Round Optimal Concurrent MPC via Strong Simulation

Saikrishna Badrinarayanan
UCLA
saikrishna@cs.ucla.edu

Dakshita Khurana
UCLA
dakshita@cs.ucla.edu

Amit Sahai
UCLA
sahai@cs.ucla.edu

Abstract

In this paper, we study the round complexity of concurrently secure multi-party computation (MPC) with super-polynomial simulation (SPS) in the plain model. In the plain model, there are known explicit attacks that show that concurrently secure MPC with polynomial simulation is impossible to achieve; SPS security is the most widely studied model for concurrently secure MPC in the plain model. We obtain the following results:

- Three-round concurrent MPC with SPS security against Byzantine adversaries, assuming sub-exponentially secure DDH and LWE.
- A barrier to constructing stand-alone two-round MPC for general functionalities, achieving any notion of meaningful security against Byzantine adversaries, from standard falsifiable assumptions.
- This barrier breaks down in the setting of input-less functionalities; where we construct two-round concurrent MPC with SPS security against Byzantine adversaries for input-less functionalities, assuming sub-exponentially secure indistinguishability obfuscation and DDH. In particular, this class includes sampling functionalities that allow parties to jointly sample a secure common reference string for cryptographic applications.

Prior to our work, to the best of our knowledge, concurrent MPC with SPS security required roughly 20 rounds, although we are not aware of any work that even gave an approximation of the constant round complexity sufficient for the multi-party setting. We also improve over the previous best round complexity for the two-party setting, where 5 rounds were needed (Garg, Kiyoshima, and Pandey, Eurocrypt 2017).

To obtain our results, we compile protocols that already achieve security against "semi-malicious" adversaries, to protocols secure against fully malicious adversaries, additionally assuming sub-exponential DDH. Our protocols develop new techniques to use two-round zero-knowledge with super-polynomial *strong* simulation, defined by Pass (EUROCRYPT 2003) and very recently realized by Khurana and Sahai (Eprint 2017). These remain zero-knowledge against adversaries running in time larger than the running time of the simulator.

Contents

1 Introduction

The round complexity of secure multi-party computation (MPC) [Yao82, Yao86, GMW87] has been a problem of fundamental interest in cryptography. The last few years have seen major advances in improving the round complexity of secure computation with dishonest majority [BMR90, KOS03, Pas04, DI05, DI06, PPV08, Wee10, Goy11, LP11, GLOV12, GMPP16, ACJ17, BHP17], culminating eventually in four round protocols for secure multi-party computation from general assumptions such as DDH and LWE [GMPP16, ACJ17, BHP17].

Intriguingly, however, when we only require security against (semi-malicious) adversaries that follow protocol specifications, recent research has also constructed MPC protocols that require even less that four rounds of simultaneous message exchange in the plain model. For instance, [DHRW16] give a two-round protocol based on indistinguishability obfuscation, while [BHP17] very recently gave a three round protocol from the hardness of the learning with errors assumption.

However, these protocols do not offer any privacy guarantees at all against Byzantine adversaries that may deviate from protocol specifications. Can we achieve meaningful security against Byzantine adversaries in two or three rounds? This question is even more interesting in the setting where parties participate in multiple executions of the MPC protocol concurrently. Indeed, as our world becomes increasingly interconnected, it is hard to imagine that future cryptographic protocols will be carried out in a standalone setting, where participants interact in only a single instance of the protocol. Thus, we ask:

"Can we achieve concurrently secure MPC in two or three rounds?"

Super-polynomial security. Indeed, even defining security against concurrent adversaries in the plain model requires care. Barak, Prabhakaran and Sahai [BPS06] give an explicit "chosen protocol attack" that rules out concurrently secure MPC with polynomial simulation in *any number of rounds* in the plain model. In fact, even in the stand-alone setting, three round secure computation with polynomial simulation and black-box reductions turns out to be impossible to achieve [GMPP16].

However, it has long been known that for MPC, a powerful security notion in the plain model is security with super-polynomial time simulation (SPS) [Pas03, PS04, BS05, MMY06, CLP10, GGJS12, GKP17, BGI$^+$17, KS17]. SPS security circumvents the impossibility results above including the chosen protocol attack in the concurrent setting, and is the most widely studied security model for concurrent MPC in the plain model.

To understand the intuition behind SPS security, it is instructive to view SPS security through the lens of the security loss inherent in all security reductions. In ordinary polynomial-time simulation, the security reduction has a polynomial security loss with respect to the ideal world. That is, an adversary in the real world has as much power as another adversary that runs in polynomially more time in the ideal world. In SPS security, the security reduction has a fixed super-polynomial security loss, for example 2^{n^ϵ}, where n is the security parameter, with respect to the ideal world. Just as in other applications in cryptography using super-polynomial assumptions, this situation still guarantees security as long as the ideal model is itself super-polynomially secure. For instance, if the ideal model hides honest party inputs *information-theoretically*, then security is maintained even with SPS. For example, this is true for applications like online auctions, where no information is leaked in the ideal world about honest party inputs beyond what can be easily computed from the output. But SPS also guarantees security for ideal worlds with cryptographic outputs, like blind signatures, as long as the security of the cryptographic output is guaranteed against super-polynomial adversaries. Indeed, SPS security was explicitly considered for blind signatures in [GRS$^+$11, GG14] with practically relevant security parameters computed in [GG14]. Additional discussion on the meaningfulness of SPS security can be found in the original works of [Pas03, PS04] that introduced SPS security in the protocol context.

Prior to our work, the best round complexity even for concurrent two-party computation with SPS security was 5 rounds [GKP17] from standard sub-exponential assumptions. For concurrent MPC with SPS security from standard sub-exponential assumptions, the previous best round complexity was perhaps approximately 20 rounds in the simultaneous message exchange model [GGJS12, KMO14], although to the best of our knowledge, no previous work even gave an approximation of the constant round complexity that is sufficient for the multi-party setting.

1.1 Our Results

We obtain several results on concurrently secure MPC in 2 or 3 rounds:

1. We obtain the following results for multi-party secure computation with SPS in *three rounds* in the simultaneous message model, against rushing adversaries.

   - A compiler that converts a large class of three round protocols secure against semi-malicious adversaries, into protocols secure against malicious adversaries, additionally assuming the sub-exponential hardness of DDH or QR or N^{th} residuosity.
   - A compiler that converts a large class of three round protocols secure against semi-malicious adversaries, into protocols secure against malicious *concurrent* adversaries, additionally assuming the sub-exponential hardness of DDH or QR or N^{th} residuosity.

   On instantiating these compilers with the three-round semi-malicious protocol in the recent work of Brakerski et al. [BHP17], we obtain the following main result.

   Informal Theorem 1. *Assuming sub-exponentially secure LWE and DDH, there exists a three-round protocol in the simultaneous message exchange model with rushing adversaries, that achieves sub-exponential concurrent SPS security for secure multi-party computation for any efficiently computable function, in which all parties can receive output.*

   The same result holds if the sub-exponential DDH assumption above is replaced with the sub-exponential QR or N^{th} residuosity assumptions.

2. A natural next question is: can we achieve security against malicious adversaries in just two rounds, in *any* meaningful setting? We first observe a barrier to constructing protocols with any security guarantees against non-uniform malicious adversaries, in just two rounds.

   We prove that any two-round MPC protocol in the simultaneous exchange model with security against non-uniform adversaries would imply a form of non-interactive non-malleable commitments with security against non-uniform adversaries. Achieving such a non-interactive non-malleable commitment from standard falsifiable sub-exponential assumptions suffers from known barriers and limitations [Pas13]. This holds not only for super-polynomial simulation, but essentially any meaningful notion of security against malicious adversaries.

3. On the other hand, somewhat surprisingly, we obtain the following results for multi-party secure computation with SPS in *two rounds* in the simultaneous message model, against rushing adversaries.

   - A compiler that converts a large class of two round protocols secure against semi-malicious adversaries, into protocols secure against malicious adversaries computing input-less *randomized* functionalities, assuming assuming sub-exponential hardness of DDH and indistinguishability obfuscation.

- A compiler that converts a large class of two round protocols secure against semi-malicious adversaries, into protocols secure against *concurrent* malicious adversaries computing input-less *randomized* functionalities, assuming assuming sub-exponential hardness of DDH and indistinguishability obfuscation.

On instantiating these compilers with the two-round semi-malicious protocol in [DHRW16], we obtain the following main result.

Informal Theorem 2. *Assuming sub-exponentially secure indistinguishability obfuscation and DDH, there exists a two-round protocol in the simultaneous message exchange model with rushing adversaries, that achieves sub-exponential concurrent SPS security for secure multi-party computation for any efficiently computable randomized input-less function, in which all parties can receive output.*

In particular, our protocols can be used to generate samples from any efficiently sampleable distribution. For example, they can be used to concurrently securely sample common reference strings from arbitrary distributions for cryptographic applications, such that the randomness used for sampling remains hidden as long as at least one of the participants is honest. Applications include generating a common reference string sufficient for building universal samplers [HJK+16]. Before our work, only the special case of multi-party coin-flipping with SPS was known to be achievable in two rounds [KS17].

2 Technical Overview

We will now give an overview of the techniques used in our work.

2.1 Three Round MPC Without Setup

A well established approach to constructing secure computation protocols against malicious adversaries in the standalone setting is to use the GMW compiler [GMW87]: compile a semi-honest protocol with zero-knowledge arguments to enforce correct behavior. Normally, such compilers involve an initial 'coin-tossing' phase, which determines the randomness that will be used by all parties in the rest of the protocol. Unfortunately, in two or three rounds, there is no scope at all to carry out an initial coin-tossing.

However, as observed by [AJL+12, MW16, BHP17], certain two and three round protocols satisfy semi-malicious security: that is, the protocol remains secure even when the adversary is allowed to chose malicious randomness, as long as the adversary behaves according to protocol specifications. When compiling semi-malicious protocols, the coin-tossing phase is no longer necessary: at a very high level, it seems like it should suffice to have all parties give proofs of correct behavior. Several difficulties arise when trying to implement such compilers in extremely few rounds. Specifically, in many parts of our protocols, we will have *only* two rounds to complete the proof of correct behavior. However, attempts to use two-round zero-knowledge with super-polynomial simulation [Pas03] run into a few key difficulties, that we now discuss.

A key concern in MPC is that malicious parties may be arbitrarily mauling the messages sent by other parties. In order to prevent this, we will use two-round non-malleable commitments, that were recently constructed in [KS17, LPS17]. In particular, we will rely on a construction of two-round concurrent non-malleable commitments with simultaneous messages, that were constructed by [KS17] assuming sub-exponential DDH.

The very first difficulty arises as soon as we try to compose non-malleable commitments with SPS-ZK.

Difficulty of using two-round SPS-ZK in few rounds with Simultaneous Messages. Standard constructions of two-round SPS zero-knowledge can be described as follows: the verifier generates a challenge that is hard to invert by adversaries running in time T, then the prover proves (via WI) that either the statement being proven is in the language, or that he knows the inverse of the challenge used by the verifier. This WI argument is such that the witness used by the prover can be extracted (via brute-force) in time $T' \ll T$. Naturally, this restricts the argument to be zero-knowledge against verifiers that run in time $T_{\mathsf{zk}} \ll T' \ll T$.

Thus, if a prover generates an accepting proof for a false statement, the WI argument can be broken in time T' to invert the challenge, leading to a contradiction. On the other hand, there exists a simulator that runs in time $T_{\mathsf{Sim}} \gg T$ to invert the receiver's challenge and simulate the proof (alternatively, such a simulator can non-uniformly obtain the inverse of the receiver's challenge). Thus, we have $T_{\mathsf{Sim}} \gg T_{\mathsf{zk}}$.

Let us now consider an SPS-ZK protocol, run simultaneously with a non-malleable commitment, as illustrated in Figure 1. The two-round concurrent non-malleable commitment scheme from [KS17] requires the committer and receiver to send simultaneous messages in the first round of the exection, followed by a single message from the committer in the second round.

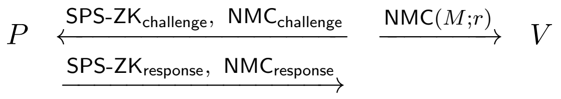
P
SPS-ZK$_{\mathsf{challenge}}$, NMC$_{\mathsf{challenge}}$
NMC$(M;r)$
V
SPS-ZK$_{\mathsf{response}}$, NMC$_{\mathsf{response}}$

Figure 1: Composing SPS-ZK with Non-malleable commitments

Let us also imagine that multiple parties running such a protocol are sending non-malleable commitments to their inputs, together with messages of the underlying semi-malicious protocol, and SPS-ZK proofs of correct behavior.

In order to begin a reduction between the real and ideal worlds, we would have to begin by simulating the proofs sent by honest parties, and then argue that adversarial parties cannot maul honest parties' inputs. However, while arguing non-malleability, *we cannot simulate proofs non-uniformly*, since that would end up also non-uniformly fixing the messages of the non-malleable commitments. Thus, we would want non-malleability of NMCom to hold even while we are sending simulated proofs in time T_{Sim}.

On the other hand, when we switch a real SPS ZK proof to being simulated, we must argue that the values within the non-malleable commitments provided by the adversary did not suddenly change. To achieve this, it must be true that the quality of the SPS ZK simulation is sufficiently high to guarantee that the messages inside the non-malleable commitments did not change. Specifically, we must be able to break the non-malleable commitments and extract from them in time that is less than T_{zk}. Putting together all these constraints, we have that non-malleable commitments should be breakable in time that is less than the time against which they remain non-malleable: this is a direct contradiction.

In order to solve this problem, we must rely on ZK argument systems where the quality of the SPS ZK simulation *exceeds* the running time of the SPS simulator, namely where $T_{\mathsf{Sim}} \ll T_{\mathsf{zk}}$. Zero-knowledge with strong simulation ([Pas03]), is roughly a primitive that satisfies exactly this constraint. We call such a ZK protocol an SPSS-ZK argument. Such a primitive was recently realized by [KS17], by constructing a new form of two-round extractable commitments. Note that if one uses SPSS-ZK instead of SPS-ZK, the contradiction described above no longer holds. This is a key insight that allows us to have significantly simpler arguments of SPS security, especially in the concurrent security setting.

However, as we already mentioned, in arguing security against malicious adversaries, we must be particularly wary of malleability attacks. In particular, we would like to ensure that while the simulator provides simulated proofs, the adversary continues to behave honestly – thereby allowing such a simulator to correctly extract the adversary's input and force the right output. This is the notion of simulation soundness [Sah99]. However, it is unknown how to build a two-round concurrently simulation-sound SPSS ZK argument. We address this by providing a mechanism to emulates two-round and three-round simulation-soundness via strong simulation, in a simultaneous message setting. This mechanism allows us to compile a semi-malicious protocol with a type of non-malleable proofs of honest behavior.

Roughly speaking, the idea behind our strategy for enforcing simulation soundness is to have each party commit not only to its input, but also all the randomness that it will use in the underlying semi-malicious secure protocol. Then, the high quality of the SPSS ZK simulation will ensure that even the joint distribution of the input, the randomness, and the protocol transcript cannot change when we move to SPS simulation. Since honest behavior can be checked by computing the correct messages using the input and randomness, the quality of the SPSS ZK simulation guarantees that adversarial behavior must remain correct. Counter-intuitively, we enforce a situation where we cannot rule out that the adversary isn't "cheating" on his ZK arguments, but nevertheless the adversary's behavior in the underlying semi-malicious MPC protocol cannot have deviated from honest behavior.

We note that our simulation strategy is uniform and straight-line. The only non-trivial use of rewinding in our protocol is in arguing non-malleability, and this is abstracted away into the underlying non-malleable commitment scheme that we invoke. This leads to a significantly simpler proof of concurrent security.

Several additional subtleties arise in the proofs of security. Please refer to Section 4 for additional details on our protocol and complete proofs.

2.2 Barriers to Two Round Secure Computation of General Functionalities

We now show that barriers exist to constructing two-round two-party SPS-secure computation of general functionalities with super-polynomial simulation, where both parties receive the output.

At first, it may seem that the beautiful round-collapsing argument in [GMPP16], for the case of multi-party functionalities, may already help us achieve a lower bound. We now recall their technique briefly. [GMPP16] prove that any k-round protocol in the simultaneous message can be "collapsed" to a $(k + 1)$-round two-party protocol with sequential messages, where both parties receive the output. This rules out protocols with less than 4 messages and standard simulation security, from black-box reductions.

While the arguments of [GMPP16] seem to follow even for the setting of super-polynomial simulation, all that this argument says in the case of 2-round SPS protocols is that any 2 round protocol can be "collapsed" to a 3-round two-party protocol with sequential messages, where both parties receive the output – and indeed, 3 round protocols with SPS, where both parties receive the output, *are possible* with sequential messages: as recently proven in [BGI+17, JKKR17].

In fact, even two-message protocols with sequential messages can be constructed [BGI+17, JKKR17] if only one party is required to receive the output. It may appear that, at least in the two-party setting, it should be possible to execute two such protocols simultaneously and in parallel with each other, just like the polynomial simulation setting, a la [GMPP16]. We show that this intuition runs up against a barrier that arises due to malleability concerns.

We will focus on protocols for the secure computation of a specific functionality $\mathcal{F}(x, y) = (x+y)$, which computes the sum of the inputs of both parties, when interpreted as natural numbers. However, our arguments also extend to all functionalities that are sensitive to the private inputs of individual parties.

For the purpose of this overview, we will restrict ourselves to two-round protocols where both parties send an encoding of their message in the first round while the next round is used to compute the output. It is not difficult to see that any protocol for two-round two-party secure computation of general functionalities, must satisfy this property, as long as security must hold against non-uniform adversaries. If the first message wasn't committing, then a non-uniform adversary could obtain a first message that is consistent with two inputs, and then by aborting in the second round, it could obtain two outputs of the function with two different inputs, violating security.

Two-Round Two-Party Secure Computation. Let Π denote a two-round secure computation protocol between parties A and B, where both parties receive the output. We will also consider a "mauling" rushing adversary that corrupts B, let us denote this corrupted party by $\widetilde{B}$. At the beginning of the protocol A sends an honest encoding of its input X. After obtaining the first round message from party A, suppose that $\widetilde{B}$ "mauls" the encoding sent by A and generates another encoding of the same input X. Because the encodings must necessarily hide the inputs of parties, the honest PPT party A cannot detect if such a mauling occurred, and sends the second message of the protocol. At this point, $\widetilde{B}$ generates its second round message on its own, *but does not send this message.* Instead, $\widetilde{B}$ computes the output of the protocol (which is guaranteed by correctness). The adversary $\widetilde{B}$ learns $2X$, and blatantly breaks security of the SPS-secure protocol. Similarly, a rushing adversary could choose to corrupt party A and launch the same attack. By security of the protocol, there should exist a (superpolynomial-time) reduction R that uses such an adversary to break an underlying challenge C.

Reduction to Non-Interactive Non-Malleable Commitments. Suppose there exists such a reduction, then we consider the following construction NMC of non-interactive non-malleable commitments with respect to commitment for two identities, that prevents copying attacks. If identity $= 0$, then the committer uses A's identity and the description of A's input encoding algorithm to generate his commitment. If identity $= 1$, then the committer uses B's identity and the description of B's input encoding algorithm to generate his commitment.

The resulting commitment NMC must be binding, by correctness of the underlying two-party computation protocol, and hiding against PPT adversaries, because of the input-hiding property of the encoding in the case where the protocol aborts after the first round. Furthermore, NMC must prevent mauling attacks. Consider a MIM that chooses identity 0 for the honest interaction and identity 1 for the man-in-the-middle's interaction. Then, given any MIM that successfully copies, the reduction $\mathcal{R}$ that is guaranteed by the two-party computation protocol against a corrupted $\widetilde{B}$ can be used to break the underlying challenge C. Similarly, if the MIM picks identity 1 for the honest interaction and 0 for the man-in-the-middle's interaction, then $\mathcal{R}$ that is guaranteed by the two-party computation protocol against a corrupted $\widetilde{A}$ can be used to break the underlying challenge C. However, Pass [Pas13, Pas16] proved the impossibility of building non-interactive non-malleable commitments from standard "falsifiable" sub-exponential assumptions, with black-box super-polynomial reductions[1]. Finally, as we describe in the next sub-section, not all is lost in the two round setting.

2.3 Two Round MPC without Setup for Input-Less Randomized Functionalities

We begin by noting that the discussion above on the hardness of two-round MPC with super-polynomial simulation does not rule out functionalities that are *not* sensitive to the private inputs of parties. In particular, let us consider input-less randomized functionalities. Even though the functionality is input-less, still each party must contribute to selecting the secret randomness on which the function is to be evaluated. At first glance, it may appear that we still have the same problem: in only two rounds, perhaps this "implied input" can be compromised. However, note that for input-less functionalities, if the adversary aborts, then even if the adversary learns the "implied inputs" of the honest parties, this does not violate security because the honest parties will not accept the output of the protocol. Thus, the honest parties' contributions to the randomness is discarded since the protocol execution is aborted. As such, we only need to guarantee security of the honest party inputs *if* the protocol terminates correctly – that is, if the adversary is able to send second-round messages that do not cause the protocol to abort.

More technically, the only actual requirement is that a super-polynomial simulator must be able to correctly and indistinguishably, force the output of the computation to an externally generated value. The security of each honest party's contribution to the randomness is implied by this forcing.

We show that this is indeed possible using only two rounds of interaction in the simultaneous message model, under suitable cryptographic assumptions. We describe a compiler that compiles a large class of two-round secure computation protocols for input-less randomized functionalities from semi-malicious to full malicious (and even concurrent) security. We consider functionalities where each party contributes some randomness, and the joint randomness of all parties is used to sample an output from some efficiently sampleable distribution.

Our protocol follows a similar template to the protocol described for the 3-round case: parties first commit to all the input and randomness that they will use throughout the execution via a non-malleable commitment. Simultaneously, parties run an underlying two-round semi-malicious protocol and by the end of the second round, provide SPSS-ZK proofs that they correctly computed all messages. We stress again that it is *only* if the adversary successfully completes both rounds of the protocol without causing an abort, that we actually need to care about hiding the shares of randomness contributed by honest parties – in order to argue overall security.

At the same time, in order to enforce correctness, the simulator would still need to extract the randomness used by the adversary at the end of the first round of the computation. Unlike our three round protocol, here, the simulator will try to extract randomness at the end of the first round anyway. This is because the simulator can afford to be optimistic: Either its extraction is correct, and it can make use of this in forcing the output. Or its extraction is incorrect, but in this case we will guarantee that the adversary will cause the protocol to abort in the second round because of the SPSS ZK argument that the adversary must give proving that it behaved honestly in the first round.

We need to take additional care when defining the simulation strategy when the simulator

[1] We note that Pass [Pas13, Pas16] rules out reductions that make polynomially many oracle queries to the adversary. In the fully non-interactive setting, additional oracle queries do not appear to be useful because the reduction is also required to extract from a committer that only generates responses from a fixed polynomial-sized set.

extracts incorrect randomness: this causes other subtleties in our proof of security. The complete constructions and proofs of standalone as well as concurrent security, can be found in Section 5.

3 Preliminaries

Here, we recall some preliminaries that will be useful in the rest of the paper. We will typically use n to denote the security parameter. We will say that $T_1(n) \gg T_2(n)$ if $T_1(n) > T_2(n) \cdot n^c$ for all constants c.

We define a T-time machine as a non-uniform Turing Machine that runs in time at most T. All honest parties in definitions below are by default uniform interactive Turing Machines, unless otherwise specified.

3.1 ZK With Superpolynomial Simulation.

We will use two message ZK arguments with strong superpolynomial simulation (SPS) and with super-polynomial strong simulation (SPSS) [Pas04].

Definition 1 (Two Message $(T_{\mathsf{Sim}}, T_{\mathsf{zk}}, \delta_{\mathsf{zk}})$-ZK Arguments With Superpolynomial Simulation)**.** *[Pas04] We say that an interactive proof (or argument) $\langle P, V\rangle$ for the language $L \in \mathsf{NP}$, with the witness relation R_L, is $(T_{\mathsf{Sim}}, T_{\mathsf{zk}}, \delta_{\mathsf{zk}})$-simulatable if for every T_{zk}-time machine V^* exists a probabilistic simulator $\mathcal{S}$ with running time bounded by T_{Sim} such that the following two ensembles are $T_{\mathsf{zk}}, \delta_{\mathsf{zk}})$-computationally indistinguishable (when the distinguishing gap is a function in $n = |x|$):*

- $\{(\langle P(y), V^*(z)\rangle(x))\}_{z\in\{0,1\}^*, x\in L}$ *for arbitrary $y \in R_L(x)$*
- $\{\mathcal{S}(x, z)\}_{z\in\{0,1\}^*, x\in L}$

That is, for every probabilistic algorithm D running in time polynomial in the length of its first input, every polynomial p, all sufficiently long $x \in L$, all $y \in R_L(x)$ and all auxiliary inputs $z \in \{0,1\}^*$ it holds that

$$\Pr[D(x, z, (\langle P(y), V^*(z)\rangle(x)) = 1] - \Pr[D(x, z, S(x, z)) = 1] < \delta_{\mathsf{zk}}(\lambda)$$

Definition 2. *We say that a two-message $(T_{\mathsf{Sim}}, T_{\mathsf{zk}}, \delta_{\mathsf{zk}})$-SPS ZK argument satisfies non-uniform simulation (for delayed statements) if we can write the simulator $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ where $\mathcal{S}_1(V^*(z))$, which outputs σ, runs in T_{Sim}-time, but where $\mathcal{S}_2(x, z, \sigma)$, which outputs the simulated view of the verifier V^*, runs in only polynomial time.*

3.2 ZK with Super-polynomial Strong Simulation

We now define zero-knowledge with strong simulation. We use the definition in [KS17].

