}\right\}_{\ell\in[3]}, \left\{\mathsf{recom}_\ell^{i\to j}\right\}_{\ell\in[3], j\in[n]\setminus\{i\}}, \left\{\mathsf{rwi}_{b,\ell}^{i\to j}\right\}_{\ell\in[2], j\in[n]\setminus\{i\}}, \left\{\mathsf{st}_b^{j\to i}\right\}_{j\in[n]\setminus\{i\}}, \mathsf{Trans}_3, \left\{\mathsf{ot}_\ell^{i\to j}\right\}_{\ell\in[4], j\in[n]\setminus\{i\}}, \mathsf{td}_{1,j}, \widetilde{\mathsf{C}}_i\Bigg)$$

   $$\mathsf{w}_b^{i\to j} := \left(\perp, \mathsf{t}_j, \mathsf{r}_{\mathsf{nmcom}}^{i\to j}\right)$$

   and compute

   $$\mathsf{wi}_3^{i\to j} \leftarrow \mathsf{WI}_3\left(\mathsf{st}_c^{i\to j}, \mathsf{w}_c^{i\to j}, \mathsf{wi}_1^{i\to j}, \mathsf{wi}_2^{i\to j}\right).$$

For each honest party P_i, send

$$\left(\mathsf{C}_i, \left\{\mathsf{wi}_3^{i\to j}, \mathsf{ot}_4^{i\to j}, \mathsf{r}_{i,\mathsf{ot}}^{j\to i}\right\}_{j\in[n]\setminus\{i\}}\right)$$

to $\mathcal{A}$.

2. **Output Computation:**

   In the main thread, for each honest party P_i, Sim does the following:

   – If $\exists j \in [n] \setminus \{i\}$, such that

   $$\mathsf{WI}_4\left(\mathsf{st}_c^{j\to i}, \mathsf{wi}_1^{j\to i}, \mathsf{wi}_2^{j\to i}, \mathsf{wi}_3^{j\to i}\right) \neq 1$$

   where $\mathsf{st}_c^{j\to i}$ is computed as earlier then abort. Also abort if any adversarial party did not send the fourth round message.

   If there is no abort, instruct the ideal functionality to deliver output to the honest parties.

Remark 5. *We note that if any round, a subprotocol outputs $\perp$, P_i broadcast $\perp$, sets output to be $\perp$ and aborts. If P_i receives a $\perp$ from another party, it sets its output to be $\perp$ and aborts.*

Running Time of the Simulator: The simulator runs in expected time polynomial in λ. The analysis follows identically from that of [BGJ+18]. The only steps that the simulator can run in exponential time are:

1. Step 2, where Sim rewinds till it gets $12 \cdot \lambda$ non-aborting transcripts. If ε denotes the probability with which Sim goes into Step 2 (i.e. did not abort in Step 1), then the expected total number of threads created are $\frac{12\cdot\lambda}{\varepsilon}$, where each thread takes only $\mathsf{poly}(\lambda)$ time.

2. Step 5, where Sim resamples the main thread. If the probability estimate is correct, then it is easy to see that this step requires the creation of at most $\frac{\lambda^2}{\varepsilon}$ (see [BGJ+18] for details) threads. This step might take time 2^λ, but that only happens with probability $\frac{1}{2^\lambda}$.

This gives a total expected running time of

$$\mathsf{poly}(\lambda) + \mathsf{poly}(\lambda) \cdot \varepsilon \left(\frac{12\cdot\lambda}{\varepsilon} + \left(1 - \frac{1}{2^\lambda}\right)\frac{\lambda^2}{\varepsilon} + 2^\lambda\left(\frac{1}{2^\lambda}\right)\right) \leq \mathsf{poly}(\lambda)$$

4.2.2 Hybrids

Assume by contradiction that there is an adversary $\mathcal{A}$ that distinguishes the real and ideal worlds with some non-negligible probability μ. μ will be used to set certain parameters in the hybrids.

$\mathsf{Hyb}_{\mathsf{REAL}}$: **Real World:** The hybrid is the same as the real world execution. We consider a simulator $\mathsf{Sim}_{\mathsf{Hyb}}$ that plays the role of the honest parties.

Hyb_0: **Determining Abort in the 3rd Round an Extraction:** In this hybrid, $\mathsf{Sim}_{\mathsf{Hyb}}$ makes the following changes:

1. $\mathsf{Sim}_{\mathsf{Hyb}}$ executes the first 3 rounds of the protocol using the honest parties' strategy. If the adversary causes an abort, $\mathsf{Sim}_{\mathsf{Hyb}}$ outputs only the view of the adversary and stops. The rest of the hybrid is skipped in this case.
2. If the "Check Abort" step succeeds, $\mathsf{Sim}_{\mathsf{Hyb}}$ checks if there is an implicit abort by extracting the RWI proofs. If there is an implicit abort, $\mathsf{Sim}_{\mathsf{Hyb}}$ extracts only the trapdoor in the subsequent step and all hybrids up until the change to the garbled circuit are skipped.
3. If there is no implicit abort, $\mathsf{Sim}_{\mathsf{Hyb}}$ rewinds back to after the completion of round 1 of the protocol and freezes the main thread. $\mathsf{Sim}_{\mathsf{Hyb}}$ creates a set of $\frac{5 \cdot n \cdot \lambda}{\mu}$ look ahead threads as described in Step 2 of Sim. Which is to say that in all the threads, $\mathsf{Sim}_{\mathsf{Hyb}}$ uses the honest parties' inputs and follows the protocol. The look ahead threads are identical to the main thread.
4. $\mathsf{Sim}_{\mathsf{Hyb}}$ now extracts the input, trapdoors and proofs from the created look-ahead threads. Specifically, it runs the "Input and Trapdoor Extraction" phase described in step 3 of the description of Sim using the first 5 look-ahead threads that are GOOD with respect to some honest party P_{i^*}.
5. $\mathsf{Sim}_{\mathsf{Hyb}}$ outputs $\perp_{\mathsf{extract}}$ if the above step fails.
6. $\mathsf{Sim}_{\mathsf{Hyb}}$ continues the execution of the main thread it had previously frozen. It does this as in the honest execution of $\mathsf{Hyb}_{\mathsf{REAL}}$. If the adversary causes an abort, $\mathsf{Sim}_{\mathsf{Hyb}}$ rewinds to the end of round 1 and re-samples the main thread honestly. This process is repeated at most $\frac{\lambda}{\mu}$ times.

Since μ is noticeable, we are guaranteed that $\mathsf{Sim}_{\mathsf{Hyb}}$ will run in polynomial in this hybrid, and subsequent hybrids, when performing this check.

Hyb_1: **Using input 0 in the Aborting Step:** In this hybrid, $\mathsf{Sim}_{\mathsf{Hyb}}$ does the "Check Abort" step using the input 0 instead of the real honest party inputs. If the adversary does cause an abort, then $\mathsf{Sim}_{\mathsf{Hyb}}$ just outputs the view of the adversary and stops. Else, it proceeds as in Hyb_0. This is done using a sequence of sub-hybrids. We only describe changes made in each sub-hybrid, with the remaining execution identical to the previous hybrid.

$\mathsf{Hyb}_{1,0}$: **Change OT receiver input to 0:** In this sub-hybrid, $\mathsf{Sim}_{\mathsf{Hyb}}$ only modifies the third round to replace the OT receiver input for all honest parties with 0. In Hyb_0, the receiver input to the OT was the third message of the RWI proof for L_b.

$\mathsf{Hyb}_{1,1}$: **Change Ecom input to 0:** In this sub-hybrid, $\mathsf{Sim}_{\mathsf{Hyb}}$ only modifies the third round to replace the Ecom input for all honest parties with 0. In $\mathsf{Hyb}_{1,0}$, the input to Ecom was the third message of the RWI proof for L_a.

$\mathsf{Hyb}_{1,2}$: **Change REComm input to 0:** In this sub-hybrid, $\mathsf{Sim}_{\mathsf{Hyb}}$ only modifies the third round to replace the REComm input for all honest parties with $(0, \mathsf{r}_i)$. In $\mathsf{Hyb}_{1,1}$, the input to Ecom for an honest party P_i was its input and randomness $(\mathsf{x}_i, \mathsf{r}_i)$ for the underlying protocol Π.

$\mathsf{Hyb}_{1,3}$: **Change Π input to 0:** In this sub-hybrid, $\mathsf{Sim}_{\mathsf{Hyb}}$ only modifies the third round to replace the Π input for all honest parties with 0. In $\mathsf{Hyb}_{1,2}$, the input to Π for an honest party P_i in the third round was x_i.

$\mathsf{Hyb}_{1,4}$: **Change Ecom input to RWI:** In this sub-hybrid, $\mathsf{Sim}_{\mathsf{Hyb}}$ only modifies the third round to replace the Ecom input for all honest parties with the correctly computed third message of the RWI proof for L_a using input 0. In $\mathsf{Hyb}_{1,3}$, the input to Ecom was 0.

$\mathsf{Hyb}_{1,5}$: **Change OT receiver input to RWI:** In this sub-hybrid, $\mathsf{Sim}_{\mathsf{Hyb}}$ only modifies the third round to replace the OT receiver input for all honest parties with the correctly computed third message of the RWI proof for L_b using. In Hyb_0, the receiver input to the OT was 0.

Note that $\mathsf{Hyb}_{1,5} \equiv \mathsf{Hyb}_1$

Hyb_2: **Using input 0 in the look-ahead threads:** In this hybrid, $\mathsf{Sim}_{\mathsf{Hyb}}$ modifies each look-ahead thread to follow the protocol but replacing the honest player inputs with 0. This is done in a sequence of hybrids, where in each sequence we only modify a single look ahead thread. Since the number of threads are T, we do the following:

$\forall k \in [T]$ the following changes are made *only to the k-th thread*:

$\mathsf{Hyb}_{2,k,0}$: **Change NMCom on k-th thread:** In this sub-hybrid, $\mathsf{Sim}_{\mathsf{Hyb}}$ only modifies the third round of the k-th thread to commit in the NMCom to the trapdoor. In Hyb_1, NMCom was a commitment to 0. Specifically, for every honest party P_i and every malicious party P_j, $\mathsf{Sim}_{\mathsf{Hyb}}$ modifies the third round NMCom message to be

$$\mathsf{nmcom}_3^{i\to j} := \mathsf{NMCom}_3\left(\mathsf{t}_j, \mathsf{nmcom}_1^{i\to j}, \mathsf{nmcom}_3^{i\to j}; \mathsf{r}_{\mathsf{nmcom}}^{i\to j}\right)$$

where t_j is a valid trapdoor extracted from the *other look-ahead threads* as in Hyb_1.

$\mathsf{Hyb}_{2,k,1}$: **Switch RWI proofs for L_b on the k-th thread:** In this sub-hybrid, $\mathsf{Sim}_{\mathsf{Hyb}}$ only modifies the third round of the k-th thread to switch to the "trapdoor witness" in the RWI proofs for L_b. Specifically, for every honest party P_i and every malicious party P_j, $\mathsf{Sim}_{\mathsf{Hyb}}$ does the following

$$\begin{aligned}
\mathsf{st}_b^{i\to j} := \Bigg(& \left\{\mathsf{recom}_\ell^{i\to k}, \mathsf{ecom}_\ell^{i\to k}, \mathsf{nmcom}_\ell^{i\to k}\right\}_{\ell\in[3], k\in[n]\setminus\{i\}}, \\
& \qquad \left\{\mathsf{msg}_{\ell,i}\right\}_{\ell\in[3]}, \left\{\mathsf{rwi}_{a,\ell}^{i\to j}\right\}_{\ell\in[2]}, \mathsf{Trans}_2, \mathsf{td}_{1,j}\Bigg) \\
\mathsf{w}_b^{i\to j} := & \left(\perp, \mathsf{t}_j, \mathsf{r}_{\mathsf{nmcom}}^{i\to j}\right)
\end{aligned}$$

and compute

$$\mathsf{rwi}_{b,3}^{i\to j} := \mathsf{RWI}_3\left(\mathsf{st}_b^{i\to j}, \mathsf{w}_b^{i\to j}, \mathsf{rwi}_{b,1}^{i\to j}, \mathsf{rwi}_{b,2}^{i\to j}; \mathsf{r}_{\mathsf{rwi},b}^{i\to j}\right).$$

$\mathsf{Hyb}_{2,k,2}$: **Switch RWI proofs for L_a on the k-th thread:** In this sub-hybrid, $\mathsf{Sim}_{\mathsf{Hyb}}$ only modifies the third round of the k-th thread to switch to the "trapdoor witness" in the RWI proofs for L_a. Specifically, for every honest party P_i and every malicious party P_j, $\mathsf{Sim}_{\mathsf{Hyb}}$ does the following

$$\begin{aligned}
\mathsf{st}_a^{i\to j} &:= \left(\left\{\mathsf{recom}_\ell^{i\to j}, \mathsf{msg}_{\ell,i}, \mathsf{nmcom}_\ell^{i\to j}\right\}_{\ell\in[3]}, \mathsf{Trans}_2, \mathsf{td}_{1,j}\right) \\
\mathsf{w}_a^{i\to j} &:= \left(\perp, \perp, \perp, \mathsf{t}_j, \mathsf{r}_{\mathsf{nmcom}}^{i\to j}\right)
\end{aligned}$$

and compute

$$\mathsf{rwi}_{a,3}^{i\to j} := \mathsf{RWI}_3\left(\mathsf{st}_a^{i\to j}, \mathsf{w}_a^{i\to j}, \mathsf{rwi}_{a,1}^{i\to j}, \mathsf{rwi}_{a,2}^{i\to j}; \mathsf{r}_{\mathsf{rwi},a}^{i\to j}\right).$$

$\mathsf{Hyb}_{2,k,3}$: **Change RECom input to 0 on the k-th thread:** In this sub-hybrid, $\mathsf{Sim}_{\mathsf{Hyb}}$ only modifies the third round of the k-th thread to replace the RECom input for all honest parties with $(0, \mathsf{r}_i)$. In $\mathsf{Hyb}_{2,k,2}$, the input to Ecom for an honest party P_i was its input and randomness $(\mathsf{x}_i, \mathsf{r}_i)$ for the underlying protocol Π. This is done by a sequence of sub-hybrids given below.

$\mathsf{Hyb}_{2,k,3,0}$: **Change Com sender's message on main thread:** In this hybrid, $\mathsf{Sim}_{\mathsf{Hyb}}$ changes the Com commitment inside the RECom in the first round of the protocol. Specifically, for every honest party P_i and malicious party P_j and for all $\ell \in [N]$, compute $\mathsf{recom}_{1,\ell} \leftarrow \mathsf{Com}(0)$. This is done since all the look ahead threads share the same first round messages with the main thread.

$\mathsf{Hyb}_{2,k,3,1}$: **Change polynomial in third round:** In this hybrid, $\mathsf{Sim}_{\mathsf{Hyb}}$ picks a new polynomial q to change the RECom third round messages. Specifically, for every honest party P_i and malicious party P_j do the following:

- for every $\ell \in [N]$, pick a new degree 4 polynomial q_ℓ such that $(\mathsf{x}_i \oplus \mathsf{p}_\ell(0)) = (0 \oplus \mathsf{q}_\ell(0))$.
- compute $\mathsf{recom}_{3,\ell}$ as $(0 \oplus \mathsf{q}_\ell(0), \mathsf{q}_\ell(\mathsf{z}_\ell))$.

$\mathsf{Hyb}_{2,k,3,2}$: **Commit to new polynomial:** In this hybrid, $\mathsf{Sim}_{\mathsf{Hyb}}$ changes the Com commitment inside the RECom in the first round of the protocol. Specifically, for every honest party P_i and malicious party P_j and for all $\ell \in [N]$, compute $\mathsf{recom}_{1,\ell} \leftarrow \mathsf{Com}(q_\ell)$.

Note that $\mathsf{Hyb}_{2,k,3,2} \equiv \mathsf{Hyb}_{2,k,3}$

$\mathsf{Hyb}_{2,k,4}$: **Change Π input to 0 on the k-th thread:** In this sub-hybrid, $\mathsf{Sim}_{\mathsf{Hyb}}$ only modifies the third round of the k-th thread to replace the Π input for all honest parties with 0. In $\mathsf{Hyb}_{2,3}$, the input to Π for an honest party P_i in the third round was x_i.

$\mathsf{Hyb}_{2,k,5}$: **Switch RWI proofs for L_a on the k-th thread:** In this sub-hybrid, $\mathsf{Sim}_{\mathsf{Hyb}}$ only modifies the third round of the k-th thread to switch back to the "honest witness" in the RWI proofs for L_a. Specifically, for every honest party P_i and every malicious party P_j, $\mathsf{Sim}_{\mathsf{Hyb}}$ does the following

$$\begin{aligned}
\mathsf{st}_a^{i\to j} &:= \left(\left\{\mathsf{recom}_\ell^{i\to j}, \mathsf{msg}_{\ell,i}, \mathsf{nmcom}_\ell^{i\to j}\right\}_{\ell\in[3]}, \mathsf{Trans}_2, \mathsf{td}_{1,j}\right) \\
\mathsf{w}_a^{i\to j} &:= \left(0, \mathsf{r}_i, \mathsf{r}_{\mathsf{recom}}^{i\to j}, \perp, \perp\right)
\end{aligned}$$

and compute

$$\mathsf{rwi}_{a,3}^{i\to j} := \mathsf{RWI}_3\left(\mathsf{st}_a^{i\to j}, \mathsf{w}_a^{i\to j}, \mathsf{rwi}_{a,1}^{i\to j}, \mathsf{rwi}_{a,2}^{i\to j}; \mathsf{r}_{\mathsf{rwi},a}^{i\to j}\right).$$

$\mathsf{Hyb}_{2,k,6}$: **Switch RWI proofs for L_b on the k-th thread:** In this sub-hybrid, $\mathsf{Sim}_{\mathsf{Hyb}}$ only modifies the third round of the k-th thread to switch back to the "honest witness" in the RWI proofs for L_b. Specifically, for every honest party P_i and every malicious party P_j, $\mathsf{Sim}_{\mathsf{Hyb}}$ does the following

$$\begin{aligned}\mathsf{st}_b^{i\to j} &:= \Bigg(\left\{\mathsf{recom}_\ell^{i\to k}, \mathsf{ecom}_\ell^{i\to k}, \mathsf{nmcom}_\ell^{i\to k}\right\}_{\ell\in[3],k\in[n]\setminus\{i\}}, \\ &\qquad\qquad\qquad \left\{\mathsf{msg}_{\ell,i}\right\}_{\ell\in[3]}, \left\{\mathsf{rwi}_{a,\ell}^{i\to j}\right\}_{\ell\in[2]}, \mathsf{Trans}_2, \mathsf{td}_{1,j}\Bigg) \\ \mathsf{w}_b^{i\to j} &:= \left(\left\{\mathsf{r}_{\mathsf{ecom}}^{i\to k}, \mathsf{r}_{\mathsf{rwi},a}^{i\to j}, \mathsf{w}_a^{i\to k}\right\}_{k\in[n]\setminus\{i\}}, \bot, \bot\right)\end{aligned}$$

and compute

$$\mathsf{rwi}_{b,3}^{i\to j} := \mathsf{RWI}_3\left(\mathsf{st}_b^{i\to j}, \mathsf{w}_b^{i\to j}, \mathsf{rwi}_{b,1}^{i\to j}, \mathsf{rwi}_{b,2}^{i\to j}; \mathsf{r}_{\mathsf{rwi},b}^{i\to j}\right).$$

$\mathsf{Hyb}_{2,k,7}$: **Change NMCom on k-th thread:** In this sub-hybrid, $\mathsf{Sim}_{\mathsf{Hyb}}$ only modifies the third round of the k-th thread to commit in the NMCom to $\bot$. Specifically, for every honest party P_i and every malicious party P_j, $\mathsf{Sim}_{\mathsf{Hyb}}$ modifies the third round NMCom message to be

$$\mathsf{nmcom}_3^{i\to j} := \mathsf{NMCom}_3\left(\bot, \mathsf{nmcom}_1^{i\to j}, \mathsf{nmcom}_3^{i\to j}; \mathsf{r}_{\mathsf{nmcom}}^{i\to j}\right).$$

Note that $\mathsf{Hyb}_{2,T,7} \equiv \mathsf{Hyb}_2$

Hyb_3: **Change NMCom on main thread:** In this hybrid, $\mathsf{Sim}_{\mathsf{Hyb}}$ only modifies the third round of the main thread to commit in the NMCom to the trapdoor. In Hyb_2, NMCom was a commitment to $\bot$. Specifically, for every honest party P_i and every malicious party P_j, $\mathsf{Sim}_{\mathsf{Hyb}}$ modifies the third round NMCom message to be

$$\mathsf{nmcom}_3^{i\to j} := \mathsf{NMCom}_3\left(\mathsf{t}_j, \mathsf{nmcom}_1^{i\to j}, \mathsf{nmcom}_3^{i\to j}; \mathsf{r}_{\mathsf{nmcom}}^{i\to j}\right)$$

where t_j is a valid trapdoor extracted from the look-ahead threads as in Hyb_2.

