Black-Box Constructions of Bounded-Concurrent Secure Computation

Sanjam Garg[1*], Xiao Liang[2†], Omkant Pandey[2†], and Ivan Visconti[3‡]

[1] University of California, Berkeley, USA
sanjamg@berkeley.edu
[2] Stony Brook University, Stony Brook, USA
{liang1,omkant}@cs.stonybrook.edu
[3] University of Salerno, Italy
visconti@unisa.it

Abstract. We construct a general purpose secure multiparty computation protocol which remains secure under (a-priori) bounded-concurrent composition and makes only black-box use of cryptographic primitives. Prior to our work, constructions of such protocols required non-black-box usage of cryptographic primitives; alternatively, black-box constructions could only be achieved for super-polynomial simulation based notions of security which offer incomparable security guarantees.
Our protocol has a constant number of rounds and relies on standard polynomial-hardness assumptions, namely, the existence of semi-honest oblivious transfers and collision-resistant hash functions. Previously, such protocols were not known even under sub-exponential assumptions.

Keywords: Multi-Party Computation · Bounded Concurrent Composition · Black-Box Construction · Straight-Line Extraction.

* Supported in part from DARPA SIEVE Award, AFOSR Award FA9550-15-1-0274, AFOSR Award FA9550-19-1-0200, AFOSR YIP Award, NSF CNS Award 1936826, DARPA and SPAWAR under contract N66001-15-C-4065, a Hellman Award, a Sloan Research Fellowship and research grants by the Okawa Foundation, Visa Inc., and Center for Long-Term Cybersecurity (CLTC, UC Berkeley). The views expressed are those of the author and do not reflect the official policy or position of the funding agencies.

† Supported in part by DARPA SIEVE Award HR00112020026, NSF grant 1907908, and a Cisco Research Award. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the United States Government, DARPA, NSF, or Cisco.

‡ This research has been supported in part by the European Union's Horizon 2020 research and innovation programme under grant agreement No 780477 (project PRIViLEDGE).

Table of Contents

1 Introduction

Secure multiparty computation (MPC) allows n players to jointly compute a functionality, while no group of (malicious parties) learn anything beyond their inputs and prescribed outputs. Introduced in the seminal works of [Yao86, GMW87], this model has since been studied extensively. General constructions for computing any functionality even when a majority of players are adversarial have been long known. The focus of this work are MPC protocols that only make a black-box use of cryptographic primitives and maintain security in the concurrent setting where several instances of the protocol may execute simultaneously.

Black-Box Constructions. General purpose MPC protocols are often *non-black-box* in nature, i.e., they use the code of the underlying cryptographic primitives at some stage of the computation. For example, a common step in such protocols is to use general-purpose zero-knowledge proofs which perform NP reductions. Non-black use of primitives is usually undesirable since not only is it computationally expensive, it also renders the protocol useless in situations where such code is not available (e.g., primitives based on hardware-tokens). One therefore seeks *black-box* constructions of such protocols which use the underlying primitives only in black-box way (i.e., only through their input/output interfaces).

Black-box constructions of general MPC protocols have received considerable attention recently. In the standalone setting, Ishai et al. [IKLP06] (together with Haitner [Hai08]) presented the first black-box construction of general purpose MPC under the minimal assumption of semi-honest oblivious Transfer (OT). Subsequently, Wee [Wee10] reduced the round complexity of these constructions to $O(\log^* n)$, and Goyal [Goy11] to only constant rounds. Very recently, Applebaum et al. [ABG+20] showed that 2-round MPC is unachievable by making only black-box use of 2-round OT. In the two-party setting, black-box construction were obtained by Pass and Wee [PW09] in constant-rounds and Ostrovsky et al. [ORS15] in 5 rounds, which is optimal w.r.t. *black-box proof techniques* [KO04]. We discuss the concurrent setting next.

Concurrent Security. The standard notion of security for MPC, also called *stand-alone security* considers only a single execution of this protocol. While this is sufficient for many applications, other situations (such as protocol executions over the Internet) require stronger notions of security. This setting, where there may be many protocols executions at the same time, is called the *concurrent* setting. Unfortunately, it is known that stand-alone security does not necessarily imply security in the concurrent setting [FS90].

To address the above issue, Canetti [Can01] proposed the notion of *universally composable* (UC) security where protocols maintain their strong simulation based security guarantees even in the presence of other arbitrary protocols. Achieving such strong notion of UC-security turned out to be impossible in the plain model [Can01, CKL03]. Moreover, Lindell [Lin03, Lin04] proved that even in the special case where only instantiations of the *same* protocol are allowed, standard notion of polynomial-time simulation is impossible to achieve. (This is called "self composition" and also corresponds to the setting in this work.)

These strong negative results motivated the study of alternative notions of security. Our focus is the *plain* model where no trusted setup is available. Two directions that are relevant to us in this model are:

– **Bounded-Concurrent Composition:** in this model, a bound m is fixed a-priori, and the protocol design may depend on m. The adversary is allowed to participate in at most m simultaneous executions of the protocol. We consider security against *dishonest majority* with *interchangeable*

roles, i.e., the adversary can choose an arbitrary subset of (all but one) parties to corrupt in each session. As in the original (unbounded) setting, the ideal-world simulator is required to run in (expected) polynomial time. Due to the a-priori bound, it is feasible to bypass the aforementioned negative results. Lindell presented a m-bounded concurrent two-party protocol in $O(m)$-rounds using black-box simulation [Lin03]. Subsequently Pass and Rosen [PR03] presented a constant round two-party protocol and Pass [Pas04] a constant round MPC protocol (under improved assumptions), using non-black-box simulation. All general-purpose secure-computation protocols in this setting make non-black-box use of the underlying cryptographic primitives.

– **Super-Polynomial Simulation:** while it is not directly relevant to this work, we build upon techniques developed in the context of super-polynomial simulation where the simulator is allowed to run in *super-polynomial time*. This relaxation provides somewhat weaker security guarantees (which are, nonetheless, meaningful for many functionalities), and allows (unbounded) concurrent composition. Three different ways to formulate this notion are super-polynomial simulation (SPS) [Pas03], angel-based security [PS04, CLP10], and security with shielded oracles [BDH+17]. Prabhakaran and Sahai [PS04] provided the initial positive result for SPS security. Although, these early results [PS04, BS05, MMY06, LPV09] relied on non-standard/sub-exponential assumptions, Canetti, Lin and Pass achieved this notion under standard polynomial-time assumptions [CLP10] in polynomially many rounds, and soon after, Garg et al. [GGJS12] in *constant* rounds. The works in [PS04, MMY06, CLP10] actually achieve angel-based security, though only [CLP10] relies on standard polynomial hardness. Subsequently, Goyal et al. [GLP+15] presented a $\widetilde{O}(\log n)$ round construction under the same assumptions.
Black-box constructions of angel-based secure computation were first presented by Lin and Pass [LP12] assuming the existence of semi-honest OT, in $O(\max(n^\epsilon, R_{\mathsf{OT}}))$ rounds, where $\epsilon > 0$ is an arbitrary constant and R_{OT} is the round complexity of the underlying OT protocol. (Hence, if the underlying OT protocol has only constant round, the round complexity is $O(n^\epsilon)$.) Subsequently, Kiyoshima [Kiy14] provided a $\tilde{O}(\log^2 n)$-round construction under the same assumption. To achieve constant round constructions, Broadnax et al. [BDH+17] proposed security with shielded oracles, a notion that lies strictly between SPS and angel-based security, along with a constant-round black-box construction under polynomial hardness assumptions. Recently, Garg, Kiyoshima, and Pandey [GKP18] presented a constant-round black-box MPC protocol which achieves SPS security under polynomial hardness assumptions (which are weaker than those in [BDH+17] at the cost of (weaker) SPS security).

State of the Art. The notion of bounded-concurrent composition requires standard polynomial-time simulation. It does not follow from security notions that rely on super-polynomial simulation (which are known to have black-box constructions). Consequently, all known constructions of bounded-concurrent secure MPC rely on non-black-box usage of underlying cryptographic primitives.

1.1 Our Contribution

In this work, we seek to construct general-purpose MPC protocols that make only black-box use of cryptographic primitives and remain secure under bounded-concurrent self composition. Furthermore, we seek constructions whose security can be proven under standard polynomial hardness assumptions (although, to the best of our knowledge, such protocols are not known even under, say, sub-exponential assumptions since the simulator must still run in polynomial time).