Definition 3 ($(T_\Pi, T_{\mathsf{Sim}}, T_{\mathsf{zk}}, T_L, \delta_{\mathsf{zk}})$-SPSS Zero Knowledge Arguments)**.** *We call an interactive protocol between a PPT prover P with input $(x, w) \in R_L$ for some language L, and PPT verifier V with input x, denoted by $\langle P, V\rangle(x, w)$, a super-polynomial strong simulation (SPSS) zero-knowledge argument if it satisfies the following properties and $T_\Pi \ll T_{\mathsf{Sim}} \ll T_{\mathsf{zk}} \ll T_L$:*

- **Completeness.** *For every $(x, w) \in R_L$, $\Pr[V$ outputs $1|\langle P, V\rangle(x, w)] \geq 1 - \mathsf{negl}(\lambda)$, where the probability is over the random coins of P and V.*

- T_Π-**Adaptive-Soundness.** *For any language L that can be decided in time at most T_L, every x, every $z \in \{0,1\}^*$, and every poly-non-uniform prover P^* running in time at most T_Π that chooses x adaptively after observing verifier message,* $\Pr[\langle P^*(z), V\rangle(x) = 1 \ \wedge \ x \notin L] \leq \mathsf{negl}(\lambda)$*, where the probability is over the random coins of $\mathcal{V}$.*
- $T_{\mathsf{Sim}}, T_{\mathsf{zk}}, \delta_{\mathsf{zk}}$-**Zero Knowledge.** *There exists a (uniform) simulator $\mathcal{S}$ that runs in time T_{Sim}, such that for every x, every non-uniform T_{zk}-verifier V^* with advice z, and every T_{zk}-distinguisher $\mathcal{D}$:* $\left|\Pr[\mathcal{D}(x, z, \mathsf{view}_{V^*}[\langle P, V^*(z)\rangle(x, w)]) = 1] \ - \Pr[\mathcal{D}(x, z, \mathcal{S}^{V^*}(x, z)) = 1]\right| \leq \delta_{\mathsf{zk}}(\lambda)$

3.3 Non-Malleability w.r.t. Commitment

Throughout this paper, we will use λ to denote the security parameter, and $\mathsf{negl}(\lambda)$ to denote any function that is asymptotically smaller than $\frac{1}{\mathsf{poly}(\lambda)}$ for any polynomial $\mathsf{poly}(\cdot)$. We will use PPT to describe a probabilistic polynomial time machine. We will also use the words "rounds" and "messages" interchangeably.

We follow the definition of non-malleable commitments introduced by Pass and Rosen [PR05] and further refined by Lin et al [LPV08] and Goyal [Goy11] (which in turn build on the original definition of [DDN91]). In the real interaction, there is a man-in-the-middle adversary MIM interacting with a committer $\mathcal{C}$ (where $\mathcal{C}$ commits to value v) in the left session, and interacting with receiver $\mathcal{R}$ in the right session. Prior to the interaction, the value v is given to C as local input. MIM receives an auxiliary input z, which might contain a-priori information about v. Then the commit phase is executed. Let $\mathsf{MIM}_{\langle C,R\rangle}(\mathsf{val}, z)$ denote a random variable that describes the value $\widetilde{\mathsf{val}}$ committed by the MIM in the right session, jointly with the view of the MIM in the full experiment. In the simulated experiment, a PPT simulator $\mathcal{S}$ directly interacts with the 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 $\mathcal{S}$. If the tags in the left and right interaction are equal, the value $\widetilde{\mathsf{val}}$ committed in the right interaction, is defined to be $\perp$ in both experiments.

Concurrent non-malleable commitment schemes consider a setting where the MIM interacts with committers in polynomially many (a-priori unbounded) left sessions, and interacts with receiver(s) in upto $\ell(n)$ right sessions. If any of the tags (in any right session) are equal to any of the tags in any left session, we set the value committed by the MIM to $\perp$ for that session. The we let $\mathsf{MIM}_{\langle C,R\rangle}(\mathsf{val}, z)^{\mathsf{many}}$ denote the joint distribution of all the values committed by the MIM in all right sessions, together with the view of the MIM in the full experiment, and $\mathsf{Sim}_{\langle C,R\rangle}(1^\lambda, z)^{\mathsf{many}}$ denotes the joint distribution of all the values committed by the simulator $\mathcal{S}$ (with access to the MIM) in all right sessions together with the view.

Definition 4 (Non-malleable Commitments w.r.t. Commitment)**.** *A commitment scheme $\langle C, R\rangle$ is said to be non-malleable if for every PPT* MIM*, there exists a PPT simulator $\mathcal{S}$ such that the following ensembles are computationally indistinguishable:*

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

Definition 5 ($\ell(n)$-Concurrent Non-malleable Commitments w.r.t. Commitment)**.** *A commitment scheme $\langle C, R\rangle$ is said to be $\ell(n)$-concurrent non-malleable if for every PPT* MIM*, there exists a PPT simulator $\mathcal{S}$ such that the following ensembles are computationally indistinguishable:*

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

We say that a commitment scheme is fully concurrent, with respect to commitment, if it is concurrent for any a-priori unbounded polynomial $\ell(n)$.

3.4 Secure Multiparty Computation

As in [Gol04], we follow the real-ideal paradigm for defining secure multi-party computation. The only difference is that our simulator can run in super-polynomial time. A formal definition can be found in Appendix B.

Semi-malicious adversary: An adversary is said to be semi-malicious if it follows the protocol correctly, but with potentially maliciously chosen randomness. We refer the reader to Appendix B.2 for more details.

Concurrent security The definition of concurrent secure multi-party computation considers an extension of the real-ideal model where the adversary participates simultaneously in many executions, corrupting subsets of parties in each execution. We refer the reader to [CLP10, GGJS12] for a detailed definition of concurrent security.

4 Three Round Malicious Secure MPC

Let f be any functionality. Consider n parties $\mathsf{P}_1, \ldots, \mathsf{P}_n$ with inputs $\mathsf{x}_1, \ldots, \mathsf{x}_n$ respectively who wish to compute f on their joint inputs by running a secure multiparty computation(MPC) protocol. Let π^{SM} be any 3 round protocol that runs without any setup for the above task and is secure against adversaries that can be completely malicious in the first round, semi-malicious in the next two rounds and can corrupt upto $(n-1)$ parties. In this section, we show how to generically transform π^{SM} into a 3 round protocol π without setup with super-polynomial simulation and secure against malicious adversaries that can corrupt upto $(n-1)$ parties. Formally, we prove the following theorem:

Theorem 1. *Assuming sub-exponentially secure:*

- *A, where $A \in \{DDH,\ Quadratic\ Residuosity,\ N^{th}\ Residuosity\}$ AND*
- *3 round MPC protocol for any functionality f that is secure against malicious adversaries in the first round and semi-malicious adversaries in the next two rounds,*

the protocol presented in Figure 2 is a 3 round MPC protocol for any functionality f, in the plain model with super-polynomial simulation.

We can instantiate the underlying MPC protocol with the construction of Brakerski et al.[BHP17], which satisfies our requirements. That is:

Imported Lemma 1. *(**[BHP17]**): There exists a 3 round MPC protocol for any functionality f based on the sub-exponential LWE assumption that is secure against malicious adversaries in the first round and semi-malicious adversaries in the next 2 rounds.*

Additionally, Dodis et al.[DHRW16] give a 2 round construction based on sub-exponentially secure indistinguishability obfuscation that is secure against semi-malicious adversaries. Of course, this can be interpreted as a 3 round construction where the first round has no message and is trivially secure against malicious adversaries in the first round. Formally, we obtain the following corollary:

Corollary 2. *Assuming sub-exponentially secure:*

- *A, where $A \in \{DDH,\ Quadratic\ Residuosity,\ N^{th}\ Residuosity\}$ AND*
- *B, where $B \in \{LWE,\ Indistinguishability\ Obfuscation\}$*

the protocol presented in Figure 2 is a 3 round MPC protocol for any functionality f, in the plain model with super-polynomial simulation.

Remark 1 (On the Semi-Malicious security of [DHRW16])**.** *We note that the protocol in [DHRW16] works in two rounds: In the first round, each party provides a suitably "spooky" homomorphic encryption of its input, under public keys chosen by each party independently. After the first round, each party carries out a deterministic homomorphic evaluation procedure that results in an encryption of $f(\mathbf{x})$, where $\mathbf{x}$ is a vector that combines inputs of all parties. In the second round, each party computes a partial decryption of this ciphertext. The result is guaranteed to be the sum of these partial decryptions in a suitable cyclic group.*

Furthermore, their protocol satisfies the invariant that given the (possibly maliciously chosen) randomness of the corrupted parties for the first round, and given the vector of ciphertexts that are fixed after the first round, it is possible to efficiently compute, at the end of the first round, the decryption shares for all corrupted parties. Thus, if there is one honest party and the other parties are corrupted, given the final output value $f(\mathbf{x})$, the first round ciphertexts and the randomness of the corrupted semi-malicious parties, it is possible to compute the unique decryption share of the honest party that would force the desired output value. This property shows that their protocol satisfies semi-malicious security, since the first round message of the simulated honest party can simply be the honest first round message corresponding to the input 0, and the second round message can be computed from $f(\mathbf{x})$, the first round ciphertexts and the randomness of the corrupted semi-malicious parties. The work of [MW16] further showed how to transform such a 2-round semi-malicious MPC protocol that handles exactly all-but-one corruptions into a 2-round semi-malicious MPC protocol that handles any number of corruptions.

4.1 High-Level Overview

Before describing our protocol formally, to help the exposition, we first give a brief overview of the construction in this subsection.

Consider n parties $\mathsf{P}_1, \ldots, \mathsf{P}_n$ with inputs $\mathsf{x}_1, \ldots, \mathsf{x}_n$ respectively who wish to run a secure MPC to compute a function f on their joint inputs. Initially, each party P_i picks some randomness r_i that it will use to run the semi-malicious protocol π^{SM}.

In the first round, each party P_i sends the first round message of the protocol π^{SM}. Then, with every other party P_j, P_i initiates two executions of the $\mathsf{SPSS.ZK}$ argument system playing the verifier's role. Additionally, P_i and P_j also initiate two executions of a non-malleable commitment scheme - each acting as the committer in one of them. P_i commits to the pair $(\mathsf{x}_i, \mathsf{r}_i)$ - that is, the input and randomness used in the protocol π^{SM}. Recall that the first round messages of π^{SM} are already secure against malicious adversaries, so intuitively, the protocol doesn't require any proofs in the first round.

In the second round, each party P_i sends the second round message of the protocol π^{SM} using input x_i and randomness r_i. Then, P_i finishes executing the non-malleable commitments (playing the committer's role) with every other party P_j, committing to $(\mathsf{x}_i, \mathsf{r}_i)$. Finally, with every other party P_j, P_i completes the execution of the $\mathsf{SPSS.ZK}$ argument by sending its second message - P_i proves that the two messages sent so far using the protocol π^{SM} were correctly generated using the pair $(\mathsf{x}_i, \mathsf{r}_i)$ committed to using the non-malleable commitment.

In the third round, each party P_i first verifies all the proofs it received in the last round and sends a global abort (asking all the parties to abort) if any proof does not verify. Then, P_i sends the third round message of the protocol π^{SM} using input x_i and randomness r_i. Finally, as before, with every other party P_j, P_i completes the execution of the $\mathsf{SPSS.ZK}$ argument by sending its second message - P_i proves that the two messages sent so far using the protocol π^{SM} were correctly generated using the pair $(\mathsf{x}_i, \mathsf{r}_i)$ committed to using the non-malleable commitment.

Each party P_i now computes its final output as follows. P_i first verifies all the proofs it received in the previous round and sends a global abort (asking all the parties to abort) if any proof does not verify. Then, P_i computes the output using the output computation algorithm of the semi-malicious protocol π^{SM}. This completes the protocol description.

Security Proof: We now briefly describe how the security proof works. Let's consider an adversary $\mathcal{A}$ who corrupts a set of parties. Recall that the goal is to move from the real world to the ideal world such that the outputs of the honest parties along with the view of the adversary is indistinguishable. We do this via a sequence of computationally indistinguishable hybrids.

The first hybrid Hyb_1, refers to the real world. In Hyb_2, the simulator extracts the adversary's input and randomness (used in protocol π^{SM}) by a brute force break of the non-malleable commitment. The simulator aborts if the extracted values don't reconstruct the protocol messages for the underlying semi-malicious protocol correctly. These two hybrids are indistinguishable because from the soundness of the proof system, except with negligible probability, the values extracted by the simulator correctly reconstruct to protocol messages.

Then, in Hyb_3, we switch the $\mathsf{SPSS.ZK}$ arguments used by all honest parties in rounds 2 and 3 to simulated ones. This hybrid is computationally indistinguishable from the previous hybrid by the security of the $\mathsf{SPSS.ZK}$ system. Notice that when we switch from real to simulated arguments, we can no longer rely on the adversary's zero knowledge arguments to argue the correctness of the values extracted by breaking the non-malleable commitment. That is, the adversary's arguments may not be simulation sound. However, recall that to check the validity of the extracted values, we only rely on the correct reconstruction of the semi-malicious protocol messages, and hence this is not a problem. Also, the running time of the simulator in these two hybrids is the time taken to break the non-malleable commitment $\mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}}$ - which must be lesser than the time against which the zero knowledge property holds - T_{ZK}.

In Hyb_4, we switch all the non-malleable commitments sent by honest parties to be commitments of 0 instead of the actual input and randomness. Recall that since the arguments of the honest parties are simulated, this doesn't violate correctness. Also, this hybrid is computationally indistinguishable from the previous hybrid by the security of the non-malleable commitment scheme. One issue that arises here is whether the simulator continues to extract the adversary's inputs correctly. Recall that to extract, the simulator has to break the non-malleable commitment for which it has to run in time $\mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}}$. However, then the reduction to the security of the non-malleable commitment only makes sense if the simulator runs in time lesser than that needed to break the non-malleable commitment. We overcome this issue by a sequence of sub-hybrids where we first switch the simulator to not extract the adversary's inputs, then switch the non-malleable commitments and then finally go back to the simulator extracting the adversary's inputs. We elaborate on this in the formal proof.

Then, in Hyb_5 we run the simulator of π^{SM} using the extracted values to generate the protocol messages. This hybrid is indistinguishable from the previous one by the security of π^{SM}. Once again, in order to ensure correctness of the extracted values, we require the running time of the simulator - which is $\mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}}$ to be lesser than the time against which the semi-malicious protocol π^{SM} is secure. This is because, then, the simulator can continue to extract the adversary's message and randomness used for the protocol π^{SM} by breaking the semi-malicious protocol. This hybrid (Hyb_5)

now corresponds to the ideal world. Notice that our simulation is in fact straight-line. There are other minor technicalities that arise and we elaborate on this in the formal proof.

4.2 Construction

We first list some notation and the primitives used before describing the construction.

Notation:

- λ denotes the security parameter.
- $\mathsf{SPSS.ZK} = (\mathsf{ZK}_1, \mathsf{ZK}_2, \mathsf{ZK}_3)$ is a two message zero knowledge argument with superior super polynomial simulation (SPSS-ZK). The zero knowledge property holds against all adversaries running in time T_{ZK}. Let $\mathsf{Sim}^{\mathsf{ZK}}$ denote the simulator that produces simulated ZK proofs and let $\mathsf{T}^{\mathsf{Sim}}_{\mathsf{ZK}}$ denote its running time. [KS17] give a construction of an $\mathsf{SPSS.ZK}$ scheme satisfying these properties that can be based on one of the following sub-exponential assumptions: 1) DDH; 2) Quadratic Residuosity; 3) N^{th} Residuosity.
- $\mathsf{NMCom} = (\mathsf{NMCom}^R_1, \mathsf{NMCom}^{\mathsf{S}}_1, \mathsf{NMCom}^S_2)$ is a two message concurrent non-malleable commitment scheme with respect to commitment in the simultaneous message model. Here, $\mathsf{NMCom}^R_1, \mathsf{NMCom}^{\mathsf{S}}_1$ denote the first message of the receiver and sender respectively while NMCom^S_2 denotes the second message of the sender. It is secure against all adversaries running in time $\mathsf{T}^{\mathsf{Sec}}_{\mathsf{Com}}$, but can be broken by adversaries running in time $\mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}}$. Let $\mathsf{Ext.Com}$ denote a brute force algorithm running in time $\mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}}$ that can break the commitment scheme. [KS17] give a construction of an NMCom scheme satisfying these properties that can be based on one of the following sub-exponential assumptions: 1) DDH; 2) Quadratic Residuosity; 3) N^{th} Residuosity.

  The NMCom we use is tagged. In the authenticated channels setting, the tag of each user performing a non-malleable commitment can just be its identity. In the general setting, in the first round, each party can choose a strong digital signature verification key VK and signing key, and then sign all its messages using this signature scheme for every message sent in the protocol. This VK is then used as the tag for all non-malleable commitments. This ensures that every adversarial party must choose a tag that is different than any tags chosen by honest parties, otherwise the adversary will not be able to sign any of its messages by the existential unforgeability property of the signature scheme. This is precisely the property that is assumed when applying NMCom. For ease of notation, we suppress writing the tags explicitly in our protocols below.
- π^{SM} is a sub-exponentially secure 3 round MPC protocol that is secure against malicious adversaries in the first round and semi-malicious adversaries in the next two rounds. This protocol is secure against all adversaries running in time T_{SM}. Let $(\mathsf{MSG}_1, \mathsf{MSG}_2, \mathsf{MSG}_3)$ denote the algorithms used by any party to compute the messages in each of the three rounds and OUT denotes the algorithm to compute the final output. Further, let's assume that this protocol π^{SM} runs over a broadcast channel. Let $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3)$ denote the straight line simulator for this protocol - that is, $\mathcal{S}_i$ is the simulator's algorithm to compute the i^{th} round messages. Also, we make the following assumptions about the protocol structure, that is satisfied by the instantiations:

  1. $\mathcal{S}_1$ and $\mathcal{S}_2$ run without any input other than the protocol transcript so far - in particular, they don't need the input, randomness and output of the malicious parties. For $\mathcal{S}_1$,

this must necessarily be true since the first round of π^{SM} is secure against malicious adversaries. We make the assumption only on $\mathcal{S}_2$.[2]

2. The algorithm MSG_3 doesn't require any new input or randomness that was not already used in the algorithms $\mathsf{MSG}_1, \mathsf{MSG}_2$. Looking ahead, this is used in our security proof when we want to invoke the simulator of this protocol π^{SM}, we need to be sure that we have fed the correct input and randomness to the simulator. This is true for all instantiantions we consider, where the semi-malicious simulator requires only the secret keys of corrupted parties (that are fixed in the second round) apart from the protocol transcript.

In order to realize our protocol, we require that $\mathsf{poly}(\lambda) < \mathsf{T}^{\mathsf{Sim}}_{\mathsf{ZK}} < \mathsf{T}^{\mathsf{Sec}}_{\mathsf{Com}} < \mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}} < \mathsf{T}_{\mathsf{ZK}}, \mathsf{T}_{\mathsf{SM}}$.

The construction of the protocol is described in Figure 2. We assume broadcast channels. In our construction, we use proofs for a some NP languages that we elaborate on below.

NP language L is characterized by the following relation R.
Statement : $\mathsf{st} = (\mathsf{c}_1, \hat{\mathsf{c}}_1, \mathsf{c}_2, \mathsf{msg}_1, \mathsf{msg}_2, \tau)$
Witness : $\mathsf{w} = (\mathsf{inp}, \mathsf{r}, \mathsf{r_c})$
$R(\mathsf{st}, \mathsf{w}) = 1$ if and only if :

- $\hat{\mathsf{c}}_1 = \mathsf{NMCom}^{\mathsf{S}}_1(\mathsf{inp}, \mathsf{r}; \mathsf{r_c})$ AND
- $\mathsf{c}_2 = \mathsf{NMCom}^{S}_2(\mathsf{inp}, \mathsf{r}, \mathsf{c}_1; \mathsf{r_c})$ AND
- $\mathsf{msg}_1 = \mathsf{MSG}_1(\mathsf{inp}; \mathsf{r})$ AND
- $\mathsf{msg}_2 = \mathsf{MSG}_2(\mathsf{inp}, \tau; \mathsf{r})$

That is, the messages $(\mathsf{c}_1, \hat{\mathsf{c}}_1, \mathsf{c}_2)$ form a non-malleable commitment of $(\mathsf{inp}, \mathsf{r})$ such that msg_2 is the second round message using input inp, randomness r by running the protocol π^{SM}, where the protocol transcript so far is τ.

NP language L_1 is characterized by the following relation R_1.
Statement : $\mathsf{st} = (\mathsf{c}_1, \hat{\mathsf{c}}_1, \mathsf{c}_2, \mathsf{msg}_3, \tau)$
Witness : $\mathsf{w} = (\mathsf{inp}, \mathsf{r}, \mathsf{r_c})$
$R(\mathsf{st}, \mathsf{w}) = 1$ if and only if :

- $\hat{\mathsf{c}}_1 = \mathsf{NMCom}^{\mathsf{S}}_1(\mathsf{inp}, \mathsf{r}; \mathsf{r_c})$ AND
- $\mathsf{c}_2 = \mathsf{NMCom}^{S}_2(\mathsf{inp}, \mathsf{r}, \mathsf{c}_1; \mathsf{r_c})$ AND
- $\mathsf{msg}_3 = \mathsf{MSG}_3(\mathsf{inp}, \tau; \mathsf{r})$

That is, the messages $(\mathsf{c}_1, \hat{\mathsf{c}}_1, \mathsf{c}_2)$ form a non-malleable commitment of $(\mathsf{inp}, \mathsf{r})$ such that msg_3 is the third round message using input inp, randomness r by running the protocol π^{SM}, where the protocol transcript so far is τ.

In the protocol, let's assume that every party has an associated identity id. For any session sid, each parties generates its non-malleable commitment using the tag $(\mathsf{id}||\mathsf{sid})$.

The correctness of the protocol follows from the correctness of the protocol π^{SM}, the non-malleable commitment scheme NMCom and the zero knowledge proof system $\mathsf{SPSS.ZK}$.

[2] This assumption can be removed by running the commitment extractor on the first round messages itself. This idea is used in Section 5.

Inputs: Each party P_i has input x_i and uses randomness r_i to compute the message in each round of the protocol π^{SM}. We now describe the messages sent by party P_i. We will use superscripts to denote the intended recipient of the message if it is not meant to be used by all parties.

1. **Round 1:**
   P_i does the following:
   - Compute $\mathsf{msg}_{1,i} \leftarrow \mathsf{MSG}_1(\mathsf{x}_i; \mathsf{r}_i)$.
   - For each $j \in [n]$ with $j \neq i$, compute:
     - $\hat{\mathsf{c}}^j_{1,i} \leftarrow \mathsf{NMCom}^{\mathsf{S}}_1(\mathsf{x}_i, \mathsf{r}_i; \mathsf{r}^j_{\mathsf{c},i})$ using a random string $\mathsf{r}^j_{\mathsf{c},i}$ and $\mathsf{c}^j_{1,i} \leftarrow \mathsf{NMCom}^R_1(1^\lambda)$.
     - $(\mathsf{ver}^j_{1,i}, \mathsf{zkst}^j_{1,i}) \leftarrow \mathsf{ZK}_1(1^\lambda)$ and $(\mathsf{ver}^j_{2,i}, \mathsf{zkst}^j_{2,i}) \leftarrow \mathsf{ZK}_1(1^\lambda)$.
   - Send $(\mathsf{msg}_{1,i}, \hat{\mathsf{c}}^j_{1,i}, \mathsf{c}^j_{1,i}, \mathsf{ver}^j_{1,i}, \mathsf{ver}^j_{2,i})$ for all j.
2. **Round 2:**
   Let τ_1 denote the protocol transcript after round 1. P_i does the following:
   - Compute $\mathsf{msg}_{2,i} \leftarrow \mathsf{MSG}_2(\mathsf{x}_i, \tau_1; \mathsf{r}_i)$.
   - For each $j \in [n]$ with $j \neq i$, compute:
     - $\mathsf{c}^j_{2,i} \leftarrow \mathsf{NMCom}^S_2(\mathsf{x}_i, \mathsf{r}_i, \mathsf{c}^i_{1,j}; \mathsf{r}^j_{\mathsf{c},i})$ using the same random string $\mathsf{r}^j_{\mathsf{c},i}$.
     - $\mathsf{prove}^j_{2,i} \leftarrow \mathsf{ZK}_2(\mathsf{ver}^i_{1,j}, \mathsf{st}^j_{2,i}, \mathsf{w}^j_{2,i})$ for the statement $\mathsf{st}^j_{2,i} = (\mathsf{c}^i_{1,j}, \hat{\mathsf{c}}^j_{1,i}, \mathsf{c}^j_{2,i}, \mathsf{msg}_{1,i}, \mathsf{msg}_{2,i}, \tau_1) \in L$ using witness $\mathsf{w}^j_{2,i} = (\mathsf{x}_i, \mathsf{r}_i, \mathsf{r}^j_{\mathsf{c},i})$.
   - Send $(\mathsf{msg}_{2,i}, \mathsf{c}^j_{2,i}, \mathsf{prove}^j_{2,i})$ for all j.
3. **Round 3:**
   Let τ_2 denote the protocol transcript after round 2. P_i does the following:
   - Compute $\mathsf{msg}_{3,i} \leftarrow \mathsf{MSG}_3(\mathsf{x}_i, \tau_2; \mathsf{r}_i)$.
   - For each $j \in [n]$ with $j \neq i$, do:
     - Abort if $\mathsf{ZK}_3(\mathsf{zkst}^j_{1,i}, \mathsf{st}^i_{2,j}) \neq 1$ where $\mathsf{st}^i_{2,j} = (\mathsf{c}^j_{1,i}, \hat{\mathsf{c}}^i_{1,j}, \mathsf{c}^i_{2,j}, \mathsf{msg}_{1,j}, \mathsf{msg}_{2,j}, \tau_1)$. In particular, send a global abort signal to all parties so that everyone aborts.
     - $\mathsf{prove}^j_{3,i} \leftarrow \mathsf{ZK}_2(\mathsf{ver}^i_{2,j}, \mathsf{st}^j_{3,i}, \mathsf{w}^j_{3,i})$ for the statement $\mathsf{st}^j_{3,i} = (\mathsf{c}^i_{1,j}, \hat{\mathsf{c}}^j_{1,i}, \mathsf{c}^j_{2,i}, \mathsf{msg}_{3,i}, \tau_2) \in L_1$ using witness $\mathsf{w}^j_{3,i} = (\mathsf{x}_i, \mathsf{r}_i, \mathsf{r}^j_{\mathsf{c},i})$.
   - Send $(\mathsf{msg}_{3,i}, \mathsf{prove}^j_{3,i})$ for all j.
4. **Output Computation:**
   Let τ_3 denote the protocol transcript after round 3. P_i does the following:
   - For each $j \in [n]$ with $j \neq i$, do:
     - Abort if $\mathsf{ZK}_3(\mathsf{zkst}^j_{2,i}, \mathsf{st}^i_{3,j}) \neq 1$ where $\mathsf{st}^i_{3,j} = (\mathsf{c}^j_{1,i}, \hat{\mathsf{c}}^i_{1,j}, \mathsf{c}^i_{2,j}, \mathsf{msg}_{3,j}, \tau_2)$. In particular, send a global abort signal to all parties so that everyone aborts.
   - Compute output $\mathsf{y}_i \leftarrow \mathsf{OUT}(\mathsf{x}_i, \tau_3; \mathsf{r}_i)$.