Hyb_4: **Switch RWI proofs for L_b on the main thread:** In this hybrid, $\mathsf{Sim}_{\mathsf{Hyb}}$ only modifies the third round of the main thread to switch to the "trapdoor witness" in the RWI proofs for L_b. Specifically, for every honest party P_i and every malicious party P_j, $\mathsf{Sim}_{\mathsf{Hyb}}$ does the following

$$\begin{aligned}\mathsf{st}_b^{i\to j} &:= \left(\left\{\mathsf{recom}_\ell^{i\to j}, \mathsf{ecom}_\ell^{i\to j}, \mathsf{msg}_{\ell,i}, \mathsf{nmcom}_\ell^{i\to j}\right\}_{\ell\in[3]}, \left\{\mathsf{rwi}_{b,\ell}^{i\to j}\right\}_{\ell\in[2]} \mathsf{Trans}_2, \mathsf{td}_{1,j}\right) \\ \mathsf{w}_b^{i\to j} &:= \left(\bot, \mathsf{t}_j, \mathsf{r}_{\mathsf{nmcom}}^{i\to j}\right)\end{aligned}$$

and compute

$$\mathsf{rwi}_{b,3}^{i\to j} := \mathsf{RWI}_3\left(\mathsf{st}_b^{i\to j}, \mathsf{w}_b^{i\to j}, \mathsf{rwi}_{b,1}^{i\to j}, \mathsf{rwi}_{b,2}^{i\to j}; \mathsf{r}_{\mathsf{rwi},b}^{i\to j}\right).$$

Hyb_5: **Switch RWI proofs for L_a on the main thread:** In this hybrid, $\mathsf{Sim}_{\mathsf{Hyb}}$ only modifies the third round of the main thread to switch to the "trapdoor witness" in the RWI proofs for L_a. Specifically, for every honest party P_i and every malicious party P_j, $\mathsf{Sim}_{\mathsf{Hyb}}$ does the following

$$\begin{aligned}\mathsf{st}_b^{i\to j} &:= \Bigg(\left\{\mathsf{recom}_\ell^{i\to k}, \mathsf{ecom}_\ell^{i\to k}, \mathsf{nmcom}_\ell^{i\to k}\right\}_{\ell\in[3],k\in[n]\setminus\{i\}}, \\ &\qquad\qquad\qquad \left\{\mathsf{msg}_{\ell,i}\right\}_{\ell\in[3]}, \left\{\mathsf{rwi}_{a,\ell}^{i\to j}\right\}_{\ell\in[2]}, \mathsf{Trans}_2, \mathsf{td}_{1,j}\Bigg) \\ \mathsf{w}_a^{i\to j} &:= \left(\bot, \bot, \bot, \mathsf{t}_j, \mathsf{r}_{\mathsf{nmcom}}^{i\to j}\right)\end{aligned}$$

and compute

$$\mathsf{rwi}_{a,3}^{i\to j} := \mathsf{RWI}_3\left(\mathsf{st}_a^{i\to j}, \mathsf{w}_a^{i\to j}, \mathsf{rwi}_{a,1}^{i\to j}, \mathsf{rwi}_{a,2}^{i\to j}; \mathsf{r}_{\mathsf{rwi},a}^{i\to j}\right).$$

Hyb_6: **Switch WI proofs for L_c on the main thread:** In this hybrid, $\mathsf{Sim}_{\mathsf{Hyb}}$ only modifies the fourth round of the main thread to switch to the "trapdoor witness" in the WI proofs for L_c. Specifically, for every honest party P_i and every malicious party P_j, $\mathsf{Sim}_{\mathsf{Hyb}}$ does the following

$$\mathsf{st}_c^{i\to j} := \left(\left\{\mathsf{recom}_\ell^{i\to j}, \mathsf{msg}_{\ell,i}, \mathsf{nmcom}_\ell^{i\to j}\right\}_{\ell\in[3]}, \left\{\mathsf{rwi}_{b,\ell}^{i\to j}\right\}_{\ell\in[2]} \mathsf{Trans}_3, \left\{\mathsf{ot}_\ell^{i\to j}\right\}_{\ell\in[4]}, \mathsf{td}_{1,j}, \mathsf{C}_i\right)$$
$$\mathsf{w}_c^{i\to j} := \left(\bot,\bot,\bot,\bot,\mathsf{t}_j,\mathsf{r}_{\mathsf{nmcom}}^{i\to j}\right)$$

and compute

$$\mathsf{wi}_3^{i\to j} := \mathsf{WI}_3\left(\mathsf{st}_c^{i\to j}, \mathsf{w}_c^{i\to j}, \mathsf{wi}_1^{i\to j}, \mathsf{wi}_2^{i\to j}; \mathsf{r}_{\mathsf{wi}}^{i\to j}\right).$$

Hyb_7: **Change RECom input to 0 on the main thread:** In this hybrid, $\mathsf{Sim}_{\mathsf{Hyb}}$ only modifies the third round of the k-th thread to replace the RECom input for all honest parties with $(0, \mathsf{r}_i)$. In Hyb_6, the input to Ecom for an honest party P_i was its input and randomness $(\mathsf{x}_i, \mathsf{r}_i)$ for the underlying protocol Π.

Hyb_8: **Simulate Π on main thread:** In this hybrid $\mathsf{Sim}_{\mathsf{Hyb}}$ only modifies the transcript of the underlying protocol Π. Specifically, $\mathsf{Sim}_{\mathsf{Hyb}}$ does the following:

1. Due to the fact that the first two simulated rounds of Π are honest computations, we do not make any changes to the first two rounds but refer to the collective first round honest inputs as the output of $\mathcal{S}_1$ with randomness $\mathsf{r}_{\mathcal{S}} := \{\mathsf{r}_i\}_{\mathsf{P}_i\in\mathcal{H}}$. Likewise for the second round messages.
2. Compute the third round messages of all honest parties in the underlying protocol Π
$$\left\{\mathsf{msg}_{3,i}\right\}_{\mathsf{P}_i\in\mathcal{H}} := \mathcal{S}_3\left(\mathsf{Trans}_2; \mathsf{r}_{\mathcal{S}}\right)$$
using the transcript obtained so far and randomness $\mathsf{r}_{\mathcal{S}}$ as defined above.
3. Compute the third round messages of all honest parties in the underlying protocol Π
$$\left\{\mathsf{msg}_{4,i}\right\}_{\mathsf{P}_i\in\mathcal{H}} \leftarrow \mathcal{S}_4\left(y, \{\mathsf{x}_j, \mathsf{r}_j\}_{\mathsf{P}_j\notin\mathcal{H}}, \mathsf{Trans}_3;\ \mathsf{r}_{\mathcal{S}}\right)$$
where i on the right hand side indexes the i-th component of the output. Note that $\mathcal{S}_4$ will not be called if there was an implicit or explicit abort.

Hyb_9: **Change GC on main thread:** In this hybrid $\mathsf{Sim}_{\mathsf{Hyb}}$ only modifies the garbled circuits of honest parties P_i if there is an implicit abort in Step 1. Specifically, if there is an implicit abort, the garbled circuit for each honest party P_i is computed as:

$$\left(\mathsf{C}_i, \overline{\mathsf{lab}}_i\right) \leftarrow \mathsf{Garble}\left(\mathsf{C}_\bot\right)$$

where $\mathsf{C}_\bot$ is the circuit with the same topology as C but always outputs $\bot$. We note that even in the case of an implicit abort, we are able to extract the trapdoor, but not necessarily the witness. For every honest party P_i,

– if there is an implicit abort, then

$$\left(\mathsf{C}_i, \overline{\mathsf{lab}_i}\right) := \mathsf{Garble}\left(\mathsf{C}_\perp; \mathsf{r}_{\mathsf{gc},i}\right)$$

– if $\exists \mathsf{P}_j \in \mathcal{A}$ such that RWI proof did not accept. i.e.

$$\mathsf{RWI}_4\left(\mathsf{rwi}_1^{j\to i}, \mathsf{rwi}_2^{j\to i}, \mathsf{rwi}_3^{j\to i}, \mathsf{st}^{j\to i}; \mathsf{r}_{\mathsf{rwi}}^{j\to i}\right) \neq 1,$$

then

$$\left(\mathsf{C}_i, \overline{\mathsf{lab}_i}\right) := \mathsf{Garble}\left(\mathsf{C}_\perp; \mathsf{r}_{\mathsf{gc},i}\right)$$

– else,

$$\left(\mathsf{C}_i, \overline{\mathsf{lab}_i}\right) := \mathsf{Garble}\left(\mathsf{C}\left[i, \mathsf{msg}_{4,i}, \left\{\mathsf{rwi}_{b,\ell}^{j\to i}\right\}_{\ell\in[2], j\in[n]\setminus\{i\}} \left\{\mathsf{st}_b^{j\to i}, \mathsf{r}_{\mathsf{rwi},b}^{j\to i}\right\}_{j\in[n]\setminus\{i\}}\right]; \mathsf{r}_{\mathsf{gc},i}\right)$$

Remark 6. *We note that if there is an implicit abort, all honest parties will have a $\perp$ encoded in the circuit.*

$\mathsf{Hyb}_{\mathsf{IDEAL}}$: **Run the actual probability estimation:** In this hybrid, the number of look-ahead threads is increased from $\frac{5\cdot n\cdot\lambda}{\mu}$ to as many as needed to estimate the probability of the adversary not aborting – ε'.

Additionally, at this point, $\mathsf{Sim}_{\mathsf{Hyb}}$ doesn't re-sample the main thread $\frac{\lambda}{\mu}$ times. Instead, $\mathsf{Sim}_{\mathsf{Hyb}}$ resamples the main thread for $\min\left(2^\lambda, \frac{\lambda^2}{\varepsilon'}\right)$ times as in the ideal world. This hybrid corresponds exactly to the ideal world.

4.2.3 Indistinguishability of Hybrids

We will maintain the following invariant across the hybrids.

Definition 15 (Invariant)**.** *Consider any malicious party P_j and any honest party P_i. $\mathsf{td}_{1,i}$ denotes the first message of the trapdoor generation protocol with P_i as the trapdoor generator. The tuple $\left(\mathsf{nmcom}_1^{j\to i}, \mathsf{nmcom}_2^{j\to i}, \mathsf{nmcom}_3^{j\to i}\right)$ denotes the messages of the non-malleable commitment with P_j as the committer and P_i is the receiver.*

This event E occurs if $\exists i, j$ such that

– $\mathsf{Ext}_{\mathsf{NMCom}}$ *outputs t_i from the non-malleable commitment* $\left(\mathsf{nmcom}_1^{j\to i}, \mathsf{nmcom}_2^{j\to i}, \mathsf{nmcom}_3^{j\to i}\right)$ *(AND)*

– $\mathsf{TDValid}(\mathsf{td}_{1,i}, \mathsf{t}_i) = 1$

That is, the event E occurs if the extractor for the non-malleable commitment outputs a valid trapdoor t_i (corresponding to the trapdoor generation protocol where P_i was the trapdoor generator) from the non-malleable commitment from player P_j to P_i.

The invariant is

$$\mathsf{Pr}\Big[\textit{Event } \mathsf{E} \textit{ occurs}\Big] \leq \mathsf{negl}(\lambda)$$

Claim 1. *Assuming the "1-rewinding security" of the trapdoor generation protocol* TDGen *and the existence of an extractor* $\mathsf{Ext}_{\mathsf{NMCom}}$ *for the non-malleable commitment scheme* NMCom*, the invariant holds in* $\mathsf{Hyb}_{\mathsf{REAL}}$.

Proof. This is proven by contradiction. Assume that the invariant doesn't hold in $\mathsf{Hyb}_{\mathsf{REAL}}$. Then there exists an adversary $\mathcal{A}$ such that for some honest party P_{i^*} and malicious party P_{j^*}, $\mathcal{A}$ causes event E to occur with non-negligible probability. We will use this adversary to create an adversary $\mathcal{A}_{\mathsf{TDGen}}$ that breaks the "1-rewinding security" of the trapdoor generation protocol TDGen with non-negligible probability.

We now describe the working of $\mathcal{A}_{\mathsf{TDGen}}$ which interacts with the challenger $\mathcal{C}_{\mathsf{TDGen}}$. $\mathcal{A}_{\mathsf{TDGen}}$ picks randomly, an honest party P_i, and a random malicious party P_j. All messages other than the trapdoor messages are computed in the same manner as $\mathsf{Sim}_{\mathsf{Hyb}}$. The trapdoor messages for P_i are exposed to the external challenger. Specifically, in round 1, set $\mathsf{td}_{1,i} = \mathsf{td}_1$ where td_1 is received from $\mathcal{C}_{\mathsf{TDGen}}$. On receiving all the values $\mathsf{td}_2^{1\to i}, \cdots, \mathsf{td}_2^{n\to i}$, including the value $\mathsf{td}_2^{j\to i}$ from $\mathcal{A}$ in round 2, $\mathcal{A}_{\mathsf{TDGen}}$ sets $\mathsf{td}_{2,i} := \left(\mathsf{td}_2^{1\to i}||\cdots||\mathsf{td}_2^{1\to i}\right)$ and this is the value forwarded to $\mathcal{C}_{\mathsf{TDGen}}$ as the second round response. Set $\mathsf{td}_{3,i} = \mathsf{td}_3$ where td_3 is received from $\mathcal{C}_{\mathsf{TDGen}}$, and compute the rest of the third round messages for $\mathcal{A}$. At this point, $\mathcal{A}_{\mathsf{TDGen}}$ rewinds $\mathcal{A}$ back to the beginning of round 2 to enable extraction from the NMCom. Specifically, $\mathcal{A}_{\mathsf{TDGen}}$ creates a look ahead thread that runs only the second and third round. As in the main thread, the trapdoor messages are received from $\mathcal{C}_{\mathsf{TDGen}}$. Recall that the "1-rewinding" property of the trapdoor generation protocol allows for a second td_2 query to $\mathcal{C}_{\mathsf{TDGen}}$.

Now $\mathcal{A}_{\mathsf{TDGen}}$ runs the extractor $\mathsf{Ext}_{\mathsf{NMCom}}$ of the non-malleable commitment scheme using the message in both the threads that correspond to the non-malleable commitment from malicious party P_j to honest party P_i. Let the output of $\mathsf{Ext}_{\mathsf{NMCom}}$ be t^*. $\mathcal{A}_{\mathsf{TDGen}}$ outputs t^* as a valid trapdoor to $\mathcal{C}_{\mathsf{TDGen}}$.

By our assumption, the invariant doesn't hold. Thus $\mathsf{Ext}_{\mathsf{NMCom}}$, on adversary P_{j^*}'s commitment to P_i, outputs a valid trapdoor t_{i^*} for the trapdoor generation messages of the honest party P_{i^*} with non-negligible probability ε. With probability at least $\frac{1}{n^2}$, where n is the total number of players, this corresponds to honest party P_i and malicious party P_j picked randomly by $\mathcal{A}_{\mathsf{TDGen}}$. Therefore, with non-negligible probability $\frac{\varepsilon}{n^2}$, $\mathcal{A}_{\mathsf{TDGen}}$ outputs t^* as a valid trapdoor to $\mathcal{C}_{\mathsf{TDGen}}$ which breaks the 1-rewinding security of the trapdoor generation protocol TDGen. Thus, it must be the case that the invariant holds in $\mathsf{Hyb}_{\mathsf{REAL}}$. □

Remark 7. *We note that if the invariant holds, it must be the case that no adversary can commit to a valid trapdoor with a non-negligible probability. This in turn implies accepting witness indistinguishable proofs cannot use a "trapdoor witness" other than with negligible probability.*

Claim 2. *The invariant holds in* Hyb_0.

Proof. Since there is no difference in the main thread in the first 3 rounds between $\mathsf{Hyb}_{\mathsf{REAL}}$ and Hyb_0, the invariant continues to hold. □

Claim 3. Hyb_0 *is indistinguishable from* $\mathsf{Hyb}_{\mathsf{REAL}}$ *except with probability at most* $\frac{\mu}{4} + \mathsf{negl}(\lambda)$.

Proof. This is argued in two cases depending on the probability with which the adversary abort.

Case 1: $\Pr[\textbf{not abort}] \geq \frac{\mu}{4}$**:**

Suppose the adversary doesn't cause an abort with probability greater that $\frac{\mu}{4}$. Let us analyze the probability with which $\perp_{\mathsf{extract}}$ is output by $\mathsf{Sim}_{\mathsf{Hyb}}$. By the Chernoff bound, in Hyb_0, except with negligible probability, in the set of $\frac{5\cdot n\cdot\lambda}{\mu}$ threads, there will be at least 5 GOOD threads with respect to some honest party P_{i^*}. Now all that's left to argue is that $\mathsf{Ext}_{\mathsf{RECom}}$ and TDExt fail to extract with negligible probability.

From the definition of RECom, algorithm $\mathsf{Ext}_{\mathsf{RECom}}$ is successful except with negligible probability if given as input $\left(\mathsf{recom}_1, \left\{\mathsf{recom}_2^k, \mathsf{recom}_3^k\right\}_{k\in[5]}\right)$ such that $\left(\mathsf{recom}_1, \mathsf{recom}_2^k, \mathsf{recom}_3^k\right)$

constitute "well-formed" and "admissible" rewinding secure extractable commitment messages. "Admissibility" follows trivially since $\mathsf{Sim}_{\mathsf{Hyb}}$ picks random challenges z for the extractable commitment. From the above claim, we've proved that the invariant holds in Hyb_0, and thus from the soundness of RWI and WI, in each GOOD thread with respect to some honest party P_{i^*}, the following holds: for every malicious P_j and every honest P_i, $\left(\mathsf{recom}_1^{j\to i}, \mathsf{recom}_2^{j\to i}, \mathsf{recom}_3^{j\to i}\right)$ is a "well formed" tuple of RECom. Thus $\mathsf{Ext}_{\mathsf{RECom}}$ fails only with negligible probability.

From the definition of TDGen, algorithm TDExt is successful except with negligible probability if given as input $\left(\mathsf{td}_1, \left\{\mathsf{td}_2^k, \mathsf{td}_3^k\right\}_{k\in[3]}\right)$ where td_1 is the first message of the protocol TDGen and $\mathsf{td}_2^k, \mathsf{td}_3^k$ denote the second and third round message of the k-th execution of TDGen using the same first round message. Since there are 5 GOOD threads, we can extract every malicious party's trapdoor except with negligible probability.

Finally, from the Chernoff bound, in the set of $\frac{\lambda}{\mu}$ re-sampled main threads, there will be at least one completed execution. Thus, the adversary's view in $\mathsf{Hyb}_{\mathsf{REAL}}$ and Hyb_0 is indistinguishable.

Case 2: $\Pr[\textbf{not abort}] < \frac{\mu}{4}$**:**

Suppose the adversary doesn't cause an abort with probability smaller than $\frac{\mu}{4}$. Then, in both hybrids, $\mathsf{Sim}_{\mathsf{Hyb}}$ aborts at the end of the "Check Abort" step except with probability $\frac{\mu}{4}$. Thus, in this case, the adversary's view in $\mathsf{Hyb}_{\mathsf{REAL}}$ and Hyb_0 is indistinguishable except with probability at most $\frac{\mu}{4} + \mathsf{negl}(\lambda)$.

□

Remark 8. *To avoid cluttering of the proof, we will assume the argument that if both adjacent hybrids have fewer than 5* GOOD *look-ahead threads with respect to all parties, the two hybrids are identical.*

Claim 4. *The invariant holds in* $\mathsf{Hyb}_{1,0}$.

Proof. Since there is no difference in the main thread in the first 3 rounds between $\mathsf{Hyb}_{1,0}$ and Hyb_0, the invariant continues to hold. □

Claim 5. *Assuming the hiding property of* OT *against malicious senders,* $\mathsf{Hyb}_{1,0}$ *is indistinguishable from* Hyb_0.

Proof. The only difference between the two hybrids is when the "Check Abort" step doesn't succeed. In that case, in Hyb_0, $\mathsf{Sim}_{\mathsf{Hyb}}$ uses as input to OT the third round message for the RWI proof for L_b, while in $\mathsf{Hyb}_{1,0}$, $\mathsf{Sim}_{\mathsf{Hyb}}$ uses input 0 for the third round of OT. This is in fact done by a sequence of hybrids, wherein only a single instance of the honest party's input to the OT is changed. There are $< n^2$ instances where an honest party is the receiver, and thus at most n^2 intermediate hybrids. Suppose there is an adversary $\mathcal{D}$ that can distinguish between any two adjacent hybrids, we will create an adversary $\mathcal{A}_{\mathsf{OT}}$ that breaks the hiding of the OT scheme. Recall that this is only in the setting that "Check Abort" doesn't succeed and hence the fourth round messages of the honest party are not sent.