Towards this goal, we first aim to construct a black-box bounded-concurrent oblivious transfer (OT) protocol. At a high level, this construction relies on non-black-box simulation to handle simulation in the bounded-concurrent setting (along the lines of [Bar01, Pas04]); to ensure that this does not result in non-black-box use of cryptographic primitives, we implement this idea using the "black-box non-black-box" protocol of Goyal et al. [GOSV14]. Once we have control over bounded-concurrent simulation, we rely on the OT protocol of Garg et al. [GKP18] to achieve the full oblivious transfer functionality. Unfortunately, implementing this idea is somewhat complex, perhaps in part because abstractions such as "straight-line simulation/extraction" are not straightforward to formalize despite their intuitive appeal. We mitigate this situation by defining a new abstraction which we call (bounded) *robust zero-knowledge*; this notion asks for simulators to work even in the presence of (bounded) external communication which cannot be "rewound" (and therefore, looks very close to UC zero-knowledge [Can01]). Similar notion has been defined by [LP09] in the context of non-malleable commitment w.r.t. an external party (see Definition 2 for more details). Zero-knowledge ($\mathcal{ZK}$) with this robust property allows us to combine the non-black-box simulation techniques with the SPS based proof techniques of [GKP18] to achieve black-box bounded-concurrent OT. An additional feature of our protocol is that it has *constant* rounds.

Along the way, we also present the first "straight-line"[4] extractable commitment scheme that only makes black-box use of semi-honest OTs. This primitive may be useful for other applications, especially for black-box constructions of MPC protocols from minimal assumption.

Having obtained bounded-concurrent security for OT, we proceed to construct bounded-concurrent MPC protocols for all functionalities. This step is executed almost identically to a similar step in [GKP18] and does not require any additional assumptions. It also maintains the black-box and constant round properties of the original OT protocol. Consequently, we obtain the first general-purpose bounded-concurrent secure MPC protocol that makes only black-box use of cryptographic primitives; furthermore, the protocol has constant rounds and relies only on standard polynomial hardness assumptions.

Theorem 1 (Informal). *Assume the existence of constant-round semi-honest oblivious transfer protocols and collision-resistant hash functions. Then, there exists a constant-round black-box construction of general-purpose MPC that achieves bounded-concurrent security.*

The formal statement is given as Theorem 6 in Section 7. This result is essentially a black-box version of Pass's result [Pas04].

1.2 Other Related Works

In addition to the works mentioned in the introduction, there are several works that study security in the concurrent setting. For SPS-security, Pass et al. [PLV12] present a constant-round non-black-box construction of MPC from constant-round semi-honest OT. Dachman-Soled et al. and Venkitasubramaniam [DMRV13, Ven14] present a non-black-box construction that satisfies adaptive security. And very recently, Badrinarayanan et al. [BGJ+17] present a non-black-box 3-round construction assuming sub-exponential hardness assumptions. For angel-based security, Kiyoshima et al. [KMO14] present a constant-round black-box construction albeit under a sub-exponential hardness assumption, and Hazay and Venkitasubramaniam [HV16] present a black-box construction that achieves adaptive security.

[4] It means the extraction strategy does not involve rewinding techniques.

We have not discussed works that focus on other security notions, e.g., input-indistinguishable computation and multiple ideal-query model [Pas04, MPR06, GJ13].

Black-box constructions have been extensively explored for several other primitives such as non-malleable or CCA-secure encryption, non-malleable commitments, zero-knowledge proofs and so on (e.g., [CHH+07, PW11, CDSMW17, GLOV12, GOSV14, OSV15]). For concurrent OT, Garay and MacKenzie [GM00] presented a protocol for independent inputs under the DDH assumption, and Garg et al. [GKOV12] proved the impossibility of this task for general input distributions.

2 Overview of Our Techniques

Before describing our approach, we first make some observations. We start by noting that in the context of concurrent secure computation, it is not possible to use rewinding-based simulation techniques since the simulator will have to provide additional outputs during rewinding but the ideal functionality does not deliver more than one output. This is in sharp contrast to concurrent zero-knowledge where the output is simply "yes" since the statement is in the language. While this can be salvaged for certain functionalities as shown by Goyal [Goy12], it is essential to move to straight-line simulators for general functionalities. In particular, in the bounded-concurrent setting we must move to non-black-box simulation techniques [Bar02].

Let us also note that in some situations, particularly in the setting of resettable zero-knowledge, a long line of work shows that it is possible to perform non-black-box simulation under one-way functions [BP12, BP13, CPS13]. Furthermore, a black-box version of these simulation techniques under one-way functions was obtained by Ostrovsky, Scafuro, and Venkitasubramaniam [OSV15]. It therefore seems possible to construct bounded-concurrent MPC under the minimal assumption of semi-honest OT in a black-box manner.[5] Unfortunately, this approach is flawed since *all* known non-black-box simulation techniques are based on rewinding and therefore cannot be applied to the concurrent MPC setting. It is also not at all clear if "straight-line" simulatable zero-knowledge based only on one-way functions can be constructed from known approaches. Therefore, we stress that *even without the requirement of black-box usage of primitives, constructing bounded-concurrent MPC under semi-honest OT only remains as a fascinating open problem.*

We therefore attempt to obtain a construction that exploits collision-resistant hash functions, in addition to the minimal assumption of semi-honest OTs. Toward this goal, we build upon techniques developed in the following two works:

1. Garg, Kiyoshima, and Pandey [GKP18] construct a constant-round black-box MPC protocol with SPS-security under polynomial hardness assumptions. The simulator works by extracting crucial information from adversary's messages via brute-force. The simulator is straight-line and such extraction steps are the only non-polynomial work in its execution.
2. Goyal et al. [GOSV14] present a black-box implementation of the non-black-box simulation techniques that rely on adversary's code [Bar01]. Such techniques often (and certainly those of [Bar01, GOSV14]) extend to situations where the adversary may receive arbitrary but *a-priori bounded* amount of external communication.

At a high level, our main idea is to use the simulation technique of [GOSV14] to replace the brute-force extraction steps in [GKP18] with polynomial-time extraction using adversary's code.

[5] In some works, when the construction is black-box but the proof of security uses non-black-box techniques (as in this paper), this is referred to as a semi-black-box protocol.

The corresponding commitment scheme will be interactive. Since this simulator is polynomial time, we can hope to get bounded-concurrent MPC (in contrast to SPS MPC). Implementing this idea turns out to be rather involved. The fact that the commitment protocol is interactive brings its own issues of non-malleability and also interferes with some key proof steps in [GKP18] which rely on rewinding. It is also not enough that the underlying commitment protocol be extractable in a "bounded-concurrent" setting; instead we need a more flexible notion (that, roughly speaking, mirrors straight-line simulation).

Although we have non-black-box simulation techniques at our disposal, we do not rely on the multiple slots approach of Pass [Pas04] to build simulation soundness directly into our protocols. Instead, by relying on the techniques in the aforementioned two works, we obtain a more modular approach where non-malleability and simulation soundness are obtained with the help of an underlying non-malleable commitment. In this sense, the structure of our bounded-concurrent protocol is fundamentally different from that of [Pas04] to achieve bounded-concurrent MPC. We now provide more details.

The high-level structure of our protocol is similar to that of [GKP18] where the MPC protocol is obtained in two steps. First, we obtain a (constant-round) black-box construction of a bounded-concurrent OT protocol. Next, we compose this OT protocol with an existing constant-round OT-hybrid UC-secure MPC protocol. We elaborate on each step below. We remark that we consider concurrent security in the interchangeable-roles setting. So, in the case of OT, the adversary can participate in a session as the sender while concurrently participating in another session as the receiver.

2.1 Black-Box (Constant-Round) Bounded-Concurrent OT

Our OT protocol is very similar to the OT protocol of [GKP18] (which in turn is based on the high-level cut-and-choose structure of [LP12] inspired from [HIK+11, CDMW09, Wee10]) except that we will implement the basic commitment scheme using a "straight-line extractable" commitment (with some other properties that we will discussion soon). At a high level, the OT protocol of [GKP18] proceeds as follows:

1. The protocol is based on cut-and-choose techniques. Therefore, as the first step of the protocol, the sender S and the receiver R commit to their challenges for future stages in advance. This step uses a two-round statistically binding commitment scheme Com. This step avoids selective opening attacks. The ideal-world simulator can extract these challenges by brute-force to perform the simulation. This is the only non-polynomial time step of this simulator (and the one we wish to replace).

2. Next, S and R execute many instances of a semi-honest OT protocol in parallel, where in each instance S and R use the inputs and the randomness that are generated by a coin-tossing protocol.

3. Next, S and R use a non-malleable commitment scheme NMCom to set up a "trapdoor statement" which, roughly speaking, commits a witness to the fact that the trapdoor statement is false. This step, following [GGJS12], makes it possible to commit to a false witness in the security proof while ensuring (due to non-malleability of NMCom) that the adversary still continues to commit to a correct witness (so that his statement is still false). The step is performed by modifying different stages of **one session at a time**. This ensures that changes in one

interactive part of the protocol are not affected by what happens in later stages of that same session.

4. Finally, S and R use OT combiner which allows them to execute an OT with their real inputs securely when most of the OT instances in the previous steps are correctly executed. To check that most of the OT instances in the previous steps were indeed correctly executed, S and R do use cut-and-choose where S (resp., R) chooses a constant fraction of the OT instances randomly and R (resp., S) reveals the input and randomness that it used in those instances so that S (resp., R) can verify whether R executed those instances correctly.