Figure 2: 3 round MPC Protocol π for functionality f.

4.3 Security Proof

In this section, we formally prove Theorem 1.

Consider an adversary $\mathcal{A}$ who corrupts t parties where $t < n$. For each party P_i, let's say that the size of input and randomness used in the protocol π^{SM} is $p(\lambda)$ for some polynomial p. That is,

$|(\mathsf{x}_i, \mathsf{r}_i)| = p(\lambda)$. The strategy of the simulator Sim against a malicious adversary $\mathcal{A}$ is described in *Figure 3*.

1. **Round 1:** For each honest party P_i, Sim does the following:
   - Compute $\mathsf{msg}_{1,i} \leftarrow \mathcal{S}_1(1^\lambda, i)$. For each $j \in [n]$ with $j \neq i$, compute:
     - $\hat{\mathsf{c}}^j_{1,i} \leftarrow \mathsf{NMCom}^{\mathsf{S}}_1(0^{p(\lambda)})$, $\mathsf{c}^j_{1,i} \leftarrow \mathsf{NMCom}^{R}_1(1^\lambda)$.
     - $(\mathsf{ver}^j_{1,i}, \mathsf{zkst}^j_{1,i}) \leftarrow \mathsf{ZK}_1(1^\lambda)$ and $(\mathsf{ver}^j_{2,i}, \mathsf{zkst}^j_{2,i}) \leftarrow \mathsf{ZK}_1(1^\lambda)$.
   - Send $(\mathsf{msg}_{1,i}, \hat{\mathsf{c}}^j_{1,i}, \mathsf{c}^j_{1,i}, \mathsf{ver}^j_{1,i}, \mathsf{ver}^j_{2,i})$ for all $j \in [n]$.
2. **Round 2:** Let τ_1 denote the protocol transcript after round 1. For each honest party P_i,
   - Compute $\mathsf{msg}_{2,i} \leftarrow \mathcal{S}_2(\tau_1, i)$. For each $j \in [n]$ with $j \neq i$, compute:
     - $\mathsf{c}^j_{2,i} \leftarrow \mathsf{NMCom}^S_2(0^{p(\lambda)}, \mathsf{c}^i_{1,j}; \mathsf{r}^j_{\mathsf{c},i})$ using a random string $\mathsf{r}^j_{\mathsf{c},i}$.
     - $\mathsf{prove}^j_{2,i} \leftarrow \mathsf{Sim}^{\mathsf{ZK}}(\mathsf{ver}^i_{1,j}, \mathsf{st}^j_{2,i})$ for $\mathsf{st}^j_{2,i} = (\mathsf{c}^i_{1,j}, \hat{\mathsf{c}}^j_{1,i}, \mathsf{c}^j_{2,i}, \mathsf{msg}_{1,i}, \mathsf{msg}_{2,i}, \tau_1) \in L$. Observe that this takes time $\mathsf{T}^{\mathsf{Sim}}_{\mathsf{ZK}}$.
   - Send $(\mathsf{msg}_{2,i}, \mathsf{c}^j_{2,i}, \mathsf{prove}^j_{2,i})$ for all $j \in [n]$.
3. **Input Extraction:** Sim does the following:
   - For each honest party P_i and for each $j \in [n]$ with $j \neq i$, do:
     - Abort if $\mathsf{ZK}_3(\mathsf{zkst}^j_{1,i}, \mathsf{st}^i_{2,j}) \neq 1$ where τ_1 is the protocol transcript after round 1 such that $\mathsf{st}^i_{2,j} = (\mathsf{c}^j_{1,i}, \hat{\mathsf{c}}^i_{1,j}, \mathsf{c}^i_{2,j}, \mathsf{msg}_{1,j}, \mathsf{msg}_{2,j}, \tau_1)$
     - Compute $(\mathsf{x}^i_j, \mathsf{r}^i_j) = \mathsf{Ext.Com}(\mathsf{c}^j_{1,i}, \hat{\mathsf{c}}^i_{1,j}, \mathsf{c}^i_{2,j})$. That is, this is the input and randomness of party P_j seen by party P_i. This step takes time $\mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}}$.
   - For each malicious party P_j, do:
     - Output "Special Abort" if the set of values $\{(\mathsf{x}^i_j, \mathsf{r}^i_j)\}$ computed in the last step, for all i corresponding to honest parties P_i is not equal. Set $(\mathsf{x}_j, \mathsf{r}_j) = (x^1_j, \mathsf{r}^1_j)$. Output "Special Abort" if $\mathsf{msg}_{1,j} \neq \mathsf{MSG}_1(\mathsf{x}_j, \mathsf{r}_j)$ and $\mathsf{msg}_{2,j} \neq \mathsf{MSG}_2(\mathsf{x}_j, \mathsf{r}_j, \tau_1)$.
     - Send all extracted x_j to the trusted functionality and receive output y.
     - Let R denote the set of all $\{\mathsf{x}_j, \mathsf{r}_j\}$.
4. **Round 3:** Let τ_2 denote the protocol transcript after round 2. For each honest party P_i, compute and send $\mathsf{msg}_{3,i} \leftarrow \mathcal{S}_3(\mathsf{y}, \mathsf{R}, \tau_2, i)$ together with $\mathsf{prove}^j_{3,i}$ for $j \in [n], j \neq i$ where $\mathsf{prove}^j_{3,i} \leftarrow \mathsf{Sim}^{\mathsf{ZK}}(\mathsf{ver}^i_{2,j}, \mathsf{st}^j_{3,i})$ for the statement $\mathsf{st}^j_{3,i} = (\mathsf{c}^i_{1,j}, \hat{\mathsf{c}}^j_{1,i}, \mathsf{c}^j_{2,i}, \mathsf{msg}_{3,i}, \tau_2) \in L_1$. Observe that this takes time $\mathsf{T}^{\mathsf{Sim}}_{\mathsf{ZK}}$.
5. **Special Abort Phase:** Sim does the following:
   - Output "Special Abort" if for each malicious party P_j, $\mathsf{msg}_{3,j} \neq \mathsf{MSG}_3(\mathsf{x}_j, \mathsf{r}_j, \tau_2)$.
6. **Output Computation:** Sim does the following:
   - For each honest party P_i and for each $j \in [n]$ with $j \neq i$, abort if $\mathsf{ZK}_3(\mathsf{zkst}^j_{2,i}, \mathsf{st}^i_{3,j}) \neq 1$ where $\mathsf{st}^i_{3,j} = (\mathsf{c}^j_{1,i}, \hat{\mathsf{c}}^i_{1,j}, \mathsf{c}^i_{2,j}, \mathsf{msg}_{3,j}, \tau_2)$.
   - Else instruct the ideal functionality to deliver output to the honest parties.

Figure 3: Simulation strategy in the 3 round protocol

Here, in the simulation, we crucially use the two assumptions about the protocol structure. The first one is easy to notice since the simulator Sim has to run the semi-malicious to produce the first and second messages before it has extracted the adversary's input and randomness. For the second assumption, observe that in order to run the simulator algorithm $\mathcal{S}_3$, Sim has to feed it the entire input and randomness of the adversary and so these have to be bound to by the end of the second round.

We now show that the simulation strategy described in Figure 3 is successful against all malicious PPT adversaries. That is, the view of the adversary along with the output of the honest parties is computationally indistinguishable in the real and ideal worlds. We will show this via a series of computationally indistinguishable hybrids where the first hybrid Hyb_1 corresponds to the real world and the last hybrid Hyb_6 corresponds to the ideal world.

1. Hyb_1**:** In this hybrid, consider a simulator $\mathsf{Sim}_{\mathsf{Hyb}}$ that plays the role of the honest parties. $\mathsf{Sim}_{\mathsf{Hyb}}$ runs in polynomial time.

2. Hyb_2**:** In this hybrid, the simulator $\mathsf{Sim}_{\mathsf{Hyb}}$ also runs the "Input Extraction" phase and the "Special Abort" phase in step3 and 5 in Figure 3. $\mathsf{Sim}_{\mathsf{Hyb}}$ runs in time $\mathsf{T}_{\mathsf{Com}}^{\mathsf{Brk}}$.

3. Hyb_3**:** This hybrid is identical to the previous hybrid except that in Rounds 2 and 3, $\mathsf{Sim}_{\mathsf{Hyb}}$ now computes simulated SPSSZK proofs as done in Round 2 in Figure 3. Once again, $\mathsf{Sim}_{\mathsf{Hyb}}$ runs in time $\mathsf{T}_{\mathsf{Com}}^{\mathsf{Brk}}$.

4. Hyb_4**:** This hybrid is identical to the previous hybrid except that $\mathsf{Sim}_{\mathsf{Hyb}}$ now computes all the $(\hat{\mathsf{c}}_{1,i}^{j}, \mathsf{c}_{2,i}^{j})$ as non-malleable commitments of $0^{p(\lambda)}$ as done in Round 2 in Figure 3. Once again, $\mathsf{Sim}_{\mathsf{Hyb}}$ runs in time $\mathsf{T}_{\mathsf{Com}}^{\mathsf{Brk}}$.

5. Hyb_5**:** This hybrid is identical to the previous hybrid except that in Round 3, $\mathsf{Sim}_{\mathsf{Hyb}}$ now computes the messages of the protocol π^{SM} using the simulator algorithms $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3)$ as done by Sim in the ideal world. $\mathsf{Sim}_{\mathsf{Hyb}}$ also instructs the ideal functionality to deliver outputs to the honest parties as done by Sim. This hybrid is now same as the ideal world. Once again, $\mathsf{Sim}_{\mathsf{Hyb}}$ runs in time $\mathsf{T}_{\mathsf{Com}}^{\mathsf{Brk}}$.

We now show that every pair of successive hybrids is computationally indistinguishable.

Lemma 1. *Assuming soundness of the* SPSS.ZK *argument system, binding of the non-malleable commitment scheme and correctness of the protocol* π^{SM}*,* Hyb_1 *is computationally indistinguishable from* Hyb_2*.*

Proof. The only difference between the two hybrids is that in Hyb_2, $\mathsf{Sim}_{\mathsf{Hyb}}$ may output "Special Abort" which doesn't happen in Hyb_1. More specifically, in Hyb_2, "Special Abort" occurs if event E described below is true.

Event E**:** Is true if : For any malicious party P_j

- All the SPSS.ZK proofs sent by P_j in round 2 and 3 verify correctly.
  (AND)

- Either of the following occur:
  - The set of values $\{(\mathsf{x}_j^i, \mathsf{r}_j^i)\}$ that are committed to using the non-malleable commitment is not same for every i where P_i is honest. (OR)
  - $\mathsf{msg}_{1,j} \neq \mathsf{MSG}_1(\mathsf{x}_j, \mathsf{r}_j)$ (OR)
  - $\mathsf{msg}_{2,j} \neq \mathsf{MSG}_2(\mathsf{x}_j, \mathsf{r}_j, \tau_1)$ where τ_1 is the protocol transcript after round 1. (OR)

– $\mathsf{msg}_{3,j} \neq \mathsf{MSG}_3(\mathsf{x}_j, \mathsf{r}_j, \tau_2)$ where τ_2 is the protocol transcript after round 2.

That is, in simpler terms, the event E occurs if for any malicious party, it gives valid ZK proofs in round 2 and 3 but its protocol transcript is not consistent with the values it committed to.

Therefore, in order to prove the indistinguishability of the two hybrids, it is enough to prove the lemma below.

Sub-Lemma 1. $\Pr[\mathsf{Event\ E\ is\ true\ in\ Hyb}_2] = \mathsf{negl}(\lambda)$.

Proof. We now prove the sub-lemma. Suppose the event E does occur. From the binding property of the commitment scheme and the correctness of the protocol π^{SM}, observe that if any of the above conditions are true, it means there exists i, j such that the statement $\mathsf{st}^i_{2,j} = (\mathsf{c}^j_{1,i}, \mathsf{c}^i_{2,j}, \mathsf{msg}_{1,j}, \mathsf{msg}_{2,j}, \tau_1) \notin L$, where P_i is honest and P_j is malicious. However, the proof for the statement verified correctly which means that the adversary has produced a valid proof for a false statement. This violates the soundness property of the SPSSZK argument system which is a contradiction. □

□

Lemma 2. *Assuming the zero knowledge property of the* $\mathsf{SPSS.ZK}$ *argument system,* Hyb_2 *is computationally indistinguishable from* Hyb_3*.*

Proof. The only difference between the two hybrids is that in Hyb_2, $\mathsf{Sim}_{\mathsf{Hyb}}$ computes the proofs in Rounds 2 and 3 honestly, by running the algorithm ZK_2 of the $\mathsf{SPSS.ZK}$ argument system, whereas in Hyb_3, a simulated proof is used. If the adversary $\mathcal{A}$ can distinguish between the two hybrids, we will use $\mathcal{A}$ to design an algorithm $\mathcal{A}_{\mathsf{ZK}}$ that breaks the zero knowledge property of the argument system.

Suppose the adversary can distinguish between the two hybrids with non-negligible probability p. Then, by a simple hybrid argument, there exists hybrids $\mathsf{Hyb}_{2,k}$ and $\mathsf{Hyb}_{2,k+1}$ that the adversary can distinguish with non-negligible probability $p' < p$ such that: the only difference between the two hybrids is in the proof sent by an honest party P_i to a (malicious) party P_j in one of the rounds . Let's say it is the proof in round 2.

$\mathcal{A}_{\mathsf{ZK}}$ performs the role of $\mathsf{Sim}_{\mathsf{Hyb}}$ in its interaction with $\mathcal{A}$ and performs all the steps exactly as in $\mathsf{Hyb}_{2,k}$ except the proof in Round 2 sent by P_i to P_j. It interacts with a challenger $\mathcal{C}$ of the $\mathsf{SPSS.ZK}$ argument system and sends the first round message $\mathsf{ver}^i_{1,j}$ it received from the adversary. $\mathcal{A}_{\mathsf{ZK}}$ receives from $\mathcal{C}$ a proof that is either honestly computed or simulated. $\mathcal{A}_{\mathsf{ZK}}$ sets this received proof as its message $\mathsf{prove}^j_{i,2}$ in Round 2 of its interaction with $\mathcal{A}$. In the first case, this exactly corresponds to $\mathsf{Hyb}_{2,k}$ while the latter exactly corresponds to $\mathsf{Hyb}_{2,k+1}$. Therefore, if $\mathcal{A}$ can distinguish between the two hybrids, $\mathcal{A}_{\mathsf{ZK}}$ can use the same distinguishing guess to distinguish the proofs: i.e, decide whether the proofs received from $\mathcal{C}$ were honest or simulated. Now, notice that $\mathcal{A}_{\mathsf{ZK}}$ runs only in time $\mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}}$ (during the input extraction phase), while the $\mathsf{SPSS.ZK}$ system is secure against adversaries running in time T_{ZK}. Since $\mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}} < \mathsf{T}_{\mathsf{ZK}}$, this is a contradiction and hence proves the lemma.
In particular, this also means the following: $\Pr[\mathsf{Event\ E\ is\ true\ in\ Hyb}_3] = \mathsf{negl}(\lambda)$. □

Lemma 3. *Assuming the non-malleability property of the non-malleable commitment scheme* NMCom*,* Hyb_3 *is computationally indistinguishable from* Hyb_4*.*

Proof. We will prove this using a series of computationally indistinguishable intermediate hybrids as follows.

- $\mathsf{Hyb}_{3,1}$**:** This is same as Hyb_3 except that the simulator $\mathsf{Sim}_{\mathsf{Hyb}}$ does not run the input extraction phase apart from verifying the $\mathsf{SPSS.ZK}$ proofs. Also, $\mathsf{Sim}_{\mathsf{Hyb}}$ does not run the special abort phase. In particular, the $\mathsf{Ext.Com}$ algorithm is not run and there is no "Special Abort". In this hybrid, $\mathsf{Sim}_{\mathsf{Hyb}}$ runs in time $\mathsf{T}^{\mathsf{Sim}}_{\mathsf{ZK}}$ which is lesser than $\mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}}$.

- $\mathsf{Hyb}_{3,2}$**:** This hybrid is identical to the previous hybrid except that in Round 2, $\mathsf{Sim}_{\mathsf{Hyb}}$ now computes all the messages $(\hat{\mathsf{c}}^j_{1,i}, \mathsf{c}^j_{2,i})$ as non-malleable commitments of $0^{p(\lambda)}$ as done by Sim in the ideal world. In this hybrid too, $\mathsf{Sim}_{\mathsf{Hyb}}$ runs in time $\mathsf{T}^{\mathsf{Sim}}_{\mathsf{ZK}}$.

- $\mathsf{Hyb}_{3,3}$**:** This is same as Hyb_3 except that the simulator does run the input extraction phase and the special abort phase. It is easy to see that $\mathsf{Hyb}_{3,3}$ is the same as Hyb_4. In this hybrid, $\mathsf{Sim}_{\mathsf{Hyb}}$ runs in time $\mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}}$ which is greater than $\mathsf{T}^{\mathsf{Sim}}_{\mathsf{ZK}}$.

We now prove the indistinguishability of these intermediate hybrids and this completes the proof of the lemma.

Sub-Lemma 2. Hyb_3 *is statistically indistinguishable from* $\mathsf{Hyb}_{3,1}$.

Proof. The only difference between the two hybrids is that in Hyb_3, the simulator might output "Special Abort" which doesn't happen in $\mathsf{Hyb}_{3,1}$. As shown in the proof of Lemma 2, the probability that Event E occurs in Hyb_3 is negligible. This means that the probability that the simulator outputs "Special Abort" in Hyb_3 is negligible and this completes the proof. □

Sub-Lemma 3. *Assuming the non-malleability property of the non-malleable commitment scheme* NMCom*,* $\mathsf{Hyb}_{3,1}$ *is computationally indistinguishable from* $\mathsf{Hyb}_{3,2}$.

Proof. The only difference between the two hybrids is that in $\mathsf{Hyb}_{3,1}$, for every honest party P_i, $\mathsf{Sim}_{\mathsf{Hyb}}$ computes the commitment messages $(\hat{\mathsf{c}}^j_{1,i}, \mathsf{c}^j_{2,i})$ as a commitment of $(\mathsf{x}_i, \mathsf{r}_i)$, whereas in $\mathsf{Hyb}_{3,2}$, they are computed as a commitment of $(0^{p(\lambda)})$. If the adversary $\mathcal{A}$ can distinguish between the two hybrids, we will use $\mathcal{A}$ to design an algorithm $\mathcal{A}_{\mathsf{NMC}}$ that breaks the security of the non-malleable commitment scheme NMCom.

$\mathcal{A}_{\mathsf{NMC}}$ acts as the man-in-the-middle adversary interacting with a challenger $\mathcal{C}$. $\mathcal{A}_{\mathsf{NMC}}$ also plays the role of $\mathsf{Sim}_{\mathsf{Hyb}}$ in its interaction with the adversary $\mathcal{A}$. It generates all the messages except the messages $\mathsf{c}^j_{1,i}$ and $(\hat{\mathsf{c}}^j_{1,i}, \mathsf{c}^j_{2,i})$ exactly as done by $\mathsf{Sim}_{\mathsf{Hyb}}$ in $\mathsf{Hyb}_{3,1}$. Corresponding to each message $\mathsf{c}^j_{1,i}$ that $\mathcal{A}_{\mathsf{NMC}}$ has to send, it receives one first round message from $\mathcal{C}$ (on the right side) corresponding to the scheme NMCom. Similarly, it receives first round messages $\hat{\mathsf{c}}^j_{1,i}$ from $\mathcal{C}$ (on the left side). $\mathcal{A}_{\mathsf{NMC}}$ forwards these messages to the adversary $\mathcal{A}$ as its first round messages $(\hat{\mathsf{c}}^j_{1,i}, \mathsf{c}^j_{1,i})$. Similarly, for each pair of messages $(\hat{\mathsf{c}}^i_{1,j}, \mathsf{c}^i_{1,j})$ it receives from $\mathcal{A}$ as part of the first round messages of the scheme NMCom, $\mathcal{A}_{\mathsf{NMC}}$ forwards the messages to $\mathcal{C}$ as its first round messages for the commitment (to the left and right side respectively). Then, for each $\mathsf{c}^j_{2,i}$ that $\mathcal{A}_{\mathsf{NMC}}$ is supposed to send to $\mathcal{A}$, it receives a second round commitment message from the challenger $\mathcal{C}$. In one case, all of these are commitments to the respective $(\mathsf{x}_i, \mathsf{r}_i)$ values while in the second case, they are all commitments to $(0^{p(\lambda)})$. $\mathcal{A}_{\mathsf{NMC}}$ forwards these messages as its commitment messages $\mathsf{c}^j_{2,i}$ to the adversary $\mathcal{A}$. Once again, it forwards each message $\mathsf{c}^i_{2,j}$ it receives from $\mathcal{A}$, as its second round commitment message in its interaction with the challenger $\mathcal{C}$. That is, these are the commitments on the right side generated by the man-in-the-middle.

Now, we can clearly see that in the first case, when $\mathcal{C}$ generates commitments to $(\mathsf{x}_i, \mathsf{r}_i)$, $\mathcal{A}$'s view corresponds to $\mathsf{Hyb}_{3,1}$ while in the latter case, it exactly corresponds to $\mathsf{Hyb}_{3,2}$. However, from the security of the non-malleable commitment scheme, the joint distribution of the value committed to by the adversary $\mathcal{A}_{\mathsf{NMC}}$ (which is the same as $\mathcal{A}$'s commitments) and its view must be indistinguishable in both cases. Therefore, if $\mathcal{A}$ can distinguish between the two hybrids, then $\mathcal{A}_{\mathsf{NMC}}$ can break the non-malleability property of the commitment scheme NMCom. However, $\mathcal{A}_{\mathsf{NMC}}$ only runs in time $\mathsf{T}^{\mathsf{Sim}}_{\mathsf{ZK}} < \mathsf{T}^{\mathsf{Sec}}_{\mathsf{Com}}$ and hence this is a contradiction, thus proving the sub-lemma.

Also, notice that since the joint distribution of the adversary $\mathcal{A}$'s committed values and his view is indistinguishable in both hybrids, this implies that Event E still occurs only with negligible probability in $\mathsf{Hyb}_{3,2}$ as well. □

Sub-Lemma 4. *$\mathsf{Hyb}_{3,2}$ is statistically indistinguishable from $\mathsf{Hyb}_{3,3}$.*

Proof. The only difference between the two hybrids is that in $\mathsf{Hyb}_{3,3}$, the simulator might output "Special Abort" which doesn't happen in $\mathsf{Hyb}_{3,2}$. As shown in the proof of Sub-Lemma 3, the probability that Event E occurs in $\mathsf{Hyb}_{3,2}$ is negligible. This means that the probability that the simulator outputs "Special Abort" in $\mathsf{Hyb}_{3,3}$ is negligible and this completes the proof. □

□

Lemma 4. *Assuming the security of the protocol π^{SM}, Hyb_4 is computationally indistinguishable from Hyb_5.*

Proof. The only difference between the two hybrids is that in Hyb_4, $\mathsf{Sim}_{\mathsf{Hyb}}$ computes the messages of protocol π^{SM} correctly using the honest parties' inputs, whereas in Hyb_5, they are computed by running the simulator $\mathcal{S}$ for protocol π^{SM}. If the adversary $\mathcal{A}$ can distinguish between the two hybrids, we will use $\mathcal{A}$ to design an algorithm $\mathcal{A}_{\mathsf{SM}}$ that can break the security of protocol π^{SM}.

$\mathcal{A}_{\mathsf{SM}}$ interacts with a challenger $\mathcal{C}$ to break the security of protocol π^{SM}. Also, $\mathcal{A}_{\mathsf{SM}}$ performs the role of $\mathsf{Sim}_{\mathsf{Hyb}}$ in its interaction with the adversary $\mathcal{A}$. Whatever parties $\mathcal{A}$ wishes to corrupt, $\mathcal{A}_{\mathsf{SM}}$ corrupts the same parties in its interaction with π^{SM}. Similarly, whatever messages $\mathcal{A}$ sends to $\mathcal{A}_{\mathsf{SM}}$ as part of the protocol π that correspond to π^{SM} messages, $\mathcal{A}_{\mathsf{SM}}$ sends the same messages to the challenger $\mathcal{C}$. Now, whatever messages $\mathcal{C}$ sends, $\mathcal{A}_{\mathsf{SM}}$ forwards the same to the adversary $\mathcal{A}$ as its messages for the π^{SM} protocol. $\mathcal{A}_{\mathsf{SM}}$ does everything else exactly as in Hyb_5.