We now describe the working of $\mathcal{A}_{\mathsf{OT}}$ which interacts with the challenger $\mathcal{C}_{\mathsf{OT}}$. Let the change in these adjacent hybrids be made for an honest party $\mathsf{P}_{\widehat{i}}$ to a party $\mathsf{P}_{\widehat{j}}$. All messages other than those of the chosen OT are computed as in the same manner as $\mathsf{Sim}_{\mathsf{Hyb}}$. First, set $\mathsf{ot}_1^{\widehat{j}\to\widehat{i}} := \mathsf{ot}_1$ where ot_1 is sent by $\mathcal{C}_{\mathsf{OT}}$. On receiving/computing[8] message $\mathsf{ot}_1^{\widehat{j}\to\widehat{i}}$, send this along with $(\mathsf{rwi}_{b,3}^{\widehat{i}\to\widehat{j}}, 0)$

[8]Since the OT sender in question may in fact be an honest party.

to $\mathcal{C}_{\mathsf{OT}}$. Where $\mathsf{rwi}_{b,3}^{\widehat{i}\to\widehat{j}}$ is computed as in the previous hybrid by $\mathsf{Sim}_{\mathsf{Hyb}}$. $\mathcal{C}_{\mathsf{OT}}$ then chooses as input one of the two values at random and sends ot_3. $\mathcal{A}_{\mathsf{OT}}$ sets $\mathsf{ot}_3^{\widehat{j}\to\widehat{i}} := \mathsf{ot}_3$. The view generated is then given to the adversary $\mathcal{D}$, wherein depending on the choice of $\mathcal{C}_{\mathsf{OT}}$, the view corresponds to one of the two adjacent hybrids. The output from $\mathcal{D}$ is set to be the output of $\mathcal{A}_{\mathsf{OT}}$.

By our assumption, views of adjacent hybrids are distinguishable with non-negligible probability ε. Therefore, with the same probability ε $\mathcal{A}_{\mathsf{OT}}$ can break the hiding property of OT. Thus, it must be the case that ε is negligible. Since there are at most n^2 intermediate hybrids, the two end hybrids, $\mathsf{Hyb}_{1,0}$ and Hyb_0, remain indistinguishable except with negligible probability. □

Claim 6. *The invariant holds in* $\mathsf{Hyb}_{1,1}$.

Proof. Since there is no difference in the main thread in the first 3 rounds between $\mathsf{Hyb}_{1,1}$ and $\mathsf{Hyb}_{1,0}$, the invariant continues to hold. □

Claim 7. *Assuming the hiding property of* Ecom*,* $\mathsf{Hyb}_{1,1}$ *is indistinguishable from* $\mathsf{Hyb}_{1,0}$.

Proof. The proof works in the same way as the proof in the previous claim. The only difference between the two hybrids is when the "Check Abort" step doesn't succeed. In that case, in $\mathsf{Hyb}_{1,0}$, $\mathsf{Sim}_{\mathsf{Hyb}}$ uses as input to Ecom the third round message for the RWI proof for L_a, while in $\mathsf{Hyb}_{1,1}$, $\mathsf{Sim}_{\mathsf{Hyb}}$ uses input 0 for the third round of Ecom. This is in fact done by a sequence of hybrids, wherein only a single instance of the honest party's input to the Ecom is changed. There are $< n^2$ instances where an honest party is the committer, and thus at most n^2 intermediate hybrids. Suppose there is an adversary $\mathcal{D}$ that can distinguish between any two adjacent hybrids, we will create an adversary $\mathcal{A}_{\mathsf{Ecom}}$ that breaks the hiding of the Ecom scheme.

We now describe the working of $\mathcal{A}_{\mathsf{Ecom}}$ which interacts with the challenger $\mathcal{C}_{\mathsf{Ecom}}$. Let the change in these adjacent hybrids be made for an honest party $\mathsf{P}_{\widehat{i}}$ to a party $\mathsf{P}_{\widehat{j}}$. All messages other than those of the chosen Ecom are computed as in the same manner as $\mathsf{Sim}_{\mathsf{Hyb}}$. First, set $\mathsf{ecom}_1^{\widehat{i}\to\widehat{j}} := \mathsf{recom}_1$ where ecom_1 is sent by $\mathcal{C}_{\mathsf{recom}}$. On receiving/computing message $\mathsf{ecom}_1^{\widehat{i}\to\widehat{j}}$, send this along with $(\mathsf{rwi}_{a,3}^{\widehat{i}\to\widehat{j}}, 0)$ to $\mathcal{C}_{\mathsf{Ecom}}$. Where $\mathsf{rwi}_{b,3}^{i\to j}$ is computed as in the previous hybrid by $\mathsf{Sim}_{\mathsf{Hyb}}$. $\mathcal{C}_{\mathsf{Ecom}}$ then commits to one of the two values at random and sends recom_3. $\mathcal{A}_{\mathsf{Ecom}}$ sets $\mathsf{ecom}_3^{\widehat{i}\to\widehat{j}} := \mathsf{ecom}_3$. The view generated is then given to the adversary $\mathcal{D}$, wherein depending on the choice of $\mathcal{C}_{\mathsf{Ecom}}$, the view corresponds to one of the two adjacent hybrids. The output from $\mathcal{D}$ is set to be the output of $\mathcal{A}_{\mathsf{Ecom}}$.

By our assumption, views of adjacent hybrids are distinguishable with non-negligible probability ε. Therefore, with the same probability ε $\mathcal{A}_{\mathsf{Ecom}}$ can break the hiding property of Ecom. Thus, it must be the case that ε is negligible. Since there are at most n^2 intermediate hybrids, the two end hybrids, $\mathsf{Hyb}_{1,1}$ and $\mathsf{Hyb}_{1,0}$, remain indistinguishable except with negligible probability. □

Claim 8. *The invariant holds in* $\mathsf{Hyb}_{1,2}$.

Proof. Since there is no difference in the main thread in the first 3 rounds between $\mathsf{Hyb}_{1,2}$ and $\mathsf{Hyb}_{1,1}$, the invariant continues to hold. □

Claim 9. *Assuming the hiding property of* RECom*,* $\mathsf{Hyb}_{1,2}$ *is indistinguishable from* $\mathsf{Hyb}_{1,1}$.

Proof. The only difference between the two hybrids is when the "Check Abort" step doesn't succeed. In that case, in $\mathsf{Hyb}_{1,1}$, $\mathsf{Sim}_{\mathsf{Hyb}}$ uses as input to RECom $(\mathsf{x}_{\widehat{i}}, \mathsf{r}_{\widehat{i}})$, while in $\mathsf{Hyb}_{1,2}$, $\mathsf{Sim}_{\mathsf{Hyb}}$ uses input 0 for the third round of RECom. This is in fact done by a sequence of hybrids, wherein only a single instance of the honest party's input to the RECom is changed. There are $< n^2$ instances where an honest party is the committer, and thus at most n^2 intermediate hybrids. Suppose there is an adversary $\mathcal{D}$ that can distinguish between any two adjacent hybrids, we will create an adversary $\mathcal{A}_{\mathsf{RECom}}$ that breaks the hiding of the RECom scheme.

We now describe the working of $\mathcal{A}_{\mathsf{RECom}}$ which interacts with the challenger $\mathcal{C}_{\mathsf{RECom}}$. Let the change in these adjacent hybrids be made for an honest party $\mathsf{P}_{\widehat{i}}$ to a party $\mathsf{P}_{\widehat{j}}$. All messages other than those of the chosen RECom are computed as in the same manner as $\mathsf{Sim}_{\mathsf{Hyb}}$. First, set $\mathsf{recom}_1^{\widehat{i}\to\widehat{j}} := \mathsf{recom}_1$ where recom_1 is sent by $\mathcal{C}_{\mathsf{recom}}$. On receiving/computing message $\mathsf{recom}_1^{\widehat{i}\to\widehat{j}}$, send this along with $\big((\mathsf{x}_{\widehat{i}}, \mathsf{r}_{\widehat{i}}), 0\big)$ to $\mathcal{C}_{\mathsf{RECom}}$. Where $(\mathsf{x}_{\widehat{i}}, \mathsf{r}_{\widehat{i}})$ is the input and randomness of $\mathsf{P}_{\widehat{i}}$ computed as in the previous hybrid by $\mathsf{Sim}_{\mathsf{Hyb}}$. $\mathcal{C}_{\mathsf{RECom}}$ then commits to one of the two values at random and sends recom_3. $\mathcal{A}_{\mathsf{RECom}}$ sets $\mathsf{recom}_3^{\widehat{i}\to\widehat{j}} := \mathsf{recom}_3$. The view generated is then given to the adversary $\mathcal{D}$, wherein depending on the choice of $\mathcal{C}_{\mathsf{RECom}}$, the view corresponds to one of the two adjacent hybrids. The output from $\mathcal{D}$ is set to be the output of $\mathcal{A}_{\mathsf{RECom}}$.

By our assumption, views of adjacent hybrids are distinguishable with non-negligible probability ε. Therefore, with the same probability ε, $\mathcal{A}_{\mathsf{RECom}}$ can break the hiding property of RECom. Thus, it must be the case that ε is negligible. Since there are at most n^2 intermediate hybrids, the two end hybrids, $\mathsf{Hyb}_{1,2}$ and $\mathsf{Hyb}_{1,1}$, remain indistinguishable except with negligible probability. □

Claim 10. *The invariant holds in* $\mathsf{Hyb}_{1,3}$.

Proof. Since there is no difference in the main thread in the first 3 rounds between $\mathsf{Hyb}_{1,3}$ and $\mathsf{Hyb}_{1,2}$, the invariant continues to hold. □

Claim 11. *Assuming the privacy of* Π*,* $\mathsf{Hyb}_{1,3}$ *is indistinguishable from* $\mathsf{Hyb}_{1,2}$.

Proof. The only difference between the two hybrids is when the "Check Abort" step doesn't succeed. In that case, in $\mathsf{Hyb}_{1,2}$, $\mathsf{Sim}_{\mathsf{Hyb}}$ uses as input to the third round of Π [9] $(\mathsf{x}_i, \mathsf{r}_i)$ for all honest parties P_i, while in $\mathsf{Hyb}_{1,3}$, $\mathsf{Sim}_{\mathsf{Hyb}}$ uses as input to the third round of Π $(0, \mathsf{r}_{\widehat{i}})$ for all honest parties P_i. This is in fact done by a sequence of hybrids, wherein only a single instance of the honest party's input to the Π is changed. There are $< n$ parties, and thus at most n^2 intermediate hybrids. Suppose there is an adversary $\mathcal{D}$ that can distinguish between any two adjacent hybrids, we will create an adversary $\mathcal{A}_{\Pi}$ that breaks the indistinguishability of Π.

We now describe the working of $\mathcal{A}_{\Pi}$ which interacts with the challenger $\mathcal{C}_{\Pi}$. Let the change in these adjacent hybrids be made for an honest party P_i. All messages other than those of the chosen Π are computed as in the same manner as $\mathsf{Sim}_{\mathsf{Hyb}}$. First, set $\mathsf{msg}_{1,i} := \mathsf{msg}_1$ where msg_1 is sent by $\mathcal{C}_{\mathsf{recom}}$. On receiving and computing message $\mathsf{msg}_{1,j}$ for all other parties P_j, send this to $\mathcal{C}_{\Pi}$. Set $\mathsf{msg}_{2,i} := \mathsf{msg}_2$ where msg_2 is sent by $\mathcal{C}_{\mathsf{recom}}$. On receiving and computing message $\mathsf{msg}_{2,j}$ for all other parties P_j, send this to $\mathcal{C}_{\Pi}$ along with $((\mathsf{x}_i, \mathsf{r}_i), (0, \mathsf{r}_i))$. Where $(\mathsf{x}_i, \mathsf{r}_i)$ is the input and randomness of P_i computed as in the previous hybrid by $\mathsf{Sim}_{\mathsf{Hyb}}$. $\mathcal{C}_{\Pi}$ then uses one of the two values at random and sends msg_3. $\mathcal{A}_{\Pi}$ sets $\mathsf{msg}_{3,i} := \mathsf{msg}_3$. The view generated is then given to the adversary $\mathcal{D}$, wherein depending on the choice of $\mathcal{C}_{\Pi}$, the view corresponds to one of the two adjacent hybrids. The output from $\mathcal{D}$ is set to be the output of $\mathcal{A}_{\Pi}$.

By our assumption, views of adjacent hybrids are distinguishable with non-negligible probability ε. Therefore, with the same probability ε $\mathcal{A}_{\Pi}$ can break the input indistinguishability property of Π. Thus, it must be the case that ε is negligible. Since there are at most n intermediate hybrids, the two end hybrids, $\mathsf{Hyb}_{1,3}$ and $\mathsf{Hyb}_{1,2}$, remain indistinguishable except with negligible probability. □

Claim 12. *The invariant holds in* $\mathsf{Hyb}_{1,4}$.

[9] This is the first round of Π that uses the input.

Proof. Since there is no difference in the main thread in the first 3 rounds between $\mathsf{Hyb}_{1,4}$ and $\mathsf{Hyb}_{1,3}$, the invariant continues to hold. $\square$

Claim 13. *Assuming the hiding property of* Ecom*,* $\mathsf{Hyb}_{1,4}$ *is indistinguishable from* $\mathsf{Hyb}_{1,3}$*.*

Proof. This proof follows identically as in Claim 7. $\square$

Claim 14. *The invariant holds in* $\mathsf{Hyb}_{1,5}$*.*

Proof. Since there is no difference in the main thread in the first 3 rounds between $\mathsf{Hyb}_{1,5}$ and $\mathsf{Hyb}_{1,4}$, the invariant continues to hold. $\square$

Claim 15. *Assuming the hiding property of* OT *against malicious senders,* $\mathsf{Hyb}_{1,5}$ *is indistinguishable from* $\mathsf{Hyb}_{1,4}$*.*

Proof. This proof follows identically as in Claim 5. $\square$

Note that $\mathsf{Hyb}_{1,5} \equiv \mathsf{Hyb}_1$. This gives us that Hyb_1 and Hyb_0 are indistinguishable other than with negligible probability.

We now prove claims for all $k \in [T]$, where we set $\mathsf{Hyb}_{2,0,7} \equiv \mathsf{Hyb}_1$

We note that we will argue that the invariant holds even in the look ahead thread that we are making changes in. Initially, since all the look ahead threads are identical to the main thread, by claim 1 we know that the invariant holds in each of them. The invariant is useful since we will argue that if the invariant holds true, the probability of the extracted RWI accepting cannot change with noticeable probability. From the soundness of RWI we are guarantees that, with the change, we are still successfully extracting from the adversary with the same probability.

Claim 16. *Assuming* NMCom *is a secure non-malleable commitment scheme with non-malleability with respect to extraction, the invariant holds in* $\mathsf{Hyb}_{2,k,0}$*.*

Proof. We know that the invariant holds $\mathsf{Hyb}_{2,k-1,7}$. The only difference between $\mathsf{Hyb}_{2,k-1,7}$ and $\mathsf{Hyb}_{2,k,0}$ is that the simulator commits to the trapdoor in the k-th look ahead thread. Assume, for the sake of contradiction, that the invariant doesn't hold in $\mathsf{Hyb}_{2,k,0}$. Then there exists an adversary $\mathcal{A}$ such that for some honest party P_{i^*} and malicious party P_{j^*}, $\mathcal{A}$ causes event E to occur with non-negligible probability. We will use this adversary to create an adversary $\mathcal{A}_{\mathsf{NMCom}}$ that breaks the security of the non-malleable commitment scheme NMCom with non-negligible probability. Specifically, we will break the property of non-malleability with respect to extraction.

We now describe the working of $\mathcal{A}_{\mathsf{NMCom}}$ which interacts with the challenger $\mathcal{C}_{\mathsf{NMCom}}$. $\mathcal{A}_{\mathsf{NMCom}}$ picks randomly an honest party P_i and a random malicious party P_j. All messages other than the chosen NMCom messages are computed in the same manner as $\mathsf{Sim}_{\mathsf{Hyb}}$. The NMCom messages from P_i to P_j are exposed to the external challenger. Specifically, in round 1, set $\mathsf{nmcom}_1^{i\to j} := \mathsf{nmcom}_1^L$ where nmcom_1^L is received from $\mathcal{C}_{\mathsf{NMCom}}$ for the left execution. On receiving $\mathsf{nmcom}_1^{j\to i}$ from $\mathcal{A}$, $\mathcal{A}_{\mathsf{NMCom}}$ forwards this to $\mathcal{C}_{\mathsf{NMCom}}$ as its first round message on the right hand side.

$\mathcal{A}_{\mathsf{NMCom}}$ creates a set of 5 look-ahead threads, in each of which, it runs rounds 2 and 3 of the protocol alone. In each look-ahead thread, $\mathcal{A}_{\mathsf{NMCom}}$ computes $\mathsf{nmcom}_3^{i\to j}$ as a commitment to $\perp$. From the definition of the NMCom scheme, $\mathcal{A}_{\mathsf{NMCom}}$ can do this even without knowing the randomness used to generate $\mathsf{nmcom}_1^{i\to j}$. These 5 threads are all GOOD with respect to some party H with noticeable probability. With the 5 threads, $\mathcal{A}_{\mathsf{NMCom}}$ can successfully run the input and trapdoor extraction phase.

On the k-th thread $\mathcal{A}_{\mathsf{NMCom}}$ receives nmcom_2^R from $\mathcal{C}_{\mathsf{NMCom}}$ as the second round message on the right side which it sets as the value $\mathsf{nmcom}_2^{j\to i}$. On receiving $\mathsf{nmcom}_2^{i\to j}$ in the k-th thread, $\mathcal{A}_{\mathsf{NMCom}}$ sends this to $\mathcal{C}_{\mathsf{NMCom}}$ as its second round message on the left side along with the pair of values $(\perp, \mathsf{t}_j)$ where t_j was obtained during the extraction phase.

$\mathcal{A}_{\mathsf{NMCom}}$ receives a third round message nmcom_3^L which is either a commitment to $\perp$ or t_j. This is sent to $\mathcal{A}$ as the value $\mathsf{nmcom}_3^{i\to j}$ in the k-th thread.

We note that $\mathcal{A}_{\mathsf{NMCom}}$ acts as an interface for the $\mathsf{Ext}_{\mathsf{NMCom}}$, rewinding $\mathcal{A}$ as necessary.

By our assumption, the invariant doesn't hold. Thus $\mathsf{Ext}_{\mathsf{NMCom}}$, on adversary P_{j^*}'s commitment to P_i, outputs a valid trapdoor t_{i^*} for the trapdoor generation messages of the honest party P_{i^*} with non-negligible probability ε. With probability at least $\frac{1}{n^2}$, where n is the total number of players, this corresponds to honest party P_i and malicious party P_j picked randomly by $\mathcal{A}_{\mathsf{NMCom}}$. Therefore, with non-negligible probability $\frac{\varepsilon}{n^2}$, $\mathsf{Ext}_{\mathsf{NMCom}}$ outputs t^* as a valid trapdoor. Since the invariant holds in $\mathsf{Hyb}_{2,k-1,7}$, if $\mathsf{Ext}_{\mathsf{NMCom}}$ outputs t^*, it must be the case that we are in $\mathsf{Hyb}_{2,k,0}$ with non-negligible probability. That is, when $\mathsf{Ext}_{\mathsf{NMCom}}$ outputs a valid trapdoor, it must correspond to $\mathcal{A}_{\mathsf{NMCom}}$ receiving a commitment to 0. This breaks the security of NMCom, which is a contradiction. Thus the invariant must also hold for $\mathsf{Hyb}_{2,k,0}$.

□

Claim 17. *Assuming hiding of* NMCom*,* $\mathsf{Hyb}_{2,k-1,7}$ *is indistinguishable from* $\mathsf{Hyb}_{2,k,0}$

Proof. Since we are only making changes in a look-ahead thread, all we need to do is argue that the extraction continues to succeed. i.e. $\mathsf{Sim}_{\mathsf{Hyb}}$ does not output $\perp_{\mathsf{extract}}$ in the extraction phase of one hybrid but not the other. The only difference between $\mathsf{Hyb}_{2,k-1,7}$ and $\mathsf{Hyb}_{2,k,0}$ is that the simulator commits to the trapdoor in the k-th look ahead thread.

Since we have already established that the invariant holds in each look-ahead thread independently, we want to use the fact that the probability that the RWI proof for L_a is accepting cannot change with non-negligible probability if the invariant is true. If this were the case, the probability $\mathsf{Sim}_{\mathsf{Hyb}}$ outputs $\perp_{\mathsf{extract}}$ will not change in the extraction phase of the two hybrids, since L_a proves honest behavior of the first 3 rounds of the protocol.

Assume, for the sake of contradiction, that this isn't true. Then there exists an adversary $\mathcal{A}$ such that for some honest party P_{i^*} and malicious party P_{j^*}, $\mathcal{A}$ commits RWI proofs for L_a in Ecom such that the probability of accept in the two cases differs by a non-negligible probability. We will use this adversary to create an adversary $\mathcal{A}_{\mathsf{NMCom}}$ that breaks the hiding of the non-malleable commitment scheme NMCom with non-negligible probability.

We now describe the working of $\mathcal{A}_{\mathsf{NMCom}}$ which interacts with the challenger $\mathcal{C}_{\mathsf{NMCom}}$. $\mathcal{A}_{\mathsf{NMCom}}$ picks randomly an honest party P_i and a random malicious party P_j. All messages other than the chosen NMCom messages are computed in the same manner as $\mathsf{Sim}_{\mathsf{Hyb}}$. The NMCom messages from P_i to P_j are exposed to the external challenger. Specifically, in round 1, set $\mathsf{nmcom}_1^{i\to j} := \mathsf{nmcom}_1$ where nmcom_1 is received from $\mathcal{C}_{\mathsf{NMCom}}$.