2.1.1 Replacing Com with straight-line extractable commitment

Our goal is to eliminate brute-force extraction using code of the adversary. In doing so, we have to ensure that (1) the interactive nature of the commitment protocol so obtained does not result into new malleability issues in the proof; and (2) the extraction step can be done in a modular fashion (especially in straight-line) so that we can keep the overall proof structure of [GKP18] where one session is modified at a time.

As a starting point, let us consider the Barak-Lindell extractable commitment scheme [BL02]. In their construction, the committer C first sends an enhanced trapdoor permutation f.[6] Then the two parties involve in the following 3-step coin tossing: (1) R sends a commitment $\mathsf{Com}(r_1)$ to a random string r_1; (2) C replies with a random string r_2; (3) R then sends r_1 with a $\mathcal{ZK}$ argument on the fact that this r_1 is indeed the random string he committed in step (1). Both parties learn the value $r = r_1 \oplus r_2$ as the output of the coin tossing. To commit to a (single-bit) message σ, C sends σ masked by the hard-core bit of $f^{-1}(r)$. An extractor can use the $\mathcal{ZK}$ simulator to bias the coin-tossing result to some value r', for which it knows the preimage of $f^{-1}(r')$. Thus, it can extract the committed value.

Straight-Line Extraction. To adapt the above scheme for our purpose, we need to ensure that the construction is black-box *and* that the committed value can be extracted in a straight-line fashion. Toward this goal, we replace R's commitment and $\mathcal{ZK}$ argument with the protocol of Goyal et al. [GOSV14]. More specifically, [GOSV14] provides a "commit-and-prove" primitive Π_{ZK} where:

- they provide a (non-interactive statistically-binding) commitment scheme[7] called VSSCom using which one can send a commitment y to a string x;
- and later, prove to a verifier, that "y is a commitment to string x such that $\phi(x) = 1$" where ϕ is an arbitrary function.

In particular, ϕ is chosen to be the $\mathcal{NP}$-relation for an $\mathcal{NP}$-complete language in [GOSV14] to get a black-box version of Barak's result [Bar01].

In our case, we will choose ϕ to be the identity function $I_x(\cdot)$.[8] Therefore, the Barak-Lindell commitment protocol mentioned above can be implemented in a black-box manner by ensuring that: (1) R uses VSSCom to prepare the commitment to r_1, and (2) protocol Π_{ZK} is the aforementioned proof protocol with $\phi := I_{r_1}(\cdot)$.

[6] In their original construction, C sends a trapdoor permutation (TDP) f and then proves in zero-knowledge that f is indeed a valid TDP. To make this step black-box, C can send an enhanced TDP instead (without the need of $\mathcal{ZK}$ proof).

[7] In [GOSV14], this commitment was required to be statistically-hiding. But it can be replaced with a statistically-binding scheme if certain modifications are made to the proof phase. See Remark 2 for more details.

[8] Note $I_x(y) = 1$ if and only if $y = x$ is well defined and the "code" of I_x requires only the knowledge of x.

At a high level, this approach meets our needs for a black-box construction that supports straight-line extraction. But more caution is needed to handle the actually simulation as we are in the (bounded) *concurrent* setting. We will address this concern in Section 2.1.2.

Removing TDPs. Since we aim to have a construction assuming only semi-honest OTs (and CRHFs), we also want to remove the reliance on the (enhanced) TDPs. As the first attempt, we ask C to secret-share the message σ to n random shares using exclusive-or. Then let the receiver learn through a special OT (e.g. an $n/2$-out-of-n OT) half of these shares. Next, we invoke the above (black-box) version of coin-tossing in Barak-Lindell protocol to determine another $n/2$ shares that C will decommit to. Due to the pseudo-randomness of the coin-tossing result, R will learn the the shares that "complement" what he learned through OT with only negligible probability. Thus, we can hope to achieve (computational) hiding. Meanwhile, an extractor could always bias the coin-tossing result to the complement shares, thus allowing it to extract the value σ.

However, there are several issues with this approach. First, the sender's (committer's) input to the OT must be the decommitment information to the secret shares. Otherwise, a malicious sender can use arbitrary values in the OT execution, which will disable our extraction strategy.[9] Also, this construction suffers from *selective opening attacks* (SOAs) as the values in the commitments are correlated. It is not clear how we can use standard techniques (e.g. asking R to commit to his challenges in advance, or using another coin-tossing to determine his challenges) to get rid of SOAs. This is because we need to keep R's challenges in this stage hidden from C (to ensure extractability).

To solve this problem, we let C commit to $2n$ secret shares of σ, denoted as $\{\mathsf{Com}(s_{i,b})\}_{i\in[n],b\in\{0,1\}}$. Then n 1-out-of-2 OT instances are executed in parallel, where R learns (the decommitment to) one share out of $(s_{i,0}, s_{i,1})$ in the i-th OT. Next, we can use the Barak-Lindell coin tossing to determine an n-bit string $r = r_1\| \ldots \|r_n$. Finally, C decommits to $\{\mathsf{Com}(s_{i,r_i})\}_{i\in[n]}$. In this construction, R's input to (a single) OT can be guessed correctly with probability $1/2$. By a careful design of hybrids, we show this is sufficient to get rid of SOAs, thus allowing us to prove hiding property (See Section 5). Moreover, the extractor can still learn all the shares by biasing r_i to the complement to its input in the i-th OT instance (for all $i \in [n]$).

Merging with [GKP18]. Finally, to ensure that the interactive nature does not create non-malleability issues, we will ask each party to commit to a long-enough random string, using the above extractable commitment. This step is done as the foremost step in our OT protocol (called "Step 0"). Then each party will use the long random string as one-time pad to "mask" the values that they want to commit to during the execution of our OT protocol. Now, we can rely on the structure of the hybrid proof of [GKP18], which first deals with all stages of a given session and then moves on to the next session in a specific order (determined by the transcript). The key observation here is that since Step 0 is performed ahead of all other steps for a fixed session s, changes in later stages of s cannot affect what happens in Step 0 (for example, issues of malleability and simulation-soundness do not arise). Furthermore, since any rewinding-based proofs of [GKP18] are only relevant to later stages, they do not rewind Step 0 of sessions s.

Remark 1. Ostrovsky et al. [OSV15] showed how to achieve the same as [GOSV14] while relaxing the assumption from CRHFs to one-way functions (OWFs). But we cannot use their approach (or any of the prior approaches that perform non-black-box simulation under OWFs) since simulators

[9] Note that we cannot ask the committer to prove in zero-knowledge that he uses the committed shares as sender's input in the OT execution, because such proof will make non-black-box use of both the commitment and OT.

in these approaches are not straight-line. It uses both the adversary's code **and** rewinding to get a OWF-based construction.

2.1.2 Robust-$\mathcal{ZK}$ for dealing with bounded concurrency

The final issue that we need to address is how the non-black-box simulation will actually be performed corresponding to protocol Π_{ZK} (in Step 0) mentioned above. The main issue is that there are concurrently many sessions of Π_{ZK} executing simultaneously. In particular, if there are m sessions of OT protocol, then there will be $\ell = 2m$ sessions of Π_{ZK}. Simply replacing the prover with the non-black-box simulator may not result in polynomial-time simulation.

An immediate idea is that if Π_{ZK} is *bounded-concurrent* $\mathcal{ZK}$ for up to ℓ sessions, then we can use the *concurrent* non-black-box simulator to simulate Step 0 of all m sessions of the OT protocol at once. This allows us to bias coin-tossing for all m sessions and then we can design hybrids exactly as in [GKP18].

Unfortunately, bounded-concurrent $\mathcal{ZK}$ only guarantees *self* composition; i.e., it can only deal with messages of protocol Π_{ZK}. In our case, Π_{ZK} is part of a larger protocol execution and the adversary receives messages from different stages of all sessions. We thus need a more robust notion of non-black-box simulation which, roughly speaking, (a) is straight-line, and (b) enables bounded-concurrent composition of $\mathcal{ZK}$ protocols *even in the presence of external messages* as long as the total communication outside the $\mathcal{ZK}$ protocol is a-priori bounded.

We formulate this notion explicitly in Section 4 and call it *robust zero-knowledge*. The notion requires that the view of a (standalone) verifier V^* who interacts with an external party B can be simulated by a simulator S only on input the code of V^*. The simulator is not allowed to rewind V^* or B. However, both B and S are allowed to see each others messages (which is essential to make sure that many concurrent instances of the simulators compose seamlessly). This yields a notion that is similar in spirit to UC zero-knowledge [Can01] and implies bounded-concurrent $\mathcal{ZK}$.