Observe that $\mathcal{A}_{\mathsf{SM}}$ runs in time $\mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}}$. If $\mathcal{C}$ sends messages that are computed correctly, this exactly corresponds to Hyb_4 in $\mathcal{A}_{\mathsf{SM}}$'s interaction with $\mathcal{A}$. On the other hand, if $\mathcal{C}$ sends simulated messages, this exactly corresponds to Hyb_5. Therefore, if $\mathcal{A}$ can distinguish between these two hybrids, $\mathcal{A}_{\mathsf{SM}}$ can use the same distinguishing guess to break the security of protocol π^{SM}. However, π^{SM} is secure against all adversaries running in time T_{SM}, where $\mathsf{T}_{\mathsf{SM}} > \mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}}$ and hence this is a contradiction. This completes the proof of the lemma. □

5 Two Round Malicious Secure MPC for Input-less Functionalities

Let f be any input-less functionality randomized functionalities. Consider n parties $\mathsf{P}_1, \ldots, \mathsf{P}_n$ who wish to compute f by running a secure multiparty computation(MPC) protocol. Let π^{SM} be any 2 round MPC protocol for f in the plain model,that is secure against semi-malicious adversaries corrupting upto $(n-1)$ parties (such a protocol for general functionalities was described in [BHP17]). In this section, we show how to generically transform π^{SM} into a 2 round protocol π_1 without setup with super-polynomial simulation and secure against malicious adversaries that can corrupt upto $(n-1)$ parties. Formally, we prove the following theorem:

Theorem 3. *Assuming sub-exponentially secure:*

- *A, where $A \in \{DDH,\ Quadratic\ Residuosity,\ N^{th}\ Residuosity\}$ AND*
- *2 round MPC protocol for any functionality f that is secure against semi-malicious adversaries,*

the protocol presented in Figure 4 is a 2 round MPC protocol for any input-less randomized functionality f, in the plain model with super-polynomial simulation.

We can instantiate the underlying MPC protocol with the 2 round construction of [DHRW16] to get the following corollary:

Corollary 4. *Assuming sub-exponentially secure:*

- *A, where $A \in \{DDH,\ Quadratic\ Residuosity,\ N^{th}\ Residuosity\}$ AND*
- *Indistinguishability Obfuscation,*

the protocol presented in Figure 4 is a 2 round MPC protocol for any input-less randomized functionality f in the plain model with super-polynomial simulation.

5.1 High-Level Overview

Before describing our protocol formally, to help the exposition, we first give a brief overview of the construction in this subsection.

Consider n parties $\mathsf{P}_1, \ldots, \mathsf{P}_n$ with no inputs who wish to run a secure MPC to compute an input-less randomized function f. Initially, each party P_i picks some randomness r_i that it will use to run the semi-malicious protocol π^{SM} for the same functionality f.

In the first round, each party P_i sends the first round message of the protocol π^{SM}. Then, with every other party P_j, P_i initiates an execution of the $\mathsf{SPSS.ZK}$ argument system playing the verifier's role. Additionally, P_i and P_j also initiate two executions of a non-malleable commitment scheme - each acting as the committer in one of them. P_i commits to the randomness r_i used in the protocol π^{SM}.

In the second round, each party P_i sends the second round message of the protocol π^{SM} using randomness r_i. Then, P_i finishes executing the non-malleable commitments (playing the committer's role) with every other party P_j, committing to rr_i. Finally, with every other party P_j, P_i completes the execution of the $\mathsf{SPSS.ZK}$ argument by sending its second message - P_i proves that the two messages sent so far using the protocol π^{SM} were correctly generated using the randomness r_i committed to using the non-malleable commitment.

Each party P_i now computes its final output as follows. P_i first verifies all the proofs it received in the last round and sends a global abort (asking all the parties to abort) if any proof does not verify. Then, P_i computes the output using the output computation algorithm of the semi-malicious protocol π^{SM}. This completes the protocol description.

Security Proof: We now briefly describe how the security proof works. Let's consider an adversary $\mathcal{A}$ who corrupts a set of parties. Recall that the goal is to move from the real world to the ideal world such that the outputs of the honest parties along with the view of the adversary is indistinguishable. We do this via a sequence of computationally indistinguishable hybrids.

In the first hybrid Hyb_1, we start with the real world.

Then, in Hyb_2, we switch the $\mathsf{SPSS.ZK}$ proofs used by all honest parties in round 2 to simulated proofs. This hybrid is computationally indistinguishable from the previous hybrid by the security of the $\mathsf{SPSS.ZK}$ system.

In Hyb_3, we switch all the non-malleable commitments sent by honest parties to be commitments of 0 rather than the randomness. Recall that since the proofs were simulated, this doesn't violate correctness. Also, this hybrid is computationally indistinguishable from the previous hybrid by the security of the non-malleable commitment scheme.

Then, in Hyb_4, the simulator extracts the adversary's randomness (used in protocol π^{SM}) by a brute force break of the non-malleable commitment. The simulator aborts if the extracted values don't reconstruct the protocol messages correctly. These two hybrids are indistinguishable because from the soundness of the proof system, the extraction works correctly except with negligible probability. One technicality here is that since we are giving simulated proofs at this point, we cannot rely on soundness anymore. To get around this, from the very first hybrid, we maintain the invariant that in every hybrid, the value committed by the adversary using the non-malleable commitments can be used to reconstruct the messages used in the semi-malicious protocol. Therefore, at this point, as in Section 4, we need the time taken to break the non-malleable commitment scheme $\mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}}$ to be lesser than the time against which the zero knowledge property holds - T_{ZK}. We elaborate on this in the formal proof.

Then, in Hyb_5 we run the simulator of π^{SM} using the extracted values to generate the protocol messages. This hybrid is indistinguishable from the previous one by the security of π^{SM}. Once again, in order to ensure correctness of the extracted values, we require the running time of the simulator - which is $\mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}}$ to be lesser than the time against which the semi-malicious protocol π^{SM} is secure. This is because, then, the simulator can continue to extract the adversary's message and randomness used for the protocol π^{SM} by breaking the semi-malicious protocol.

Finally, Hyb_5 corresponds to the ideal world. Notice that our simulation is in fact straight-line. There are some slight technicalities that arise and we elaborate on this in the formal proof. We now refer the reader to the formal protocol construction.

5.2 Construction

As in Section 4, we first list some notation and the primitives used before describing the construction.

Notation:

- λ denotes the security parameter.
- $\mathsf{SPSS.ZK} = (\mathsf{ZK}_1, \mathsf{ZK}_2, \mathsf{ZK}_3)$ is a two message zero knowledge argument with superior super polynomial simulation (SPSS-ZK). The zero knowledge property holds against all adversaries running in time T_{ZK}. Let $\mathsf{Sim}^{\mathsf{ZK}}$ denote the simulator that produces simulated ZK proofs and let $\mathsf{T}^{\mathsf{Sim}}_{\mathsf{ZK}}$ denote its running time. [KS17] give a construction of an $\mathsf{SPSS.ZK}$ scheme satisfying these properties that can be based on one of the following sub-exponential assumptions: 1) DDH; 2) Quadratic Residuosity; 3) N^{th} Residuosity.
- $\mathsf{NMCom} = (\mathsf{NMCom}^{R}_1, \mathsf{NMCom}^{\mathsf{S}}_1, \mathsf{NMCom}^{S}_2)$ is a two message concurrent non-malleable commitment scheme with respect to commitment in the simultaneous message model. Here, $\mathsf{NMCom}^{R}_1, \mathsf{NMCom}^{\mathsf{S}}_1$ denote the first message of the receiver and sender respectively while NMCom^{S}_2 denotes the second message of the sender. It is secure against all adversaries running in time $\mathsf{T}^{\mathsf{Sec}}_{\mathsf{Com}}$, but can be broken by adversaries running in time $\mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}}$. Let $\mathsf{Ext.Com}$ denote a brute force algorithm running in time $\mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}}$ that can break the commitment scheme just using the first round messages. [KS17] give a construction of an NMCom scheme satisfying these properties that can be based on one of the following sub-exponential assumptions: 1) DDH; 2) Quadratic Residuosity; 3) N^{th} Residuosity.

- π^{SM} is a sub-exponentially secure 2 round MPC protocol that is secure against semi-malicious adversaries. This protocol is secure against all adversaries running in time T_{SM}. Let $(\mathsf{MSG}_1, \mathsf{MSG}_2)$ denote the algorithms used by any party to compute the messages in each of the two rounds and OUT denotes the algorithm to compute the final output. Further, let's assume that this protocol π^{SM} runs over a broadcast channel. Let $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ denote the simulator for the protocol π^{SM} - that is, $\mathcal{S}_i$ is the simulator's algorithm to compute the i^{th} round messages. Also, we make the following assumptions about the protocol structure that is satisfied by the instantiations:

  1. Since the protocol is for input-less functionalities, we assume that $\mathcal{S}_1$ is identical to the algorithm MSG_1 used by honest parties to generate their first message.
  2. The algorithm MSG_2 doesn't use any new randomness that was not already used in the algorithm MSG_1. This is similar to the assumption used in Section 4.

In order to realize our protocol, we require that $\mathsf{poly}(\lambda) < \mathsf{T}^{\mathsf{Sim}}_{\mathsf{ZK}} < \mathsf{T}^{\mathsf{Sec}}_{\mathsf{Com}} < \mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}} < \mathsf{T}_{\mathsf{ZK}}, \mathsf{T}_{\mathsf{SM}}$.

The construction of the protocol is described in Figure 4. We assume broadcast channels. In our construction, we use proofs for a some NP languages that we elaborate on below.

NP language L is characterized by the following relation R.
Statement : $\mathsf{st} = (\mathsf{c}_1, \hat{\mathsf{c}}_1, \mathsf{c}_2, \mathsf{msg}_1, \mathsf{msg}_2, \tau)$
Witness : $\mathsf{w} = (\mathsf{r}, \mathsf{r_c})$
$R(\mathsf{st}, \mathsf{w}) = 1$ if and only if :

- $\hat{\mathsf{c}}_1 = \mathsf{NMCom}^{\mathsf{S}}_1(\mathsf{r}; \mathsf{r_c})$ AND
- $\mathsf{c}_2 = \mathsf{NMCom}^{S}_2(\mathsf{r}, \mathsf{c}_1; \mathsf{r_c})$ AND
- $\mathsf{msg}_1 = \mathsf{MSG}_1(\perp; \mathsf{r})$ AND
- $\mathsf{msg}_2 = \mathsf{MSG}_2(\perp, \tau; \mathsf{r})$

That is, the messages $(\mathsf{c}_1, \hat{\mathsf{c}}_1, \mathsf{c}_2)$ form a non-malleable commitment of $(\mathsf{inp}, \mathsf{r})$ such that msg_2 is the second round message using input inp, randomness r by running the protocol π^{SM}, where the protocol transcript so far is τ.

In the protocol, let's assume that every party has an associated identity id. For any session sid, each parties generates its non-malleable commitment using the tag $(\mathsf{id}||\mathsf{sid})$.

The correctness of the protocol follows from the correctness of the protocol π^{SM}, the non-malleable commitment scheme NMCom and the zero knowledge proof system $\mathsf{SPSS.ZK}$.

5.3 Security Proof

In this section, we formally prove Theorem 3.
Consider an adversary $\mathcal{A}$ who corrupts t parties where $t < n$. For each party P_i, let's say that the size of randomness used in the protocol π^{SM} is $p(\lambda)$ for some polynomial p. That is, $|\mathsf{r}_i| = p(\lambda)$. The strategy of the simulator Sim against a malicious adversary $\mathcal{A}$ is described in *Figure 5*.

Here, notice that since there is no input, the simulator gets the output from the ideal functionality - y right at the beginning. It still has to instruct the functionality to deliver output to the honest party.

We now show that the simulation strategy described in Figure 5 is successful against all malicious PPT adversaries. That is, the view of the adversary along with the output of the honest parties

Inputs: Each party P_i uses randomness r_i to compute the message in each round of the protocol π^{SM}. To make the exposition easier, we think of each party's input as being $\perp$. We now describe the messages sent by party P_i. We will use superscripts to denote the intended recipient of the message if it isn't meant to be used by all parties.

1. **Round 1:**
   P_i does the following:
   - Compute $\mathsf{msg}_{1,i} \leftarrow \mathsf{MSG}_1(\perp; \mathsf{r}_i)$.
   - For each $j \in [n]$ with $j \neq i$, compute:
     - $\hat{\mathsf{c}}^j_{1,i} \leftarrow \mathsf{NMCom}^{\mathsf{S}}_1(\mathsf{r}_i; \mathsf{r}^j_{\mathsf{c},i})$ using a random string $\mathsf{r}^j_{\mathsf{c},i}$. $\mathsf{c}^j_{1,i} \leftarrow \mathsf{NMCom}^R_1(1^\lambda)$.
     - $(\mathsf{ver}^j_{1,i}, \mathsf{zkst}^j_{1,i}) \leftarrow \mathsf{ZK}_1(1^\lambda)$.
   - Send $(\mathsf{msg}_{1,i}, \hat{\mathsf{c}}^j_{1,i}, \mathsf{c}^j_{1,i}, \mathsf{ver}^j_{1,i})$ for all j.
2. **Round 2:**
   Let τ_1 denote the protocol transcript after round 1. P_i does the following:
   - Compute $\mathsf{msg}_{2,i} \leftarrow \mathsf{MSG}_2(\perp, \tau_1; \mathsf{r}_i)$.
   - For each $j \in [n]$ with $j \neq i$, compute:
     - $\mathsf{c}^j_{2,i} \leftarrow \mathsf{NMCom}^S_2(\mathsf{r}_i, \mathsf{c}^i_{1,j}; \mathsf{r}^j_{\mathsf{c},i})$ using the same random string $\mathsf{r}^j_{\mathsf{c},i}$.
     - $\mathsf{prove}^j_{2,i} \leftarrow \mathsf{ZK}_2(\mathsf{ver}^i_{1,j}, \mathsf{st}^j_{2,i}, \mathsf{w}^j_{2,i})$ for the statement $\mathsf{st}^j_{2,i} = (\mathsf{c}^i_{1,j}, \hat{\mathsf{c}}^j_{1,i}, \mathsf{c}^j_{2,i}, \mathsf{msg}_{1,i}, \mathsf{msg}_{2,i}, \tau_1) \in L$ using witness $\mathsf{w}^j_{2,i} = (\mathsf{r}_i, \mathsf{r}^j_{\mathsf{c},i})$.
   - Send $(\mathsf{msg}_{2,i}, \mathsf{c}^j_{2,i}, \mathsf{prove}^j_{2,i})$ for all j.
3. **Output Computation:**
   Let τ_2 denote the protocol transcript after round 2. P_i does the following:
   - For each $j \in [n]$ with $j \neq i$, do:
     - Abort if $\mathsf{ZK}_3(\mathsf{zkst}^j_{1,i}, \mathsf{st}^i_{2,j}) \neq 1$ where $\mathsf{st}^i_{2,j} = (\mathsf{c}^j_{1,i}, \hat{\mathsf{c}}^i_{1,j}, \mathsf{c}^i_{2,j}, \mathsf{msg}_{1,j}, \mathsf{msg}_{2,j}, \tau_1)$. In particular, send a global abort signal to all parties so that everyone aborts.
   - Compute output $\mathsf{y}_i \leftarrow \mathsf{OUT}(\perp, \tau_2; \mathsf{r}_i)$.

Figure 4: 2 round MPC Protocol π_1 for input-less randomized functionality f.

is computationally indistinguishable in the real and ideal worlds. We will show this via a series of computationally indistinguishable hybrids where the first hybrid Hyb_1 corresponds to the real world and the last hybrid Hyb_6 corresponds to the ideal world.

1. Hyb_1**:** In this hybrid, consider a simulator $\mathsf{Sim}_{\mathsf{Hyb}}$ that plays the role of the honest parties. $\mathsf{Sim}_{\mathsf{Hyb}}$ runs in polynomial time.

2. Hyb_2**:** This hybrid is identical to the previous hybrid except that in Round 2, $\mathsf{Sim}_{\mathsf{Hyb}}$ now computes simulated SPSSZK proofs as done in Round 2 in Figure 5. Here, $\mathsf{Sim}_{\mathsf{Hyb}}$ runs in time $\mathsf{T}^{\mathsf{Sim}}_{\mathsf{ZK}}$.

3. Hyb_3**:** This hybrid is identical to the previous hybrid except that $\mathsf{Sim}_{\mathsf{Hyb}}$ now computes all the $(\hat{\mathsf{c}}^j_{1,i}, \mathsf{c}^j_{2,i})$ as non-malleable commitments of $0^{p(\lambda)}$ as done in Round 2 in Figure 5. Once again, $\mathsf{Sim}_{\mathsf{Hyb}}$ runs in time $\mathsf{T}^{\mathsf{Sim}}_{\mathsf{ZK}}$.

4. Hyb_4**:** In this hybrid, the simulator $\mathsf{Sim}_{\mathsf{Hyb}}$ also runs the "Randomness Extraction" phase and the "Special Abort" phase in steps 2 and 4 in Figure 5. Now, $\mathsf{Sim}_{\mathsf{Hyb}}$ runs in time $\mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}}$.

1. **Round 1:** For each honest party P_i, Sim does the following:
    - Compute $\mathsf{msg}_{1,i} \leftarrow \mathsf{MSG}_1(\perp; \mathsf{r}_i)$ using some random string r_i. Recall that this is identical to running the simulator $\mathcal{S}_1(1^\lambda, i)$. For each $j \in [n]$ with $j \neq i$, compute $\hat{\mathsf{c}}^j_{1,i} \leftarrow \mathsf{NMCom}^{\mathsf{S}}_1(0^{p(\lambda)})$, $\mathsf{c}^j_{1,i} \leftarrow \mathsf{NMCom}^R_1(1^\lambda)$ and $(\mathsf{ver}^j_{1,i}, \mathsf{zkst}^j_{1,i}) \leftarrow \mathsf{ZK}_1(1^\lambda)$.
    - Send $(\mathsf{msg}_{1,i}, \hat{\mathsf{c}}^j_{1,i}, \mathsf{c}^j_{1,i}, \mathsf{ver}^j_{1,i})$ for all $j \in [n]$.
2. **Randomness Extraction:** Sim does the following:
    - For each honest party P_i and for each $j \in [n]$ with $j \neq i$, do:
        - Compute $(\mathsf{r}^i_j) = \mathsf{Ext.Com}(\mathsf{c}^j_{1,i}, \hat{\mathsf{c}}^i_{1,j})$. That is, this is the randomness of party P_j seen by party P_i. This step takes time $\mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}}$.
    - Initialize a variable $\mathsf{correct} = 1$. Then, for each malicious party P_j, do:
        - Set $\mathsf{correct} = 0$ if the set of values $\{\mathsf{r}^i_j\}$, for all i corresponding to honest parties P_i is not equal. Set $\mathsf{r}_j = \mathsf{r}^1_j$ and let R denote the set of all $\{\mathsf{r}_j\}$.
        - Set $\mathsf{correct} = 0$ if $\mathsf{msg}_{1,j} \neq \mathsf{MSG}_1(\perp, \mathsf{r}_j)$.
3. **Round 2:** Let τ_1 denote the protocol transcript after round 1. Sim does the following:
    - For each honest party P_i:
        - If $\mathsf{correct} = 1$, compute $\mathsf{msg}_{2,i} \leftarrow \mathcal{S}_2(\tau_1, \mathsf{R}, i)$.
        - Else, compute $\mathsf{msg}_{2,i} \leftarrow \mathsf{MSG}_2(\perp, \tau_1; \mathsf{r}_i)$ where r_i was used in round 1.
    - For each honest party P_i and for each $j \in [n]$ with $j \neq i$, compute:
        - $\mathsf{c}^j_{2,i} \leftarrow \mathsf{NMCom}^S_2(0^{p(\lambda)}, \mathsf{c}^i_{1,j}; \mathsf{r}^j_{\mathsf{c},i})$ using a random string $\mathsf{r}^j_{\mathsf{c},i}$.
        - $\mathsf{prove}^j_{2,i} \leftarrow \mathsf{Sim}^{\mathsf{ZK}}(\mathsf{ver}^i_{1,j}, \mathsf{st}^j_{2,i})$ for $\mathsf{st}^j_{2,i} = (\mathsf{c}^i_{1,j}, \hat{\mathsf{c}}^j_{1,i}, \mathsf{c}^j_{2,i}, \mathsf{msg}_{1,i}, \mathsf{msg}_{2,i}, \tau_1) \in L$. Observe that this takes time $\mathsf{T}^{\mathsf{Sim}}_{\mathsf{ZK}}$.
    - Send $(\mathsf{msg}_{2,i}, \mathsf{c}^j_{2,i}, \mathsf{prove}^j_{2,i})$ for all $j \in [n]$.
4. **Special Abort Phase:** For each malicious party P_j:
    - Output "Special Abort" if $\mathsf{correct} = 0$.
    - Also, output "Special Abort" if $\mathsf{msg}_{2,j} \neq \mathsf{MSG}_2(\perp, \mathsf{r}_j, \tau_1)$.
5. **Output Computation:** Sim does the following:
    - For each honest party P_i and for each $j \in [n]$ with $j \neq i$, abort if $\mathsf{ZK}_3(\mathsf{zkst}^j_{1,i}, \mathsf{st}^i_{2,j}) \neq 1$ where $\mathsf{st}^i_{2,j} = (\mathsf{c}^j_{1,i}, \hat{\mathsf{c}}^i_{1,j}, \mathsf{c}^i_{2,j}, \mathsf{msg}_{1,j}, \mathsf{msg}_{2,j}, \tau_1)$.
    - Else, instruct the ideal functionality to deliver output to the honest parties.

Figure 5: Simulation strategy in the 2 round protocol

5. Hyb_5**:** In this hybrid, if the value of the variable $\mathsf{correct} = 1$, $\mathsf{Sim}_{\mathsf{Hyb}}$ now computes the second round message of the protocol π^{SM} using the simulator algorithms $\mathcal{S}_2$ as done by Sim in the ideal world. $\mathsf{Sim}_{\mathsf{Hyb}}$ also instructs the ideal functionality to deliver outputs to the honest parties as done by Sim. This hybrid is now same as the ideal world. Once again, $\mathsf{Sim}_{\mathsf{Hyb}}$ runs in time $\mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}}$.

We now show that every pair of successive hybrids is computationally indistinguishable. Additionally, we also prove some claims about certain hybrids that aids in the proof.
Before that, we will define the following event E that is useful in the proofs.

Event E: Occurs if, for any malicious party P_j :

- All the $\mathsf{SPSS.ZK}$ proofs sent by P_j in round 2 verify correctly.
(AND)
- Either of the following occur:
  - The set of values $\{\mathsf{r}_j^i\}$ that are committed to using the non-malleable commitment is not same for every i where P_i is honest. (OR)
  - $\mathsf{msg}_{1,j} \neq \mathsf{MSG}_1(\perp, \mathsf{r}_j)$ (OR)
  - $\mathsf{msg}_{2,j} \neq \mathsf{MSG}_2(\perp, \mathsf{r}_j, \tau_1)$ where τ_1 is the protocol transcript after round 1.

Lemma 5. *Assuming soundness of the* $\mathsf{SPSS.ZK}$ *argument system, binding of the non-malleable commitment scheme and correctness of the protocol* π^{SM}, $\mathsf{Pr}[\mathsf{Event\ E\ is\ true\ in\ Hyb}_1] = \mathsf{negl}(\lambda)$.

Proof. Suppose the event E does occur in Hyb_1. From the binding property of the commitment scheme and the correctness of the protocol π^{SM}, observe that if any of the conditions that cause event E to occur are true, it means there exists i, j such that the statement $\mathsf{st}_{2,j}^i = (\mathsf{c}_{1,i}^j, \mathsf{c}_{2,j}^i, \mathsf{msg}_{1,j}, \mathsf{msg}_{2,j}, \tau_1) \notin L$, where P_i is honest and P_j is malicious. However, the proof for the statement verified correctly which means that the adversary has produced a valid proof for a false statement. This violates the soundness property of the SPSSZK argument system which is a contradiction. □

Lemma 6. *Assuming the zero knowledge property of the* $\mathsf{SPSS.ZK}$ *argument system,* Hyb_1 *is computationally indistinguishable from* Hyb_2.

Proof. The only difference between the two hybrids is that in Hyb_1, $\mathsf{Sim}_{\mathsf{Hyb}}$ computes the proofs in Rounds 2 honestly, by running the algorithm ZK_2 of the $\mathsf{SPSS.ZK}$ argument system, whereas in Hyb_3, a simulated proof is used. If the adversary $\mathcal{A}$ can distinguish between the two hybrids, we will use $\mathcal{A}$ to design an algorithm $\mathcal{A}_{\mathsf{ZK}}$ that breaks the zero knowledge property of the argument system.

Suppose the adversary can distinguish between the two hybrids with non-negligible probability p. Then, by a simple hybrid argument, there exists hybrids $\mathsf{Hyb}_{1,k}$ and $\mathsf{Hyb}_{1,k+1}$ that the adversary can distinguish with non-negligible probability $p' < p$ such that: the only difference between the two hybrids is in the proof sent by an honest party P_i to a (malicious) party P_j in round 2.

$\mathcal{A}_{\mathsf{ZK}}$ performs the role of $\mathsf{Sim}_{\mathsf{Hyb}}$ in its interaction with $\mathcal{A}$ and performs all the steps exactly as in $\mathsf{Hyb}_{1,k}$ except the proof in Round 2 sent by P_i to P_j. It interacts with a challenger $\mathcal{C}$ of the $\mathsf{SPSS.ZK}$ argument system and sends the first round message $\mathsf{ver}_{1,j}^i$ it received from the adversary. $\mathcal{A}_{\mathsf{ZK}}$ receives from $\mathcal{C}$ a proof that is either honestly computed or simulated. $\mathcal{A}_{\mathsf{ZK}}$ sets this received proof as its message $\mathsf{prove}_{i,2}^j$ in Round 2 of its interaction with $\mathcal{A}$. In the first case, this exactly corresponds to $\mathsf{Hyb}_{1,k}$ while the latter exactly corresponds to $\mathsf{Hyb}_{1,k+1}$. Therefore, if $\mathcal{A}$ can distinguish between the two hybrids, $\mathcal{A}_{\mathsf{ZK}}$ can use the same distinguishing guess to distinguish the proofs: i.e, decide whether the proofs received from $\mathcal{C}$ were honest or simulated. Now, notice that $\mathcal{A}_{\mathsf{ZK}}$ runs only in polynomial, while the $\mathsf{SPSS.ZK}$ system is secure against adversaries running in time T_{ZK} which is much larger. Thus, this is a contradiction and proves the lemma. □

Lemma 7. *Assuming the zero knowledge property of the* $\mathsf{SPSS.ZK}$ *argument system,*

$$|\mathsf{Pr}[\mathsf{Event\ E\ is\ true\ in\ Hyb}_1] - \mathsf{Pr}[\mathsf{Event\ E\ is\ true\ in\ Hyb}_2]| = \mathsf{negl}(\lambda).$$

Proof. Suppose the claim is not true. That is, there exists some adversary $\mathcal{A}$ for which the difference in the probability of the event E occurring between the two hybrids is some non-negligible probability p. Then we will design an algorithm $\mathcal{A}_{\mathsf{ZK}}$ that breaks the zero knowledge property of the argument system.