$\mathcal{A}_{\mathsf{NMCom}}$ creates a set of 5 look-ahead threads, in each of which, it runs rounds 2 and 3 of the protocol alone. In each look-ahead thread, $\mathcal{A}_{\mathsf{NMCom}}$ computes $\mathsf{nmcom}_3^{i\to j}$ as a commitment to $\perp$. From the definition of the NMCom scheme, $\mathcal{A}_{\mathsf{NMCom}}$ can do this even without knowing the randomness used to generate $\mathsf{nmcom}_1^{i\to j}$. These 5 threads are all GOOD with respect to some party H with noticeable probability. With the 5 threads, $\mathcal{A}_{\mathsf{NMCom}}$ can successfully run the input and trapdoor extraction phase.

On receiving $\mathsf{nmcom}_1^{i\to j}$, $\mathcal{A}_{\mathsf{NMCom}}$ forwards it to $\mathcal{C}_{\mathsf{NMCom}}$ along with pair of values $(0, \mathsf{t}_j)$ where t_j was obtained during the extraction phase.

$\mathcal{A}_{\mathsf{NMCom}}$ receives a third round message nmcom_3^L which is either a commitment to 0 or t_j. This is sent to $\mathcal{A}$ as the value $\mathsf{nmcom}_3^{i\to j}$ on the k-th thread. On receiving the third round messages from $\mathcal{A}$, from 2 GOOD look ahead threads with respect to P_i, extract $\mathsf{rwi}_{a,3}^{j\to i}$ from Ecom[10]. From the definition of Ecom, the extracted value can be verified to be correctly extracted. $\mathcal{A}_{\mathsf{NMCom}}$ now checks if

$$\mathsf{RWI}_4\left(\mathsf{st}_a^{j\to i}, \mathsf{rwi}_{a,1}^{j\to i}, \mathsf{rwi}_{a,2}^{j\to i}, \mathsf{rwi}_{a,3}^{j\to i}; \mathsf{r}_{\mathsf{rwi},a}^{j\to i}\right) = 1.$$

If so, it guesses that the commitment was to 0. Otherwise, it guesses that the commitment was to t_j. Let us define Trap as the event that the commitment was to the trapdoor and $\overline{\mathsf{Trap}}$ as the even that the commitment was to $\perp$. From the challenge game, we know $\Pr[\mathsf{Trap}] = \Pr\left[\overline{\mathsf{Trap}}\right] = \frac{1}{2}$

$$\begin{aligned}
\Pr[\text{guess correct}] &= \Pr\left[\text{guess correct} \;\middle|\; \mathsf{Trap}\right] \cdot \Pr[\mathsf{Trap}] + \Pr\left[\text{guess correct} \;\middle|\; \overline{\mathsf{Trap}}\right] \cdot \Pr\left[\overline{\mathsf{Trap}}\right] \\
&= \Pr\left[\text{guess correct} \;\middle|\; \mathsf{Trap}\right] \cdot \frac{1}{2} + \Pr\left[\text{guess correct} \;\middle|\; \overline{\mathsf{Trap}}\right] \cdot \frac{1}{2} \\
&= \frac{1}{2} \cdot \left(\Pr\left[\mathsf{RWI} \text{ proof accepts} \;\middle|\; \mathsf{Trap}\right] + \Pr\left[\mathsf{RWI} \text{ proof rejects} \;\middle|\; \overline{\mathsf{Trap}}\right]\right) \\
&= \frac{1}{2} \cdot \left(\Pr\left[\mathsf{RWI} \text{ proof accepts} \;\middle|\; \mathsf{Trap}\right] + 1 - \Pr\left[\mathsf{RWI} \text{ proof accepts} \;\middle|\; \overline{\mathsf{Trap}}\right]\right) \\
&= \frac{1}{2} + \frac{1}{2} \cdot \left(\Pr\left[\mathsf{RWI} \text{ proof accepts} \;\middle|\; \mathsf{Trap}\right] - \Pr\left[\mathsf{RWI} \text{ proof accepts} \;\middle|\; \overline{\mathsf{Trap}}\right]\right)
\end{aligned}$$

By our assumption, the adversary P_{j^*}'s acceptance probability of the RWI proof for L_a to P_{i^*} differs non-negligible probability ε. With probability at least $\frac{1}{n^2}$, where n is the total number of players, this corresponds to honest party P_i and malicious party P_j picked randomly by $\mathcal{A}_{\mathsf{NMCom}}$. Therefore, P_j's acceptance probability of the RWI proof for L_a to P_i differs non-negligible probability $\frac{\varepsilon}{n^2}$. Now, the extractor $\mathsf{Ext}_{\mathsf{Ecom}}$ is successful with some non-negligible probability ε'. Therefore, with non-negligible advantage $\frac{\varepsilon\cdot\varepsilon'}{2\cdot n^2}$, $\mathcal{A}_{\mathsf{NMCom}}$ wins the challenge game with $\mathcal{C}_{\mathsf{NMCom}}$ which breaks the hiding property of NMCom. Thus, ε must be negligible, and thus the views are indistinguishable. □

Claim 18. *Assuming Assuming that* RWI *is a bounded rewinding secure protocol, and the existence of an extractor* $\mathsf{Ext}_{\mathsf{NMCom}}$*, the invariant holds in* $\mathsf{Hyb}_{2,k,1}$.

Proof. We know that the invariant holds $\mathsf{Hyb}_{2,k,0}$. The only difference between $\mathsf{Hyb}_{2,k,0}$ and $\mathsf{Hyb}_{2,k,1}$ is that the simulator switches the witness in the RWI for L_b. Assume, for the sake of contradiction, that the invariant doesn't hold in $\mathsf{Hyb}_{2,k,1}$. Then there exists an adversary $\mathcal{A}$ such that for some honest party P_{i^*} and malicious party P_{j^*}, $\mathcal{A}$ causes event E to occur with non-negligible probability. We will use this adversary to create an adversary $\mathcal{A}_{\mathsf{RWI}}$ that breaks the bounded rewinding security of RWI with non-negligible probability.

We now describe the working of $\mathcal{A}_{\mathsf{RWI}}$ which interacts with the challenger $\mathcal{C}_{\mathsf{RWI}}$. $\mathcal{A}_{\mathsf{RWI}}$ picks randomly an honest party P_i and a random malicious party P_j. All messages other than the chosen RWI messages are computed in the same manner as $\mathsf{Sim}_{\mathsf{Hyb}}$. The RWI messages from P_i to P_j are exposed to the external challenger. Specifically, in round 1, set $\mathsf{rwi}_{b,1}^{i\to j} := \mathsf{rwi}_1$ where rwi_1 is received from $\mathcal{C}_{\mathsf{RWI}}$.

After receiving $\mathsf{rwi}_{b,2}^{i\to j}$ from $\mathcal{A}$, $\mathcal{A}_{\mathsf{RWI}}$ creates a set of 5 look-ahead threads, in each of which, it runs rounds 2 and 3 of the protocol alone. In each look-ahead thread, $\mathcal{A}_{\mathsf{RWI}}$ on receiving $\mathsf{rwi}_{b,1}^{i\to j}$

[10]The extraction in fact does not require further rewinds since mask already extracted in the "Check Abort" phase. But for simplicity, we ignore this point for now.

forwards it to $\mathcal{C}_{\mathsf{RWI}}$ as its second round message. For each thread, $\mathcal{A}_{\mathsf{RWI}}$ also sends the statement

$$\mathsf{st}_b^{i\to j} := \left(\left\{ \mathsf{recom}_\ell^{i\to k}, \mathsf{ecom}_\ell^{i\to k}, \mathsf{nmcom}_\ell^{i\to k} \right\}_{\ell\in[3],k\in[n]\setminus\{i\}}, \left\{ \mathsf{msg}_{\ell,i} \right\}_{\ell\in[3]}, \left\{ \mathsf{rwi}_{a,\ell}^{i\to j} \right\}_{\ell\in[2]}, \mathsf{Trans}_2, \mathsf{td}_{1,j} \right)$$

where the other values are generated as in $\mathsf{Hyb}_{2,k,0}$.

In the main thread, $\mathcal{A}_{\mathsf{RWI}}$ also sends the pair of witnesses $\left(\left\{ \mathsf{r}_{\mathsf{ecom}}^{i\to k}, \mathsf{r}_{\mathsf{rwi},a}^{i\to j}, \mathsf{w}_a^{i\to k} \right\}_{k\in[n]\setminus\{i\}}, \bot, \bot \right)$ and $\left(\bot, \mathsf{t}_j, \mathsf{r}_{\mathsf{nmcom}}^{i\to j} \right)$ where t_j is obtained in the input extraction phase, and $\mathsf{w}_a^{i\to k}$ is computed as defined. For each thread, $\mathcal{A}_{\mathsf{RWI}}$ receives rwi_3 which is set as $\mathsf{rwi}_{b,3}^{i\to j}$.

Recall that RWI is secure even in the presence of 6 total threads. Now $\mathcal{A}_{\mathsf{RWI}}$ runs the extractor $\mathsf{Ext}_{\mathsf{NMCom}}$ of the non-malleable commitment scheme using the message in both the threads that correspond to the non-malleable commitment from malicious party P_j to honest party P_i. Let the output of $\mathsf{Ext}_{\mathsf{NMCom}}$ be t^*. $\mathcal{A}_{\mathsf{RWI}}$ checks if $\mathsf{TDValid}(t^*, \mathsf{td}_{1,i}) = 1$. If so, it outputs 1 to indicate $\mathsf{Hyb}_{2,k,1}$ and 0 otherwise. Let us denote this output by $\widetilde{b}$, and let the challenge bit be b. Then,

$$\begin{aligned}
\Pr\left[\widetilde{b} = b\right] &= \Pr\left[\widetilde{b} = 0 \;\middle|\; b = 0\right] \cdot \Pr\left[b = 0\right] + \Pr\left[\widetilde{b} = 1 \;\middle|\; b = 1\right] \cdot \Pr\left[b = 1\right] \\
&= \Pr\left[\widetilde{b} = 0 \;\middle|\; b = 0\right] \cdot \frac{1}{2} + \Pr\left[\widetilde{b} = 1 \;\middle|\; b = 1\right] \cdot \frac{1}{2} \\
&= \frac{1}{2} \cdot \left(1 - \Pr\left[\widetilde{b} = 1 \;\middle|\; b = 0\right] + \Pr\left[\widetilde{b} = 1 \;\middle|\; b = 1\right]\right) \\
&= \frac{1}{2} + \frac{1}{2} \cdot \left(\Pr\left[\widetilde{b} = 1 \;\middle|\; b = 1\right] - \Pr\left[\widetilde{b} = 1 \;\middle|\; b = 0\right]\right) \\
&= \frac{1}{2} + \frac{1}{2} \cdot \left(\Pr\left[\mathsf{EXT} \;\middle|\; b = 1\right] - \Pr\left[\mathsf{EXT} \;\middle|\; b = 0\right]\right)
\end{aligned}$$

where EXT denotes the even that the extractor outputs a valid trapdoor. By our assumption, the invariant doesn't hold. Thus $\mathsf{Ext}_{\mathsf{NMCom}}$, on adversary P_{j^*}'s commitment to P_i, outputs a valid trapdoor t_{i^*} for the trapdoor generation messages of the honest party P_{i^*} with non-negligible probability ε. With probability at least $\frac{1}{n^2}$, where n is the total number of players, this corresponds to honest party P_i and malicious party P_j picked randomly by $\mathcal{A}_{\mathsf{RWI}}$. Therefore, with non-negligible probability $\frac{\varepsilon}{n^2}$, $\mathsf{Ext}_{\mathsf{NMCom}}$ outputs t^* as a valid trapdoor. Since the invariant holds in $\mathsf{Hyb}_{2,k,0}$, if $\mathsf{Ext}_{\mathsf{NMCom}}$ outputs t^*, it must be the case that we are in $\mathsf{Hyb}_{2,k,1}$ with non-negligible probability. That is, when $\mathsf{Ext}_{\mathsf{NMCom}}$ outputs a valid trapdoor, it must correspond to $\mathcal{A}_{\mathsf{RWI}}$ receiving a proof using the trapdoor witness. This breaks the security of RWI, which is a contradiction. Thus the invariant must also hold for $\mathsf{Hyb}_{2,k,1}$. $\square$

Claim 19. *Assuming the bounded rewinding witness indistinguishability* RWI*,* $\mathsf{Hyb}_{2,k,0}$ *is indistinguishable from* $\mathsf{Hyb}_{2,k,1}$

Proof. Since we're only making changes in a look-ahead thread, all we need to do is argue that the extraction continues to succeed. i.e. $\mathsf{Sim}_{\mathsf{Hyb}}$ does not output $\bot_{\mathsf{extract}}$ in the extraction phase of one hybrid and not the other. The only difference between $\mathsf{Hyb}_{2,k,0}$ and $\mathsf{Hyb}_{2,k,1}$ is that the simulator switches the witness in the RWI for L_b.

Assume, for the sake of contradiction, that this isn't true. Then there exists an adversary $\mathcal{A}$ such that for some honest party P_{i^*} and malicious party P_{j^*}, $\mathcal{A}$ commits RWI proofs for L_b in Ecom such that the probability of accept in the two cases in non-negligible. We will use this adversary to create an adversary $\mathcal{A}_{\mathsf{RWI}}$ that breaks the bounded rewinding security of RWI with non-negligible probability.

The proof is similar to that of Claim 17 and Claim 18. We note that we use the fact that RWI is secure even in the presence of the 2 total threads used for extracting from Ecom. $\square$

Claim 20. *Assuming Assuming that* RWI *is a bounded rewinding secure protocol, and the existence of an extractor* $\mathsf{Ext}_{\mathsf{NMCom}}$*, the invariant holds in* $\mathsf{Hyb}_{2,k,2}$.

Proof. We know that the invariant holds $\mathsf{Hyb}_{2,k,1}$. The only difference between $\mathsf{Hyb}_{2,k,1}$ and $\mathsf{Hyb}_{2,k,2}$ is that the simulator switches the witness in the RWI for L_a. Assume, for the sake of contradiction, that the invariant doesn't hold in $\mathsf{Hyb}_{2,k,2}$. Then there exists an adversary $\mathcal{A}$ such that for some honest party P_{i^*} and malicious party P_{j^*}, $\mathcal{A}$ causes event E to occur with non-negligible probability. We will use this adversary to create an adversary $\mathcal{A}_{\mathsf{RWI}}$ that breaks the bounded rewinding security of RWI with non-negligible probability. The rest of the proof is similar to that of Claim 18. $\square$

Claim 21. *Assuming the bounded rewinding witness indistinguishability* RWI*,* $\mathsf{Hyb}_{2,k,1}$ *is indistinguishable from* $\mathsf{Hyb}_{2,k,2}$

Proof. Since we're only making changes in a look-ahead thread, all we need to do is argue that the extraction continues to succeed. i.e. $\mathsf{Sim}_{\mathsf{Hyb}}$ does not output $\bot_{\mathsf{extract}}$ in the extraction phase of one hybrid and not the other. The only difference between $\mathsf{Hyb}_{2,k,1}$ and $\mathsf{Hyb}_{2,k,2}$ is that the simulator switches the witness in the RWI for L_a.

Assume, for the sake of contradiction, that this isn't true. Then there exists an adversary $\mathcal{A}$ such that for some honest party P_{i^*} and malicious party P_{j^*}, $\mathcal{A}$ commits RWI proofs for L_a in Ecom such that the probability of accept in the two cases in non-negligible. We will use this adversary to create an adversary $\mathcal{A}_{\mathsf{RWI}}$ that breaks the bounded rewinding security of RWI with non-negligible probability.

The proof is similar to that of Claim 17 and Claim 18. $\square$

Claim 22. *Assuming that* Com *is a secure commitment scheme, and the existence of an extractor* $\mathsf{Ext}_{\mathsf{NMCom}}$*, the invariant holds in* $\mathsf{Hyb}_{2,k,3}$.

Proof. We prove this by a sequence of sub-claims.

Sub-Claim 23. *Assuming that* Com *is a secure commitment scheme, and the existence of an extractor* $\mathsf{Ext}_{\mathsf{NMCom}}$*, the invariant holds in* $\mathsf{Hyb}_{2,k,3,0}$.

Proof. We know that the invariant holds $\mathsf{Hyb}_{2,k,2}$. The only difference between $\mathsf{Hyb}_{2,k,2}$ and $\mathsf{Hyb}_{2,k,3,0}$ is that the simulator switches the commitment in Com from polynomials p to 0. This is in fact done by a sequence of hybrids where only a single Com is changed at a time. For simplicity, we proceed with the assumption that in this hybrid, only a single commitment was changed. Assume, for the sake of contradiction, that the invariant doesn't hold in $\mathsf{Hyb}_{2,k,3}$. Then there exists an adversary $\mathcal{A}$ such that for some honest party P_{i^*} and malicious party P_{j^*}, $\mathcal{A}$ causes event E to occur with non-negligible probability. We will use this adversary to create an adversary $\mathcal{A}_{\mathsf{Com}}$ that breaks the hiding property of Com with non-negligible probability.

We now describe the working of $\mathcal{A}_{\mathsf{Com}}$ which interacts with the challenger $\mathcal{C}_{\mathsf{Ecom}}$. $\mathcal{A}_{\mathsf{Com}}$ picks randomly an honest party P_i and a random malicious party P_j. All messages other than the chosen Com messages are computed in the same manner as $\mathsf{Sim}_{\mathsf{Hyb}}$. The Com messages from P_i to P_j are exposed to the external challenger. Specifically, $\mathcal{A}_{\mathsf{Com}}$ sends two challenges $(\mathsf{p}_\ell, 0)$ to $\mathcal{C}$. And sets $\mathsf{recom}_{1,\ell}^{i\to j} := \mathsf{com}$ where com is received from $\mathcal{C}_{\mathsf{Com}}$. Depending on the challenge used by $\mathcal{C}_{\mathsf{Com}}$, we are either in $\mathsf{Hyb}_{2,k,2}$ or $\mathsf{Hyb}_{2,k,3,0}$.

$\mathcal{A}_{\mathsf{Com}}$ creates sufficiently many look ahead threads where it runs rounds 2 and 3 of the protocol alone. Now $\mathcal{A}_{\mathsf{Com}}$ runs the extractor $\mathsf{Ext}_{\mathsf{NMCom}}$ of the non-malleable commitment scheme using the message in both the threads that correspond to the non-malleable commitment from malicious party P_j to honest party P_i. Let the output of $\mathsf{Ext}_{\mathsf{NMCom}}$ be t^*. $\mathcal{A}_{\mathsf{Com}}$ checks if $\mathsf{TDValid}(t^*, \mathsf{td}_{1,i}) = 1$. If so, it outputs 1 to indicate $\mathsf{Hyb}_{2,k,3}$ and 0 otherwise.

By our assumption, the invariant doesn't hold. Thus $\mathsf{Ext}_{\mathsf{NMCom}}$, on adversary P_{j^*}'s commitment to P_i, outputs a valid trapdoor t_{i^*} for the trapdoor generation messages of the honest party P_{i^*} with non-negligible probability ε. With probability at least $\frac{1}{n^2}$, where n is the total number of players, this corresponds to honest party P_i and malicious party P_j picked randomly by $\mathcal{A}_{\mathsf{Com}}$. Therefore, with non-negligible probability $\frac{\varepsilon}{n^2}$, $\mathsf{Ext}_{\mathsf{NMCom}}$ outputs t^* as a valid trapdoor. Since the invariant holds in $\mathsf{Hyb}_{2,k,2}$, if $\mathsf{Ext}_{\mathsf{NMCom}}$ outputs t^*, it must be the case that we're in $\mathsf{Hyb}_{2,k,3}$ with non-negligible probability. That is, when $\mathsf{Ext}_{\mathsf{NMCom}}$ outputs a valid trapdoor, it must correspond to $\mathcal{A}_{\mathsf{Com}}$ receiving input 0. This breaks the security of Com, which is a contradiction. Thus the invariant must also hold for $\mathsf{Hyb}_{2,k,3}$.

□

Sub-Claim 24. *The invariant holds in* $\mathsf{Hyb}_{2,k,3,1}$.

Proof. The change from is statistical $\mathsf{Hyb}_{2,k,3,0}$ when there are fewer than $\mathsf{B}_{\mathsf{recom}}$ rewinds when extracting from the NMCom. This follows from the fact that the degree of the polynomial is set to be $\mathsf{B}_{\mathsf{recom}}$, and thus statistically undetermined by the number of rewinds $\leq \mathsf{B}_{\mathsf{recom}}$. By our setting of parameters, we know that number of rewinds $\leq \mathsf{B}_{\mathsf{recom}}$. Thus The invariant holds in $\mathsf{Hyb}_{2,k,3,1}$. □

Sub-Claim 25. *Assuming that* Com *is a secure commitment scheme, and the existence of an extractor* $\mathsf{Ext}_{\mathsf{NMCom}}$*, the invariant holds in* $\mathsf{Hyb}_{2,k,3,2}$.

Proof. The proof follows identically as in Sub-Claim 23. □

Thus we have that the invariant holds for $\mathsf{Hyb}_{2,k,3}$.