We remark that most $\mathcal{ZK}$ protocols based on non-black-box simulation, with suitable adjustment of parameters, can actually handle *arbitrary* external messages (and not just the messages of the same protocol) without any modification. This observation was first used in Barak's original work [Bar01], and finds applications in other places [BL02, PR03, Pas04]. In particular, it also holds for the protocol of Goyal et al. [GOSV14] and is implicit in their security proof. Thus, these protocols already achieve the (bounded) robust-$\mathcal{ZK}$ notion. Robust-$\mathcal{ZK}$ is just a convenient tool to help in the hybrid proofs.

By setting the parameters of Π_{ZK} so that it is ℓ-robust-$\mathcal{ZK}$ allows us to replace the provers of Π_{ZK} with simulator instances in Step 0 of any given session s while maintaining the overall structure and sequence of hybrids in [GKP18] where stages of one session are handled at any given time. This gives us m-bounded concurrent OT.

2.2 Composition of OT with OT-hybrid MPC

The final step of our construction is the same as in [GKP18]. Namely, we compose our bounded-concurrent OT protocol with a OT-hybrid UC-secure MPC protocol (i.e., replace each invocation of the ideal OT functionality in the latter with an execution of the former), thereby obtaining a MPC protocol in the plain model. While selecting the parameters, we have to ensure we adjust the parameters of Π_{ZK} to allow long enough messages so that simulation can be performed for the MPC protocol instead of the OT protocol. Since we only proved bounded-concurrent self composition for

OT (not full UC-security), we do not get a proof for the MPC protocol right away. Hence, we prove the security by analyzing the MPC protocol directly. In essence, what we do is to observe that the security proof for our OT protocol (which consists of a hybrid argument from the real world to the ideal world) still works even after the OT protocol is composed with a OT-hybrid MPC protocol.

3 Preliminaries

We denote the security parameter by n. We use $\overset{\text{c}}{\approx}$ to denote computational indistinguishability between two distributions. For a set S, we use $x \overset{\$}{\leftarrow} \mathsf{S}$ to mean that x is sampled uniformly at random from S. PPT denotes probabilistic polynomial time and $\mathsf{negl}(\cdot)$ denotes negligible functions. Some basic terminologies and definitions (e.g. secret sharing schemes, commitment schemes, and extractable commitment schemes) are given in Section A. In the following, we present the formal definition for *Non-Malleable Commitments* and *Bounded-Concurrent MPC (with Interchangeable Roles)*.

3.1 Non-Malleable Commitment Schemes.

We recall the definition of non-malleable commitment schemes from [LP09]. Let $\langle C, R\rangle$ be a tag-based commitment scheme (i.e., a commitment scheme that takes a n-bit string (a *tag*) as an additional input). For any man-in-the-middle adversary $\mathcal{M}$, consider the following experiment. On input security parameter 1^n and auxiliary input $z \in \{0,1\}^*$, $\mathcal{M}$ participates in one left and one right interactions simultaneously. In the left interaction, $\mathcal{M}$ interacts with the committer of $\langle C, R\rangle$ and receives a commitment to value v using identity $\mathsf{id} \in \{0,1\}^n$ of its choice. In the right interaction, $\mathcal{M}$ interacts with the receiver of $\langle C, R\rangle$ and gives a commitment using identity $\widetilde{\mathsf{id}}$ of its choice. Let $\widetilde{v}$ be the value that $\mathcal{M}$ commits to on the right. If the right commitment is invalid or undefined, $\widetilde{v}$ is defined to be $\perp$. If $\mathsf{id} = \widetilde{\mathsf{id}}$, value $\widetilde{v}$ is also defined to be $\perp$. Let $\mathsf{mim}(\langle C, R\rangle, \mathcal{M}, v, z)$ be a random variable representing $\widetilde{v}$ and the view of $\mathcal{M}$ in the above experiment.

Definition 1. *A commitment scheme $\langle C, R\rangle$ is **non-malleable** if for any* PPT *adversary $\mathcal{M}$, the following are computationally indistinguishable.*

- $\{\mathsf{mim}(\langle C, R\rangle, \mathcal{M}, v, z)\}_{n\in\mathbb{N}, v\in\{0,1\}^n, v'\in\{0,1\}^n, z\in\{0,1\}^*}$
- $\{\mathsf{mim}(\langle C, R\rangle, \mathcal{M}, v', z)\}_{n\in\mathbb{N}, v\in\{0,1\}^n, v'\in\{0,1\}^n, z\in\{0,1\}^*}$

The above definition can be generalized naturally so that the adversary gives multiple commitments *in parallel* in the right interaction. The non-malleability in this generalized setting is called *parallel non-malleability*. (It is known that this "one-many" definition implies the "many-many" one, where the adversary receives multiple commitments in the left session [LPV08].)

Robust non-malleability. We next recall the definition of k-robust non-malleability (a.k.a. non-malleability w.r.t. k-round protocols) [LP09]. Consider a man-in-the-middle adversary $\mathcal{M}$ that participates in one left interaction—communicating with a machine B—and one right interaction—communicating with a receiver a commitment scheme $\langle C, R\rangle$. As in the standard definition of non-malleability, $\mathcal{M}$ can choose the identity in the right interaction. We denote by $\mathsf{mim}^{B,\mathcal{M}}_{\langle C,R\rangle}(y, z)$ the random variable consisting of the view of $\mathcal{M}(z)$ in a man-in-the-middle execution when communicating with $B(y)$ on the left and an honest receiver on the right, combined with the value $\mathcal{M}(z)$ commits to on the right. Intuitively, $\langle C, R\rangle$ is non-malleable w.r.t. B if $\mathsf{mim}^{B,\mathcal{M}}_{\langle C,R\rangle}(y_1, z)$ and

$\mathsf{mim}^{B,\mathcal{M}}_{\langle C,R\rangle}(y_2,z)$ are indistinguishable whenever interactions with $B(y_1)$ and $B(y_2)$ are indistinguishable.

Definition 2. *Let $\langle C,R\rangle$ be a commitment scheme and B be a* PPT *ITM. We say that a commitment scheme $\langle C,R\rangle$ is* ***non-malleable w.r.t.*** *B if the following holds: For every two sequences $\{y^1_n\}_{n\in\mathbb{N}}$ and $\{y^2_n\}_{n\in\mathbb{N}}$ such that $y^1_n, y^2_n \in \{0,1\}^n$, if it holds that for any* PPT *ITM $\mathcal{A}$,*

$$\left\{\langle B(y^1_n),\mathcal{A}(z)\rangle(1^n)\right\}_{n\in\mathbb{N},z\in\{0,1\}^*} \overset{c}{\approx} \left\{\langle B(y^2_n),\mathcal{A}(z)\rangle(1^n)\right\}_{n\in\mathbb{N},z\in\{0,1\}^*} ,$$

it also holds that for any PPT *man-in-the-middle adversary $\mathcal{M}$,*

$$\left\{\mathsf{mim}^{B,\mathcal{M}}_{\langle C,R\rangle}(y_1,z)\right\}_{n\in\mathbb{N},z\in\{0,1\}^*} \overset{c}{\approx} \left\{\mathsf{mim}^{B,\mathcal{M}}_{\langle C,R\rangle}(y_2,z)\right\}_{n\in\mathbb{N},z\in\{0,1\}^*} .$$

$\langle C,R\rangle$ is *k-robust* if $\langle C,R\rangle$ is non-malleable w.r.t. any machine that interacts with the adversary in k rounds. We define parallel k-robustness naturally.

Black-box instantiation. There exists a constant-round black-box construction of a parallel (actually, concurrent) non-malleable commitment scheme based on one-way functions [GLOV12]. Furthermore, Garg, Kiyoshima, and Pandey [GKP18] show that any parallel non-malleable commitment can be transformed into a parallel k-robust non-malleable one in the black-box way by using collision-resistant hash functions (more precisely, by using statistically hiding commitment schemes, which can be constructed from collision-resistant hash functions). If k is constant, the round complexity of their transformation increases only by a constant factor in this transformation. Thus, there exists a $O(1)$-round parallel $O(1)$-robust nonmalleable commitment scheme assuming the existence of CRHFs [GLOV12, GKP18].

3.2 Bounded-Concurrent MPC with Interchangeable Roles

We recall the definition of m-bounded concurrent secure computation. Parts of this section are taken verbatim from [Pas04] with minor modification, following [GGS15], to allow for *interchangeable* roles; these in turn are a slight generalization of "security with abort and no fairness" of [GL02] and concurrent secure two-party computation with adaptive inputs of [Lin04]. The basic formulation and setup of secure computation follows [GL91, MR92, Bea91, Can00].

We consider the case of self composition where m simultaneous executions of the same MPC protocol Π take place. We will consider security against *interchangeable* roles where a party controlled by the adversary can play different roles in different sessions (see description below). We will only consider the *malicious* and *static* setting where the set of corrupted parties is fixed at the beginning of the protocol and the corrupted parties execute the instructions provided by the adversary. The scheduling of message delivery is decided by the adversary. Since security against interchangeable roles is impossible without identities, we assume each party has a unique identity $\mathsf{id} \in \{0,1\}^n$. Since we do not consider fairness, the adversary will always receive its own output and can then decide when (if at all) the honest parties will receive their output.