Let's say the number of proofs given by an honest party to a malicious party is q. Then, consider a set of intermediate hybrids $\mathsf{Hyb}_{1,1}, \ldots, \mathsf{Hyb}_{1,q}$ such that $\mathsf{Hyb}_{1,q} = \mathsf{Hyb}_2$ where the difference between $\mathsf{Hyb}_{1,k-1}$ and $\mathsf{Hyb}_{1,k}$ is that we switch the k^{th} proof alone from honest to simulated. By a simple hybrid argument, there exists a k such that the difference in the probability of the event E occurring between the two hybrids $\mathsf{Hyb}_{1,k-1}$ and $\mathsf{Hyb}_{1,k}$ is some non-negligible probability $p' < p$. Let the proof that is different between the two hybrids be that sent by a honest party P_i to a malicious party P_j.

$\mathcal{A}_{\mathsf{ZK}}$ performs the role of $\mathsf{Sim}_{\mathsf{Hyb}}$ in its interaction with $\mathcal{A}$ and performs all the steps exactly as in $\mathsf{Hyb}_{1,k-1}$ except the proof in Round 2 sent by P_i to P_j. It interacts with a challenger $\mathcal{C}$ of the $\mathsf{SPSS.ZK}$ argument system and sends the first round message $\mathsf{ver}^i_{1,j}$ it received from the adversary. $\mathcal{A}_{\mathsf{ZK}}$ receives from $\mathcal{C}$ a proof that is either honestly computed or simulated. $\mathcal{A}_{\mathsf{ZK}}$ sets this received proof as its message $\mathsf{prove}^j_{i,2}$ in Round 2 of its interaction with $\mathcal{A}$. In the first case, this exactly corresponds to $\mathsf{Hyb}_{1,k-1}$ while the latter exactly corresponds to $\mathsf{Hyb}_{1,k}$.

After completing the experiment, $\mathcal{A}_{\mathsf{ZK}}$ runs the brute force extractor $\mathsf{UC\text{-}Com}$ on each of the adversary's messages to break the commitment. It then checks the adversary's protocol messages to see if the event E did occur. If the event E did occur, it outputs to the challenger $\mathcal{C}$ that the proof was simulated and if the event E did not occur, it says real. $\mathcal{A}_{\mathsf{ZK}}$ takes time $\mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}}$ to run.

Therefore, if $\mathcal{A}$'s probability of making E occur is non-negligibly different in both the hybrids, $\mathcal{A}_{\mathsf{ZK}}$ can distinguish honest proofs from simulated ones with the same probability. Now, notice that $\mathcal{A}_{\mathsf{ZK}}$ runs only in time $\mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}}$, while the $\mathsf{SPSS.ZK}$ system is secure against adversaries running in time T_{ZK} which is much larger. Thus, this is a contradiction and proves the lemma. □

Lemma 8. $\mathsf{Pr}[\mathsf{Event\ E\ is\ true\ in\ Hyb}_2] = \mathsf{negl}(\lambda)$.

Proof. This follows by combining Lemma 5 and Lemma 8. □

Lemma 9. *Assuming the non-malleability property of the non-malleable commitment scheme* NMCom*,* Hyb_2 *is computationally indistinguishable from* Hyb_3*.*

Proof. The only difference between the two hybrids is that in Hyb_2, for every honest party P_i, $\mathsf{Sim}_{\mathsf{Hyb}}$ computes the commitment messages $(\hat{\mathsf{c}}^j_{1,i}, \mathsf{c}^j_{2,i})$ as a commitment of (r_i), whereas in Hyb_3, they are computed as a commitment of $(0^{p(\lambda)})$. If the adversary $\mathcal{A}$ can distinguish between the two hybrids, we will use $\mathcal{A}$ to design an algorithm $\mathcal{A}_{\mathsf{NMC}}$ that breaks the security of the non-malleable commitment scheme NMCom.

$\mathcal{A}_{\mathsf{NMC}}$ acts as the man-in-the-middle adversary interacting with a challenger $\mathcal{C}$. $\mathcal{A}_{\mathsf{NMC}}$ also plays the role of $\mathsf{Sim}_{\mathsf{Hyb}}$ in its interaction with the adversary $\mathcal{A}$. It generates all the messages except the messages $\mathsf{c}^j_{1,i}$ and $(\hat{\mathsf{c}}^j_{1,i}, \mathsf{c}^j_{2,i})$ exactly as done by $\mathsf{Sim}_{\mathsf{Hyb}}$ in Hyb_2. Corresponding to each message $\mathsf{c}^j_{1,i}$ that $\mathcal{A}_{\mathsf{NMC}}$ has to send, it receives one first round message from $\mathcal{C}$ (on the right side) corresponding to the scheme NMCom. Similarly, it receives first round messages $\hat{\mathsf{c}}^j_{1,i}$ from $\mathcal{C}$ (on the left side). $\mathcal{A}_{\mathsf{NMC}}$ forwards these messages to the adversary $\mathcal{A}$ as its first round messages $(\hat{\mathsf{c}}^j_{1,i}, \mathsf{c}^j_{1,i})$. Similarly, for each pair of messages $(\hat{\mathsf{c}}^i_{1,j}, \mathsf{c}^i_{1,j})$ it receives from $\mathcal{A}$ as part of the first round messages of the scheme NMCom, $\mathcal{A}_{\mathsf{NMC}}$ forwards the messages to $\mathcal{C}$ as its first round messages for the commitment (to the left and right side respectively). Then, for each $\mathsf{c}^j_{2,i}$ that $\mathcal{A}_{\mathsf{NMC}}$ is supposed to send to $\mathcal{A}$, it receives a second round commitment message from the challenger $\mathcal{C}$. In one case, all of these are commitments to the respective (r_i) values while in the second case, they are all commitments to

$(0^{p(\lambda)})$. $\mathcal{A}_{\mathsf{NMC}}$ forwards these messages as its commitment messages $\mathsf{c}_{2,i}^{j}$ to the adversary $\mathcal{A}$. Once again, it forwards each message $\mathsf{c}_{2,j}^{i}$ it receives from $\mathcal{A}$, as its second round commitment message in its interaction with the challenger $\mathcal{C}$. That is, these are the commitments on the right side generated by the man-in-the-middle.

Now, we can clearly see that in the first case, when $\mathcal{C}$ generates commitments to r_i, $\mathcal{A}$'s view corresponds to Hyb_2 while in the latter case, it exactly corresponds to Hyb_3. However, from the security of the non-malleable commitment scheme, the joint distribution of the value committed to by the adversary $\mathcal{A}_{\mathsf{NMC}}$ (which is the same as $\mathcal{A}$'s commitments) and its view must be indistinguishable in both cases. Therefore, if $\mathcal{A}$ can distinguish between the two hybrids, then $\mathcal{A}_{\mathsf{NMC}}$ can break the non-malleability property of the commitment scheme NMCom. However, $\mathcal{A}_{\mathsf{NMC}}$ only runs in time $\mathsf{T}_{\mathsf{ZK}}^{\mathsf{Sim}} < \mathsf{T}_{\mathsf{Com}}^{\mathsf{Sec}}$ and hence this is a contradiction, thus proving the sub-lemma.

Also, notice that since the joint distribution of the adversary $\mathcal{A}$'s committed values and his view is indistinguishable in both hybrids, this implies that Event E still occurs only with negligible probability in Hyb_3 as well. □

Lemma 10. Hyb_3 *is statistically indistinguishable from* Hyb_4*.*

Proof. The only difference between the two hybrids is that in Hyb_4, the simulator might output "Special Abort" which doesn't happen in Hyb_3. As shown in the above proof (of Lemma 9), the probability that Event E occurs in Hyb_3 is negligible. Notice from the description of the simulator in Figure 5, the output "Special Abort" occurs exactly if the event E occurs. This means that the probability that the simulator outputs "Special Abort" in Hyb_4 is negligible and this completes the proof. □

Lemma 11. *Assuming the security of the protocol* π^{SM}*,* Hyb_4 *is computationally indistinguishable from* Hyb_5*.*

Proof. The difference between the two hybrids is in the messages of protocol π^{SM}. In Hyb_4, $\mathsf{Sim}_{\mathsf{Hyb}}$ computes the messages of protocol π^{SM} correctly using the honest parties' strategy. In Hyb_5, if $\mathsf{correct} = 1$, they are computed by running the simulator $\mathcal{S}$ for protocol π^{SM} and if $\mathsf{correct} = 0$, they are computed using the honest parties' strategy. Therefore, the only difference is if $\mathsf{correct} = 1$. If the adversary $\mathcal{A}$ can distinguish between the two hybrids, we will use $\mathcal{A}$ to design an algorithm $\mathcal{A}_{\mathsf{SM}}$ that can break the security of protocol π^{SM}.

$\mathcal{A}_{\mathsf{SM}}$ interacts with a challenger $\mathcal{C}$ to break the security of protocol π^{SM}. Also, $\mathcal{A}_{\mathsf{SM}}$ performs the role of $\mathsf{Sim}_{\mathsf{Hyb}}$ in its interaction with the adversary $\mathcal{A}$. Whatever parties $\mathcal{A}$ wishes to corrupt, $\mathcal{A}_{\mathsf{SM}}$ corrupts the same parties in its interaction with π^{SM}. Similarly, whatever messages $\mathcal{A}$ sends to $\mathcal{A}_{\mathsf{SM}}$ as part of the protocol π that correspond to π^{SM} messages, $\mathcal{A}_{\mathsf{SM}}$ sends the same messages to the challenger $\mathcal{C}$. Now, whatever messages $\mathcal{C}$ sends, $\mathcal{A}_{\mathsf{SM}}$ forwards the same to the adversary $\mathcal{A}$ as its messages for the π^{SM} protocol. $\mathcal{A}_{\mathsf{SM}}$ does everything else exactly as in Hyb_5.

Observe that $\mathcal{A}_{\mathsf{SM}}$ runs in time $\mathsf{T}_{\mathsf{Com}}^{\mathsf{Brk}}$. If $\mathcal{C}$ sends messages that are computed correctly, this exactly corresponds to Hyb_4 in $\mathcal{A}_{\mathsf{SM}}$'s interaction with $\mathcal{A}$. On the other hand, if $\mathcal{C}$ sends simulated messages, this exactly corresponds to Hyb_5. Therefore, if $\mathcal{A}$ can distinguish between these two hybrids, $\mathcal{A}_{\mathsf{SM}}$ can use the same distinguishing guess to break the security of protocol π^{SM}. However, π^{SM} is secure against all adversaries running in time T_{SM}, where $\mathsf{T}_{\mathsf{SM}} > \mathsf{T}_{\mathsf{Com}}^{\mathsf{Brk}}$ and hence this is a contradiction. This completes the proof of the lemma.

Further, since the two hybrids are indistinguishable, the probability that $\mathsf{Sim}_{\mathsf{Hyb}}$ outputs "Special Abort" in hybrid 5 continues to remain negligible. □

Finally, let us discuss why the simulation strategy in fact does correspond to the ideal world. Notice that if the value of the variable $\mathsf{correct} = 1$, then the simulator forces the output correctly by invoking the semi-malicious protocol's simulator. On the other hand, if $\mathsf{correct} = 0$, and the ZK proof verifies in round 2, then the simulator outputs "Special Abort". However, we proved that the probability of "Special Abort" occurring is negligible. Therefore, if $\mathsf{correct} = 0$, then the ZK proof doesn't verify correctly. As a result, the adversary causes an abort in both the real and ideal worlds and hence the simulator's objective is not to force any output.

6 Three Round Concurrently Secure MPC

Let f be any functionality. Consider n parties $\mathsf{P}_1, \ldots, \mathsf{P}_n$ with inputs $\mathsf{x}_1, \ldots, \mathsf{x}_n$ respectively who wish to compute f on their joint inputs by running a concurrently secure multiparty computation(MPC) protocol. Let π^{SM} be any 3 round protocol that runs without any setup for the above task and is secure against adversaries that can be completely malicious in the first round, semi-malicious in the next two rounds and can corrupt upto $(n-1)$ parties. In this section, we show how to generically transform π^{SM} into a 3 round concurrently secure protocol π^{Conc} without setup with super-polynomial simulation that is secure against malicious adversaries which can corrupt upto $(n-1)$ parties. Formally, we prove the following theorem:

Theorem 5. *Assuming sub-exponentially secure:*

- *A, where $A \in \{DDH,\ Quadratic\ Residuosity,\ N^{th}\ Residuosity\}$ AND*
- *3 round MPC protocol for any functionality f that is stand-alone secure against malicious adversaries in the first round and semi-malicious adversaries in the next two rounds,*

the protocol presented in Figure 6 is a 3 round concurrently secure MPC protocol without any setup with super-polynomial simulation for any functionality f, secure against malicious adversaries.

We can instantiate the underlying MPC protocol with the constructions of [DHRW16, BHP17] to get the following corollary:

Corollary 6. *Assuming sub-exponentially secure:*

- *A, where $A \in \{DDH,\ Quadratic\ Residuosity,\ N^{th}\ Residuosity\}$ AND*
- *B, where $B \in \{LWE,\ Indistinguishability\ Obfuscation\}$*

the protocol presented in Figure 6 is a 3 round concurrently secure MPC protocol without any setup with super-polynomial simulation for any functionality f, secure against malicious adversaries.

We essentially prove that the same protocol from Section 4 is also concurrently secure. The proof is fairly simple and not too different from the proof of stand-alone security, because the simulation strategy as well as all reductions are straight-line. The only use of rewinding occurs (implicitly) within the proof of non-malleability, which we carefully combine with identities to ensure that the protocol remains concurrently secure. For the sake of completeness, we write out the protocol and the proof in their entirety. As in Section 4, we first list some notation and the primitives used before describing the construction.

Notation:

- λ denotes the security parameter.

- $\mathsf{SPSS.ZK} = (\mathsf{ZK}_1, \mathsf{ZK}_2, \mathsf{ZK}_3)$ is a two message zero knowledge argument with superior super polynomial simulation (SPSS-ZK). The zero knowledge property holds against all adversaries running in time T_{ZK}. Let $\mathsf{Sim}^{\mathsf{ZK}}$ denote the simulator that produces simulated ZK proofs and let $\mathsf{T}^{\mathsf{Sim}}_{\mathsf{ZK}}$ denote its running time. [KS17] give a construction of an $\mathsf{SPSS.ZK}$ scheme satisfying these properties that can be based on one of the following sub-exponential assumptions: 1) DDH; 2) Quadratic Residuosity; 3) N^{th} Residuosity.

- $\mathsf{NMCom} = (\mathsf{NMCom}^R_1, \mathsf{NMCom}^{\mathsf{S}}_1, \mathsf{NMCom}^S_2)$ is a two message concurrent non-malleable commitment scheme with respect to commitment in the simultaneous message model. Here, $\mathsf{NMCom}^R_1, \mathsf{NMCom}^{\mathsf{S}}_1$ denote the first message of the receiver and sender respectively while NMCom^S_2 denotes the second message of the sender. It is secure against all adversaries running in time $\mathsf{T}^{\mathsf{Sec}}_{\mathsf{Com}}$, but can be broken by adversaries running in time $\mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}}$. Let $\mathsf{Ext.Com}$ denote a brute force algorithm running in time $\mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}}$ that can break the commitment scheme. [KS17] give a construction of an NMCom scheme satisfying these properties that can be based on one of the following sub-exponential assumptions: 1) DDH; 2) Quadratic Residuosity; 3) N^{th} Residuosity.

- π^{SM} is a sub-exponentially secure 3 round stand-alone MPC protocol that is secure against malicious adversaries in the first round and semi-malicious adversaries in the next two rounds. This protocol is secure against all adversaries running in time T_{SM}. Let $(\mathsf{MSG}_1, \mathsf{MSG}_2, \mathsf{MSG}_3)$ denote the algorithms used by any party to compute the messages in each of the three rounds and OUT denotes the algorithm to compute the final output. Further, let's assume that this protocol π^{SM} runs over a broadcast channel. Let $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3)$ denote the straight line simulator for this protocol - that is, $\mathcal{S}_i$ is the simulator's algorithm to compute the i^{th} round messages. Also, we make the following assumptions about the protocol structure that is satisfied by the instantiations:

  1. $\mathcal{S}_1$ and $\mathcal{S}_2$ run without any input other than the protocol transcript so far - in particular, they don't need the input, randomness and output of the malicious parties. For $\mathcal{S}_1$, this must necessarily be true since the first round of π^{SM} is secure against malicious adversaries. We make the assumption only on $\mathcal{S}_2$.[3]
  2. The algorithm MSG_3 doesn't use any new input or randomness that was not already used in the algorithms $\mathsf{MSG}_1, \mathsf{MSG}_2$. Looking ahead, this is used in our security proof when we want to invoke the simulator of this protocol π^{SM}, we need to be sure that we have fed the correct input and randomness to the simulator.

In order to realize our protocol, we require that $\mathsf{poly}(\lambda) < \mathsf{T}^{\mathsf{Sim}}_{\mathsf{ZK}} < \mathsf{T}^{\mathsf{Sec}}_{\mathsf{Com}} < \mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}} < \mathsf{T}_{\mathsf{ZK}}, \mathsf{T}_{\mathsf{SM}}$.

The construction of the protocol is described in Figure 6. We assume broadcast channels. In our construction, we use proofs for a some NP languages that we elaborate on below.

NP language L is characterized by the following relation R.
Statement : $\mathsf{st} = (\mathsf{c}_1, \hat{\mathsf{c}}_1, \mathsf{c}_2, \mathsf{tag}, \mathsf{msg}_1, \mathsf{msg}_2, \tau)$
Witness : $\mathsf{w} = (\mathsf{inp}, \mathsf{r}, \mathsf{r}_\mathsf{c})$
$R(\mathsf{st}, \mathsf{w}) = 1$ if and only if :

- $\hat{\mathsf{c}}_1 = \mathsf{NMCom}^{\mathsf{S}}_1((\mathsf{inp}, \mathsf{r}), \mathsf{tag}; \mathsf{r}_\mathsf{c})$ AND

- $\mathsf{c}_2 = \mathsf{NMCom}^S_2((\mathsf{inp}, \mathsf{r}), \mathsf{tag}, \mathsf{c}_1; \mathsf{r}_\mathsf{c})$ AND

[3] This assumption can be removed by running the commitment extractor on the first round messages itself as used in Section 5.

- $\mathsf{msg}_1 = \mathsf{MSG}_1(\mathsf{inp};\mathsf{r})$ AND
- $\mathsf{msg}_2 = \mathsf{MSG}_2(\mathsf{inp},\tau;\mathsf{r})$

That is, the messages $(\mathsf{c}_1,\hat{\mathsf{c}}_1,\mathsf{c}_2)$ form a non-malleable commitment of $(\mathsf{inp},\mathsf{r})$ such that msg_2 is the second round message using input inp, randomness r by running the protocol π^{SM}, where the protocol transcript so far is τ.

NP language L_1 is characterized by the following relation R_1.
Statement : $\mathsf{st} = (\mathsf{c}_1,\hat{\mathsf{c}}_1,\mathsf{c}_2,\mathsf{tag},\mathsf{msg}_3,\tau)$
Witness : $\mathsf{w} = (\mathsf{inp},\mathsf{r},\mathsf{r}_\mathsf{c})$
$R(\mathsf{st},\mathsf{w}) = 1$ if and only if :

- $\hat{\mathsf{c}}_1 = \mathsf{NMCom}^{\mathsf{S}}_1((\mathsf{inp},\mathsf{r}),\mathsf{tag};\mathsf{r}_\mathsf{c})$ AND
- $\mathsf{c}_2 = \mathsf{NMCom}^{S}_2((\mathsf{inp},\mathsf{r}),\mathsf{tag},\mathsf{c}_1;\mathsf{r}_\mathsf{c})$ AND
- $\mathsf{msg}_3 = \mathsf{MSG}_3(\mathsf{inp},\tau;\mathsf{r})$

That is, the messages $(\mathsf{c}_1,\hat{\mathsf{c}}_1,\mathsf{c}_2)$ form a non-malleable commitment of $(\mathsf{inp},\mathsf{r})$ such that msg_3 is the third round message using input inp, randomness r by running the protocol π^{SM}, where the protocol transcript so far is τ.

In the protocol, let's assume that every party has an associated identity id. For any session sid, each parties generates its non-malleable commitment using the tag $(\mathsf{id}||\mathsf{sid})$. We make the tags explicit in the protocol here to illustrate concurrency.

The correctness of the protocol follows from the correctness of the protocol π^{SM}, the non-malleable commitment scheme NMCom and the zero knowledge proof system $\mathsf{SPSS.ZK}$.

6.1 Security Proof

In this section, we formally prove Theorem 5.
Let the total number of sessions be m where in each session, a set of n parties take part. Note that m is polynomial in the security parameter λ. Consider an adversary $\mathcal{A}$ who corrupts several parties in each session and can schedule the messages across various sessions arbitrarily. For any session s_k, let's say the adversary $\mathcal{A}$ corrupts t_k parties in this session where $t_k < n$. For each party P_i, let's say that the size of input and randomness used in the protocol π^{SM} is $p(\lambda)$ for some polynomial p. That is, $|(\mathsf{x}_i,\mathsf{r}_i)| = p(\lambda)$. The strategy of the simulator Sim against a malicious adversary $\mathcal{A}$ is described in *Figure 7*.

Here, in the simulation, we crucially use the two assumptions about the protocol structure. The first one is easy to notice since the simulator Sim, in each session, has to run the semi-malicious to produce the first and second messages before it has extracted the adversary's input and randomness. For the second assumption, observe that in order to run the simulator algorithm $\mathcal{S}_3$, Sim has to feed it the entire input and randomness of the adversary and so these have to be bound to by the end of the second round.

We now show that the simulation strategy described in Figure 7 is successful against all malicious PPT adversaries. That is, the view of the adversary along with the output of the honest parties is computationally indistinguishable in the real and ideal worlds. We will show this via a series of computationally indistinguishable hybrids where the first hybrid Hyb_1 corresponds to the real world and the last hybrid Hyb_6 corresponds to the ideal world.

Inputs: Each party P_i has input x_i and uses randomness r_i to compute the message in each round of the protocol π^{SM}. We now describe the messages sent by party P_i. We will use superscripts to denote the intended recipient of the message if it isn't meant to be used by all parties.

1. **Round 1:**
   P_i does the following:
   - Compute $\mathsf{msg}_{1,i} \leftarrow \mathsf{MSG}_1(\mathsf{x}_i; \mathsf{r}_i)$.
   - For each $j \in [n]$ with $j \neq i$, compute:
     - $\hat{\mathsf{c}}^j_{1,i} \leftarrow \mathsf{NMCom}^{\mathsf{S}}_1((\mathsf{x}_i, \mathsf{r}_i), \mathsf{tag}_{i,j}; \mathsf{r}^j_{\mathsf{c},i})$ using a random string $\mathsf{r}^j_{\mathsf{c},i}$ and $\mathsf{c}^j_{1,i} \leftarrow \mathsf{NMCom}^R_1(1^\lambda)$.
     - $(\mathsf{ver}^j_{1,i}, \mathsf{zkst}^j_{1,i}) \leftarrow \mathsf{ZK}_1(1^\lambda)$ and $(\mathsf{ver}^j_{2,i}, \mathsf{zkst}^j_{2,i}) \leftarrow \mathsf{ZK}_1(1^\lambda)$.
   - Send $(\mathsf{msg}_{1,i}, \hat{\mathsf{c}}^j_{1,i}, \mathsf{c}^j_{1,i}, \mathsf{ver}^j_{1,i}, \mathsf{ver}^j_{2,i})$ for all j.
2. **Round 2:**
   Let τ_1 denote the protocol transcript after round 1. P_i does the following:
   - Compute $\mathsf{msg}_{2,i} \leftarrow \mathsf{MSG}_2(\mathsf{x}_i, \tau_1; \mathsf{r}_i)$.
   - For each $j \in [n]$ with $j \neq i$, compute:
     - $\mathsf{c}^j_{2,i} \leftarrow \mathsf{NMCom}^S_2((\mathsf{x}_i, \mathsf{r}_i), \mathsf{tag}_{i,j}, \mathsf{c}^i_{1,j}; \mathsf{r}^j_{\mathsf{c},i})$ using the same random string $\mathsf{r}^j_{\mathsf{c},i}$.
     - $\mathsf{prove}^j_{2,i} \leftarrow \mathsf{ZK}_2(\mathsf{ver}^i_{1,j}, \mathsf{st}^j_{2,i}, \mathsf{w}^j_{2,i})$ for the statement $\mathsf{st}^j_{2,i} = (\mathsf{c}^i_{1,j}, \hat{\mathsf{c}}^j_{1,i}, \mathsf{c}^j_{2,i}, \mathsf{tag}_{i,j}, \mathsf{msg}_{1,i}, \mathsf{msg}_{2,i}, \tau_1) \in L$ using witness $\mathsf{w}^j_{2,i} = (\mathsf{x}_i, \mathsf{r}_i, \mathsf{r}^j_{\mathsf{c},i})$.
   - Send $(\mathsf{msg}_{2,i}, \mathsf{c}^j_{2,i}, \mathsf{prove}^j_{2,i})$ for all j.
3. **Round 3:**
   Let τ_2 denote the protocol transcript after round 2. P_i does the following:
   - Compute $\mathsf{msg}_{3,i} \leftarrow \mathsf{MSG}_3(\mathsf{x}_i, \tau_2; \mathsf{r}_i)$.
   - For each $j \in [n]$ with $j \neq i$, do:
     - Abort if $\mathsf{ZK}_3(\mathsf{zkst}^j_{1,i}, \mathsf{st}^i_{2,j}) \neq 1$ where $\mathsf{st}^i_{2,j} = (\mathsf{c}^j_{1,i}, \hat{\mathsf{c}}^i_{1,j}, \mathsf{c}^i_{2,j}, \mathsf{tag}_{i,j}, \mathsf{msg}_{1,j}, \mathsf{msg}_{2,j}, \tau_1)$. In particular, send a global abort signal to all parties so that everyone aborts.
     - $\mathsf{prove}^j_{3,i} \leftarrow \mathsf{ZK}_2(\mathsf{ver}^i_{2,j}, \mathsf{st}^j_{3,i}, \mathsf{w}^j_{3,i})$ for the statement $\mathsf{st}^j_{3,i} = (\mathsf{c}^i_{1,j}, \hat{\mathsf{c}}^j_{1,i}, \mathsf{c}^j_{2,i}, \mathsf{tag}_{i,j}, \mathsf{msg}_{3,i}, \tau_2) \in L_1$ using witness $\mathsf{w}^j_{3,i} = (\mathsf{x}_i, \mathsf{r}_i, \mathsf{r}^j_{\mathsf{c},i})$.
   - Send $(\mathsf{msg}_{3,i}, \mathsf{prove}^j_{3,i})$ for all j.
4. **Output Computation:**
   Let τ_3 denote the protocol transcript after round 3. P_i does the following:
   - For each $j \in [n]$ with $j \neq i$, do:
     - Abort if $\mathsf{ZK}_3(\mathsf{zkst}^j_{2,i}, \mathsf{st}^i_{3,j}) \neq 1$ where $\mathsf{st}^i_{3,j} = (\mathsf{c}^j_{1,i}, \hat{\mathsf{c}}^i_{1,j}, \mathsf{c}^i_{2,j}, \mathsf{tag}_{i,j}, \mathsf{msg}_{3,j}, \tau_2)$. In particular, send a global abort signal to all parties so that everyone aborts.
   - Compute output $\mathsf{y}_i \leftarrow \mathsf{OUT}(\mathsf{x}_i, \tau_3; \mathsf{r}_i)$.