□

Claim 26. *Assuming that* Com *is a secure commitment scheme,* $\mathsf{Hyb}_{2,k,2}$ *is indistinguishable from* $\mathsf{Hyb}_{2,k,3}$

Proof. Since we're only making changes in a look-ahead thread, all we need to do is argue that the extraction continues to succeed. i.e. $\mathsf{Sim}_{\mathsf{Hyb}}$ does not output $\perp_{\mathsf{extract}}$ in the extraction phase of one hybrid and not the other. The only difference between $\mathsf{Hyb}_{2,k,2}$ and $\mathsf{Hyb}_{2,k,3}$ is that the simulator switches commitment in RECom to 0.

The proof is similar to that of Claim 17 and Claim 22. □

Claim 27. *Assuming that* Π *is a rewinding secure protocol for the first three rounds, and the existence of an extractor* $\mathsf{Ext}_{\mathsf{NMCom}}$*, the invariant holds in* $\mathsf{Hyb}_{2,k,4}$.

Proof. We know that the invariant holds $\mathsf{Hyb}_{2,k,3}$. The only difference between $\mathsf{Hyb}_{2,k,3}$ and $\mathsf{Hyb}_{2,k,4}$ is that the simulator switches the input in Π from (x, r) to 0 for each honest party $\mathsf{P}_{\widehat{i}}$. This is in fact done by a sequence of hybrids where only a single party's input is changed at a time. For simplicity, we proceed with the assumption that in this hybrid, only a single party's input was changed. Assume, for the sake of contradiction, that the invariant doesn't hold in $\mathsf{Hyb}_{2,k,4}$. Then there exists an adversary $\mathcal{A}$ such that for some honest party P_{i^*} and malicious party P_{j^*}, $\mathcal{A}$ causes event E to occur with non-negligible probability. We will use this adversary to create an adversary $\mathcal{A}_\Pi$ that breaks the bounded rewinding security of the first three round of Π with non-negligible probability.

We now describe the working of $\mathcal{A}_\Pi$ which interacts with the challenger $\mathcal{C}_\Pi$. $\mathcal{A}_\Pi$ picks randomly an honest party P_i and a random malicious party P_j. All messages other than the chosen Π messages for $\mathsf{P}_{\widehat{i}}$ are computed in the same manner as $\mathsf{Sim}_{\mathsf{Hyb}}$. The Π messages for $\mathsf{P}_{\widehat{i}}$ are exposed to the external challenger. Specifically, in round 1, set $\mathsf{msg}_{1,\widehat{i}} := \mathsf{msg}_1$ where msg_1 is received from $\mathcal{C}_\Pi$.

After generating/receiving Trans_1 from $\mathcal{A}$, $\mathcal{A}_\Pi$ creates a set of 5 look-ahead threads, in each of which, it runs rounds 2 and 3 of the protocol alone. In each look-ahead thread, $\mathcal{A}_\Pi$ on computing Trans_2 forwards it to $\mathcal{C}_\Pi$.

In the main thread (k-th look-ahead thread), $\mathcal{A}_\Pi$ also sends the pair of inputs $(\mathsf{x}_i, \mathsf{r}_i)$ and 0. For the look-ahead threads for extraction, $\mathcal{A}_\Pi$ sends the input $(\mathsf{x}_i, \mathsf{r}_i)$. For each thread, $\mathcal{A}_\Pi$ receives msg_3 which is set as $\mathsf{msg}_{3,\widehat{i}}$. Depending on the input used by $\mathcal{C}_\Pi$, we are either in $\mathsf{Hyb}_{2,k,3}$ or $\mathsf{Hyb}_{2,k,4}$.

Recall that Π is secure even in the presence of 3 total threads. Now $\mathcal{A}_\Pi$ runs the extractor $\mathsf{Ext}_{\mathsf{NMCom}}$ of the non-malleable commitment scheme using the message in both the threads that correspond to the non-malleable commitment from malicious party P_j to honest party P_i. Let the output of $\mathsf{Ext}_{\mathsf{NMCom}}$ be t^*. $\mathcal{A}_\Pi$ checks if $\mathsf{TDValid}(t^*, \mathsf{td}_{1,i}) = 1$. If so, it outputs 1 to indicate $\mathsf{Hyb}_{2,k,4}$ and 0 otherwise.

By our assumption, the invariant doesn't hold. Thus $\mathsf{Ext}_{\mathsf{NMCom}}$, on adversary P_{j^*}'s commitment to P_i, outputs a valid trapdoor t_{i^*} for the trapdoor generation messages of the honest party P_{i^*} with non-negligible probability ε. With probability at least $\frac{1}{n^2}$, where n is the total number of players, this corresponds to honest party P_i and malicious party P_j picked randomly by $\mathcal{A}_\Pi$. Therefore, with non-negligible probability $\frac{\varepsilon}{n^2}$, $\mathsf{Ext}_{\mathsf{NMCom}}$ outputs t^* as a valid trapdoor. Since the invariant holds in $\mathsf{Hyb}_{2,k,3}$, if $\mathsf{Ext}_{\mathsf{NMCom}}$ outputs t^*, it must be the case that we're in $\mathsf{Hyb}_{2,k,4}$ with non-negligible probability. That is, when $\mathsf{Ext}_{\mathsf{NMCom}}$ outputs a valid trapdoor, it must correspond to $\mathcal{A}_\Pi$ receiving the messages using input 0. This breaks the security of Π, which is a contradiction. Thus the invariant must also hold for $\mathsf{Hyb}_{2,k,4}$.

□

Claim 28. *Assuming that Π is a rewinding secure protocol for the first three rounds,* $\mathsf{Hyb}_{2,k,3}$ *is indistinguishable from* $\mathsf{Hyb}_{2,k,4}$

Proof. Since we're only making changes in a look-ahead thread, all we need to do is argue that the extraction continues to succeed. i.e. $\mathsf{Sim}_{\mathsf{Hyb}}$ does not output $\perp_{\mathsf{extract}}$ in the extraction phase of one hybrid and not the other. The only difference between $\mathsf{Hyb}_{2,k,3}$ and $\mathsf{Hyb}_{2,k,4}$ is that the simulator switches input to 0.

Assume, for the sake of contradiction, that this isn't true. Then there exists an adversary $\mathcal{A}$ such that for some honest party P_{i^*} and malicious party P_{j^*}, $\mathcal{A}$ commits RWI proofs for L_a in Ecom such that the probability of accept in the two cases in non-negligible. We will use this adversary to create an adversary $\mathcal{A}_\Pi$ that breaks the bounded rewinding security of Π with non-negligible probability.

The proof is similar to that of Claim 17 and Claim 27. □

Claim 29. *Assuming Assuming that* RWI *is a bounded rewinding secure protocol, and the existence of an extractor* $\mathsf{Ext}_{\mathsf{NMCom}}$*, the invariant holds in* $\mathsf{Hyb}_{2,k,5}$.

Proof. Proof is identical to that of Claim 20. □

Claim 30. *Assuming the bounded rewinding witness indistinguishability* RWI*,* $\mathsf{Hyb}_{2,k,4}$ *is indistinguishable from* $\mathsf{Hyb}_{2,k,5}$

Proof. Proof is identical to that of Claim 21. □

Claim 31. *Assuming Assuming that* RWI *is a bounded rewinding secure protocol, and the existence of an extractor* $\mathsf{Ext}_{\mathsf{NMCom}}$*, the invariant holds in* $\mathsf{Hyb}_{2,k,6}$*.*

Proof. Proof is identical to that of Claim 18. □

Claim 32. *Assuming the bounded rewinding witness indistinguishability* RWI*,* $\mathsf{Hyb}_{2,k,5}$ *is indistinguishable from* $\mathsf{Hyb}_{2,k,6}$

Proof. Proof is identical to that of Claim 19. □

Claim 33. *Assuming* NMCom *is a secure non-malleable commitment scheme with respect to extraction, the invariant holds in* $\mathsf{Hyb}_{2,k,7}$*.*

Proof. Proof is identical to that of Claim 16. □

Claim 34. *Assuming* NMCom *is a secure non-malleable commitment scheme,* $\mathsf{Hyb}_{2,k,6}$ *is indistinguishable from* $\mathsf{Hyb}_{2,k,7}$

Proof. Proof is identical to that of Claim 17. □

Claim 35. *Assuming* NMCom *is a secure non-malleable commitment scheme with respect to extraction, the invariant holds in* Hyb_3*.*

Proof. Proof is identical to that of Claim 16. □

Claim 36. *Assuming* NMCom *is a secure non-malleable commitment scheme,* Hyb_3 *is indistinguishable from* Hyb_2

Proof. The only difference between Hyb_3 and Hyb_2 is that the simulator commits to the trapdoor in the main look ahead thread.

Assume, for the sake of contradiction, that that the views are distinguishable. Then there exists an adversary $\mathcal{D}$ such that $\mathcal{D}$ can distinguish between Hyb_3 and Hyb_2 with non-negligible advantage. We will use this adversary to create an adversary $\mathcal{A}_{\mathsf{NMCom}}$ that breaks the hiding of the non-malleable commitment scheme NMCom with non-negligible probability.

We now describe the working of $\mathcal{A}_{\mathsf{NMCom}}$ which interacts with the challenger $\mathcal{C}_{\mathsf{NMCom}}$. $\mathcal{A}_{\mathsf{NMCom}}$ picks randomly an honest party P_i and a random malicious party P_j. All messages other than the chosen NMCom messages are computed in the same manner as $\mathsf{Sim}_{\mathsf{Hyb}}$. The NMCom messages from P_i to P_j are exposed to the external challenger. Specifically, in round 1, set $\mathsf{nmcom}_1^{i\to j} := \mathsf{nmcom}_1$ where nmcom_1 is received from $\mathcal{C}_{\mathsf{NMCom}}$.

$\mathcal{A}_{\mathsf{NMCom}}$ creates a set of 5 look-ahead threads, in each of which, it runs rounds 2 and 3 of the protocol alone. In each look-ahead thread, $\mathcal{A}_{\mathsf{NMCom}}$ computes $\mathsf{nmcom}_3^{i\to j}$ as a commitment to $\perp$. From the definition of the NMCom scheme, $\mathcal{A}_{\mathsf{NMCom}}$ can do this even without knowing the randomness used to generate $\mathsf{nmcom}_1^{i\to j}$. These 5 threads are all GOOD with respect to some party H with noticeable probability. With the 5 threads, $\mathcal{A}_{\mathsf{NMCom}}$ can successfully run the input and trapdoor extraction phase.

On receiving $\mathsf{nmcom}_1^{i\to j}$, $\mathcal{A}_{\mathsf{NMCom}}$ forwards it to $\mathcal{C}_{\mathsf{NMCom}}$ along with pair of values $(\perp, \mathsf{t}_j)$ where t_j was obtained during the extraction phase.

$\mathcal{A}_{\mathsf{NMCom}}$ receives a third round message nmcom_3^L which is either a commitment to $\perp$ or t_j. This is sent to $\mathcal{A}$ as the value $\mathsf{nmcom}_3^{i\to j}$ on the main thread. The rest of the messages are obtained in the same manner as $\mathsf{Sim}_{\mathsf{Hyb}}$. Depending on which value was committed we are either in Hyb_3 or Hyb_2. On completion of the execution, the view is input to $\mathcal{D}$ and the output returned is the output of $\mathcal{A}_{\mathsf{NMCom}}$

By our assumption, $\mathcal{D}$ can distinguish between the two hybrids with noticeable probability ε. Therefore, with non-negligible advantage $\frac{\varepsilon}{n^2}$, $\mathcal{A}_{\mathsf{NMCom}}$ wins the challenge game with $\mathcal{C}_{\mathsf{NMCom}}$ which breaks the hiding property of NMCom. Thus, ε must be negligible, and thus the views are indistinguishable. □

Claim 37. *Assuming the bounded rewinding witness indistinguishability* RWI*, the invariant holds in* Hyb_4.

Proof. Proof is identical to that of Claim 18. □

Claim 38. *Assuming the bounded rewinding witness indistinguishability* RWI*,* Hyb_4 *is indistinguishable from* Hyb_3

Proof. The only difference between Hyb_4 and Hyb_3 is that the simulator switches the witness in the RWI for L_b.

Assume, for the sake of contradiction, that this isn't true. Then there exists an adversary $\mathcal{D}$ can distinguish between Hyb_4 and Hyb_3 with non-negligible advantage. We will use this adversary to create an adversary $\mathcal{A}_{\mathsf{RWI}}$ that breaks the bounded rewinding security of RWI with non-negligible probability.

The proof is similar to that of Claim 36 and Claim 37. □

Claim 39. *Assuming the bounded rewinding witness indistinguishability* RWI*, the invariant holds in* Hyb_5.

Proof. Proof is identical to that of Claim 20. □

Claim 40. *Assuming the bounded rewinding witness indistinguishability* RWI*,* Hyb_5 *is indistinguishable from* Hyb_4

Proof. The only difference between Hyb_5 and Hyb_4 is that the simulator switches the witness in the RWI for L_a.

Assume, for the sake of contradiction, that this isn't true. Then there exists an adversary $\mathcal{D}$ can distinguish between Hyb_5 and Hyb_4 with non-negligible advantage. We will use this adversary to create an adversary $\mathcal{A}_{\mathsf{RWI}}$ that breaks the bounded rewinding security of RWI with non-negligible probability.

The proof is similar to that of Claim 36 and Claim 37. □

Claim 41. *The invariant holds in* Hyb_6.

Proof. The claim is trivially true since the change is made only in the fourth round. □

Claim 42. *Assuming the witness indistinguishability* WI*,* Hyb_6 *is indistinguishable from* Hyb_5

Proof. The only difference between Hyb_6 and Hyb_5 is that the simulator switches the witness in the WI for L_c.

Assume, for the sake of contradiction, that this isn't true. Then there exists an adversary $\mathcal{D}$ can distinguish between Hyb_6 and Hyb_5 with non-negligible advantage. We will use this adversary to create an adversary $\mathcal{A}_{\mathsf{WI}}$ that breaks the witness indistinguishability of WI with non-negligible probability.

The proof is similar to that of Claim 36 and Claim 37. We point out that since only the second round of WI overlaps with the rewinding rounds, we don't need the external challenger to handle rewinds since the responses on the look-ahead threads, that are run only till the end of third round, are discarded. □

Claim 43. *Assuming the rewinding security of* RECom*,* Hyb_7 *is indistinguishable from* Hyb_6

Proof. This is proved via a sequence of hybrids given below.

This is done by a sequence of hybrids mentioned below. We note that we separate the look-ahead threads into two separate types: (i) to extract trapdoor, (ii) to extract input. In our hybrids, we shall only make changes to type (ii) threads.

$\mathsf{Hyb}_{7,0}$: **Change main thread RECom to random:** In this hybrid, $\mathsf{Sim}_{\mathsf{Hyb}}$ modifies the third round of the main thread to send "junk" responses. Specifically, for every honest party P_i and malicious party P_j do the following:

- for every $\ell \in [N]$, pick a new degree 4 polynomial q_ℓ.
- compute $\mathsf{recom}_{3,\ell}$ as $(0 \oplus \mathsf{q}_\ell(0), \mathsf{q}_\ell(\mathsf{z}_\ell))$.

Given that we changed our RECom to random, we want to claim that the adversary's input has not also become random.

Claim 44. *Assuming the security of* Com *and the existence of* $\mathsf{Ext}_{\mathsf{NMCom}}$*, the invariant holds in* $\mathsf{Hyb}_{7,0}$*.*

Proof. We prove that the invariant holds in the look-ahead threads that we make the changes in. We know that the invariant holds Hyb_6. The only difference between Hyb_6 and $\mathsf{Hyb}_{7,0}$ is that the simulator uses random polynomials to compute the third round messages of RECom on the main thread. An alternate way to think of this is that either the polynomials used inside Com and that used to compute the third round of RECom are the same, or they're independently sample random polynomials. Thus we think of the change as $\mathsf{Sim}_{\mathsf{Hyb}}$ switching the commitment in Com from polynomials p to q while using p to compute the third round of RECom. This is in fact done by a sequence of hybrids where only a single Com is changed at a time. For simplicity, we proceed with the assumption that in this hybrid, only a single commitment was changed. Assume, for the sake of contradiction, that the invariant doesn't hold in $\mathsf{Hyb}_{7,0}$. Then there exists an adversary $\mathcal{A}$ such that for some honest party P_{i^*} and malicious party P_{j^*}, $\mathcal{A}$ causes event E to occur with non-negligible probability. We will use this adversary to create an adversary $\mathcal{A}_{\mathsf{Com}}$ that breaks the hiding property of Com with non-negligible probability.

We now describe the working of $\mathcal{A}_{\mathsf{Com}}$ which interacts with the challenger $\mathcal{C}_{\mathsf{Com}}$. $\mathcal{A}_{\mathsf{Com}}$ picks randomly an honest party P_i and a random malicious party P_j. All messages other than the chosen Com messages are computed in the same manner as $\mathsf{Sim}_{\mathsf{Hyb}}$. The Com messages from P_i to P_j are exposed to the external challenger. Specifically, $\mathcal{A}_{\mathsf{Com}}$ sends two challenges $(\mathsf{p}_\ell, \mathsf{q}_\ell)$ to $\mathcal{C}$. And sets $\mathsf{recom}_{1,\ell}^{i\to j} := \mathsf{com}$ where com is received from $\mathcal{C}_{\mathsf{Com}}$. Depending on the challenge used by $\mathcal{C}_{\mathsf{Com}}$, we are either in Hyb_6 or $\mathsf{Hyb}_{7,0}$.

$\mathcal{A}_{\mathsf{Com}}$ creates 2 look ahead threads where it runs rounds 2 and 3 of the protocol alone. Now $\mathcal{A}_{\mathsf{Com}}$ runs the extractor $\mathsf{Ext}_{\mathsf{NMCom}}$ of the non-malleable commitment scheme using the message in both the threads that correspond to the non-malleable commitment from malicious party P_j to honest party P_i. Let the output of $\mathsf{Ext}_{\mathsf{NMCom}}$ be t^*. $\mathcal{A}_{\mathsf{Com}}$ checks if $\mathsf{TDValid}(t^*, \mathsf{td}_{1,i}) = 1$. If so, it outputs 1 to indicate $\mathsf{Hyb}_{7,2}$ and 0 otherwise.

By our assumption, the invariant doesn't hold. Thus $\mathsf{Ext}_{\mathsf{NMCom}}$, on adversary P_{j^*}'s commitment to P_i, outputs a valid trapdoor t_{i^*} for the trapdoor generation messages of the honest party P_{i^*} with non-negligible probability ε. With probability at least $\frac{1}{n^2}$, where n is the total number of players, this corresponds to honest party P_i and malicious party P_j picked randomly by $\mathcal{A}_{\mathsf{Com}}$. Therefore, with non-negligible probability $\frac{\varepsilon}{n^2}$, $\mathsf{Ext}_{\mathsf{NMCom}}$ outputs t^* as a valid trapdoor. Since the invariant holds in Hyb_0, if $\mathsf{Ext}_{\mathsf{NMCom}}$ outputs t^*, it must be the case that we're in $\mathsf{Hyb}_{7,2}$ with non-negligible probability. That is, when $\mathsf{Ext}_{\mathsf{NMCom}}$ outputs a valid trapdoor, it must correspond to $\mathcal{A}_{\mathsf{Com}}$ receiving the challenge using input q. This breaks the security of Com, which is a contradiction. Thus the invariant must also hold for $\mathsf{Hyb}_{7,2}$.

This works because as long as the number of threads created to extract from NMCom is less than B_{recom}, which is in fact true, since otherwise, the "random" polynomial no longer appears random. It should be noted that we don't need to extract the adversary's input for the reduction, and thus no use of creating any Type (ii) threads. □

Claim 45. *Assuming the security of* Com, $\mathsf{Hyb}_{7,0}$ *is indistinguishable from* Hyb_6

Proof. Since we're only making changes in a look-ahead thread, all we need to do is argue that the adversary doesn't switch to "junk" commitments when we make the change. The only difference between Hyb_6 and $\mathsf{Hyb}_{7,0}$ is that the simulator uses random polynomials to compute the third round messages of RECom look-ahead threads.

Assume, for the sake of contradiction, that this isn't true. Then there exists an adversary $\mathcal{A}$ such that for some honest party P_{i^*} and malicious party P_{j^*}, $\mathcal{A}$ commits RWI proofs for L_a in Ecom such that the probability of accept in the two cases in non-negligible. We will use this adversary to create an adversary $\mathcal{A}_{\mathsf{Com}}$ that breaks the security of Com with non-negligible probability.

The proof is similar to that of Claim 17 and Claim 44. □

$\mathsf{Hyb}_{7,1}$: **Create Type (ii) look-ahead thread:** In this hybrid, $\mathsf{Sim}_{\mathsf{Hyb}}$ creates Type (ii) threads that are identical to the main thread. These will be used to extract the adversary's input. We create as many needed for the extraction of the adversary's input.

Claim 46. *Assuming the security of* Com *and the existence of* $\mathsf{Ext}_{\mathsf{NMCom}}$*, the invariant holds in* $\mathsf{Hyb}_{7,1}$.