Multi-party computation. A multi-party protocol problem for k parties $P_1,\ldots,P_k$ is cast by specifying a random process that maps vectors of inputs to vectors of outputs (one input and one output for each party). We refer to such a process as a k-ary functionality and denote it $f:(\{0,1\}^*)^k \to (\{0,1\}^*)^k$, where $f=(f_1,...,f_k)$. That is, for every vector of inputs $\overline{x}=(x_1,...,x_k)$, the output-vector is a random variable $\big(f_1(\overline{x}),...,f_k(\overline{x})\big)$ ranging over vectors of strings. The output

of the i'th party (with input x_i) is defined to be $f_i(\overline{x})$. In the context of concurrent composition, each party actually uses many inputs (one for each execution) and these may be chosen adaptively based on previous outputs. The fact that m-bounded concurrency is considered relates to the allowed scheduling of messages by the adversary in the protocol executions; see the description of the real model below.

Concurrent execution in the ideal model. Next, we describe the concurrent execution of the protocol in the ideal world. Unlike the stand-alone setting, here the trusted party computes the functionality many times, each time upon different inputs.

Let $\Pi := (P_1, \ldots, P_k)$ be an MPC protocol for computing a k-ary functionality f and n be the security parameter. For simplicity we assume that the length of the inputs of each party is n. In total, let there be N parties: $Q_1, \ldots, Q_N$ and let P_i^j denote the party playing the role of P_i in session j (for $i \in [k], j \in [m]$). The adversary can corrupt an arbitrary subset of these parties.

Let $I \subset [N]$ denote the subset of corrupted parties. An ideal execution with an adversary who controls the parties I proceeds as follows:

Inputs: The inputs of the parties $Q_1, ..., Q_N$ in each session j are determined using PPT machines $M_1, ..., M_k$ which take as input the session number j, some inputs $x_1, ..., x_N$, and the outputs that were obtained from executions that have already concluded. Note that the number of previous outputs range from zero (when no previous outputs have been obtained) to some polynomial in n that depends on the number of sessions initiated by the adversary.

Session initiation: When the adversary initiates the session number $j \in [m]$ by sending a (start-session, j) to the trusted party, the trusted party sends (start-session, j) to parties P_i^j where $i \in [k]$.

Honest parties send inputs to trusted party: Upon receiving (start-session, j) from the trusted party, each honest party P_i^j applies its input-selecting machine M_i to its initial input x_i, the session number j and its previous outputs, and obtains a new input $x_{i,j}$. In the first session $x_{i,1} = M_i(x, 1)$. In later sessions j, $x_{i,j} = M_i(x, j, \alpha_{i,1}...\alpha_{i,\omega})$ where ω sessions have concluded and the outputs of P_i^j were $\alpha_{i,1}, ..., \alpha_{i,\omega}$. Each honest party P_i^j then sends $(j, x_{i,j})$ to the trusted party.

Corrupted parties send inputs to trusted party: Whenever the adversary wishes it may ask a corrupted party P_i^j to send a message $(j, x'_{i,j})$ to the trusted third party, for any $x'_{i,j} \in \{0,1\}^n$ of its choice. A corrupted party P_i^j can send the pairs $(j, x'_{i,j})$ in any order it wishes and can also send them *adaptively* (i.e., choosing inputs based on previous outputs). The only limitation is that for any j, at most one pair indexed by j can be sent to the trusted party.

Trusted party answers corrupted parties: When the trusted third party has received messages $(j, x'_{i,j})$ from all parties (both honest and corrupted) it sets $\overline{x}_j = (x'_{1,j}, ..., x'_{k,j})$. It then computes $f(\overline{x}_j)$ and sends $(j, f_i(\overline{x}'_j))$ to every corrupted P_i^j.

Adversary instructs the trusted party to answer honest parties: When the adversary sends a message of the type (send-output, j, i) to the trusted party, the trusted party directly sends (j, $f_i(x'_j)$) to party P_i^j. If all inputs for session j have not yet been received by the trusted party the message is ignored. If the output has already been delivered to the honest party, or i is the index so that P_i^j is a corrupted party, the message is ignored as well.

Outputs: Each honest party always outputs the vector of outputs that it received from the trusted party. The corrupted parties may output an arbitrary (probabilistic polynomial-time computable) function of its initial input and the messages obtained from the trusted party.

Let $f : (\{0,1\}^*)^k \to (\{0,1\}^*)^k$ be a k-ary functionality, where $f = (f_1, ..., f_k)$. Let S be a non-uniform PPT machine (representing the ideal-model adversary) and let $I \subset [N]$ (the set of corrupted parties) be such that for every $i \in I$, the adversary S controls Q_i. Then the *ideal execution* of f with security parameter n, input-selecting machines $M = M_1, ..., M_k$, initial inputs $\overline{x} = (x_1, ..., x_N)$ and auxiliary input z to S, denoted $\text{IDEAL}_{f,I,S,M}(n, \overline{x}, z)$, is defined as the output vector of the parties and S resulting from the ideal process described above.

We remark that the definition of the ideal model includes the bound m on the concurrency although it is possible to define it without it.

Execution in the real model. We next consider the execution of Π in the real world. We assume that the parties communicate through an *asynchronous* fully connected and authentic point-to-point channel but without guaranteed delivery of messages.

Let f, I be as above and let Π be a multi-party protocol for computing f. Furthermore, let $\mathcal{A}$ be a non-uniform PPT machine such that for every $i \in I$, the adversary $\mathcal{A}$ controls Q_i. Then, the real m-bounded concurrent execution of Π with security parameter n, input-selecting machines $M = M_1, ..., M_k$, initial inputs $\overline{x} = (x_1, ..., x_N)$ and auxiliary input z to $\mathcal{A}$, denoted $\text{REAL}^m_{\Pi,I,A,M}(n, \overline{x}, z)$, is defined as the output vector of the honest parties and the adversary $\mathcal{A}$ resulting from the following process. The parties run concurrent executions of the protocol, where every party initiates a new session whenever it receives a start-session from the adversary. The honest parties then apply their input-selecting machines to their initial input, the session number and their previously received outputs, and obtain the input for this new session. The scheduling of all messages throughout the executions is controlled by the adversary.

Security as emulation of a real execution in the ideal model. The security of Π under bounded composition is defined by saying that for every real-model adversary there exists an ideal model adversary that can simulate an execution of the secure real-model protocol. Formally:

Definition 3 (m-Bounded Concurrent Security in the Malicious Model). *Let $m = m(n)$ be a polynomial and let f, k, N and Π be as above. Protocol Π is said to securely compute f under* m-bounded concurrent composition *if for every real-model non-uniform* PPT *adversary $\mathcal{A}$, there exists an ideal-model non-uniform probabilistic expected polynomial-time adversary S, such that for all input-selecting machines $M = M_1, ..., M_k$, every $z \in \{0,1\}^*$, every $\overline{x} = (x_1, ..., x_N)$, where $x_1, ..., x_N \in \{0,1\}^n$ and every $I \subset [N]$,*

$$\left\{\text{IDEAL}_{f,I,S,M}(n, \overline{x}, z)\right\}_{n\in\mathbb{N}} \overset{c}{\approx} \left\{\text{REAL}^m_{\Pi,I,A,M}(n, \overline{x}, z)\right\}_{n\in\mathbb{N}}$$

That is, concurrent executions of Π with $\mathcal{A}$ cannot be distinguished from concurrent invocations of f with S in the ideal model.

4 Robust Zero-Knowledge Commit-and-Prove Protocols

Goyal et al. [GOSV14] present a new non-black-box zero-knowledge argument for $\mathcal{NP}$. Their protocol (with slight modification for the "commit-and-prove" form) is presented in Protocol 1. We recall briefly how their protocol works.

They first construct a black-box size-hiding commit-and-prove protocol (BBCom, BBProve). In Protocol 1, the committer commits to the secret shares of the witness via BBCom. The Proof Phase combines PCP of Proximity (PCPP) [BGH+04] and Barak's non-black-box $\mathcal{ZK}$ protocol [Bar01].

Protocol 1 ℓ-Robust Commit-and-Prove for ϕ [GOSV14]

Common Input: Security parameter 1^n, robustness parameter ℓ, property ϕ
Auxiliary Input to C': String $w \in \{0,1\}^n$ to be committed.
Commit Phase:

1. C' generate VSS representation of w: $\mathsf{VSS}^w = (w_1^{\mathsf{VSS}}, ..., w_n^{\mathsf{VSS}})$.
2. C' creates commitments to each share with independent randomness $\rho_i \in \{0,1\}^n$, to get $c_i = \mathsf{Com}(w_i^{\mathsf{VSS}}; \rho_i)$ for $i = 1, ..., n$.
3. C' sends $\mathsf{VSSCom}(w) := (c_1, ..., c_n)$.