Figure 6: 3 round concurrently secure MPC Protocol π^{Conc} for functionality f.

1. Hyb_1**:** In this hybrid, consider a simulator $\mathsf{Sim}_{\mathsf{Hyb}}$ that plays the role of the honest parties in each sesion. $\mathsf{Sim}_{\mathsf{Hyb}}$ runs in polynomial time.

2. Hyb_2**:** In this hybrid, for each session, the simulator $\mathsf{Sim}_{\mathsf{Hyb}}$ also runs the "Input Extraction"

Corresponding to every session, the simulator stores a database of the private coins and state used in that session. After each round in the protocol for a particular session, the simulator updates the database. For any new session s_k that the adversary initiates, the simulator does the following:

1. **Round 1:** For each honest party P_i, Sim does the following:
    - Compute $\mathsf{msg}_{1,i} \leftarrow \mathcal{S}_1(1^\lambda, i)$. For each $j \in [n]$ with $j \neq i$, compute:
        - $\hat{\mathsf{c}}^j_{1,i} \leftarrow \mathsf{NMCom}^{\mathsf{S}}_1(0^{p(\lambda)}, \mathsf{tag}_{i,j})$, $\mathsf{c}^j_{1,i} \leftarrow \mathsf{NMCom}^R_1(1^\lambda)$.
        - $(\mathsf{ver}^j_{1,i}, \mathsf{zkst}^j_{1,i}) \leftarrow \mathsf{ZK}_1(1^\lambda)$ and $(\mathsf{ver}^j_{2,i}, \mathsf{zkst}^j_{2,i}) \leftarrow \mathsf{ZK}_1(1^\lambda)$.
    - Send $(\mathsf{msg}_{1,i}, \hat{\mathsf{c}}^j_{1,i}, \mathsf{c}^j_{1,i}, \mathsf{ver}^j_{1,i}, \mathsf{ver}^j_{2,i})$ for all $j \in [n]$.

Whenever the adversary schedules round 2 (or any subsequent round) for a session s_k, the simulator, using the appropriate values in the stored database, does the following:

2. **Round 2:** Let τ_1 denote the protocol transcript after round 1. For each honest party P_i,
    - Compute $\mathsf{msg}_{2,i} \leftarrow \mathcal{S}_2(\tau_1, i)$. For each $j \in [n]$ with $j \neq i$, compute:
        - $\mathsf{c}^j_{2,i} \leftarrow \mathsf{NMCom}^S_2(0^{p(\lambda),\mathsf{tag}_{i,j}}, \mathsf{c}^i_{1,j}; \mathsf{r}^j_{\mathsf{c},i})$ using a random string $\mathsf{r}^j_{\mathsf{c},i}$.
        - $\mathsf{prove}^j_{2,i} \leftarrow \mathsf{Sim}^{\mathsf{ZK}}(\mathsf{ver}^i_{1,j}, \mathsf{st}^j_{2,i})$ for $\mathsf{st}^j_{2,i} = (\mathsf{c}^i_{1,j}, \hat{\mathsf{c}}^j_{1,i}, \mathsf{c}^j_{2,i}, \mathsf{tag}_{i,j}, \mathsf{msg}_{1,i}, \mathsf{msg}_{2,i}, \tau_1) \in L$.
    - Send $(\mathsf{msg}_{2,i}, \mathsf{c}^j_{2,i}, \mathsf{prove}^j_{2,i})$ for all $j \in [n]$.
3. **Input Extraction:** Sim does the following:
    - For each honest party P_i and for each $j \in [n]$ with $j \neq i$, do:
        - Abort if $\mathsf{ZK}_3(\mathsf{zkst}^j_{1,i}, \mathsf{st}^i_{2,j}) \neq 1$ where τ_1 is the protocol transcript after round 1 such that $\mathsf{st}^i_{2,j} = (\mathsf{c}^j_{1,i}, \hat{\mathsf{c}}^i_{1,j}, \mathsf{c}^i_{2,j}, \mathsf{tag}_{i,j}, \mathsf{msg}_{1,j}, \mathsf{msg}_{2,j}, \tau_1)$
        - Compute $(\mathsf{x}^i_j, \mathsf{r}^i_j) = \mathsf{Ext.Com}(\mathsf{c}^j_{1,i}, \hat{\mathsf{c}}^i_{1,j}, \mathsf{c}^i_{2,j})$. That is, this is the input and randomness of party P_j seen by party P_i. This step takes time $\mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}}$.
    - For each malicious party P_j, do:
        - Output "Special Abort" if the set of values $\{(\mathsf{x}^i_j, \mathsf{r}^i_j)\}$ computed in the last step, for all i corresponding to honest parties P_i is not equal. Set $(\mathsf{x}_j, \mathsf{r}_j) = (x^1_j, \mathsf{r}^1_j)$. Output "Special Abort" if $\mathsf{msg}_{1,j} \neq \mathsf{MSG}_1(\mathsf{x}_j, \mathsf{r}_j)$ and $\mathsf{msg}_{2,j} \neq \mathsf{MSG}_2(\mathsf{x}_j, \mathsf{r}_j, \tau_1)$.
        - Send all extracted x_j to the trusted functionality and receive output y.
        - Let R denote the set of all $\{\mathsf{x}_j, \mathsf{r}_j\}$.
4. **Round 3:** Let τ_2 denote the protocol transcript after round 2. For each honest party P_i, compute and send $\mathsf{msg}_{3,i} \leftarrow \mathcal{S}_3(\mathsf{y}, \mathsf{R}, \tau_2, i)$ together with $\mathsf{prove}^j_{3,i}$ for $j \in [n], j \neq i$ where $\mathsf{prove}^j_{3,i} \leftarrow \mathsf{Sim}^{\mathsf{ZK}}(\mathsf{ver}^i_{2,j}, \mathsf{st}^j_{3,i})$ for the statement $\mathsf{st}^j_{3,i} = (\mathsf{c}^i_{1,j}, \hat{\mathsf{c}}^j_{1,i}, \mathsf{c}^j_{2,i}, \mathsf{tag}_{i,j}, \mathsf{msg}_{3,i}, \tau_2) \in L_1$. Observe that this takes time $\mathsf{T}^{\mathsf{Sim}}_{\mathsf{ZK}}$.
5. **Special Abort Phase:** Sim does the following:
    - Output "Special Abort" if for each malicious party P_j, $\mathsf{msg}_{3,j} \neq \mathsf{MSG}_3(\mathsf{x}_j, \mathsf{r}_j, \tau_2)$.
6. **Output Computation:** Sim does the following:
    - For each honest party P_i and for each $j \in [n]$ with $j \neq i$, abort if $\mathsf{ZK}_3(\mathsf{zkst}^j_{2,i}, \mathsf{st}^i_{3,j}) \neq 1$ where $\mathsf{st}^i_{3,j} = (\mathsf{c}^j_{1,i}, \hat{\mathsf{c}}^i_{1,j}, \mathsf{c}^i_{2,j}, \mathsf{tag}_{i,j}, \mathsf{msg}_{3,j}, \tau_2)$.
    - Else instruct the ideal functionality to deliver output to the honest parties.

Figure 7: Simulation strategy in the 3 round concurrently secure protocol

phase and the "Special Abort" phase in step3 and 5 in Figure 7. $\mathsf{Sim}_{\mathsf{Hyb}}$ runs in time $\mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}}$.

3. Hyb_3: This hybrid is identical to the previous hybrid except that in each session, in Rounds 2 and 3, $\mathsf{Sim}_{\mathsf{Hyb}}$ now computes simulated SPSSZK proofs as done in Round 2 in Figure 7. Once again, $\mathsf{Sim}_{\mathsf{Hyb}}$ runs in time $\mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}}$.

4. Hyb_4: This hybrid is identical to the previous hybrid except that in each session, $\mathsf{Sim}_{\mathsf{Hyb}}$ now computes all the $(\hat{\mathsf{c}}^j_{1,i}, \mathsf{c}^j_{2,i})$ as non-malleable commitments of $0^{p(\lambda)}$ as done in Round 2 in Figure 7. Once again, $\mathsf{Sim}_{\mathsf{Hyb}}$ runs in time $\mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}}$.

5. Hyb_5: This hybrid is identical to the previous hybrid except that in each session, in Round 3, $\mathsf{Sim}_{\mathsf{Hyb}}$ now computes the messages of the protocol π^{SM} using the simulator algorithms $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3)$ as done by Sim in the ideal world. $\mathsf{Sim}_{\mathsf{Hyb}}$ also instructs the ideal functionality to deliver outputs to the honest parties as done by Sim session by session. This hybrid is now same as the ideal world. Once again, $\mathsf{Sim}_{\mathsf{Hyb}}$ runs in time $\mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}}$.

We now show that every pair of successive hybrids is computationally indistinguishable.

Lemma 12. *Assuming soundness of the* $\mathsf{SPSS.ZK}$ *argument system, binding of the non-malleable commitment scheme and correctness of the protocol* π^{SM}, Hyb_1 *is computationally indistinguishable from* Hyb_2.

Proof. The only difference between the two hybrids is that in Hyb_2, $\mathsf{Sim}_{\mathsf{Hyb}}$ may output "Special Abort" in some session which doesn't happen in Hyb_1. More specifically, in Hyb_2, "Special Abort" occurs in some session if event E described below is true in that session.

Event E: Is true if: For any malicious party P_j

- All the $\mathsf{SPSS.ZK}$ proofs sent by P_j in round 2 and 3 verify correctly.
  (AND)

- Either of the following occur:
  - The set of values $\{(\mathsf{x}^i_j, \mathsf{r}^i_j)\}$ that are committed to using the non-malleable commitment is not same for every i where P_i is honest. (OR)
  - $\mathsf{msg}_{1,j} \neq \mathsf{MSG}_1(\mathsf{x}_j, \mathsf{r}_j)$ (OR)
  - $\mathsf{msg}_{2,j} \neq \mathsf{MSG}_2(\mathsf{x}_j, \mathsf{r}_j, \tau_1)$ where τ_1 is the protocol transcript after round 1. (OR)
  - $\mathsf{msg}_{3,j} \neq \mathsf{MSG}_3(\mathsf{x}_j, \mathsf{r}_j, \tau_2)$ where τ_2 is the protocol transcript after round 2.

That is, in simpler terms, the event E occurs if for any malicious party, it gives valid ZK proofs in round 2 and 3 but its protocol transcript is not consistent with the values it committed to.

Therefore, in order to prove the indistinguishability of the two hybrids, it is enough to prove the lemma below.

Sub-Lemma 5. $\Pr[\text{Event } \mathsf{E} \text{ is true in } \mathsf{Hyb}_2] = \mathsf{negl}(\lambda)$.

Proof. We now prove the sub-lemma. Suppose the event E does occur with non-negligible probability p. Then there exists some session s such that the event E occurs in that session with non-negligible probability $p' < p$. Let's focus on that session s.

From the binding property of the commitment scheme and the correctness of the protocol π^{SM}, observe that if any of the above conditions are true, it means there exists i, j such that the statement $\mathsf{st}^i_{2,j} = (\hat{\mathsf{c}}^i_{1,j}, \mathsf{c}^j_{1,i}, \mathsf{c}^i_{2,j}, \mathsf{tag}_{j,i}, \mathsf{msg}_{1,j}, \mathsf{msg}_{2,j}, \tau_1) \notin L$, where P_i is honest and P_j is malicious. However, the proof for the statement verified correctly which means that the adversary has produced a valid proof for a false statement. This violates the soundness property of the SPSSZK argument system which is a contradiction. □

□

Lemma 13. *Assuming the zero knowledge property of the* $\mathsf{SPSS.ZK}$ *argument system,* Hyb_2 *is computationally indistinguishable from* Hyb_3*.*

Proof. The only difference between the two hybrids is that in Hyb_2, for every session, $\mathsf{Sim}_{\mathsf{Hyb}}$ computes the proofs in Rounds 2 and 3 honestly, by running the algorithm ZK_2 of the $\mathsf{SPSS.ZK}$ argument system, whereas in Hyb_3, a simulated proof is used for every session. If the adversary $\mathcal{A}$ can distinguish between the two hybrids, we will use $\mathcal{A}$ to design an algorithm $\mathcal{A}_{\mathsf{ZK}}$ that breaks the zero knowledge property of the argument system.

Suppose the adversary can distinguish between the two hybrids with non-negligible probability p. Then, by a simple hybrid argument, there exists hybrids $\mathsf{Hyb}_{4,s}$ and $\mathsf{Hyb}_{4,s+1}$ that can be distinguished by $\mathcal{A}$ with some non-negligible probability $p' < p$ where the difference between these two hybrids is that only in session s, the proofs are computed differently.

by a simple hybrid argument, there exists some session s, an honest party P_i and a (malicious) party P_j such that the adversary can distinguish the proof sent by P_i to P_j in one of the rounds with non-negligible probability $p' < p$. Let's say it is the proof in round 2.

$\mathcal{A}_{\mathsf{ZK}}$ performs the role of $\mathsf{Sim}_{\mathsf{Hyb}}$ in its interaction with $\mathcal{A}$ and performs all the steps exactly as in Hyb_2 except the proof in Round 2 sent by P_i to P_j in session s. It interacts with a challenger $\mathcal{C}$ of the $\mathsf{SPSS.ZK}$ argument system and sends the first round message $\mathsf{ver}^i_{1,j}$ it received from the adversary. $\mathcal{A}_{\mathsf{ZK}}$ receives from $\mathcal{C}$ a proof that is either honestly computed or simulated. $\mathcal{A}_{\mathsf{ZK}}$ sets this received proof as its message $\mathsf{prove}^j_{i,2}$ in Round 2 of its interaction with $\mathcal{A}$ in session s. In the first case, this exactly corresponds to Hyb_2 while the latter exactly corresponds to Hyb_3. Therefore, if $\mathcal{A}$ can distinguish between the two hybrids, $\mathcal{A}_{\mathsf{ZK}}$ can use the same distinguishing guess to distinguish the proofs: i.e, decide whether the proofs received from $\mathcal{C}$ were honest or simulated. Now, notice that $\mathcal{A}_{\mathsf{ZK}}$ runs only in time $\mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}}$ (during the input extraction phase), while the $\mathsf{SPSS.ZK}$ system is secure against adversaries running in time T_{ZK}. Since $\mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}} < \mathsf{T}_{\mathsf{ZK}}$, this is a contradiction and hence proves the lemma.
In particular, this also means the following: $\Pr[\mathsf{Event\ E\ is\ true\ in\ Hyb}_3] = \mathsf{negl}(\lambda)$. □

Lemma 14. *Assuming the non-malleability property of the non-malleable commitment scheme* NMCom*,* Hyb_3 *is computationally indistinguishable from* Hyb_4*.*

Proof. We will prove this using a series of computationally indistinguishable intermediate hybrids as follows.

- $\mathsf{Hyb}_{3,1}$**:** This is same as Hyb_3 except that the simulator $\mathsf{Sim}_{\mathsf{Hyb}}$, in each session, does not run the input extraction phase apart from verifying the $\mathsf{SPSS.ZK}$ proofs. Also, $\mathsf{Sim}_{\mathsf{Hyb}}$ does not run the special abort phase. In particular, the $\mathsf{Ext.Com}$ algorithm is not run and there is no "Special Abort". In this hybrid, $\mathsf{Sim}_{\mathsf{Hyb}}$ runs in time $\mathsf{T}^{\mathsf{Sim}}_{\mathsf{ZK}}$ which is lesser than $\mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}}$.

- $\mathsf{Hyb}_{3,2}$**:** This hybrid is identical to the previous hybrid except that in each session, in Round 2, $\mathsf{Sim}_{\mathsf{Hyb}}$ now computes all the messages $(\hat{\mathsf{c}}^j_{1,i}, \mathsf{c}^j_{2,i})$ as non-malleable commitments of $0^{p(\lambda)}$ as done by Sim in the ideal world. In this hybrid too, $\mathsf{Sim}_{\mathsf{Hyb}}$ runs in time $\mathsf{T}^{\mathsf{Sim}}_{\mathsf{ZK}}$.

- $\mathsf{Hyb}_{3,3}$**:** This is same as Hyb_3 except that in each session, the simulator does run the input extraction phase and the special abort phase. It is easy to see that $\mathsf{Hyb}_{3,3}$ is the same as Hyb_4. In this hybrid, $\mathsf{Sim}_{\mathsf{Hyb}}$ runs in time $\mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}}$ which is greater than $\mathsf{T}^{\mathsf{Sim}}_{\mathsf{ZK}}$.

We now prove the indistinguishability of these intermediate hybrids and this completes the proof of the lemma.

Sub-Lemma 6. Hyb_3 *is statistically indistinguishable from* $\mathsf{Hyb}_{3,1}$.

Proof. The only difference between the two hybrids is that in Hyb_3, the simulator might output "Special Abort" in some session which doesn't happen in $\mathsf{Hyb}_{3,1}$. As shown in the proof of Lemma 13, the probability that Event E occurs in Hyb_3 is negligible. This means that the probability that the simulator outputs "Special Abort" in any session in Hyb_3 is negligible and this completes the proof. □

Sub-Lemma 7. *Assuming the non-malleability property of the non-malleable commitment scheme* NMCom*,* $\mathsf{Hyb}_{3,1}$ *is computationally indistinguishable from* $\mathsf{Hyb}_{3,2}$.

Proof. The only difference between the two hybrids is that in $\mathsf{Hyb}_{3,1}$, in each session, for every honest party P_i, $\mathsf{Sim}_{\mathsf{Hyb}}$ computes the commitment messages $(\hat{\mathsf{c}}^j_{1,i}, \mathsf{c}^j_{2,i})$ as a commitment of $(\mathsf{x}_i, \mathsf{r}_i)$, whereas in $\mathsf{Hyb}_{3,2}$, they are computed as a commitment of $(0^{p(\lambda)})$. If the adversary $\mathcal{A}$ can distinguish between the two hybrids, we will use $\mathcal{A}$ to design an algorithm $\mathcal{A}_{\mathsf{NMC}}$ that breaks the security of the non-malleable commitment scheme NMCom.

Suppose $\mathcal{A}$ can distinguish between these two hybrids with some non-negligible probability p. Then by a simple hybrid argument, there exists hybrids $\mathsf{Hyb}_{3,1,s}$ and $\mathsf{Hyb}_{3,1,s+1}$ that can be distinguished by $\mathcal{A}$ with some non-negligible probability $p' < p$ where the difference between these two hybrids is that only in session s, the commitments are computed differently.

$\mathcal{A}_{\mathsf{NMC}}$ acts as the man-in-the-middle adversary interacting with a challenger $\mathcal{C}$. $\mathcal{A}_{\mathsf{NMC}}$ also plays the role of $\mathsf{Sim}_{\mathsf{Hyb}}$ in its interaction with the adversary $\mathcal{A}$. It generates all the messages except the messages $\mathsf{c}^j_{1,i}$ and $(\hat{\mathsf{c}}^j_{1,i}, \mathsf{c}^j_{2,i})$ in session s exactly as done by $\mathsf{Sim}_{\mathsf{Hyb}}$ in $\mathsf{Hyb}_{3,1,s}$. Corresponding to each message $\mathsf{c}^j_{1,i}$ that $\mathcal{A}_{\mathsf{NMC}}$ has to send in session s, it receives one first round message from $\mathcal{C}$ (on the right side) corresponding to the scheme NMCom. Similarly, it receives first round messages $\hat{\mathsf{c}}^j_{1,i}$ from $\mathcal{C}$ (on the left side). $\mathcal{A}_{\mathsf{NMC}}$ forwards these messages to the adversary $\mathcal{A}$ as its first round messages $(\hat{\mathsf{c}}^j_{1,i}, \mathsf{c}^j_{1,i})$ in session s. Similarly, for each pair of messages $(\hat{\mathsf{c}}^i_{1,j}, \mathsf{c}^i_{1,j})$ it receives from $\mathcal{A}$ as part of the first round messages of the scheme NMCom in session s, $\mathcal{A}_{\mathsf{NMC}}$ forwards the messages to $\mathcal{C}$ as its first round messages for the commitment (to the left and right side respectively). Then, for each $\mathsf{c}^j_{2,i}$ that $\mathcal{A}_{\mathsf{NMC}}$ is supposed to send to $\mathcal{A}$ in session s, it receives a second round commitment message from the challenger $\mathcal{C}$. In one case, all of these are commitments to the respective $(\mathsf{x}_i, \mathsf{r}_i)$ values while in the second case, they are all commitments to $(0^{p(\lambda)})$. $\mathcal{A}_{\mathsf{NMC}}$ forwards these messages as its commitment messages $\mathsf{c}^j_{2,i}$ to the adversary $\mathcal{A}$ in session s. Once again, it forwards each message $\mathsf{c}^i_{2,j}$ it receives from $\mathcal{A}$, as its second round commitment message in its interaction with the challenger $\mathcal{C}$. That is, these are the commitments on the right side generated by the man-in-the-middle.

Now, we can clearly see that in the first case, when $\mathcal{C}$ generates commitments to $(\mathsf{x}_i, \mathsf{r}_i)$ in session s, $\mathcal{A}$'s view corresponds to $\mathsf{Hyb}_{3,1,s}$ while in the latter case, it exactly corresponds to $\mathsf{Hyb}_{3,1,s+1}$. However, from the security of the non-malleable commitment scheme, the joint distribution of the value committed to by the adversary $\mathcal{A}_{\mathsf{NMC}}$ (which is the same as $\mathcal{A}$'s commitments) and its view must be indistinguishable in both cases. Therefore, if $\mathcal{A}$ can distinguish between the two hybrids, then $\mathcal{A}_{\mathsf{NMC}}$ can break the non-malleability property of the commitment scheme NMCom. However, $\mathcal{A}_{\mathsf{NMC}}$ only runs in time $\mathsf{T}^{\mathsf{Sim}}_{\mathsf{ZK}} < \mathsf{T}^{\mathsf{Sec}}_{\mathsf{Com}}$ and hence this is a contradiction, thus proving the sub-lemma.

Also, notice that since the joint distribution of the adversary $\mathcal{A}$'s committed values and his view is indistinguishable in both hybrids, this implies that Event E still occurs only with negligible probability in $\mathsf{Hyb}_{3,2}$ as well. □

Sub-Lemma 8. $\mathsf{Hyb}_{3,2}$ *is statistically indistinguishable from* $\mathsf{Hyb}_{3,3}$.

Proof. The only difference between the two hybrids is that in $\mathsf{Hyb}_{3,3}$, the simulator might output "Special Abort" in some session which doesn't happen in $\mathsf{Hyb}_{3,2}$. As shown in the proof of Sub-Lemma 7, the probability that Event E occurs in $\mathsf{Hyb}_{3,2}$ is negligible. This means that the probability that the simulator outputs "Special Abort" in $\mathsf{Hyb}_{3,3}$ is negligible and this completes the proof. □

□

Lemma 15. *Assuming the security of the protocol* π^{SM}, Hyb_4 *is computationally indistinguishable from* Hyb_5.

Proof. The only difference between the two hybrids is that in Hyb_4, in every session, $\mathsf{Sim}_{\mathsf{Hyb}}$ computes the messages of protocol π^{SM} correctly using the honest parties' inputs, whereas in Hyb_5, they are computed by running the simulator $\mathcal{S}$ for protocol π^{SM}. If the adversary $\mathcal{A}$ can distinguish between the two hybrids, we will use $\mathcal{A}$ to design an algorithm $\mathcal{A}_{\mathsf{SM}}$ that can break the security of protocol π^{SM}.

Once again, suppose $\mathcal{A}$ can distinguish between these two hybrids with some non-negligible probability p. Then by a simple hybrid argument, there exists hybrids $\mathsf{Hyb}_{4,s}$ and $\mathsf{Hyb}_{4,s+1}$ that can be distinguished by $\mathcal{A}$ with some non-negligible probability $p' < p$ where the difference between these two hybrids is that only in session s, the protocol messages are computed differently.