Proof. This trivially follows from the fact that invariant holds in $\mathsf{Hyb}_{7,0}$ are identical to the main thread. □

Claim 47. *Assuming the security of* Com, $\mathsf{Hyb}_{7,1}$ *is indistinguishable from* Hyb_6

Proof. This follows as in the proof of Claim 45. □

$\mathsf{Hyb}_{7,2}$: **Change main thread RECom to 0:** In this hybrid, $\mathsf{Sim}_{\mathsf{Hyb}}$ modifies the third round of the main thread to commit to 0. Specifically, for every honest party P_i and malicious party P_j do the following:

– compute $\mathsf{recom}_{3,\ell}$ as $(0 \oplus \mathsf{p}_\ell(0), \mathsf{poly}(\lambda)_\ell(\mathsf{z}_\ell)$.

where p_ℓ are the polynomials committed to in the first round.

Claim 48. *Assuming the security of* Com *and the existence of* $\mathsf{Ext}_{\mathsf{NMCom}}$*, the invariant holds in* $\mathsf{Hyb}_{7,2}$.

Proof. The proof follows as in 44. □

Claim 49. *Assuming the security of* Com*,* $\mathsf{Hyb}_{7,2}$ *is indistinguishable from* Hyb_6

Proof. The proof follows as in 45 □

Note that $\mathsf{Hyb}_{7,2} \equiv \mathsf{Hyb}_7$

Thus Hyb_7 is indistinguishable from Hyb_6. □

Claim 50. *Assuming that* Π *is a secure protocol instantiated with rewinding secure* ROT*, hiding of* OT *against malicious senders, hiding of* Ecom*, bounded rewind witness indistinguishability of* RWI*,* Hyb_8 *is indistinguishable from* Hyb_7

Proof. This is proved via a sequence of hybrids. The reasoning behind this sub-division is that if we directly make changes to the protocol on the main thread, a subtle issue shows up during reduction. Namely, the look ahead threads currently employ an honest strategy using the inputs 0 for the underlying MPC. Additionally, they use the same first round message as the main thread. Although, no proof is sent in the clear, honest behavior is proven via the commitment and the OT receiver message on these look ahead thread, which requires knowledge of the randomness used for the underlying MPC. This is problematic during a reduction to the security of the underlying MPC. We will utilize the fact that our proofs are not sent in the clear to get around this issue.

Recall that we separate out the look ahead threads based on their purpose. Type (i) look-ahead threads are used to extract the trapdoor, while the type (ii) look-ahead threads are used to extract the inputs.

$\mathsf{Hyb}_{8,0}$: **Change type (i) threads** Ecom **to 0:** In this hybrid, $\mathsf{Sim}_{\mathsf{Hyb}}$ modifies the third round of the type (i) threads to commit to 0 instead of commitment to RWI third round messages corresponding to L_a.

Claim 51. *Assuming the hiding of* Ecom*,* $\mathsf{Hyb}_{8,0}$ *is indistinguishable from* Hyb_7

Proof. Since we're making changes only to type (i) threads used to extract trapdoor, we need to ensure we're still able to extract the trapdoor with this change. This can be done without having to rewind to the extract the trapdoor. On receiving the third round of the adversary's message on these threads, we can use TDOut to check if the trapdoor messages sent by the adversary satisfy validity. If there is a noticeable change in the validity condition being satisfied, we can break the hiding property of Ecom.

Assume there exists an adversary $\mathcal{D}$ that results in the trapdoor validity check being passed in $\mathsf{Hyb}_{8,0}$ and Hyb_7 with non-negligible difference. We will use this to create an adversary $\mathcal{A}_{\mathsf{Ecom}}$ to break the hiding property of Ecom with non-negligible probability. Note that we make changes to these threads one at a time.

$\mathcal{A}_{\mathsf{Ecom}}$ picks randomly an honest party P_i and a random malicious party P_j. All messages in the main and look ahead threads other than the Ecom messages from P_i to P_j on the changed look ahead thread are identical. The Ecom messages from P_i to P_j are exposed to the external challenger. The first two rounds are forwarded to and from the adversary. Then, $\mathcal{A}_{\mathsf{Ecom}}$ sends to the challenger the pair of values $(0, \mathsf{rwi}_{a,3}^{i \to j})$

$\mathcal{A}_{\mathsf{Ecom}}$ receives a third round message ecom_3 which is either a commitment to 0 or $\mathsf{rwi}_{a,3}^{i \to j}$. This is sent to $\mathcal{A}$ as the third round message. The rest of the messages are obtained in the same manner as $\mathsf{Sim}_{\mathsf{Hyb}}$. Depending on which value was committed we are either in $\mathsf{Hyb}_{8,0}$ or Hyb_7. On completion of the execution, check the validity condition of the trapdoor messages sent using TDOut. If the validity check passes, output 0 (to indicate we're in Hyb_7), else output 1.

By our assumption, $\mathcal{D}$ results in non-negligible difference in the validity condition being verified it the two hybrids with noticeable probability difference ε. Therefore, with non-negligible advantage $\frac{\varepsilon}{n^2}$, $\mathcal{A}_{\mathsf{Ecom}}$ wins the challenge game with $\mathcal{C}_{\mathsf{Ecom}}$ which breaks the hiding property of Ecom. Thus, ε must be negligible, and thus we continue to extract the trapdoor.

Since we continue to extract trapdoor, $\mathsf{Hyb}_{8,0}$ is indistinguishable from Hyb_7.

□

Note that we don't need to argue invariant here to argue indistinguishability since the look ahead threads we extract inputs from are unchanged.

$\mathsf{Hyb}_{8,1}$: **Change Type (i) threads receiver input to 0 in OT:** In this hybrid, $\mathsf{Sim}_{\mathsf{Hyb}}$ modifies the third round of the type (i) threads to use receiver input 0 in OT instead of the third round messages of RWI corresponding to L_b.

Claim 52. *Assuming the hiding of* OT *against malicious senders,* $\mathsf{Hyb}_{8,1}$ *is indistinguishable from* $\mathsf{Hyb}_{8,0}$

Proof. The proofs follows identically as in Claim 51. The only difference being the now the OT messages are exposed to the external challenger. Since we're still able to extract the trapdoor, $\mathsf{Hyb}_{8,1}$ is indistinguishable from $\mathsf{Hyb}_{8,0}$.

□

We now make changes to look ahead threads of Type (ii)

$\mathsf{Hyb}_{8,2}$: **Switch RWI proofs for L_a on Type (ii) threads:** In this hybrid, $\mathsf{Sim}_{\mathsf{Hyb}}$ modifies the third round of the type (ii) threads to switch to the "trapdoor witness" in the RWI proofs for L_a. This is done a single thread at a time.

We need to ensure that we still continue extracting the inputs of the adversary. This is done by proving the invariant holds in each of the threads when we make this change.

Claim 53. *Assuming bounded rewind witness indistinguishability of* RWI*, and the existence of an extractor* $\mathsf{Ext}_{\mathsf{NMCom}}$ *the invariant holds in* $\mathsf{Hyb}_{8,2}$ *.*

Proof. The proof follows identically as in Claim 18. □

Claim 54. *Assuming bounded rewind witness indistinguishability of* RWI*,* $\mathsf{Hyb}_{8,2}$ *is indistinguishable from* $\mathsf{Hyb}_{8,1}$*.*

Proof. The proof follows identically as in Claim 19. □

$\mathsf{Hyb}_{8,3}$: **Switch RWI proofs for L_b on Type (ii) threads:** In this hybrid, $\mathsf{Sim}_{\mathsf{Hyb}}$ modifies the third round of the type (ii) threads to switch to the "trapdoor witness" in the RWI proofs for L_b. This is done a single thread at a time.

Claim 55. *Assuming bounded rewind witness indistinguishability of* RWI*, and the existence of an extractor* $\mathsf{Ext}_{\mathsf{NMCom}}$ *the invariant holds in* $\mathsf{Hyb}_{8,3}$ *.*

Proof. The proof follows identically as in Claim 18. □

Claim 56. *Assuming bounded rewind witness indistinguishability of* RWI*,* $\mathsf{Hyb}_{8,3}$ *is indistinguishable from* $\mathsf{Hyb}_{8,2}$*.*

Proof. The proof follows identically as in Claim 19. □

$\mathsf{Hyb}_{8,4}$: **Switch RWI proofs for L_b on Type (ii) threads:** In this hybrid, $\mathsf{Sim}_{\mathsf{Hyb}}$ modifies the third round of the type (ii) threads to switch to the "trapdoor witness" in the RWI proofs for L_b. This is done a single thread at a time.
$\mathsf{Hyb}_{8,5}$: **Simulate Π on main thread:** In this hybrid, $\mathsf{Sim}_{\mathsf{Hyb}}$ modifies the transcript of the underlying protocol Π. For a complete description of the changes, refer to the description of Hyb_8.

Claim 57. *Assuming that Π is a secure protocol instantiated with rewinding secure* ROT*, the invariant holds in* $\mathsf{Hyb}_{8,5}$*.*

Proof. Here, since the invariant only depends on the first three rounds, we need to prove that the invariant holds conditioned on the view of the first three rounds. The proof is similar to Claim 27. □

Claim 58. *Assuming that Π is a secure protocol instantiated with rewinding secure* ROT*,* $\mathsf{Hyb}_{8,5}$ *is indistinguishable from* $\mathsf{Hyb}_{8,4}$*.*

Proof. The only difference between $\mathsf{Hyb}_{8,5}$ and $\mathsf{Hyb}_{8,4}$ is how the transcript of the underlying protocol Π is computed.

Assume, for the sake of contradiction, that that the views are distinguishable. Then there exists an adversary $\mathcal{D}$ such that $\mathcal{D}$ can distinguish between $\mathsf{Hyb}_{8,5}$ and $\mathsf{Hyb}_{8,4}$ with non-negligible advantage. We will use this adversary to create an adversary $\mathcal{A}_\Pi$ that breaks the indistinguishability of Π when instantiated with bounded rewinding secure ROT with non-negligible probability. Essentially, we rely on the fact that if the ROT is rewinding secure, then the transcript of Π for an honest and simulated transcript are indistinguishable.

We now describe the working of $\mathcal{A}_\Pi$ which interacts with the challenger $\mathcal{C}_\Pi$. All messages other than the Π messages are computed in the same manner as $\mathsf{Sim}_{\mathsf{Hyb}}$. The Π messages from are exposed to the external challenger. Specifically, in round 1, set $\left\{\mathsf{msg}_{1,i}\right\}_{\mathsf{P}_i\in\mathcal{H}} := \overrightarrow{\mathsf{msg}}_1$ where $\overrightarrow{\mathsf{msg}}_1$ is received from $\mathcal{C}_\Pi$. Send to $\mathcal{C}_\Pi$ $\left\{\mathsf{msg}_{1,i}\right\}_{\mathsf{P}_i\notin\mathcal{H}}$ that is sent by $\mathcal{A}$. The response from $\mathcal{C}_\Pi$, $\overrightarrow{\mathsf{msg}}_2$ is parsed as $\left\{\mathsf{msg}_{2,i}\right\}_{\mathsf{P}_i\in\mathcal{H}} := \overrightarrow{\mathsf{msg}}_2$.

$\mathcal{A}_\Pi$ then creates a set of 3 type (i) look-ahead threads, in each of which, it runs rounds 2 and 3 of the protocol alone. This is done to extract the trapdoor. This doesn't require generating proofs on these look-ahead threads. Now, with the extracted trapdoor, $\mathcal{A}_\Pi$ then creates a set of 5 type (ii) look-ahead threads, in each of which, it runs rounds 2 and 3 of the protocol alone. In each look-ahead thread, $\mathcal{A}_\Pi$ forwards the $\left\{\mathsf{msg}_{2,i}\right\}_{\mathsf{P}_i\notin\mathcal{H}}$ sent by $\mathcal{A}$ in each look-ahead thread to $\mathcal{C}_\Pi$. These are simply ROT messages and will be responded to by $\mathcal{C}_\Pi$. The response is likewise forwarded to $\mathcal{A}$. These 5 threads are all GOOD with respect to some party H with noticeable probability. With the 5 threads, $\mathcal{A}_\Pi$ can successfully run the extraction phase. Note that in these threads, $\mathcal{A}_\Pi$ can use the "trapdoor witness" extracted using the type (i) threads.

On completion of the extraction phase, prior to the third round on the main thread, $\mathcal{A}_\Pi$ sends to $\mathcal{C}_\Pi$ all parties inputs $(\{\mathsf{x}_i, \mathsf{r}_i\}_{i\in[n]}, y)$ to $\mathcal{C}_\Pi$. $\mathcal{C}_\Pi$ then either responds with the simulated last message or the honest execution for the rest of the transcript. The rest of the messages are obtained in the same manner as $\mathsf{Sim}_{\mathsf{Hyb}}$. Depending on the choice of $\mathcal{C}_\Pi$ we are either in Hyb_8 or Hyb_7. On completion of the execution, the view is input to $\mathcal{D}$ and the output returned is the output of $\mathcal{A}_\Pi$

By our assumption, $\mathcal{D}$ can distinguish between the two hybrids with non-negligible probability ε. Therefore, with non-negligible advantage ε, $\mathcal{A}_\Pi$ wins the challenge game with $\mathcal{C}_\Pi$ which breaks the security of Π when rewinding security of ROT is maintained. Thus, ε must be negligible, and thus the views are indistinguishable.

□

Now we undo the changes made to the look ahead threads. The proofs follow identically as argued above, and are skipped.

$\mathsf{Hyb}_{8,6}$: **Switch RWI proofs for L_b on Type (ii) threads:** In this hybrid, $\mathsf{Sim}_{\mathsf{Hyb}}$ modifies the third round of the type (ii) threads to switch back to the "real witness" in the RWI proofs for L_b. This is done a single thread at a time.

$\mathsf{Hyb}_{8,7}$: **Change Type (i) threads receiver input to RWI message in OT:** In this hybrid, $\mathsf{Sim}_{\mathsf{Hyb}}$ modifies the third round of the type (i) threads to use receiver input to be the third round message of RWI, corresponding to L_b, in OT, instead of 0.

$\mathsf{Hyb}_{8,8}$: **Change Type (i) threads Ecom to RWI message:** In this hybrid, $\mathsf{Sim}_{\mathsf{Hyb}}$ modifies the third round of the type (i) threads to commit to RWI third round messages corresponding to L_a instead of the commitment to 0.

Note that $\mathsf{Hyb}_{8,8} \equiv \mathsf{Hyb}_8$

□

Claim 59. *The invariant holds in* Hyb_9*.*

Proof. The claim is trivially true since the change is made only in the fourth round. □

Claim 60. *Assuming the security of* GC *and sender's* OT *messages,* Hyb_9 *is indistinguishable from* Hyb_8

Proof. This is established by the creating the following sub-hybrids.

$\mathsf{Hyb}_{9,0}$: **Change OT sender's message on main thread:** In this hybrid, $\mathsf{Sim}_{\mathsf{Hyb}}$ changes how the sender OT is computed. We extract from ot to obtain the adversary's receiver message. Use the receiver value extracted from the ot to change the sender OT to include only a single label of the garbled circuit. Specifically, $\forall j \in [n] \setminus \{i\}$, compute

$$\mathsf{ot}_4^{i\to j} := \mathsf{OT}_4\left(\left(\mathsf{lab}_{i,v|_j}, \mathsf{lab}_{i,v|_j}\right), \mathsf{ot}_1^{i\to j}, \mathsf{ot}_2^{i\to j}, \mathsf{ot}_3^{i\to j}; \mathsf{r}_{i,\mathsf{ot}}^{i\to j}\right).$$

where v is the extracted receiver string from $\mathsf{ot}_3^{i\to j}$.

Claim 61. *Assuming the security of sender's* OT *messages,* $\mathsf{Hyb}_{9,0}$ *is indistinguishable from* Hyb_8

Proof. The only difference between $\mathsf{Hyb}_{9,0}$ and Hyb_8 is that the simulator $\mathsf{Sim}_{\mathsf{Hyb}}$ switches the sender OT input to using the same label twice P_i if it receives a non-accepting RWI proof for L_a.

Assume, for the sake of contradiction, that that the views are distinguishable. Then there exists an adversary $\mathcal{D}$ such that $\mathcal{D}$ can distinguish between $\mathsf{Hyb}_{9,0}$ and Hyb_8 with non-negligible advantage. We will use this adversary to create an adversary $\mathcal{A}_{\mathsf{OT}}$ that breaks the sender's security in OT with non-negligible probability.

We now describe the working of $\mathcal{A}_{\mathsf{OT}}$ which interacts with the challenger $\mathcal{C}_{\mathsf{OT}}$. $\mathcal{A}_{\mathsf{OT}}$ picks randomly an honest party P_i and a random malicious party P_j. All messages other than the chosen OT messages are computed in the same manner as $\mathsf{Sim}_{\mathsf{Hyb}}$. The OT messages from P_i to P_j are exposed to the external challenger. Specifically, in round 1, send to $\mathcal{C}_{\mathsf{OT}}$ the first round OT message $\mathsf{ot}_1^{i\to j}$ sent by $\mathcal{A}$. Receive ot_2 and set $\mathsf{ot}_2^{i\to j} := \mathsf{ot}_2$.

$\mathcal{A}_{\mathsf{OT}}$ creates sufficiently many look-ahead threads, in each of which, it runs rounds 2 and 3 of the protocol alone. In each look-ahead thread, $\mathcal{A}_{\mathsf{OT}}$ re-sends the same $\mathsf{ot}_2^{i\to j}$ message in the second round. Only $\mathsf{ot}_3^{i\to j}$ on the main thread is forwarded to $\mathcal{C}_{\mathsf{OT}}$. With the the look ahead threads threads, $\mathcal{A}_{\mathsf{OT}}$ can successfully run the extraction phase to extract the OT receiver bit from P_j to be v. Send $\left(\widetilde{\mathsf{lab}}_{i|_j}, \widetilde{\mathsf{lab}}_{i|_j}\right)$ and $\left(\mathsf{lab}_{i,v|_j}, \mathsf{lab}_{i,v|_j}\right)$ to $\mathcal{C}_{\mathsf{OT}}$ as challenges. Note that we don't need rewind security here since the look ahead threads are only executed up to the third round. And for the alternating message OT, a new adversarial receiver message doesn't have to be answered on the look ahead threads.

The rest of the messages are obtained in the same manner as $\mathsf{Sim}_{\mathsf{Hyb}}$. Depending on pair was used as sender input we are either in $\mathsf{Hyb}_{9,0}$ or Hyb_8. On completion of the execution, the view is input to $\mathcal{D}$ and the output returned is the output of $\mathcal{A}_{\mathsf{OT}}$

By our assumption, $\mathcal{D}$ can distinguish between the two hybrids with noticeable probability ε. Therefore, with non-negligible advantage $\frac{\varepsilon}{n^2}$, $\mathcal{A}_{\mathsf{OT}}$ wins the challenge game with $\mathcal{C}_{\mathsf{OT}}$ which breaks the sender security of OT. Thus, ε must be negligible, and thus the views are indistinguishable. $\square$

$\mathsf{Hyb}_{9,1}$: **Simulate garbled circuit:** In this hybrid, $\mathsf{Sim}_{\mathsf{Hyb}}$ computes a garbled circuit to output $\perp$. Specifically,

$$\left(\mathsf{C}_i, \widetilde{\mathsf{lab}}_i\right) \leftarrow \mathsf{Garble}\left(\mathsf{C}_\perp\right)$$

Claim 62. *Assuming the security of* GC*,* $\mathsf{Hyb}_{9,1}$ *is indistinguishable from* $\mathsf{Hyb}_{9,0}$

Proof. The only difference between $\mathsf{Hyb}_{9,1}$ and $\mathsf{Hyb}_{9,0}$ is that the simulator $\mathsf{Sim}_{\mathsf{Hyb}}$ switches the garbled circuit to a circuit for each relevant P_i. Note that this is a functionally equivalent circuit given the condition we choose to switch. Namely, either there is an implicit abort, or that P_i receives a wrong RWI proof via OT. The changes are made through a sequence of sub-hybrids, where in each sub-hybrid only a single circuit is switched.

Assume, for the sake of contradiction, that that the views are distinguishable. Then there exists an adversary $\mathcal{D}$ such that $\mathcal{D}$ can distinguish between $\mathsf{Hyb}_{9,1}$ and $\mathsf{Hyb}_{9,0}$ with non-negligible advantage. We will use this adversary to create an adversary $\mathcal{A}_{\mathsf{GC}}$ that breaks GC security with non-negligible probability.

We now describe the working of $\mathcal{A}_{\mathsf{GC}}$ which interacts with the challenger $\mathcal{C}_{\mathsf{GC}}$. All messages other than the garbled circuit are computed in the same manner as $\mathsf{Sim}_{\mathsf{Hyb}}$. The GC messages from P_i are exposed to the external challenger. Specifically, in round four, it sends as challenges to $\mathcal{C}_{\mathsf{GC}}$ $\mathsf{C}_\perp$ and $\left(\mathsf{C}\left[i, \mathsf{msg}_{4,i}, \left\{\mathsf{rwi}_{b,\ell}^{j\to i}\right\}_{\ell\in[2], j\in[n]\setminus\{i\}} \left\{\mathsf{st}_b^{j\to i}\right\}_{j\in[n]\setminus\{i\}}\right], v\right)$ where v is the concatenation of all extracted/generated receiver values for all parties other than P_i. $\mathcal{C}_{\mathsf{GC}}$ then returns a garbled circuit $\widetilde{C}$ and labels $\widetilde{\mathsf{lab}}$. These are set as $\widetilde{\mathsf{C}}_i := \widetilde{C}$ and $\widetilde{\mathsf{lab}}_i := \widetilde{\mathsf{lab}}$. The rest of the messages are obtained in the same manner as $\mathsf{Sim}_{\mathsf{Hyb}}$. Depending on challenge bit used by $\mathcal{C}_{\mathsf{GC}}$ we are either in $\mathsf{Hyb}_{9,1}$ or $\mathsf{Hyb}_{9,0}$. On completion of the execution, the view is input to $\mathcal{D}$ and the output returned is the output of $\mathcal{A}_{\mathsf{GC}}$.