Comment: Note that ℓ, ϕ are not required in this phase. In [GOSV14], the commit-phase is actually a part of the "proof phase" since the goal is to describe a system for $\mathcal{NP}$. We choose this form to emphasize the commit-and-prove nature of their protocol.

Proof Phase:

1. Trapdoor-generation:
   (a) C' runs $\mathsf{BBCom}(0^n)$ with R'. Let z be the commitment so obtained.
   (b) R' sends a random string r **of length** $n + \ell(n)$. The public theorem a is defined as: $a = (z, r, t)$. This message is referred to as the **long message**.
2. Actual proof for ϕ:
   (a) *Commitment of PCPP:* C' runs $\mathsf{BBCom}(0^n)$ and sends the commitments.
   (b) *PCPP Queries:* R' sends random tapes $r_1, ..., r_{\ell_d}$ from which C' and R' compute $(q_i^j, p_i^j) = Q_{\mathsf{pcpx}}(a, r_j, i)$ with $i \in [k]$, where k is the security parameter for the PCPP. Let $I_j^M = \{q_1^j, ..., q_k^j\}$ and $I_j^\pi = \{p_1^j, ..., p_k^j\}$.
   (c) *Proof.* C' runs $\mathsf{BBProve}(\psi, I^M, I^\pi)$, where the predicate ψ is true iff:
      – D_{pcpx} outputs 1 on selected positions of M and π; **or**
      – There exist $\{(w_i^{\mathsf{VSS}}, \rho_i)\}_{i=1}^n$ such that $c_i = \mathsf{Com}(w_i^{\mathsf{VSS}}; \rho_i)$ for all i and $\phi\left(\mathsf{Recon}(w_1^{\mathsf{VSS}}, ..., w_n^{\mathsf{VSS}})\right) = 1$.

   R' accepts the proof if and only if the verifier of $\mathsf{BBProve}$ accepts.

The committer C' (the prover) first sends z which is supposed to be a commitment to a Turing machine M. An honest prover will just commit to 0^n. Once R' replies with a string r, the trapdoor theorem is set to a of the pair language $\mathcal{L}_{\mathcal{P}} = \{(a \coloneqq (z, r, t), Y) : \exists M \in \{0,1\}^* \text{ s.t. } Y \leftarrow \mathsf{ECC}(M),$ and $M(z) = r$ within t steps.$\}$ (where $\mathsf{ECC}(\cdot)$ is a binary error correcting code tolerating a constant fraction $\delta > 0$ errors). Then C' uses $\mathsf{BBProve}$ to prove either the trapdoor theorem is true or $\phi(w) = 1$.

Note that the proof for the trapdoor theorem is conducted via PCPP. Specifically, commitment to PCPP proof π is sent to R' (honest prover commits to 0^n, as shown in 2-(a) of Proof Phase). R' generates PCPP queries on Y (the private theorem) and π by running algorithm Q_{pcpx}. C' then proves that the PCPP decision algorithm D_{pcpx} verifies to 1. Details of component protocols such as $\mathsf{BBCom}, \mathsf{BBProve}$, etc. are not necessary and omitted; see [GOSV14] for their details.

This protocol makes only black-box use of CRHF; it is also public-coin, constant-rounds, and has negligible soundness error. In fact, it enjoys the following properties:

(i) The protocol is actually a "commit-and-prove" protocol for arbitrary polynomial-size circuits ϕ. That is, it consists of two phases: in the "commit" phase, the committer commits an arbitrary string $w \in \{0,1\}^n$ using a special commitment scheme called VSSCom, and later, in the "proof" phase, it can prove in zero-knowledge that the committed string satisfies ϕ; i.e., $\phi(w) = 1$

where w is uniquely determined from the transcript of the commit phase. For concreteness, the "commit-and-prove" form of [GOSV14] ZK is depicted in Protocol 1.[10]

(ii) To prove zero-knowledge, the simulator relies on Barak's technique of committing the verifier's code [Bar01]. Consequently, the protocol inherits several properties of Barak's original protocol (e.g., public-coin and constant rounds). In particular, the protocol has a "preamble" phase where the verifier sends a random string r; the simulator is "straight-line" *even in the presence of arbitrary (external) communication of a-priori bounded length* $\ell(n)$ provided that $|r|$ is sufficiently bigger than $\ell(n)$.

Remark 2 (On the Hiding Property of VSSCom*).* In the **Commit Phase** of Protocol 1, we define VSSCom, which consists of *statistically-binding* commitments Com on each VSS shares of the value to be committed to (the witness w in our protocol). However, in the original construction of the [GOSV14] ZK, the underlying Com actually needs to be *statistically-hiding*. This is because that their construction relies on the MPC-in-the-head technique, where a subset (verifier's challenge set) of the commitments are revealed to the verifier for the view-consistency checking. Moreover, the security of their construction relies on the hiding of the remaining unopened commitments. Since the challenge set is picked by the verifier, a statically-hiding commitments must be used to resolve the selective-opening problem [Hof11].

We remark that there are two alternative ways to avoid the selective-opening problem, while relying only on statistically-binding commitment:

(1) Ask V to commit to the challenge sets before the P's first message, and to decommit to the challenge sets once it receives P's first message. This approach is taken by, e.g., [PW09, GLOV12, Kiy14].

(2) After P's first message, determine the challenge set by a coin-tossing protocol between P and V, instead of letting V pick the challenge set. This approach appears in [Lin13, CLP20a] (see [CLP20b, Section 4.1] for a detailed demonstration in the MPC-in-the-head setting).

Both of these approaches can be taken if one wants to replace the statistically-hiding commitment in VSSCom and BBCom, while maintaining the security of the [GOSV14] construction. But they were not exploited as [GOSV14] pursued a public-coin construction. As another concern, these approaches only lead to *computational* ZK property, while the original construction in [GOSV14] is *statistical* ZK.

In contrast, we are able to make use of these approaches in the current work. We take approach (2) as it not only gives a cleaner construction, but also maintains the (weak) Proof-of-Knowledge property of the original [GOSV14] ZK. Concretely, we modify the BBProve such that V's challenge set will be determined by the following coin-tossing:

- V first sends an extractable commitment to a random string r_1,
- P responds with a random string r_1
- V then sends r_1 with the decommitment information.

Other parts of BBProve remain unchanged, except that the challenge set is now defined by the (pseudo)random string $r_1 \oplus r_2$.

[10] The protocol for proving $x \in L$ for $L \in \mathcal{NP}$ is obtained by setting w to be a witness for x (under an appropriate relation R for L) and committing to it as the first step of the proof using "commit" phase, followed by the "proof" phase for $\phi(\cdot) := R(x, \cdot)$.

Such a modified BBProve allows us to replace the statistically-hiding commitment with a statistically-binding one in *both* BBCom and VSSCom. Thus, we can safely use the statically-binding VSSCom (as currently presented in Protocol 1).

We also remark that BBCom will not be statistically-binding, even though its underlying commitment is replaced by a statistically-binding one. This is because that BBCom applies the (underlying) commitment to (the paths of) a Merkle hashing tree on the target value, resulting in information loss.

4.1 Robust Zero-Knowledge

To capture the above property (ii) (i.e., "straight-line simulation in the presence of bounded external communication"), we define the notion of *robust zero-knowledge*. It roughly captures the fact that the simulator does not rewind the external party to perform the simulation. This property is implicit in the relations defined for bounded-concurrent simulation in [Bar01, PR03]; a related but very different notion of robustness appears explicitly in the context of non-malleability in [LPV09, GLP+15]. This notion is useful in constructing security proofs even though it follows from [Bar01] (and similar protocols).

Let $L \in \mathcal{NP}$ with witness relation R_L, and let $R_L(x) := \{w : R_L(x, w) = 1\}$. Let $\Pi := \langle P, V \rangle$ be an (efficient) interactive argument system for L and B be an arbitrary PPT ITM.

For $n \in \mathbb{N}, L \in \mathcal{NP}, x \in L, w \in R_L(x), z \in \{0,1\}^*$ and $y \in \{0,1\}^*$, we define the following two experiments:

Real Experiment: The experiment starts the execution of V^* on input $(1^n, x, z)$ where z denotes the auxiliary input of V^*. During its execution, V^* can simultaneously participate in two interactions (1) an execution of Π with the honest prover machine $P(1^n, x, w)$ and (2) arbitrary (unspecified) interaction with the machine $B(1^n, y)$.

The interaction occurs over a network where each message is processed as follows:

- If V^* sends a message of Π (resp., for B), it is delivered to P (resp., to B).
- If P receives a message from V^*, it prepares the next message of Π, denoted a; a is then sent to **both V^* as well as B**.
- If B receives a message from V^*, it prepares the next message (according to the unspecified interaction protocol between B and V^*), say b; message b is then sent to V^*.