$\mathcal{A}_{\mathsf{SM}}$ interacts with a challenger $\mathcal{C}$ to break the security of protocol π^{SM}. Also, $\mathcal{A}_{\mathsf{SM}}$ performs the role of $\mathsf{Sim}_{\mathsf{Hyb}}$ in its interaction with the adversary $\mathcal{A}$ exactly as in $\mathsf{Hyb}_{3,s}$ except for session s. Whatever parties $\mathcal{A}$ wishes to corrupt in session s, $\mathcal{A}_{\mathsf{SM}}$ corrupts the same parties in its interaction with π^{SM}. Similarly, whatever messages $\mathcal{A}$ sends to $\mathcal{A}_{\mathsf{SM}}$ as part of the protocol π in session s that correspond to π^{SM} messages, $\mathcal{A}_{\mathsf{SM}}$ sends the same messages to the challenger $\mathcal{C}$. Now, whatever messages $\mathcal{C}$ sends, $\mathcal{A}_{\mathsf{SM}}$ forwards the same to the adversary $\mathcal{A}$ as its messages for the π^{SM} protocol in session s.

Observe that $\mathcal{A}_{\mathsf{SM}}$ runs in time $\mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}}$. If $\mathcal{C}$ sends messages that are computed correctly, this exactly corresponds to $\mathsf{Hyb}_{4,s}$ in $\mathcal{A}_{\mathsf{SM}}$'s interaction with $\mathcal{A}$. On the other hand, if $\mathcal{C}$ sends simulated messages, this exactly corresponds to $\mathsf{Hyb}_{4,s+1}$. Therefore, if $\mathcal{A}$ can distinguish between these two hybrids, $\mathcal{A}_{\mathsf{SM}}$ can use the same distinguishing guess to break the security of protocol π^{SM}. However, π^{SM} is secure against all adversaries running in time T_{SM}, where $\mathsf{T}_{\mathsf{SM}} > \mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}}$ and hence this is a contradiction. This completes the proof of the lemma. □

7 Two Round Concurrently Secure MPC for Input-less Functionalities

Let f be any input-less functionality randomized functionalities. Consider n parties $\mathsf{P}_1, \ldots, \mathsf{P}_n$ who wish to compute f by running a concurrently secure multiparty computation(MPC) protocol. Let π^{SM} be any 2 round protocol that runs without any setup for the above task and is secure against semi-malicious adversaries that can corrupt upto $(n-1)$ parties. In this section, we show how to generically transform π^{SM} into a 2 round concurrently secure protocol π_1^{Conc} without setup with super-polynomial simulation and secure against malicious adversaries that can corrupt upto $(n-1)$ parties. Formally, we prove the following theorem:

Theorem 7. *Assuming sub-exponentially secure:*

- *A, where $A \in \{DDH,\ Quadratic\ Residuosity,\ N^{th}\ Residuosity\}$ AND*
- *2 round MPC protocol for any functionality f that is stand-alone secure against semi-malicious adversaries,*

the protocol presented in Figure 8 is a 2 round concurrently secure MPC protocol without any setup with super-polynomial simulation for any input-less randomized functionality f, secure against malicious adversaries.

We can instantiate the underlying MPC protocol with the 2 round construction of [DHRW16] to get the following corollary:

Corollary 8. *Assuming sub-exponentially secure:*

- *A, where $A \in \{DDH,\ Quadratic\ Residuosity,\ N^{th}\ Residuosity\}$ AND*
- *Indistinguishability Obfuscation,*

the protocol presented in Figure 8 is a 2 round concurrently secure MPC protocol without any setup with super-polynomial simulation for any input-less randomized functionality f.

We essentially prove that the same protocol from Section 5 is also concurrently secure. The proof is fairly simple and not too different from the proof of stand-alone security, because the simulation strategy as well as all reductions are straight-line. The only use of rewinding occurs (implicitly) within the proof of non-malleability, which we carefully combine with identities to ensure that the protocol remains concurrently secure. For the sake of completeness, we write out the protocol and the proof in their entirety. As in Section 5, we first list some notation and the primitives used before describing the construction.

Notation:

- λ denotes the security parameter.
- $\mathsf{SPSS.ZK} = (\mathsf{ZK}_1, \mathsf{ZK}_2, \mathsf{ZK}_3)$ is a two message zero knowledge argument with superior super polynomial simulation (SPSS-ZK). The zero knowledge property holds against all adversaries running in time T_{ZK}. Let $\mathsf{Sim}^{\mathsf{ZK}}$ denote the simulator that produces simulated ZK proofs and let $\mathsf{T}^{\mathsf{Sim}}_{\mathsf{ZK}}$ denote its running time. [KS17] give a construction of an $\mathsf{SPSS.ZK}$ scheme satisfying these properties that can be based on one of the following sub-exponential assumptions: 1) DDH; 2) Quadratic Residuosity; 3) N^{th} Residuosity.
- $\mathsf{NMCom} = (\mathsf{NMCom}^R_1, \mathsf{NMCom}^{\mathsf{S}}_1, \mathsf{NMCom}^S_2)$ is a two message concurrent non-malleable commitment scheme with respect to commitment in the simultaneous message model. Here, $\mathsf{NMCom}^R_1, \mathsf{NMCom}^{\mathsf{S}}_1$ denote the first message of the receiver and sender respectively while NMCom^S_2 denotes the second message of the sender. It is secure against all adversaries running in time $\mathsf{T}^{\mathsf{Sec}}_{\mathsf{Com}}$, but can be broken by adversaries running in time $\mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}}$. Let $\mathsf{Ext.Com}$ denote a brute force algorithm running in time $\mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}}$ that can break the commitment scheme just using the first round messages. [KS17] give a construction of an NMCom scheme satisfying these properties that can be based on one of the following sub-exponential assumptions: 1) DDH; 2) Quadratic Residuosity; 3) N^{th} Residuosity.

- π^{SM} is a sub-exponentially secure 2 round MPC protocol that is secure against semi-malicious adversaries. This protocol is secure against all adversaries running in time T_{SM}. Let $(\mathsf{MSG}_1, \mathsf{MSG}_2)$ denote the algorithms used by any party to compute the messages in each of the two rounds and OUT denotes the algorithm to compute the final output. Further, let's assume that this protocol π^{SM} runs over a broadcast channel. Let $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ denote the simulator for the protocol π^{SM} - that is, $\mathcal{S}_i$ is the simulator's algorithm to compute the i^{th} round messages. Also, we make the following assumptions about the protocol structure that is satisfied by the instantiations:

  1. Since the protocol is for input-less functionalities, we assume that $\mathcal{S}_1$ is identical to the algorithm MSG_1 used by honest parties to generate their first message.
  2. The algorithm MSG_2 doesn't use any new randomness that was not already used in the algorithm MSG_1. This is similar to the assumption used in Section 4.

In order to realize our protocol, we require that $\mathsf{poly}(\lambda) < \mathsf{T}_{\mathsf{ZK}}^{\mathsf{Sim}} < \mathsf{T}_{\mathsf{Com}}^{\mathsf{Sec}} < \mathsf{T}_{\mathsf{Com}}^{\mathsf{Brk}} < \mathsf{T}_{\mathsf{ZK}}, \mathsf{T}_{\mathsf{SM}}$.

The construction of the protocol is described in Figure 8. We assume broadcast channels. In our construction, we use proofs for a some NP languages that we elaborate on below.

NP language L is characterized by the following relation R.
Statement : $\mathsf{st} = (\mathsf{c}_1, \hat{\mathsf{c}}_1, \mathsf{c}_2, \mathsf{tag}, \mathsf{msg}_1, \mathsf{msg}_2, \tau)$
Witness : $\mathsf{w} = (\mathsf{r}, \mathsf{r_c})$
$R(\mathsf{st}, \mathsf{w}) = 1$ if and only if :

- $\hat{\mathsf{c}}_1 = \mathsf{NMCom}_1^{\mathsf{S}}(\mathsf{r}, \mathsf{tag}; \mathsf{r_c})$ AND
- $\mathsf{c}_2 = \mathsf{NMCom}_2^{S}(\mathsf{r}, \mathsf{tag}, \mathsf{c}_1; \mathsf{r_c})$ AND
- $\mathsf{msg}_1 = \mathsf{MSG}_1(\perp; \mathsf{r})$ AND
- $\mathsf{msg}_2 = \mathsf{MSG}_2(\perp, \tau; \mathsf{r})$

That is, the messages $(\mathsf{c}_1, \hat{\mathsf{c}}_1, \mathsf{c}_2)$ form a non-malleable commitment of $(\mathsf{inp}, \mathsf{r})$ such that msg_2 is the second round message using input inp, randomness r by running the protocol π^{SM}, where the protocol transcript so far is τ.

In the protocol, let's assume that every party has an associated identity id. For any session sid, each parties generates its non-malleable commitment using the tag $(\mathsf{id}||\mathsf{sid})$. We make the tags explicit in the protocol here to illustrate concurrency.

The correctness of the protocol follows from the correctness of the protocol π^{SM}, the non-malleable commitment scheme NMCom and the zero knowledge proof system $\mathsf{SPSS.ZK}$.

7.1 Security Proof

In this section, we formally prove Theorem 7.
Let the total number of sessions be m where in each session, a set of n parties take part. Note that m is polynomial in the security parameter λ. Consider an adversary $\mathcal{A}$ who corrupts several parties in each session and can schedule the messages across various sessions arbitrarily. For any session s_k, let's say the adversary $\mathcal{A}$ corrupts t_k parties in this session where $t_k < n$. For each party P_i, let's say that the size of randomness used in the protocol π^{SM} is $p(\lambda)$ for some polynomial p. That is, $|\mathsf{r}_i| = p(\lambda)$. The strategy of the simulator Sim against a malicious adversary $\mathcal{A}$ is described in *Figure 9*.

Inputs: Each party P_i uses randomness r_i to compute the message in each round of the protocol π^{SM}. To make the exposition easier, we think of each party's input as being $\bot$. We now describe the messages sent by party P_i. We will use superscripts to denote the intended recipient of the message if it isn't meant to be used by all parties.

1. **Round 1:**
   P_i does the following:
   - Compute $\mathsf{msg}_{1,i} \leftarrow \mathsf{MSG}_1(\bot; \mathsf{r}_i)$.
   - For each $j \in [n]$ with $j \neq i$, compute:
     - $\hat{\mathsf{c}}^j_{1,i} \leftarrow \mathsf{NMCom}^{\mathsf{S}}_1(\mathsf{r}_i, \mathsf{tag}_{i,j}; \mathsf{r}^j_{\mathsf{c},i})$ using a random string $\mathsf{r}^j_{\mathsf{c},i}$. $\mathsf{c}^j_{1,i} \leftarrow \mathsf{NMCom}^R_1(1^\lambda)$.
     - $(\mathsf{ver}^j_{1,i}, \mathsf{zkst}^j_{1,i}) \leftarrow \mathsf{ZK}_1(1^\lambda)$.
   - Send $(\mathsf{msg}_{1,i}, \hat{\mathsf{c}}^j_{1,i}, \mathsf{c}^j_{1,i}, \mathsf{ver}^j_{1,i})$ for all j.
2. **Round 2:**
   Let τ_1 denote the protocol transcript after round 1. P_i does the following:
   - Compute $\mathsf{msg}_{2,i} \leftarrow \mathsf{MSG}_2(\bot, \tau_1; \mathsf{r}_i)$.
   - For each $j \in [n]$ with $j \neq i$, compute:
     - $\mathsf{c}^j_{2,i} \leftarrow \mathsf{NMCom}^S_2(\mathsf{r}_i, \mathsf{tag}_{i,j}, \mathsf{c}^i_{1,j}; \mathsf{r}^j_{\mathsf{c},i})$ using the same random string $\mathsf{r}^j_{\mathsf{c},i}$.
     - $\mathsf{prove}^j_{2,i} \leftarrow \mathsf{ZK}_2(\mathsf{ver}^i_{1,j}, \mathsf{st}^j_{2,i}, \mathsf{w}^j_{2,i})$ for the statement $\mathsf{st}^j_{2,i} = (\mathsf{c}^i_{1,j}, \hat{\mathsf{c}}^j_{1,i}, \mathsf{c}^j_{2,i}, \mathsf{tag}_{i,j}, \mathsf{msg}_{1,i}, \mathsf{msg}_{2,i}, \tau_1) \in L$ using witness $\mathsf{w}^j_{2,i} = (\mathsf{r}_i, \mathsf{r}^j_{\mathsf{c},i})$.
   - Send $(\mathsf{msg}_{2,i}, \mathsf{c}^j_{2,i}, \mathsf{prove}^j_{2,i})$ for all j.
3. **Output Computation:**
   Let τ_2 denote the protocol transcript after round 2. P_i does the following:
   - For each $j \in [n]$ with $j \neq i$, do:
     - Abort if $\mathsf{ZK}_3(\mathsf{zkst}^j_{1,i}, \mathsf{st}^i_{2,j}) \neq 1$ where $\mathsf{st}^i_{2,j} = (\mathsf{c}^j_{1,i}, \hat{\mathsf{c}}^i_{1,j}, \mathsf{c}^i_{2,j}, \mathsf{tag}_{i,j}, \mathsf{msg}_{1,j}, \mathsf{msg}_{2,j}, \tau_1)$. In particular, send a global abort signal to all parties so that everyone aborts.
   - Compute output $\mathsf{y}_i \leftarrow \mathsf{OUT}(\bot, \tau_2; \mathsf{r}_i)$.

Figure 8: 2 round concurrently secure MPC Protocol π^{Conc}_1 for input-less randomized functionality f.

Here, notice that since there is no input, the simulator gets the output from the ideal functionality - y right at the beginning. It still has to instruct the functionality to deliver output to the honest party.

We now show that the simulation strategy described in Figure 9 is successful against all malicious PPT adversaries. That is, the view of the adversary along with the output of the honest parties is computationally indistinguishable in the real and ideal worlds. We will show this via a series of computationally indistinguishable hybrids where the first hybrid Hyb_1 corresponds to the real world and the last hybrid Hyb_6 corresponds to the ideal world.

1. Hyb_1**:** In this hybrid, consider a simulator $\mathsf{Sim}_{\mathsf{Hyb}}$ that plays the role of the honest parties. $\mathsf{Sim}_{\mathsf{Hyb}}$ runs in polynomial time.

2. Hyb_2**:** This hybrid is identical to the previous hybrid except that in Round 2, $\mathsf{Sim}_{\mathsf{Hyb}}$ now computes simulated SPSSZK proofs as done in Round 2 in Figure 9. Here, $\mathsf{Sim}_{\mathsf{Hyb}}$ runs in time $\mathsf{T}^{\mathsf{Sim}}_{\mathsf{ZK}}$.

Corresponding to every session, the simulator stores a database of the private coins and state used in that session. After each round in the protocol for a particular session, the simulator updates the database. For any new session s_k that the adversary initiates, the simulator does the following:

1. **Round 1:** For each honest party P_i, Sim does the following:
    - Compute $\mathsf{msg}_{1,i} \leftarrow \mathsf{MSG}_1(\bot; \mathsf{r}_i)$ using some random string r_i. Recall that this is identical to running the simulator $\mathcal{S}_1(1^\lambda, i)$. For each $j \in [n]$ with $j \neq i$, compute $\hat{\mathsf{c}}^j_{1,i} \leftarrow \mathsf{NMCom}^{\mathsf{S}}_1(0^{p(\lambda)}, \mathsf{tag}_{i,j})$, $\mathsf{c}^j_{1,i} \leftarrow \mathsf{NMCom}^R_1(1^\lambda)$ and $(\mathsf{ver}^j_{1,i}, \mathsf{zkst}^j_{1,i}) \leftarrow \mathsf{ZK}_1(1^\lambda)$.
    - Send $(\mathsf{msg}_{1,i}, \hat{\mathsf{c}}^j_{1,i}, \mathsf{c}^j_{1,i}, \mathsf{ver}^j_{1,i})$ for all $j \in [n]$.
2. **Randomness Extraction:** Sim does the following:
    - For each honest party P_i and for each $j \in [n]$ with $j \neq i$, do:
        - Compute $(\mathsf{r}^i_j) = \mathsf{Ext.Com}(\mathsf{c}^j_{1,i}, \hat{\mathsf{c}}^i_{1,j})$. That is, this is the randomness of party P_j seen by party P_i. This step takes time $\mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}}$.
    - Initialize a variable $\mathsf{correct} = 1$. Then, for each malicious party P_j, do:
        - Set $\mathsf{correct} = 0$ if the set of values $\{\mathsf{r}^i_j\}$, for all i corresponding to honest parties P_i is not equal. Set $\mathsf{r}_j = \mathsf{r}^1_j$ and let R denote the set of all $\{\mathsf{r}_j\}$.
        - Set $\mathsf{correct} = 0$ if $\mathsf{msg}_{1,j} \neq \mathsf{MSG}_1(\bot, \mathsf{r}_j)$.

Whenever the adversary schedules round 2 for a session s_k, the simulator, using the appropriate values in the stored database, does the following:

2. **Round 2:** Let τ_1 denote the protocol transcript after round 1. Sim does the following:
    - For each honest party P_i:
        - If $\mathsf{correct} = 1$, compute $\mathsf{msg}_{2,i} \leftarrow \mathcal{S}_2(\tau_1, \mathsf{R}, i)$.
        - Else, compute $\mathsf{msg}_{2,i} \leftarrow \mathsf{MSG}_2(\bot, \tau_1; \mathsf{r}_i)$ where r_i was used in round 1.
    - For each honest party P_i and for each $j \in [n]$ with $j \neq i$, compute:
        - $\mathsf{c}^j_{2,i} \leftarrow \mathsf{NMCom}^S_2(0^{p(\lambda)}, \mathsf{tag}_{i,j}, \mathsf{c}^i_{1,j}; \mathsf{r}^j_{\mathsf{c},i})$ using a random string $\mathsf{r}^j_{\mathsf{c},i}$.
        - $\mathsf{prove}^j_{2,i} \leftarrow \mathsf{Sim}^{\mathsf{ZK}}(\mathsf{ver}^i_{1,j}, \mathsf{st}^j_{2,i})$ for $\mathsf{st}^j_{2,i} = (\mathsf{c}^i_{1,j}, \hat{\mathsf{c}}^j_{1,i}, \mathsf{c}^j_{2,i}, \mathsf{tag}_{i,j}, \mathsf{msg}_{1,i}, \mathsf{msg}_{2,i}, \tau_1) \in L$. Observe that this takes time $\mathsf{T}^{\mathsf{Sim}}_{\mathsf{ZK}}$.
    - Send $(\mathsf{msg}_{2,i}, \mathsf{c}^j_{2,i}, \mathsf{prove}^j_{2,i})$ for all $j \in [n]$.
3. **Special Abort Phase:** For each malicious party P_j:
    - Output "Special Abort" if $\mathsf{correct} = 0$.
    - Also, output "Special Abort" if $\mathsf{msg}_{2,j} \neq \mathsf{MSG}_2(\bot, \mathsf{r}_j, \tau_1)$.
4. **Output Computation:** Sim does the following:
    - For each honest party P_i and for each $j \in [n]$ with $j \neq i$, abort if $\mathsf{ZK}_3(\mathsf{zkst}^j_{1,i}, \mathsf{st}^i_{2,j}) \neq 1$ where $\mathsf{st}^i_{2,j} = (\mathsf{c}^j_{1,i}, \hat{\mathsf{c}}^i_{1,j}, \mathsf{c}^i_{2,j}, \mathsf{tag}_{i,j}, \mathsf{msg}_{1,j}, \mathsf{msg}_{2,j}, \tau_1)$.
    - Else, instruct the ideal functionality to deliver output to the honest parties.

Figure 9: Simulation strategy in the 2 round concurrently secure protocol

3. Hyb_3**:** This hybrid is identical to the previous hybrid except that $\mathsf{Sim}_{\mathsf{Hyb}}$ now computes all the $(\hat{\mathsf{c}}^j_{1,i}, \mathsf{c}^j_{2,i})$ as non-malleable commitments of $0^{p(\lambda)}$ as done in Round 2 in Figure 9. Once again, $\mathsf{Sim}_{\mathsf{Hyb}}$ runs in time $\mathsf{T}^{\mathsf{Sim}}_{\mathsf{ZK}}$.

4. Hyb_4**:** In this hybrid, the simulator $\mathsf{Sim}_{\mathsf{Hyb}}$ also runs the "Randomness Extraction" phase and the "Special Abort" phase in steps 2 and 4 in Figure 9. Now, $\mathsf{Sim}_{\mathsf{Hyb}}$ runs in time $\mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}}$.

5. Hyb_5**:** In this hybrid, if the value of the variable $\mathsf{correct} = 1$, $\mathsf{Sim}_{\mathsf{Hyb}}$ now computes the second round message of the protocol π^{SM} using the simulator algorithms $\mathcal{S}_2$ as done by Sim in the ideal world. $\mathsf{Sim}_{\mathsf{Hyb}}$ also instructs the ideal functionality to deliver outputs to the honest parties as done by Sim. This hybrid is now same as the ideal world. Once again, $\mathsf{Sim}_{\mathsf{Hyb}}$ runs in time $\mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}}$.

We now show that every pair of successive hybrids is computationally indistinguishable. Additionally, we also prove some claims about certain hybrids that aids in the proof.
Before that, we will define the following event E that is useful in the proofs.

Event E**:** In any given session s_k, the event occurs if, for any malicious party P_j :

- All the $\mathsf{SPSS.ZK}$ proofs sent by P_j in round 2 verify correctly.
  (AND)

- Either of the following occur:
  - The set of values $\{\mathsf{r}^i_j\}$ that are committed to using the non-malleable commitment is not same for every i where P_i is honest. (OR)
  - $\mathsf{msg}_{1,j} \neq \mathsf{MSG}_1(\perp, \mathsf{r}_j)$ (OR)
  - $\mathsf{msg}_{2,j} \neq \mathsf{MSG}_2(\perp, \mathsf{r}_j, \tau_1)$ where τ_1 is the protocol transcript after round 1.

Lemma 16. *Assuming soundness of the* $\mathsf{SPSS.ZK}$ *argument system, binding of the non-malleable commitment scheme and correctness of the protocol* π^{SM}, $\mathsf{Pr}[\mathsf{Event\ E\ is\ true\ in\ Hyb}_1] = \mathsf{negl}(\lambda)$.

Proof. Suppose the event E does occur in Hyb_1 with non-negligible probability p. Then there exists some session s such that the event E occurs in that session with non-negligible probability $p' < p$. Let's focus on that session s.

From the binding property of the commitment scheme and the correctness of the protocol π^{SM}, observe that if any of the conditions that cause event E to occur are true, it means there exists i, j such that the statement $\mathsf{st}^i_{2,j} = (\mathsf{c}^j_{1,i}, \mathsf{c}^i_{2,j}, \mathsf{msg}_{1,j}, \mathsf{msg}_{2,j}, \tau_1) \notin L$, where P_i is honest and P_j is malicious. However, the proof for the statement verified correctly which means that the adversary has produced a valid proof for a false statement. This violates the soundness property of the SPSSZK argument system which is a contradiction. □

Lemma 17. *Assuming the zero knowledge property of the* $\mathsf{SPSS.ZK}$ *argument system,* Hyb_1 *is computationally indistinguishable from* Hyb_2.

Proof. The only difference between the two hybrids is that in Hyb_1, $\mathsf{Sim}_{\mathsf{Hyb}}$ computes the proofs in Rounds 2 honestly, by running the algorithm ZK_2 of the $\mathsf{SPSS.ZK}$ argument system, whereas in Hyb_3, a simulated proof is used. If the adversary $\mathcal{A}$ can distinguish between the two hybrids, we will use $\mathcal{A}$ to design an algorithm $\mathcal{A}_{\mathsf{ZK}}$ that breaks the zero knowledge property of the argument system.

Suppose the adversary can distinguish between the two hybrids with non-negligible probability p. Then, by a simple hybrid argument, there exists hybrids $\mathsf{Hyb}_{1,k}$ and $\mathsf{Hyb}_{1,k+1}$ that the adversary can distinguish with non-negligible probability $p' < p$ such that: the only difference between the two hybrids is the proof sent by an honest party P_i to a (malicious) party P_j in round 2 in a particular session s.

$\mathcal{A}_{\mathsf{ZK}}$ performs the role of $\mathsf{Sim}_{\mathsf{Hyb}}$ in its interaction with $\mathcal{A}$ and performs all the steps exactly as in $\mathsf{Hyb}_{1,k}$ except the proof in Round 2 sent by P_i to P_j in session s. It interacts with a challenger $\mathcal{C}$ of the SPSS.ZK argument system and sends the first round message $\mathsf{ver}^i_{1,j}$ it received from the adversary in session s. $\mathcal{A}_{\mathsf{ZK}}$ receives from $\mathcal{C}$ a proof that is either honestly computed or simulated. $\mathcal{A}_{\mathsf{ZK}}$ sets this received proof as its message $\mathsf{prove}^j_{i,2}$ in Round 2 of its interaction with $\mathcal{A}$ in session s. In the first case, this exactly corresponds to $\mathsf{Hyb}_{1,k}$ while the latter exactly corresponds to $\mathsf{Hyb}_{1,k+1}$. Therefore, if $\mathcal{A}$ can distinguish between the two hybrids, $\mathcal{A}_{\mathsf{ZK}}$ can use the same distinguishing guess to distinguish the proofs: i.e, decide whether the proofs received from $\mathcal{C}$ were honest or simulated. Now, notice that $\mathcal{A}_{\mathsf{ZK}}$ runs only in polynomial, while the SPSS.ZK system is secure against adversaries running in time T_{ZK} which is much larger. Thus, this is a contradiction and proves the lemma. □

Lemma 18. *Assuming the zero knowledge property of the* SPSS.ZK *argument system,*

$$|\Pr[\text{Event E is true in } \mathsf{Hyb}_1] - \Pr[\text{Event E is true in } \mathsf{Hyb}_2]| = \mathsf{negl}(\lambda).$$

Proof. Suppose the claim is not true. That is, there exists some adversary $\mathcal{A}$ for which the difference in the probability of the event E occurring between the two hybrids is some non-negligible probability p. Then we will design an algorithm $\mathcal{A}_{\mathsf{ZK}}$ that breaks the zero knowledge property of the argument system.