By our assumption, $\mathcal{D}$ can distinguish between the two hybrids with noticeable probability ε. Therefore, with non-negligible advantage, $\mathcal{A}_{\mathsf{GC}}$ wins the challenge game with $\mathcal{C}_{\mathsf{GC}}$ which breaks the security of GC. Thus, ε must be negligible, and thus the views are indistinguishable. $\square$

$\mathsf{Hyb}_{9,2}$: **Change OT sender's message on main thread:** In this hybrid, $\mathsf{Sim}_{\mathsf{Hyb}}$ changes how the sender OT is computed.

Change the sender OT to include back to include both labels of the garbled circuit. Specifically, $\forall j \in [n] \setminus \{i\}$, compute

$$\mathsf{ot}_4^{i\to j} := \mathsf{OT}_4\left(\left(\mathsf{lab}_{i,0|_j}, \mathsf{lab}_{i,1|_j}\right), \mathsf{ot}_1^{i\to j}, \mathsf{ot}_2^{i\to j}, \mathsf{ot}_3^{i\to j}; \mathsf{r}_{i,\mathsf{ot}}^{i\to j}\right).$$

Claim 63. *Assuming the security of sender's* OT *messages,* $\mathsf{Hyb}_{9,2}$ *is indistinguishable from* Hyb_8

Proof. The proof follows identically as in Claim 61. $\square$

Note that $\mathsf{Hyb}_{9,2} \equiv \mathsf{Hyb}_9$.

$\square$

Claim 64. *The invariant holds in* $\mathsf{Hyb}_{\mathsf{IDEAL}}$.

Proof. The claim is trivially true since the main thread remains unchanged. $\square$

Claim 65. Hyb_9 *is indistinguishable from* $\mathsf{Hyb}_{\mathsf{IDEAL}}$ *except with probability at most* $\frac{\mu}{4} + \mathsf{negl}(\lambda)$.

Proof. This is argued in two cases depending on the probability with which the adversary abort.

Case 1: $\Pr[\textbf{not abort}] \geq \frac{\mu}{4}$:

Suppose the adversary doesn't cause an abort with probability greater that $\frac{\mu}{4}$. Let us analyze the probability with which $\perp_{\mathsf{extract}}$ is output by $\mathsf{Sim}_{\mathsf{Hyb}}$. By the Chernoff bound, in Hyb_{10}, except with negligible probability, in the set of $\frac{5\cdot n\cdot\lambda}{\mu}$ threads, there will be at least 5 GOOD threads with respect to some honest party P_{i^*}. Also in $\mathsf{Hyb}_{\mathsf{IDEAL}}$, $\mathsf{Sim}_{\mathsf{Hyb}}$ will run an expected polynomial number of threads to get 12λ (which is greater than $5\cdot n$) GOOD threads. Thus the extractions will be successful in except with negligible probability.

Therefore the only difference between $\mathsf{Hyb}_{\mathsf{REAL}}$ and Hyb_{10} is that in Hyb_{10}, after extraction, $\mathsf{Sim}_{\mathsf{Hyb}}$ samples the main thread $\frac{\lambda}{\mu}$ times while in $\mathsf{Hyb}_{\mathsf{REAL}}$, $\mathsf{Sim}_{\mathsf{Hyb}}$ first estimates the probability of not aborting to be ε' and then re-samples the main thread $\min\left(2^{\lambda}, \frac{\lambda^2}{\varepsilon'}\right)$ times. The rest of the proof follows in a very similar manner to the proof of claim 5.8 in [Lin16]. That is, we show that if "Check Abort" step succeeds, the simulator in $\mathsf{Hyb}_{\mathsf{IDEAL}}$ fails only with negligible probability using the claim in [Lin16]. Also, by a Markov argument, we know that Hyb_{10}, if the "Check Abort" step succeeds, the simulation successfully forces the output and hence, this completes the proof.

Case 2: $\Pr[\textbf{not abort}] < \frac{\mu}{4}$:

Suppose the adversary doesn't cause an abort with probability smaller than $\frac{\mu}{4}$. Then, in both hybrids, $\mathsf{Sim}_{\mathsf{Hyb}}$ aborts at the end of the "Check Abort" step except with probability $\frac{\mu}{4}$. Thus, in this case, the adversary's view in $\mathsf{Hyb}_{\mathsf{IDEAL}}$ and Hyb_{10} is indistinguishable except with probability at most $\frac{\mu}{4} + \mathsf{negl}(\lambda)$.

$\square$

We now calculate the probability that the adversary can distinguish between $\mathsf{Hyb}_{\mathsf{REAL}}$ and $\mathsf{Hyb}_{\mathsf{IDEAL}}$.

Except in two cases, every pair of hybrids are indistinguishable except with negligible probability. In the two special cases, the hybrids are indistinguishable except with probability $\frac{\mu}{4} + \mathsf{negl}(\lambda)$. Thus, $\mathsf{Hyb}_{\mathsf{REAL}}$ and $\mathsf{Hyb}_{\mathsf{IDEAL}}$ are indistinguishable except with probability $\frac{\mu}{2} + \mathsf{negl}(\lambda)$. This contradicts our assumption that there must be an adversary $\mathcal{A}$ that can distinguish the REAL and IDEAL executions with probability at least μ.

5 Acknowledgments

We would like to thank Alex Lombardi and Luke Schaeffer for pointing out to us that any OT with perfect completeness implies non-interactive commitment schemes, and, for suggesting to us to revise our theorem statements to reflect this observation.

Vipul Goyal is supported by a grant from Northrop Grumman, a gift from DOS Network, and, a gift from Ripple.

Arka Rai Choudhuri and Abhishek Jain are supported in part by a DARPA/ARL Safeware Grant W911NF-15-C-0213, and a subaward from NSF CNS-1414023.

Rafail Ostrovsky is supported in part by NSF grant 1619348, BSF grant 2015782, DARPA SafeWare subcontract to Galois Inc., DARPA SPAWAR contract N66001-15-1C-4065, JP Morgan Faculty Research Award, OKAWA Foundation Research Award, IBM Faculty Research Award, Xerox Faculty Research Award, B. John Garrick Foundation Award, Teradata Research Award, and Lockheed-Martin Corporation Research Award. The views expressed are those of the authors and do not reflect position of the Department of Defense or the U.S. Government.

Michele Ciampi was also partially supported by H2020 project PRIVILEDGE #780477.

References

[ACJ17] Prabhanjan Ananth, Arka Rai Choudhuri, and Abhishek Jain. A new approach to round-optimal secure multiparty computation. In Jonathan Katz and Hovav Shacham, editors, *CRYPTO 2017, Part I*, volume 10401 of *LNCS*, pages 468–499, Santa Barbara, CA, USA, August 20–24, 2017. Springer, Heidelberg, Germany.

[BGJ+18] Saikrishna Badrinarayanan, Vipul Goyal, Abhishek Jain, Yael Tauman Kalai, Dakshita Khurana, and Amit Sahai. Promise zero knowledge and its applications to round optimal MPC. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part II*, volume 10992 of *LNCS*, pages 459–487, Santa Barbara, CA, USA, August 19–23, 2018. Springer, Heidelberg, Germany.

[BHP17] Zvika Brakerski, Shai Halevi, and Antigoni Polychroniadou. Four round secure computation without setup. In Yael Kalai and Leonid Reyzin, editors, *TCC 2017, Part I*, volume 10677 of *LNCS*, pages 645–677, Baltimore, MD, USA, November 12–15, 2017. Springer, Heidelberg, Germany.

[BL18] Fabrice Benhamouda and Huijia Lin. k-round multiparty computation from k-round oblivious transfer via garbled interactive circuits. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part II*, volume 10821 of *LNCS*, pages 500–532, Tel Aviv, Israel, April 29 – May 3, 2018. Springer, Heidelberg, Germany.

[BMR90] Donald Beaver, Silvio Micali, and Phillip Rogaway. The round complexity of secure protocols (extended abstract). In *STOC*, pages 503–513, 1990.

[CGJ19] Arka Rai Choudhuri, Vipul Goyal, and Abhishek Jain. On round optimal secure multiparty computation from minimal assumptions. Cryptology ePrint Archive, Report 2019/216, 2019. https://eprint.iacr.org/2019/216.

[CO19] Michele Ciampi and Rafail Ostrovsky. Four-round secure multiparty computation from general assumptions. Cryptology ePrint Archive, Report 2019/214, 2019. https://eprint.iacr.org/2019/214.

[COSV16] Michele Ciampi, Rafail Ostrovsky, Luisa Siniscalchi, and Ivan Visconti. Concurrent non-malleable commitments (and more) in 3 rounds. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part III*, volume 9816 of *LNCS*, pages 270–299, Santa Barbara, CA, USA, August 14–18, 2016. Springer, Heidelberg, Germany.

[COSV17a] Michele Ciampi, Rafail Ostrovsky, Siniscalchi, and Ivan Visconti. Round-optimal secure two-party computation from trapdoor permutations. In *TCC*, 2017.

[COSV17b] Michele Ciampi, Rafail Ostrovsky, Luisa Siniscalchi, and Ivan Visconti. Delayed-input non-malleable zero knowledge and multi-party coin tossing in four rounds. In Yael Kalai and Leonid Reyzin, editors, *TCC 2017, Part I*, volume 10677 of *LNCS*, pages 711–742, Baltimore, MD, USA, November 12–15, 2017. Springer, Heidelberg, Germany.

[DDN91] Danny Dolev, Cynthia Dwork, and Moni Naor. Non-malleable cryptography (extended abstract). In *23rd ACM STOC*, pages 542–552, New Orleans, LA, USA, May 6–8, 1991. ACM Press.

[DN00] Cynthia Dwork and Moni Naor. Zaps and their applications. In *41st FOCS*, pages 283–293, Redondo Beach, CA, USA, November 12–14, 2000. IEEE Computer Society Press.

[GGSW13] Sanjam Garg, Craig Gentry, Amit Sahai, and Brent Waters. Witness encryption and its applications. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *45th ACM STOC*, pages 467–476, Palo Alto, CA, USA, June 1–4, 2013. ACM Press.

[GK96a] Oded Goldreich and Ariel Kahan. How to construct constant-round zero-knowledge proof systems for NP. *J. Cryptology*, 9(3):167–190, 1996.

[GK96b] Oded Goldreich and Hugo Krawczyk. On the composition of zero-knowledge proof systems. *SIAM J. Comput.*, 25(1):169–192, 1996.

[GKM+00] Yael Gertner, Sampath Kannan, Tal Malkin, Omer Reingold, and Mahesh Viswanathan. The relationship between public key encryption and oblivious transfer. In *41st FOCS*, pages 325–335, Redondo Beach, CA, USA, November 12–14, 2000. IEEE Computer Society Press.

[GMPP16a] Sanjam Garg, Pratyay Mukherjee, Omkant Pandey, and Antigoni Polychroniadou. The exact round complexity of secure computation. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part II*, volume 9666 of *LNCS*, pages 448–476, Vienna, Austria, May 8–12, 2016. Springer, Heidelberg, Germany.

[GMPP16b] Sanjam Garg, Pratyay Mukherjee, Omkant Pandey, and Antigoni Polychroniadou. The exact round complexity of secure computation. In *EUROCRYPT*, pages 448–476, 2016.

[GMR88] Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM J. Comput.*, 17(2):281–308, 1988.

[GMR89] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. *SIAM J. Comput.*, 1989.

[GMW87] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or A completeness theorem for protocols with honest majority. In *STOC*, pages 218–229, 1987.

[Gol04] Oded Goldreich. *The Foundations of Cryptography - Volume 2, Basic Applications*. Cambridge University Press, 2004.

[Goy11] Vipul Goyal. Constant round non-malleable protocols using one way functions. In *STOC*, pages 695–704, 2011.

[GPR16] Vipul Goyal, Omkant Pandey, and Silas Richelson. Textbook non-malleable commitments. In Daniel Wichs and Yishay Mansour, editors, *48th ACM STOC*, pages 1128–1141, Cambridge, MA, USA, June 18–21, 2016. ACM Press.

[GR19] Vipul Goyal and Silas Richelson. Non-malleable commitments using goldreich-levin list decoding. In *Under submission*, 2019.

[GS18] Sanjam Garg and Akshayaram Srinivasan. Two-round multiparty secure computation from minimal assumptions. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part II*, volume 10821 of *LNCS*, pages 468–499, Tel Aviv, Israel, April 29 – May 3, 2018. Springer, Heidelberg, Germany.

[HHPV18] Shai Halevi, Carmit Hazay, Antigoni Polychroniadou, and Muthuramakrishnan Venkitasubramaniam. Round-optimal secure multi-party computation. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part II*, volume 10992 of *LNCS*, pages 488–520, Santa Barbara, CA, USA, August 19–23, 2018. Springer, Heidelberg, Germany.

[IKOS07] Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Zero-knowledge from secure multiparty computation. In *STOC*, pages 21–30, 2007.

[IPS08] Yuval Ishai, Manoj Prabhakaran, and Amit Sahai. Founding cryptography on oblivious transfer - efficiently. In David Wagner, editor, *CRYPTO 2008*, volume 5157 of *LNCS*, pages 572–591, Santa Barbara, CA, USA, August 17–21, 2008. Springer, Heidelberg, Germany.

[JKKR17] Abhishek Jain, Yael Tauman Kalai, Dakshita Khurana, and Ron Rothblum. Distinguisher-dependent simulation in two rounds and its applications. In *CRYPTO*, 2017.

[Kil88] Joe Kilian. Founding cryptography on oblivious transfer. In *20th ACM STOC*, pages 20–31, Chicago, IL, USA, May 2–4, 1988. ACM Press.

[KO04] Jonathan Katz and Rafail Ostrovsky. Round-optimal secure two-party computation. In *CRYPTO*, pages 335–354, 2004.

[KOS03] Jonathan Katz, Rafail Ostrovsky, and Adam D. Smith. Round efficiency of multi-party computation with a dishonest majority. In *EUROCRYPT*, pages 578–595, 2003.

[Lin16] Yehuda Lindell. How to simulate it - A tutorial on the simulation proof technique. Cryptology ePrint Archive, Report 2016/046, 2016. http://eprint.iacr.org/2016/046.

[LPV08] Huijia Lin, Rafael Pass, and Muthuramakrishnan Venkitasubramaniam. Concurrent non-malleable commitments from any one-way function. In Ran Canetti, editor, *TCC 2008*, volume 4948 of *LNCS*, pages 571–588, San Francisco, CA, USA, March 19–21, 2008. Springer, Heidelberg, Germany.

[LS90] Dror Lapidot and Adi Shamir. Publicly verifiable non-interactive zero-knowledge proofs. In *CRYPTO*, pages 353–365, 1990.

[LS91] Dror Lapidot and Adi Shamir. Publicly verifiable non-interactive zero-knowledge proofs. In Alfred J. Menezes and Scott A. Vanstone, editors, *CRYPTO'90*, volume 537 of *LNCS*, pages 353–365, Santa Barbara, CA, USA, August 11–15, 1991. Springer, Heidelberg, Germany.

[LS19] Alex Lombardi and Luke Schaeffer. A note on key agreement and non-interactive commitments. Cryptology ePrint Archive, Report 2019/279, 2019. https://eprint.iacr.org/2019/279.

[Pas04] Rafael Pass. Bounded-concurrent secure multi-party computation with a dishonest majority. In *Proceedings of the 36th Annual ACM Symposium on Theory of Computing, Chicago, IL, USA, June 13-16, 2004*, pages 232–241, 2004.

[PR05] Rafael Pass and Alon Rosen. Concurrent non-malleable commitments. In *46th FOCS*, pages 563–572, Pittsburgh, PA, USA, October 23–25, 2005. IEEE Computer Society Press.

[PRS02] Manoj Prabhakaran, Alon Rosen, and Amit Sahai. Concurrent zero knowledge with logarithmic round-complexity. In *43rd FOCS*, pages 366–375, Vancouver, British Columbia, Canada, November 16–19, 2002. IEEE Computer Society Press.

[PW10] Rafael Pass and Hoeteck Wee. Constant-round non-malleable commitments from sub-exponential one-way functions. In *EUROCRYPT*, pages 638–655, 2010.

[Ros04] Alon Rosen. A note on constant-round zero-knowledge proofs for NP. In Moni Naor, editor, *TCC 2004*, volume 2951 of *LNCS*, pages 191–202, Cambridge, MA, USA, February 19–21, 2004. Springer, Heidelberg, Germany.

[Sah99] Amit Sahai. Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. In *40th FOCS*, pages 543–553, New York, NY, USA, October 17–19, 1999. IEEE Computer Society Press.

[Wee10] Hoeteck Wee. Black-box, round-efficient secure computation via non-malleability amplification. In *FOCS*, pages 531–540, 2010.

[Yao86] Andrew Chi-Chih Yao. How to generate and exchange secrets (extended abstract). In *FOCS*, 1986.

A Delayed-Input Rewind Secure Witness Indistinguishable Proofs from [GR19]

For completeness, we discuss the rewind secure delay-input witness indistinguishable proofs from the unpublished work of Goyal and Richelson [GR19]. This section is taken verbatim from their paper. Throughout, we let λ denote the security parameter, and we write $\mathsf{negl}(\lambda)$ for functions which tend to zero faster than λ^{-c} for any constant c.

A.1 Preliminaries

MPC-in-the-Head [IKOS07]. As in [BGJ+18], we make black-box use of a 3-round zero knowledge protocol (non delayed-input) with bounded rewinding security. The soundness error of the protocol would depend upon the rewinding parameter B.

Definition 16 (3-Round ZK with Bounded Rewinding Security). *[BGJ+18] Fix a positive integer B. A delayed-input 3-round interactive argument (as defined in Definition 7) for an NP language L, with an NP relation R_L is said to have B-*Rewinding Security *if there exists a simulator* Sim *such that for every non-uniform PPT interactive Turing Machine V^*, it holds that $\{\mathsf{REAL}^{V^*}(1^\lambda)\}_\lambda$ and $\{\mathsf{IDEAL}^{V^*}(1^\lambda)\}_\lambda$ are computationally indistinguishable, where the random variable $\mathsf{REAL}^{V^*}(1^\lambda)$ is defined via the following experiment. In what follows we denote by P_1 the prover's algorithm in the first round, and similarly we denote by P_3 his algorithm in the third round.*

Experiment $\mathsf{REAL}^{V^*}(1^\lambda)$ *is defined as follows:*

1. *Run $P_1(1^\lambda, x, w; r)$ and obtain output rwi_1 to be sent to the verifier.*
2. *Run the verifier $V^*(1^\lambda, \mathsf{rwi}_1)$ and interpret its output as message rwi_2.*
3. *Run $P_3(1^\lambda, \mathsf{rwi}_2, x, w; r)$, where P_3 is the (honest) prover's algorithm for generating the third message of the WI protocol, and send its output rwi_3 to V^*.*
4. *Set a counter $i = 0$.*
5. *If $i < B$, then set $i = i + 1$, and V^* (given all the information so far) generates another message rwi_2^i, and receives the (honest) prover's message $P_3(\mathsf{rwi}_2^i, x, w; r)$. Repeat this step until $i = B$.*
6. *The output of the experiment is the view of V^*.*

Experiment $\mathsf{IDEAL}^{V^*}(1^\lambda)$ *is the output of the experiment* $\mathsf{Sim}^{V^*}(1^\lambda, x; r)$.

Imported Theorem 3. *[IKOS07, BGJ+18] Assume the existence of non-interactive commitments. Then, for any (polynomial) rewinding parameter B, there exists a 3-round zero-knowledge protocol for proving NP statements that is simulatable under B-bounded rewinding according to Definition 16.*

If B is a constant, the soundness error of the above protocol will be a constant. If $B = \mathrm{poly}(\lambda)$, the soundness error $\epsilon \leq 1 - q(\lambda)$ where q is also a polynomial.

A.2 The Construction

Building Blocks. Our construction will make use of two crucial building blocks: the 3-round delayed-input WI protocol in [LS90], and, the bounded rewinding secure 3-round "MPC in the head" based 3-round protocol of [IKOS07].

Theorem 6. *Assuming non-interactive commitments, for every (polynomial) rewinding parameter B, there exists a three round delayed-input witness-indistinguishable proof system* RWI *with B-rewinding security.*

The soundness of our protocol depends upon the rewinding parameter B and can be amplified via parallel repetition while preserving the WI property. Our protocol RWI will consists of 4 algorithms $(\mathrm{SWI1}, \mathrm{SWI2}, \mathrm{SWI3}, \mathrm{SWI4})$ where the first 3 denote the algorithms used by the prover and verifier to send their messages and the last is the final verification algorithm. We use the protocol from [IKOS07]. We denote its algorithms by $\mathsf{Head.ZK} = (\mathsf{zk1}, \mathsf{zk2}, \mathsf{zk3}, \mathsf{zk4})$, where the first 3 denote the algorithms used by the prover and verifier to generate their messages, and the last is the final verification algorithm. The simulator of the protocol $\mathsf{Head.ZK}$ is denoted by S_{zk}. We will also use the delayed-input WI protocol from [LS90] and denote its algorithms by $\mathsf{DIWI} = (\mathsf{DWI}_1, \mathsf{DWI}_2, \mathsf{DWI}_3, \mathsf{DWI}_4)$, where the first 3 denote the algorithms used by the prover and verifier to generate their messages, and the last is the final verification algorithm.