The output of this experiment is the *(joint) view* of V^*, and denoted as:

$$\mathsf{Rview}^{B(y)}_{\Pi,n,x}\langle P(w), V^*(z)\rangle.$$

Simulated Experiment: This experiment is identical to the real experiment except that: (1) the honest prover algorithm $P(1^n, x, w)$ is *replaced* with a "simulator" algorithm S which receives the code of V^* as input, and (2) any message V^* receives from B is also provided to S.

Formally, the experiment starts an execution of $V^*(1^n, x, z)$; V^* can simultaneously participate in two interactions (1) an execution of Π with the *simulator machine* $S(1^n, x, \mathsf{code}[V^*], z)$ and (2) arbitrary (unspecified) interaction with the machine $B(1^n, y)$.

The interaction occurs over a network where each message is processed as follows:

- If V^* sends a message of Π (resp., for B), it is delivered to S (resp., to B).

– If S receives a message from V^*, it prepares the next message, denoted a; a is sent to **both** V^* **as well as** B.

– If B receives a message from V^*, it prepares the next message (according to the unspecified interaction protocol between B and V^*), say b; message b is then sent to **both** V^* **and** S.

The output of this experiment is the *(joint) view* of V^*, and denoted by:

$$\mathsf{Sview}^{B(y)}_{\Pi,n,x}\langle S(\mathsf{code}[V^*],z),V^*(z)\rangle.$$

Remark 3. Two important remarks are in order. First, the simulated experiment *does not allow rewinding* by definition. Instead, it requires S to "act like the prover" of protocol Π; the only help S has is the code of V^* as well as immediate access to all messages that V^* receives. In particular, rewinding V^* may involve rewinding B and this is not allowed by the experiment.

Second, both B and S have access to all messages V^* *receives* from the network. S must have access to all such messages to simulate in "straight line" (since it does not have the code of B). B is given access to these messages to facilitate (bounded concurrent) *composition*. In particular, B has access to all message S (or P) sends to V^* and S has access to all messages B sends to V^*.

Protocol Π is robust zero-knowledge if V^* cannot tell whether it is in the real experiment or the simulated one. If it is robust w.r.t. only machines B that send at most ℓ bits, it is called ℓ-robust zero-knowledge. Formally:

Definition 4 (Robust Zero-Knowledge). *An interactive argument system* Π *for a language* $L\in\mathcal{NP}$ *is* robust $\mathcal{ZK}$ w.r.t. a PPT ITM B *if for all* PPT ITM V^* *there exists a* PPT ITM S *(called the* robust simulator*), such that:*

$$\left\{\mathsf{Rview}^{B(y)}_{\Pi,n,x}\langle P(w),V^*(z)\rangle\right\}_{n,x,w,z,y}\overset{c}{\approx}\left\{\mathsf{Sview}^{B(y)}_{\Pi,n,x}\langle S(\mathsf{code}[V^*],z),V^*(z)\rangle\right\}_{n,x,z,y}.$$

where $n\in\mathbb{N},x\in L,w\in R_L(x),z\in\{0,1\}^*,y\in\{0,1\}^*$.

For a polynomial $\ell:\mathbb{N}\to\mathbb{N}$, Π *is* ℓ-robust zero-knowledge *if it is robust w.r.t.* every PPT ITM B *that sends at most* $\ell(n)$ *bits.* Π *is* robust zero-knowledge *if it is* ℓ*-robust zero-knowledge for* every *polynomial* ℓ*.*

Remark 4. We remark that robust (i.e., unbounded) $\mathcal{ZK}$ is actually impossible (for non-trivial languages) in the plain model since, if unbounded external communication was allowed with B, V^* can just be a "dummy" adversary so that access to its code provides no advantage to the simulator to complete the proof. This is akin to the use of dummy adversary in UC setting and impossibility of UC-$\mathcal{ZK}$ for languages outside of BPP [Can01, GK90].

4.1.1 (Bounded) robust $\mathcal{ZK}$ implies bounded $c\mathcal{ZK}$

We now demonstrate the flexibility of using robust $\mathcal{ZK}$ in concurrent settings. More specifically, we show that any ℓ-robust $\mathcal{ZK}$ protocol Π remains $\mathcal{ZK}$ under bounded composition of ℓ' instances for sufficiently large ℓ.

Recall that in the ℓ'-bounded $c\mathcal{ZK}$ composition of protocol Π, an adversarial verifier V^* participates in ℓ' simultaneous executions of Π while controlling the scheduling of messages of various sessions. For simplicity (only) we assume that all provers prove the same statement x using same witness w and let $\mathsf{view}_{\Pi,n,x,w,z}$ denote the view of $V^*(n,x,z)$ in this concurrent execution. We say

that Π is ℓ'-bounded-$c\mathcal{ZK}$ for language L if for every such V^* there exists a simulator S_{V^*} such that for all $x \in L, w \in R_L(x), z \in \{0,1\}^*$:

$$\left\{\mathsf{view}_{\Pi,n,x,w,z}\right\}_{n,x,w,z} \overset{c}{\approx} \left\{S_{V^*}(n,x,z)\right\}_{n,x,z}.$$

Claim 1. *If a protocol Π is ℓ-robust zero-knowledge, then it is ℓ'-bounded $c\mathcal{ZK}$, for any ℓ' such that $\ell' \cdot m \leq \ell$ where m is the length of all messages sent by the prover of protocol Π.*

Proof. We show that a simple composition of individual robust-$\mathcal{ZK}$ simulators for each session yields a simulator for bounded-concurrent composition of Π.

Let V^* be a concurrent verifier participating in ℓ' concurrent sessions of Π. Let S be the robust-$\mathcal{ZK}$ simulator for Π. The bounded-concurrent simulator S_{V^*}, on input the code of V^*, x, and z, proceeds as follows:

- For each session i, S_{V^*} prepares the "fake" prover algorithm S_i which behaves identically to the algorithm $S(n, x, \mathsf{code}[V^*], z)$ with fresh randomness and interacts with V^* in session i.
- S_{V^*} initiates an execution of V^* with fresh randomness, relaying messages between V^* and fake provers $(S_1, \ldots, S_{\ell'})$ as in the bounded-concurrent execution.
- When V^* halts, S_{V^*} outputs its view.

It is straightforward to see that S_{V^*} runs in polynomial time since each S_i and V^* are polynomial time. To prove indistinguishability, consider hybrids $H_0, \ldots, H_{\ell'}$:

Hybrid H_0. The real experiment where V^* concurrently interacts with $(P_1, .., P_{\ell'})$, where P_i $(i \in [\ell'])$ denotes the i-th prover instance of Π on input $(1^n, x, w)$.

Hybrid H_k (for $(k \in [\ell'])$. This hybrid is same as H_{k-1} except that prover instance P_k is replaced by the simulator instance of S_k (defined above). Therefore, V^* interacts with algorithms $(S_1, ..., S_k, P_{k+1}, ...P_{\ell'})$, as the "provers."

Note that $H_{\ell'}$ is the simulator S_{V^*}. It is easy to see that each H_k is polynomial time. We prove that $H_{k-1} \approx_c H_k$ using the robust-$\mathcal{ZK}$ property of Π, where $k \in [\ell']$.

Let B_k be the following machine: B_k incorporates $(S_1, ..., S_{k-1}, P_{k+1}, ..., P_{\ell'})$, and interacts with V^* in the robust-$\mathcal{ZK}$ experiment as follows: B_k proceeds identically to H_{k-1} so that messages of all sessions $i \neq k$, are received from or sent to B_k (which internally simulates H_{k-1}). All prover messages of the k-th session are expected to come from an external machine, say M. If M is the prover instance P_k, the view of V^* is distributed identically to H_{k-1}. Note that a copy of each message of P_k in this case is also sent to B_k at the same time as V^*; consequently, the internal execution of B_k (which includes $S_1, \ldots, S_{k-1}$) continues without any problems. Likewise if M is the simulator instance S_k, V^*'s view is distributed identically to H_k; note that a copy of each message of S_k (resp., B_k) in this case is also sent to B_k (resp., S_k) at the same time as V^*. Consequently executions of both S_k and B_k continues without any problems.

Finally, if m is the total communication from a single prover instance, the external communication to V^* from B_k is bounded by $m\ell' \leq \ell$; furthermore, this condition also holds from the point of view of each S_i instance internal to B_k (as desired). It follows that if Π is ℓ-robust-$\mathcal{ZK}$, it is also ℓ'-bounded concurrent. □

4.2 Constructions of ℓ-Robust $\mathcal{ZK}$

As noted earlier, Barak's bounded $c\mathcal{ZK}$ protocol is also ℓ-robust $\mathcal{ZK}$, although it requires non-black-box use of hash functions [Bar01]. The variant of Barak's technique by Goyal et al. [GOSV14] makes only black-box use of such functions and achieves the same result.[11] To summarize, we have the following theorem from [GOSV14] (restated in our language).