Let's say the number of proofs given by an honest party to a malicious party is q across all sessions. Then, consider a set of intermediate hybrids $\mathsf{Hyb}_{1,1}, \ldots, \mathsf{Hyb}_{1,q}$ such that $\mathsf{Hyb}_{1,q} = \mathsf{Hyb}_2$ where the difference between $\mathsf{Hyb}_{1,k-1}$ and $\mathsf{Hyb}_{1,k}$ is that we switch the k^{th} proof alone from honest to simulated. By a simple hybrid argument, there exists a k such that the difference in the probability of the event E occurring between the two hybrids $\mathsf{Hyb}_{1,k-1}$ and $\mathsf{Hyb}_{1,k}$ is some non-negligible probability $p' < p$. Let the proof that is different between the two hybrids be that sent by a honest party P_i to a malicious party P_j in a session s.

$\mathcal{A}_{\mathsf{ZK}}$ performs the role of $\mathsf{Sim}_{\mathsf{Hyb}}$ in its interaction with $\mathcal{A}$ and performs all the steps exactly as in $\mathsf{Hyb}_{1,k-1}$ except the proof in Round 2 sent by P_i to P_j in session s. It interacts with a challenger $\mathcal{C}$ of the SPSS.ZK argument system and sends the first round message $\mathsf{ver}^i_{1,j}$ it received from the adversary. $\mathcal{A}_{\mathsf{ZK}}$ receives from $\mathcal{C}$ a proof that is either honestly computed or simulated. $\mathcal{A}_{\mathsf{ZK}}$ sets this received proof as its message $\mathsf{prove}^j_{i,2}$ in Round 2 of its interaction with $\mathcal{A}$ in session s. In the first case, this exactly corresponds to $\mathsf{Hyb}_{1,k-1}$ while the latter exactly corresponds to $\mathsf{Hyb}_{1,k}$.

After completing the experiment, $\mathcal{A}_{\mathsf{ZK}}$ runs the brute force extractor UC-Com on each of the adversary's messages to break the commitment. It then checks the adversary's protocol messages to see if the event E did occur. If the event E did occur, it outputs to the challenger $\mathcal{C}$ that the proof was simulated and if the event E did not occur, it says real. $\mathcal{A}_{\mathsf{ZK}}$ takes time $\mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}}$ to run.

Therefore, if $\mathcal{A}$'s probability of making E occur is non-negligibly different in both the hybrids, $\mathcal{A}_{\mathsf{ZK}}$ can distinguish honest proofs from simulated ones with the same probability. Now, notice that $\mathcal{A}_{\mathsf{ZK}}$ runs only in time $\mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}}$, while the SPSS.ZK system is secure against adversaries running in time T_{ZK} which is much larger. Thus, this is a contradiction and proves the lemma. □

Lemma 19. $\Pr[\text{Event E is true in } \mathsf{Hyb}_2] = \mathsf{negl}(\lambda)$.

Proof. This follows by combining Lemma 5 and Lemma 8. □

Lemma 20. *Assuming the non-malleability property of the non-malleable commitment scheme* NMCom*,* Hyb_2 *is computationally indistinguishable from* Hyb_3*.*

Proof. The only difference between the two hybrids is that in Hyb_2, for every honest party P_i, $\mathsf{Sim}_{\mathsf{Hyb}}$ computes the commitment messages $(\hat{\mathsf{c}}^j_{1,i}, \mathsf{c}^j_{2,i})$ as a commitment of (r_i), whereas in Hyb_3, they are computed as a commitment of $(0^{p(\lambda)})$. If the adversary $\mathcal{A}$ can distinguish between the two hybrids, we will use $\mathcal{A}$ to design an algorithm $\mathcal{A}_{\mathsf{NMC}}$ that breaks the security of the non-malleable commitment scheme NMCom.

Suppose $\mathcal{A}$ can distinguish between these two hybrids with some non-negligible probability p. Then by a simple hybrid argument, there exists hybrids $\mathsf{Hyb}_{2,s}$ and $\mathsf{Hyb}_{2,s+1}$ that can be distinguished by $\mathcal{A}$ with some non-negligible probability $p' < p$ where the difference between these two hybrids is that only in session s, the commitments are computed differently.

$\mathcal{A}_{\mathsf{NMC}}$ acts as the man-in-the-middle adversary interacting with a challenger $\mathcal{C}$. $\mathcal{A}_{\mathsf{NMC}}$ also plays the role of $\mathsf{Sim}_{\mathsf{Hyb}}$ in its interaction with the adversary $\mathcal{A}$. It generates all the messages except the messages $\mathsf{c}^j_{1,i}$ and $(\hat{\mathsf{c}}^j_{1,i}, \mathsf{c}^j_{2,i})$ in session s exactly as done by $\mathsf{Sim}_{\mathsf{Hyb}}$ in $\mathsf{Hyb}_{2,s}$. Corresponding to each message $\mathsf{c}^j_{1,i}$ that $\mathcal{A}_{\mathsf{NMC}}$ has to send in session s, it receives one first round message from $\mathcal{C}$ (on the right side) corresponding to the scheme NMCom. Similarly, it receives first round messages $\hat{\mathsf{c}}^j_{1,i}$ from $\mathcal{C}$ (on the left side). $\mathcal{A}_{\mathsf{NMC}}$ forwards these messages to the adversary $\mathcal{A}$ as its first round messages $(\hat{\mathsf{c}}^j_{1,i}, \mathsf{c}^j_{1,i})$ in session s. Similarly, for each pair of messages $(\hat{\mathsf{c}}^i_{1,j}, \mathsf{c}^i_{1,j})$ it receives from $\mathcal{A}$ as part of the first round messages of the scheme NMCom in session s, $\mathcal{A}_{\mathsf{NMC}}$ forwards the messages to $\mathcal{C}$ as its first round messages for the commitment (to the left and right side respectively). Then, for each $\mathsf{c}^j_{2,i}$ that $\mathcal{A}_{\mathsf{NMC}}$ is supposed to send to $\mathcal{A}$ in session s, it receives a second round commitment message from the challenger $\mathcal{C}$. In one case, all of these are commitments to the respective (r_i) values while in the second case, they are all commitments to $(0^{p(\lambda)})$. $\mathcal{A}_{\mathsf{NMC}}$ forwards these messages as its commitment messages $\mathsf{c}^j_{2,i}$ to the adversary $\mathcal{A}$ in session s. Once again, it forwards each message $\mathsf{c}^i_{2,j}$ it receives from $\mathcal{A}$ in session s, as its second round commitment message in its interaction with the challenger $\mathcal{C}$. That is, these are the commitments on the right side generated by the man-in-the-middle.

Now, we can clearly see that in the first case, when $\mathcal{C}$ generates commitments to r_i, $\mathcal{A}$'s view corresponds to $\mathsf{Hyb}_{2,s}$ while in the latter case, it exactly corresponds to $\mathsf{Hyb}_{2,s+1}$. However, from the security of the non-malleable commitment scheme, the joint distribution of the value committed to by the adversary $\mathcal{A}_{\mathsf{NMC}}$ (which is the same as $\mathcal{A}$'s commitments) and its view must be indistinguishable in both cases. Therefore, if $\mathcal{A}$ can distinguish between the two hybrids, then $\mathcal{A}_{\mathsf{NMC}}$ can break the non-malleability property of the commitment scheme NMCom. However, $\mathcal{A}_{\mathsf{NMC}}$ only runs in time $\mathsf{T}^{\mathsf{Sim}}_{\mathsf{ZK}} < \mathsf{T}^{\mathsf{Sec}}_{\mathsf{Com}}$ and hence this is a contradiction, thus proving the sub-lemma.

Also, notice that since the joint distribution of the adversary $\mathcal{A}$'s committed values and his view is indistinguishable in both hybrids, this implies that Event E still occurs only with negligible probability in Hyb_3 as well. □

Lemma 21. Hyb_3 *is statistically indistinguishable from* Hyb_4.

Proof. The only difference between the two hybrids is that in Hyb_4, the simulator might output "Special Abort" which doesn't happen in Hyb_3. As shown in the above proof (of Lemma 20), the probability that Event E occurs in Hyb_3 is negligible. Notice from the description of the simulator in Figure 9, the output "Special Abort" occurs exactly if the event E occurs. This means that the probability that the simulator outputs "Special Abort" in Hyb_4 is negligible and this completes the proof. □

Lemma 22. *Assuming the security of the protocol* π^{SM}, Hyb_4 *is computationally indistinguishable from* Hyb_5.

Proof. The difference between the two hybrids is in the messages of protocol π^{SM}. In Hyb_4, in every session, $\mathsf{Sim}_{\mathsf{Hyb}}$ computes the messages of protocol π^{SM} correctly using the honest parties' strategy. In Hyb_5, in each session, if $\mathsf{correct} = 1$, they are computed by running the simulator $\mathcal{S}$ for protocol π^{SM} and if $\mathsf{correct} = 0$, they are computed using the honest parties' strategy. Therefore, the only difference in any session is if $\mathsf{correct} = 1$. If the adversary $\mathcal{A}$ can distinguish between the two hybrids, we will use $\mathcal{A}$ to design an algorithm $\mathcal{A}_{\mathsf{SM}}$ that can break the security of protocol π^{SM}.

Once again, suppose $\mathcal{A}$ can distinguish between these two hybrids with some non-negligible probability p. Then by a simple hybrid argument, there exists hybrids $\mathsf{Hyb}_{4,s}$ and $\mathsf{Hyb}_{4,s+1}$ that can be distinguished by $\mathcal{A}$ with some non-negligible probability $p' < p$ where the difference between these two hybrids is that only in session s, the protocol messages are computed differently.

$\mathcal{A}_{\mathsf{SM}}$ interacts with a challenger $\mathcal{C}$ to break the security of protocol π^{SM}. Also, $\mathcal{A}_{\mathsf{SM}}$ performs the role of $\mathsf{Sim}_{\mathsf{Hyb}}$ in its interaction with the adversary $\mathcal{A}$ exactly as in $\mathsf{Hyb}_{4,s}$ except for session s. Whatever parties $\mathcal{A}$ wishes to corrupt in session s, $\mathcal{A}_{\mathsf{SM}}$ corrupts the same parties in its interaction with π^{SM}. Similarly, whatever messages $\mathcal{A}$ sends to $\mathcal{A}_{\mathsf{SM}}$ as part of the protocol π in session s that correspond to π^{SM} messages, $\mathcal{A}_{\mathsf{SM}}$ sends the same messages to the challenger $\mathcal{C}$. Now, whatever messages $\mathcal{C}$ sends, $\mathcal{A}_{\mathsf{SM}}$ forwards the same to the adversary $\mathcal{A}$ as its messages for the π^{SM} protocol in session s.

Observe that $\mathcal{A}_{\mathsf{SM}}$ runs in time $\mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}}$. If $\mathcal{C}$ sends messages that are computed correctly, this exactly corresponds to $\mathsf{Hyb}_{4,s}$ in $\mathcal{A}_{\mathsf{SM}}$'s interaction with $\mathcal{A}$. On the other hand, if $\mathcal{C}$ sends simulated messages, this exactly corresponds to $\mathsf{Hyb}_{4,s+1}$. Therefore, if $\mathcal{A}$ can distinguish between these two hybrids, $\mathcal{A}_{\mathsf{SM}}$ can use the same distinguishing guess to break the security of protocol π^{SM}. However, π^{SM} is secure against all adversaries running in time T_{SM}, where $\mathsf{T}_{\mathsf{SM}} > \mathsf{T}^{\mathsf{Brk}}_{\mathsf{Com}}$ and hence this is a contradiction. This completes the proof of the lemma.

Further, since the two hybrids are indistinguishable, the probability that $\mathsf{Sim}_{\mathsf{Hyb}}$ outputs "Special Abort" in hybrid 5 continues to remain negligible. □

As in Section 5, we can see why this simulation strategy corresponds to the ideal world.

Acknowledgements: We thank Ron Rothblum for useful discussions.

References

[ACJ17] Prabhanjan Ananth, Arka Rai Choudhuri, and Abhishek Jain. A new approach to round-optimal secure multiparty computation. *CRYPTO*, 2017. 3

[AJL+12] Gilad Asharov, Abhishek Jain, Adriana López-Alt, Eran Tromer, Vinod Vaikuntanathan, and Daniel Wichs. Multiparty computation with low communication, computation and interaction via threshold FHE. In *Advances in Cryptology - EUROCRYPT 2012 - 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK, April 15-19, 2012. Proceedings*, pages 483–501, 2012. 5, 54

[BGI+17] Saikrishna Badrinarayanan, Sanjam Garg, Yuval Ishai, Amit Sahai, and Akshay Wadia. Two-message witness indistinguishability and secure computation in the plain model from new assumptions. *IACR Cryptology ePrint Archive*, 2017. 3, 8

[BHP17] Zvika Brakerski, Shai Halevi, and Antigoni Polychroniadou. Four round secure computation without setup. *IACR Cryptology ePrint Archive*, 2017:386, 2017. 3, 4, 5, 12, 22, 31

[BMR90] Donald Beaver, Silvio Micali, and Phillip Rogaway. The round complexity of secure protocols (extended abstract). In *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing, May 13-17, 1990, Baltimore, Maryland, USA*, pages 503–513, 1990. 3

[BPS06] Boaz Barak, Manoj Prabhakaran, and Amit Sahai. Concurrent non-malleable zero knowledge. In *47th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2006), 21-24 October 2006, Berkeley, California, USA, Proceedings*, pages 345–354, 2006. 3

[BS05] Boaz Barak and Amit Sahai. How to play almost any mental game over the net - concurrent composition via super-polynomial simulation. In *46th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2005), 23-25 October 2005, Pittsburgh, PA, USA, Proceedings*, pages 543–552, 2005. 3

[CLP10] Ran Canetti, Huijia Lin, and Rafael Pass. Adaptive hardness and composable security in the plain model from standard assumptions. In *51th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2010, October 23-26, 2010, Las Vegas, Nevada, USA*, pages 541–550, 2010. 3, 12

[DDN91] Danny Dolev, Cynthia Dwork, and Moni Naor. Non-malleable cryptography (extended abstract). In *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing, May 5-8, 1991, New Orleans, Louisiana, USA*, pages 542–552, 1991. 11

[DHRW16] Yevgeniy Dodis, Shai Halevi, Ron D. Rothblum, and Daniel Wichs. Spooky encryption and its applications. In *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part III*, pages 93–122, 2016. 3, 5, 12, 13, 23, 31, 40

[DI05] Ivan Damgård and Yuval Ishai. Constant-round multiparty computation using a black-box pseudorandom generator. In *Advances in Cryptology - CRYPTO 2005: 25th Annual International Cryptology Conference, Santa Barbara, California, USA, August 14-18, 2005, Proceedings*, pages 378–394, 2005. 3

[DI06] Ivan Damgård and Yuval Ishai. Scalable secure multiparty computation. In *Advances in Cryptology - CRYPTO 2006, 26th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 2006, Proceedings*, pages 501–520, 2006. 3

[GG14] Sanjam Garg and Divya Gupta. Efficient round optimal blind signatures. In *Advances in Cryptology - EUROCRYPT 2014 - 33rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Copenhagen, Denmark, May 11-15, 2014. Proceedings*, pages 477–495, 2014. 3

[GGJS12] Sanjam Garg, Vipul Goyal, Abhishek Jain, and Amit Sahai. Concurrently secure computation in constant rounds. In *Advances in Cryptology - EUROCRYPT 2012 - 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK, April 15-19, 2012. Proceedings*, pages 99–116, 2012. 3, 4, 12

[GKP17] Sanjam Garg, Susumu Kiyoshima, and Omkant Pandey. On the exact round complexity of self-composable two-party computation. In *Advances in Cryptology - EUROCRYPT*

2017 - 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Paris, France, April 30 - May 4, 2017, Proceedings, Part II, pages 194–224, 2017. 3, 4

[GLOV12] Vipul Goyal, Chen-Kuei Lee, Rafail Ostrovsky, and Ivan Visconti. Constructing non-malleable commitments: A black-box approach. In *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012*, pages 51–60, 2012. 3

[GMPP16] Sanjam Garg, Pratyay Mukherjee, Omkant Pandey, and Antigoni Polychroniadou. The exact round complexity of secure computation. In *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part II*, pages 448–476, 2016. 3, 7, 8

[GMW87] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or A completeness theorem for protocols with honest majority. In *Proceedings of the 19th Annual ACM Symposium on Theory of Computing, 1987, New York, New York, USA*, pages 218–229, 1987. 3, 5

[Gol04] Oded Goldreich. *The Foundations of Cryptography - Volume 2, Basic Applications*. Cambridge University Press, 2004. 12, 52

[Goy11] Vipul Goyal. Constant round non-malleable protocols using one way functions. In *Proceedings of the 43rd ACM Symposium on Theory of Computing, STOC 2011, San Jose, CA, USA, 6-8 June 2011*, pages 695–704, 2011. 3, 11

[GRS$^+$11] Sanjam Garg, Vanishree Rao, Amit Sahai, Dominique Schröder, and Dominique Unruh. Round optimal blind signatures. In *Advances in Cryptology - CRYPTO 2011 - 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings*, pages 630–648, 2011. 3

[HJK$^+$16] Dennis Hofheinz, Tibor Jager, Dakshita Khurana, Amit Sahai, Brent Waters, and Mark Zhandry. How to generate and use universal samplers. In *Advances in Cryptology - ASIACRYPT 2016 - 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part II*, pages 715–744, 2016. 5

[JKKR17] Abhishek Jain, Yael Tauman Kalai, Dakshita Khurana, and Ron Rothblum. Distinguisher-dependent simulation in two rounds and its applications. *IACR Cryptology ePrint Archive*, 2017:330, 2017. 8

[KMO14] Susumu Kiyoshima, Yoshifumi Manabe, and Tatsuaki Okamoto. Constant-round black-box construction of composable multi-party computation protocol. In *Theory of Cryptography - 11th Theory of Cryptography Conference, TCC 2014, San Diego, CA, USA, February 24-26, 2014. Proceedings*, pages 343–367, 2014. 4

[KOS03] Jonathan Katz, Rafail Ostrovsky, and Adam D. Smith. Round efficiency of multi-party computation with a dishonest majority. In *Advances in Cryptology - EUROCRYPT 2003, International Conference on the Theory and Applications of Cryptographic Techniques, Warsaw, Poland, May 4-8, 2003, Proceedings*, pages 578–595, 2003. 3

[KS17] Dakshita Khurana and Amit Sahai. Two-message non-malleable commitments from standard sub-exponential assumptions. *IACR Cryptology ePrint Archive*, 2017:291, 2017. 3, 5, 6, 7, 10, 15, 24, 32, 40

[LP11] Huijia Lin and Rafael Pass. Constant-round non-malleable commitments from any one-way function. In *Proceedings of the 43rd ACM Symposium on Theory of Computing, STOC 2011, San Jose, CA, USA, 6-8 June 2011*, pages 705–714, 2011. 3

[LPS17] Huijia Lin, Rafael Pass, and Pratik Soni. Two-round concurrent non-malleable commitment from time-lock puzzles. *IACR Cryptology ePrint Archive*, 2017:273, 2017. 6

[LPV08] Huijia Lin, Rafael Pass, and Muthuramakrishnan Venkitasubramaniam. Concurrent non-malleable commitments from any one-way function. In *Theory of Cryptography, Fifth Theory of Cryptography Conference, TCC 2008, New York, USA, March 19-21, 2008.*, pages 571–588, 2008. 11

[MMY06] Tal Malkin, Ryan Moriarty, and Nikolai Yakovenko. Generalized environmental security from number theoretic assumptions. In *Theory of Cryptography, Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006, Proceedings*, pages 343–359, 2006. 3

[MW16] Pratyay Mukherjee and Daniel Wichs. Two round multiparty computation via multi-key FHE. In *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part II*, pages 735–763, 2016. 5, 13

[Pas03] Rafael Pass. Simulation in quasi-polynomial time, and its application to protocol composition. In *Advances in Cryptology - EUROCRYPT 2003, International Conference on the Theory and Applications of Cryptographic Techniques, Warsaw, Poland, May 4-8, 2003, Proceedings*, pages 160–176, 2003. 3, 4, 5, 7

[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. 3, 10

[Pas13] Rafael Pass. Unprovable security of perfect NIZK and non-interactive non-malleable commitments. In *Theory of Cryptography - 10th Theory of Cryptography Conference, TCC 2013, Tokyo, Japan, March 3-6, 2013. Proceedings*, pages 334–354, 2013. 4, 9

[Pas16] Rafael Pass. Unprovable security of perfect NIZK and non-interactive non-malleable commitments. *Computational Complexity*, 25(3):607–666, 2016. 9

[PPV08] Omkant Pandey, Rafael Pass, and Vinod Vaikuntanathan. Adaptive one-way functions and applications. In *Advances in Cryptology - CRYPTO 2008, 28th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2008. Proceedings*, pages 57–74, 2008. 3

[PR05] Rafael Pass and Alon Rosen. Concurrent non-malleable commitments. In *46th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2005), 23-25 October 2005, Pittsburgh, PA, USA, Proceedings*, pages 563–572, 2005. 11

[PS04] Manoj Prabhakaran and Amit Sahai. New notions of security: achieving universal composability without trusted setup. In *Proceedings of the 36th Annual ACM Symposium on Theory of Computing, Chicago, IL, USA, June 13-16, 2004*, pages 242–251, 2004. 3, 4

[Sah99] Amit Sahai. Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. In *40th Annual Symposium on Foundations of Computer Science, FOCS '99, 17-18 October, 1999, New York, NY, USA*, pages 543–553, 1999. 7

[Wee10] Hoeteck Wee. Black-box, round-efficient secure computation via non-malleability amplification. In *51th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2010, October 23-26, 2010, Las Vegas, Nevada, USA*, pages 531–540, 2010. 3

[Yao82] Andrew Chi-Chih Yao. Protocols for secure computations (extended abstract). In *FOCS*, 1982. 3

[Yao86] Andrew Chi-Chih Yao. How to generate and exchange secrets (extended abstract). In *FOCS*, 1986. 3

A Barriers to Two-Round Secure Computation

In this section, we prove that two-round secure computation of general functionalities is impossible in the simultaneous model, even for the case of two parties (this barrier directly extends to 3 or more parties with two corruptions, in a setting where only one party receives the output). We have the following theorem.

Theorem 9. *Consider two-party computation for the functionality $\mathcal{F}(x, y) = (x + y)$, where $x, y \in \mathbb{Z}_p$ for some prime p. Any two-round two-party protocol Π for computing $\mathcal{F}$ implies the existence of a non-interactive non-malleable commitment (with respect to commitment) Π' for two tags.*

Proof. (Sketch) Suppose Π is a two-round secure computation protocol between parties A and B, where both parties receive the output. We will also consider a "mauling" rushing adversary that statistically corrupts B, let us denote this corrupted party by $\widetilde{B}$. At the beginning of the protocol A sends an honest encoding of its input X. On obtaining the first round message from party A, $\widetilde{B}$ "mauls" the encoding sent by A and generates another encoding of the same input X. Because the encodings must necessarily hide the inputs of parties, the honest PPT party A cannot detect if such a mauling occured, and sends the second message of the protocol. $\widetilde{B}$ at this point, generates its second round message on its own, *but does not send this message.* Instead, $\widetilde{B}$ computes the output of the protocol (which is guaranteed by correctness). Such an adversary, $\widetilde{B}$ learns $2X$, and blatantly breaks security of the SPS-secure protocol. Similarly, a rushing adversary could choose to corrupt party A and launch an identical attack. By security of the protocol, a (superpolynomial-time) reduction R must be able to use such an adversary to break an underlying challenge C.

Then we consider the following construction Π' of non-interactive non-malleable commitments with respect to commitment for two identities, that prevents copying attacks. If identity $= 0$, then the committer uses A's identity and the description of A's input encoding algorithm to generate his commitment. If identity $= 1$, then the committer uses B's identity and the description of B's input encoding algorithm to generate his commitment.

The resulting commitment NMC must be binding, by correctness of the underlying two-party computation protocol, and hiding against PPT adversaries, because of the input-hiding property of the encoding. Furthermore, NMC must be a prevent mauling attacks. Consider a MIM that chooses identity 0 for the honest interaction and identity 1 for the man-in-the-middle's interaction. Then, given any MIM that successfully copies, the reduction $\mathcal{R}$ that is guaranteed by the two-party computation protocol against a corrupted $\widetilde{B}$ can be used to break the underlying challenge C. Similarly, if the MIM picks identity 1 for the honest interaction and 0 for the man-in-the-middle's interaction, then $\mathcal{R}$ that is guaranteed by the two-party computation protocol against a corrupted $\widetilde{A}$ can be used to break the underlying challenge C. □

B Secure Multiparty Computation

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 $\mathsf{poly}(\lambda)$ 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.

B.1 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 $\mathbf{x} = (x_1, \ldots, x_n)$.

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 $\mathsf{REAL}_{\pi,\mathcal{A}(z),I}(k, \mathbf{x})$ 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 $\{\mathsf{REAL}_{\pi,\mathcal{A}(z),I}(k,\mathbf{x})\}_{k\in\mathsf{N},\langle\mathbf{x},z\rangle\in\{0,1\}^*}$.

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 $\mathsf{IDEAL}_{f_\perp,\mathcal{A}(z)}(k,\mathbf{x})$ as above,and we let $\{\mathsf{IDEAL}_{f_\perp,\mathcal{A}(z),I}(k,\mathbf{x})\}_{k\in\mathsf{N},\langle\mathbf{x},z\rangle\in\{0,1\}^*}$ 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 6. *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:*

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

   where $\mathbf{x} = \{x_i\}_{i\in[n]} \in \{0,1\}^*$ 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 super-polynomial time adversary* Sim *such that for any $I \subset [n]$ with $|I| \leq t$ the following quantity is negligible:*

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

B.2 Security Against Semi-Malicious Adversaries

We take this definition almost verbatim from [AJL+12]. We define a notion of a semi-malicious adversary that is stronger than the standard notion of semi-honest adversary and formalize security against semi-malicious 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 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 P_k up to that point, including the new message m, must exactly match the honest protocol specification for 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 P_k in any step of the interaction.

Definition 7. *We say that a protocol π securely realizes f for semi-malicious adversaries if it satisfies Definition 6 when we only quantify over all semi-malicious adversaries A.*