Let λ be the statistical security parameter. We define parameter $N = B^2\lambda^4$.

Inputs: At the beginning of the third round, the prover P gets as input (x, w); V gets only x.

1. **Round 1: Prover message:**
   - P prepares and sends commitments $c_1, \ldots, c_N$ where $c_i = \mathsf{Com}(0)$ for all i.
   - P also prepares and sends a first round message $\mathsf{hzk}_1^{P \to V}$ for a single instance of $\mathsf{Head.ZK}$, using $\mathsf{zk1}$. The statement for $\mathsf{Head.ZK}$ is that each $c_i, i \in [N]$ is indeed a commitment to 0; P uses the commitment openings as its witness.
   - P also prepares and sends first round messages $\{\mathsf{dwi}_{1,i}^{P \to V}\}_{i \in [N]}$ for N separate instances of DIWI. The statements for these DIWI instances will come in the third round.
2. **Round 2: Verifier message:**
   - The verifier samples a challenge bit ch and sends it to P.
   - If $ch = 0$, V in addition executes $\mathsf{zk2}$ to sample $\mathsf{hzk}_2^{V \to P}$ and sends it to V.
   - If $ch = 1$, V executes DWI_2 on $\{\mathsf{dwi}_{1,i}^{P \to V}\}_{i \in [N]}$ to get $\{\mathsf{dwi}_{2,i}^{V \to P}\}_{i \in [N]}$ and sends to P.
3. **Round 3: Prover message:**

   If $ch = 0$, P generates $\mathsf{hzk}_3^{P \to V}$ by running $\mathsf{zk3}$ and sends it to P. If $ch = 1$, P proceeds as follows:
   - Following [IKOS07], emulate an MPC computation of the circuit representing the witness relation with λ players. The input of each player will be a share of the witness w. Let the view of the i-th player be V_i. For $i \in [\lambda]$, compute $cv_i = \mathsf{Com}(V_i)$ and send it to V.
   - Select a set of $\lambda(\lambda - 1)$ distinct random indices $\{k_{i,j} \in [N]\}_{i \neq j, i \in [\lambda], j \in [\lambda]}$. Represent these set of indices by SI and send them to V.
   - Use $\{\mathsf{dwi}_{1,i}^{P \to V}, \mathsf{dwi}_{2,i}^{V \to P}\}_{i \in SI}$ and the algorithm DWI_3 to generate $\{\mathsf{dwi}_{3,i}^{P \to V}\}_{i \in SI}$ and send them to V. For each $k_{i,j} \in SI$, the message $\mathsf{dwi}_{3,k_{i,j}}^{P \to V}$ prove that either (a) $c_{k_{i,j}}$ is a commitment to 1, or, (b) the views (V_i, V_j) are honest and "consistent" with each other. That is, there exist input (w_i, r_i) (resp (w_j, r_j)) s.t. V_i (resp. V_j) is computed and committed honestly using (w_i, r_i) (resp (w_j, r_j)). Furthermore, each outgoing message sent to the j-th player in V_i is consistent with each incoming message from the i-th player in V_j, and, vice-versa. The honest prover P uses the witness corresponding to (b) to compute $\mathsf{dwi}_{3,k_{i,j}}^{P \to V}$.
4. **Verifier Output:**
   - If $ch = 0$, compute the output of the algorithm $\mathsf{zk4}$ on $(\mathsf{hzk}_1^{P \to V}, \mathsf{hzk}_2^{V \to P}, \mathsf{hzk}_3^{P \to V})$ and the private randomness of V. Output whatever $\mathsf{zk4}$ outputs.
   - If $ch = 1$, for each $i \in SI$, execute the algorithm DWI_4 on $(\mathsf{dwi}_{1,i}^{P \to V}, \mathsf{dwi}_{2,i}^{V \to P}, \mathsf{dwi}_{3,i}^{P \to V})$. If all executions of DWI_4 accept, then output accept and reject otherwise.

Figure 6: 3 round Bounded Rewinding Secure WI

A.3 Security Analysis

Proving Soundness. We prove that our protocol RWI has soundness $\delta/2$ where δ is the soundness parameter of the $\mathsf{Head.ZK}$ construction. Suppose $x \notin L$. Consider the following two cases:

1. **Case 1:** There exists $i \in [N]$ s.t. $c_i \neq \mathsf{Com}(0)$. In this case, we claim that V will reject the execution with probability at least $\delta/2$. This is because with probability $1/2$, the challenge ch will be 0. If so, by the soundness of $\mathsf{Head.ZK}$, V is guaranteed to reject the execution with probability at least δ.

2. **Case 2:** For all $i \in [N]$, c_i is indeed a commitment to 0. Assume that the verifier accepts all $\lambda(\lambda-1)$ executions of the DIWIprotocol. Then w.h.p, the prepared views $V_1, \ldots, V_\lambda$ are such that each pair (V_i, V_j) is consistent. This follows from the soundness of the DIWI protocol (which has negligible soundness error). Since the underlying MPC construction has perfect correctness, it follows that $x \in L$ which is a contradiction. Hence, w.h.p, the verifier must reject at least one execution of the DIWI protocol.

Suppose the probability of Case 1 and Case 2 are p and $1-p$ respectively. Then RWI has soundness $p\delta/2 + (1-p)\cdot(1-\mathsf{negl}(\lambda)(\lambda)) \geq \delta/2$.

Witness Indistinguishability under B rewinds: We will now prove that RWI satisfies witness indistinguishability under B rewinds where B is the rewinding parameter of the $\mathsf{Head.ZK}$ construction. Consider the following sequence of hybrid experiments.

Hybrid H_0: This hybrid experiment corresponds to the honest protocol execution where the prover uses witnesses $w^1, \ldots, w_0^B$ to prove the statements $x^1, \ldots, x^B$ respectively in B rewound executions.

Hybrid H_1: In this hybrid experiment, the prover starts using the simulator S_{zk} to simulate the execution of the protocol $\mathsf{Head.ZK}$ across all executions. In more details, the prover runs S_{zk} to get the message $\mathsf{hzk}_1^{P\to V}$. Prover then prepares the first message of the protocol honestly except for using $\mathsf{hzk}_1^{P\to V}$ given by S_{zk} and sends it to V^*. In all the B execution, the prover handles the messages of $\mathsf{Head.ZK}$ as follows. If $ch = 0$, prover forwards the verifier message of $\mathsf{Head.ZK}$ to S_{zk} and forwards the response back to V^*. If $ch = 1$, the prover aborts this particular execution with S_{zk} since there will be no further message of $\mathsf{Head.ZK}$ in this execution. All messages other than messages of $\mathsf{Head.ZK}$ are computed honestly as in H_0.

By the zero-knowledge property of $\mathsf{Head.ZK}$, it follows that the view produced by S_{zk} across the B executions will be indistinguishable from that in H_0. Hence, the view of V^* in H_1 is indistinguishable from that in H_0.

Hybrid H_2: The prover now selects a random set of $\lambda(\lambda-1)$ distinct indices (from N indices) for each of the B executions even before the protocol starts. Denote these sets by $SI_1, \ldots, SI_B$. Define a set SU which consists of all the indices which appear in *more than 1* of these B sets $SI_1, \ldots, SI_B$. In hybrid H_2, the prover is identical to that in H_1 except that for each $i \notin SU$, the prover sets $c_i = \mathsf{Com}(1)$. (The remaining commitments are commitments of 0 as before.)

The indistinguishability of this hybrid follows directly from the hiding property of Com. Observe that in this experiment, the openings of the commitments $c_1, \ldots, c_N$ are not being used by the prover in any of the B executions.

We also prove the following lemma.

Lemma 5. *Suppose $N = B^2\lambda^4$. Except with negligible probability over the random tape of the prover, $|SU| \leq \frac{\lambda}{6}$.*

Proof. Define $T = B\lambda^2$. We consider the following experiment. First pick T independent and random indices from the set N. The (multi)set of indices is denoted by ST and the indices themselves are denoted by $E_1, \ldots, E_T$. Since the indices are picked independently, it is possible that some of them maybe the same (and hence ST is a multiset rather than a set). We now construct sets $SI_1, \ldots, SI_B$ from ST as follows. SI_1 will simply consist of the first $\lambda(\lambda-1)$ mutually distinct elements from ST (starting with element E_1). SI_2 will consist of the second $\lambda(\lambda-1)$ mutually distinct elements from ST, and so on. Note that for all i, all elements within SI_i must be distinct. However, two sets SI_i and SI_j with $i \neq j$ may have non-zero intersection. To be able to successfully construct $SI_1, \ldots, SI_B$, it is sufficient (though not necessary) for ST to have at least $B\lambda(\lambda-1)$ distinct elements. The distribution of sets $SI_1, \ldots, SI_B$ constructed using this algorithm is identical to the distribution when $SI_1, \ldots, SI_B$ are picked one at a time by randomly picking $\lambda(\lambda-1)$ distinct indices out of N. We now prove that, in fact, most elements in ST are distinct.

Claim 66. *Multiset ST has at least $T - \lambda/6$ distinct elements except with negligible probability.*

Proof. Since all elements of ST are picked independently and uniformly, the probability that the i-th element is identical to any other element in ST is at most $\frac{T}{N}$. Define random variable X_i s.t. $X_i = 1$ if $\exists j \neq i$ s.t. $E_i = E_j$, and, $X_i = 0$ otherwise. Clearly, the expectation $\mathbb{E}[X_i] \leq \frac{T}{N}$. Denote $X = \sum_i X_i$. By linearly of expectation, $\mathbb{E}[X] \leq \frac{T^2}{N} = 1$.

Denote $\mathbb{E}[X]$ by μ. Set $\delta = \frac{\lambda}{7}$. By Chernoff bounds, we have that $\Pr[X > (1+\delta)\mu] \leq \mathsf{negl}(\lambda)\,(\lambda)$. Thus, $\Pr[X > \frac{\lambda}{6}] \leq \mathsf{negl}(\lambda)\,(\lambda)$.

□

If ST has T elements and at least $T - \lambda/6$ are distinct, at most $\lambda/6$ elements appear multiple times in ST. This also means that at most $\lambda/6$ elements appear multiple times across the sets $SI_1, \ldots, SI_B$. Thus, $|SU| \leq \frac{\lambda}{6}$.

□

Hybrid H_3: This hybrid is identical to the previous except in the way prover computes $\{\mathsf{dwi}_{3,i}^{P\to V}\}_{i \notin SU}$ in the last round. Note that if $i \notin SU$, $c_i = \mathsf{Com}(1)$. Hence, the prover now has an alternative witness to prove the statement. The prover switches to using this witness to compute $\{\mathsf{dwi}_{3,i}^{P\to V}\}_{i \notin SU}$ in all executions.

Now observe the following. By definition of SU, if $i \notin SU$, then the message $\{\mathsf{dwi}_{3,i}^{P\to V}\}_{i \notin SU}$ is actually required to be sent in *at most* one execution. That is, $i \notin SU$, the i-th parallel instance of DIWI is only executed at most once (without any rewinding). Hence, the indistinguishability of the view of V^* between H_2 and H_3 follows from the witness indistinguishability of DIWI.

Hybrid H_4: We now define a set $S_{leak} \subset [\lambda]$ of the views as follows. Start with an empty S_{leak}. For all $k_{i,j} \in SU$, add i and j to S_{leak}. Clearly, since $|SU| \leq \frac{\lambda}{6}$, it follows that $|S_{leak}| \leq \frac{\lambda}{3}$.

This hybrid is identical to the previous except now for all $i \notin S_{leak}$, the prover sets cv_i to be $\mathsf{Com}(0)$ as opposed to $\mathsf{Com}(V_i)$ (in all executions). Now observe that if $i \notin S_{leak}$, the opening of cv_i was not being used as a witness in any DIWI execution. This is because any DIWI instance which could have used V_i has already been switched to using the alternate witness. Thus, the indistinguishability of the view of V^* between H_3 and H_4 directly follows from the hiding of the commitment scheme Com.

Hybrid H_5: This hybrid is identical to the previous one except in how the views are computed by the prover in the last round. We note that in each rewound execution, the prover only needs to construct a view V_i if $i \in S_{leak}$. However since $|S_{leak}| \leq \frac{\lambda}{3}$, the prover needs to construct at most $\frac{\lambda}{3}$ views. The prover stops using the supplied witness at this point and instead starts using the MPC simulator to generate all the required views. Observe that we are using an MPC protocol with perfect correct and perfect security which is capable to simulating the view of up to $\frac{\lambda}{3}$ players. Thus, the indistinguishability of the view of V^* between H_4 and H_5 follows from indistinguishability of real and simulated views in the underlying MPC construction.

We now observe that in hybrid H_5, our prover is no longer using the supplied witnesses in any of the B execution. Hence, our construction RWI is, in fact, zero-knowledge under B rewinds. This in particular implies that our construction satisfies the notion of WI with bounded rewind security. We also note that although not necessary in our application, the parallel repetition of RWI can also be shown to have the proof of knowledge property.

B Bidirectional to Alternating message model

In [GMPP16b] the authors prove that there does not exist a 3-round protocol in the bidirectional message model for tossing $\omega(\log \lambda)$ coins which can be proven secure via blackbox simulation. To prove this theorem, the authors show how to reschedule a 3-round protocol in the bidirectional message into a 4-round non-simultaneous protocol thus contradicting the impossibility of [KO04]. In this section we extend the proof approach used in [GMPP16b] to show the following. Let $\Pi^{\leftrightarrow} = (A^{\leftrightarrow}, B^{\leftrightarrow})$ be a k-round two-party protocol (2PC) that securely computes the function f in the bidirectional message model. f takes the inputs of the parties $A^{\leftrightarrow}$ and $B^{\leftrightarrow}$, that we denote with $x_{A^{\leftrightarrow}}$ and $x_{B^{\leftrightarrow}}$ respectively, and outputs $y_{A^{\leftrightarrow}}$ and $y_{B^{\leftrightarrow}}$, where $y_{A^{\leftrightarrow}}$ corresponds to the output of $A^{\leftrightarrow}$ and $y_{B^{\leftrightarrow}}$ corresponds to the output of $B^{\leftrightarrow}$. We show how to obtain a k-round 2PC protocol $\Pi^{\leftrightarrows} = (A, B)$ in the alternating message model, in which *at least* one party gets the output.

Theorem 7. *Any k-round two party protocol (2PC) $\Pi^{\leftrightarrow}$ that securely computes f in the bidirectional message model, proven secure via blackbox simulation, can be turned into a k-round two party protocol in the alternating message model, in which* at least *one party gets the output of f.*

Proof. We show how to obtain a k-round 2PC protocol $\Pi^{\leftrightarrows} = (A, B)$, in the alternating message model, that securely computes f in which only one party gets the output. Without loss of generality, we assume that only the party B gets the output. We denote with m_i^A the message that the party $A^{\leftrightarrow}$ sends in the i-th round of $\Pi^{\leftrightarrow}$, and with m_i^B the message that the party $B^{\leftrightarrow}$ in the i-th round of $\Pi^{\leftrightarrow}$ with $1 \leq i \leq k$.

In $\Pi^{\leftrightarrows}$ the party A computes its messages by internally running $A^{\leftrightarrow}$, and the same does B with $B^{\leftrightarrow}$. We provide an high level description of $\Pi^{\leftrightarrows}$ in Fig 7. The main observation that makes possible to reschedule the messages of $\Pi^{\leftrightarrow}$ in the alternating message model is that the message that $B^{\leftrightarrow}$ sends to $A^{\leftrightarrow}$ in the last round of $\Pi^{\leftrightarrow}$ can be removed given that A does not need to compute the output. Moreover, the security of $\Pi^{\leftrightarrow}$ is proved by considering a rushing adversary. This means that a message that an honest party sends in the round i-th of $\Pi^{\leftrightarrow}$ has to be independent from the message that the other party sends in the i-th round. We propose a more formal description of $\Pi^{\leftrightarrows}$ in Fig. 8.

We start by considering the case in which B is corrupted (we denote a corrupted party P with $P^{\star}$). Then we need to build an expected PPT simulator $\mathcal{S}^{\leftrightarrows}$ that satisfies the Definition 1. Since $\Pi^{\leftrightarrow}$ is secure, then there exists a simulator $\mathcal{S}^{\leftrightarrow}$ in the ideal world for any corrupt $B^{\leftrightarrow\star}$ executing the simultaneous message exchange protocol $\Pi^{\leftrightarrow}$. Our simulator $\mathcal{S}^{\leftrightarrows}$ is constructed using $\mathcal{S}^{\leftrightarrow}$ and works as follows.

1. $\mathcal{S}^{\leftrightarrows}$, upon receiving m_1^B from $B^{\star}$, forwards m_1^B to $\mathcal{S}^{\leftrightarrow}$. $\mathcal{S}^{\leftrightarrow}$ outputs (m_1^A, m_2^A) (note that the inner simulator must be able to produce m_2^A even before seeing the second message m_2^B of party $B^{\leftrightarrow\star}$ given that the $B^{\leftrightarrow\star}$ is rushing). Moreover, $\mathcal{S}^{\leftrightarrows}$ acts as a proxy between $\mathcal{S}^{\leftrightarrow}$ and the ideal functionality and whenever $\mathcal{S}^{\leftrightarrow}$ asks to rewind the adversary, $\mathcal{S}^{\leftrightarrows}$ rewinds $B^{\star}$

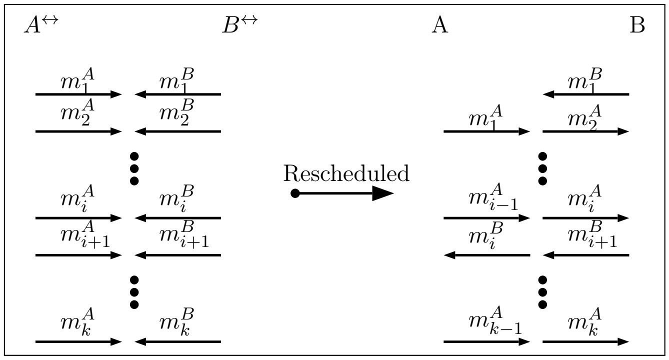


Figure 7: High level description of the rescheduled messages.

Round-1: B runs $\Pi^{\leftrightarrow}$ on behalf of $B^{\leftrightarrow}$ thus obtaining the m_1^B and sends it to A

Round-i with $1 < i < k$, $i \mod 2 = 0$: A runs $\Pi^{\leftrightarrow}$ on behalf of $A^{\leftrightarrow}$ to compute the messages (m_{i-1}^A, m_i^A) and sends them to B .

Round-i with $1 < i < k, i \mod 2 \neq 0$: B runs $\Pi^{\leftrightarrow}$ on behalf of $B^{\leftrightarrow}$ to compute the messages (m_{i-1}^B, m_i^B) and sends them to A.

Round-k: A runs $\Pi^{\leftrightarrow}$ on behalf of $A^{\leftrightarrow}$ to compute the messages (m_{k-1}^A, m_k^A) and sends them to B.

Figure 8: $\Pi^{\leftrightarrows}$ description.

2. Upon receiving the message $m_i = (m, m')$ from $B^\star$ in the i-th round, with $1 < i < k-1$, $\mathcal{S}^{\leftrightarrows}$, sends m to $\mathcal{S}^{\leftrightarrow}$, receives m_{i-1}^A and sends also m' to $\mathcal{S}^{\leftrightarrow}$. $\mathcal{S}^{\leftrightarrow}$ now outputs m_i^A which $\mathcal{S}^{\leftrightarrows}$ sends to $B^\star$.

3. Upon receiving the message $m_{k-1} = (m, m')$ from $B^\star$ in the k-th round, $\mathcal{S}^{\leftrightarrows}$ sends m to $\mathcal{S}^{\leftrightarrow}$, receives m_{k-1}^A and sends also m' to $\mathcal{S}^{\leftrightarrow}$. $\mathcal{S}^{\leftrightarrow}$ now outputs m_k^A and $\mathcal{S}^{\leftrightarrows}$ sends (m_{k-1}, m_k) to $B^\star$. In the end $\mathcal{S}^{\leftrightarrows}$ sends an abort message to $\mathcal{S}^{\leftrightarrow}$ (to indicate that the adversary has not sent the last message) and outputs what $\mathcal{S}^{\leftrightarrow}$ outputs.

It should be easy to see that $\mathcal{S}^{\leftrightarrows}$ emulates correctly $B^{\leftrightarrow\star}$ and hence $\mathcal{S}^{\leftrightarrows}$ represents a good adversary for the ideal world. The proof for the case in which A is corrupted is similar, with the difference that the last message output by the inner simulator $\mathcal{S}^{\leftrightarrow}$ is not forwarded to $A^\star$. $\square$