Theorem 2 (Black-Box ℓ-Robust Zero-Knowledge for $\mathcal{NP}$). *If there exists a family $\mathcal{H}$ of collision-resistant hash functions, then for every polynomial ℓ, there exists a constant round public coin ℓ-robust zero-knowledge interactive argument for $\mathcal{NP}$ which requires only oracle access to functions in $\mathcal{H}$.*

As noted earlier, the preceding theorem is actually a corollary of the more general theorem that proof-phase of the commit-and-prove protocol depicted in Protocol 1 is ℓ-robust. We refer the reader to [GOSV14] for a formal definition of "commit-and-prove" protocols. We only recall the following properties for Protocol 1:

- The proof-phase is performed only for the statement defined by the transcript of the commit-phase.
- For each transcript, the receiver gets *only one* (interactive) proof from the committer during the proof-phase. The zero-knowledge property (as well as the implicit ℓ-robust zero-knowledge) is then required only for this single execution of the proof-phase. This suffices for Theorem 2 (by simply repeating the commit-phase before every proof-phase, see footnote 10).
- To get the ℓ-robust $\mathcal{ZK}$ property, the length of the challenge from the verifier is modified to be sufficiently larger than ℓ (as in bounded $c\mathcal{ZK}$ in Barak [Bar01]). Note that this requires modifying the pair language for the universal argument (and PCPP) to allow strings of length at most ℓ. In particular, this language is the following:
  "$\mathcal{L}_{\mathcal{P}} = \{(a = (z, r, t), (Y)) : \exists M \in \{0,1\}^*$ and $\exists y \in \{0,1\}^*$ such that $Y \leftarrow \mathsf{ECC}(M), M(z, y) = r$ within t steps, and $|y| \leq |r| - n.\}$"

  where $\mathsf{ECC}(\cdot)$ is a binary error correcting code tolerating a constant fraction $\delta > 0$ of errors, M is the description of a Turing machine and n is the security parameter. We use $R_{\mathcal{L}_{\mathcal{P}}}$ to denote the relation defined on $\mathcal{L}_{\mathcal{P}}$.

To summarize, we have the following theorem (from [GOSV14]):

Theorem 3 (Black-Box ℓ-Robust Commit-and-Prove). *If there exists a family $\mathcal{H}$ of collision-resistant hash functions, then for every polynomial ℓ and every polynomial-size circuit ϕ, there exists a commit-and-prove protocol such that the commit-phase is statistically binding (with at most two rounds), the proof-phase is a constant-round public-coin ℓ-robust zero-knowledge interactive argument for ϕ, and both phases require only oracle access to functions in $\mathcal{H}$.*

Remark 5. We note that [GOSV14] also provides a size-hiding commitment scheme (which cannot be statistically-binding) along with a ℓ-robust $\mathcal{ZK}$ proof-phase for every ϕ. However, we will not need this version of their protocol.

[11] Although only standalone case is discussed in [GOSV14], their security proof (just like Barak's) also works for bounded-concurrent case by increasing the length of verifier's challenge and slightly modifying the relation for the UARG appropriately.

5 Straight-Line Extractable Commitments

In this section, we construct an extractable commitment scheme, assuming black-box access to any semi-honest oblivious transfer. The construction (shown in Protocol 2) makes black-box use of a statistically-binding commitment Com and a maliciously-secure oblivious transfer OT. For the OT, we require (computational) indistinguishability-based security against malicious senders, and simulation-based security (ideal/real paradigm) against malicious receivers. Such OTs can be constructed in a black-box manner from any semi-honest OT [Hai08]. To ease the presentation, we show in Protocol 2 a single-bit commitment, and talk about how to extend it to commit to strings toward the end of this section (Remark 7).

Theorem 4. *Protocol 2 is a straight-line extractable statistically-binding commitment scheme, which only accesses the underlying primitives in a black-box manner.*

5.1 Proof of Theorem 4

The construction is black-box as we use the black-box commit-and-prove protocol from [GOSV14] (presented in Protocol 1 in Section 4) in the coin-tossing step. Statistically-binding property follows directly from that of the Step-2 commitment scheme Com. Next, we focus on computationally-hiding property and extractability.

5.1.1 Computationally-Hiding

Let σ be an arbitrary bit in $\{0,1\}$. For any PPT receiver R^*, we denote by $\mathcal{V}^{R^*}(n,\sigma)$ the distribution over R^*'s view from an execution $\langle C(\sigma), R^*\rangle$ of Protocol 2, where the honest C commits to the value σ to R^*. To prove the hiding property, we need to show that for any PPT machine $\mathcal{D}$,

$$\mathsf{Adv}_n^{\mathcal{D}} \coloneqq \left|\Pr[\mathcal{D}\big(\mathcal{V}^{R^*}(n,1)\big)=1] - \Pr[\mathcal{D}\big(\mathcal{V}^{R^*}(n,0)\big)=1]\right| \leq \mathsf{negl}(n). \tag{1}$$

In the following, we prove Inequality (1) by a sequence of hybrids.

Hybrid $H_0(n,\sigma)$: in this hybrid, we change the way the values $\{s_{i,b}\}$ are chosen. Specifically, the hybrid does the following:

(a) It samples independently at random a bit $\eta \xleftarrow{\$} \{0,1\}$ and a bit $\mathsf{g} \xleftarrow{\$} \{0,1\}$.

(b) For $i \in [n-1]$ and $b \in \{0,1\}$, it samples independently $s_{i,b} \xleftarrow{\$} \{0,1\}$.

(c) It defines $s_{n,1-\mathsf{g}} \coloneqq \eta \oplus \sigma$ and

$$s_{n,\mathsf{g}} \coloneqq (s_{1,0} \oplus s_{1,1}) \oplus \ldots \oplus (s_{n-1,0} \oplus s_{n-1,1}) \oplus \eta$$

(d) It then uses the honest commiter's strategy and $\{s_{i,b}\}$ defined above to finish Step 2 to 6 in Protocol 2.

(e) Once R^* terminates, H_0 gets the view $\mathcal{V}_0^{R^*}(n,\sigma)$ of R^* in this execution. It invokes $\mathcal{D}$ on input $\mathcal{V}_0^{R^*}(n,\sigma)$, and outputs whatever $\mathcal{D}$ outputs. Let $H_0(n,\sigma)$ also denote the output of this hybrid.

[12] We remark that this step can actually happen in parallel with the OT instances in Step 3. It is put here (only) to ease the presentation of the security proof. In Remark 6, we talk about how the proof can be modified to accommodate the case where the Step-4 OT happens in parallel with Step 3.

Protocol 2 ℓ-Robust Extractable statistically-Binding Commitment

Common Input: Security parameter 1^n, robustness parameter ℓ
Auxiliary Input to C: A bit $\sigma \in \{0,1\}$ to be committed
Commit Phase:

1. C samples $2n$ random bits $\{s_{i,b}\}_{i\in[n],b\in\{0,1\}}$, whose exclusive-or equals σ.
2. C and R involves in $2n$ independent executions of Com in parallel, where C commits to each values in $\{s_{i,b}\}_{i\in[n],b\in\{0,1\}}$ separately. Let $c_{i,b}$ denote the commitment to $s_{i,b}$. Let $d_{i,b}$ denote the decommitment information w.r.t. $c_{i,b}$.
3. R samples independently $n-1$ random bits $\tau_1,\ldots,\tau_{n-1} \xleftarrow{\$} \{0,1\}^{n-1}$. C and R involves in n independent executions of OT in parallel. For the i-th OT execution ($i \in [n-1]$), C acts as the sender with the two private input set to $\mathsf{Inp}_0^{(i)} = d_{i,0}$ and $\mathsf{Inp}_1^{(i)} = d_{i,1}$. R acts as the receiver with input τ_i. Note that at the end of this stage R learns $\{d_{i,\tau_i}\}_{i\in[n-1]}$. R rejects if any of these decommitments are invalid.
4. R samples uniformly at random a bit $\tau_n \xleftarrow{\$} \{0,1\}$. C and R involves in an execution of OT where C acts as the sender with the two private input set to $\mathsf{Inp}_0^{(n)} = d_{n,0}$ and $\mathsf{Inp}_1^{(n)} = d_{n,1}$. R acts as the receiver with input τ_n. Note that at the end of this stage R learns d_{n,τ_n}. R rejects if d_{n,τ_n} is not a valid decommitment w.r.t. c_{n,τ_n}.[12]
5. C and R run a coin-tossing protocol:
   (a) R samples a random string $r_1 \xleftarrow{\$} \{0,1\}^n$ and runs the VSS **Commit Phase** of Protocol 1 to generate $c_{r_1} = \mathsf{VSSCom}(r_1)$. R sends c_{r_1}.
   (b) C chooses a random string $r_2 \xleftarrow{\$} \{0,1\}^n$ and sends r_2.
   (c) R sends r_1 (without decommitment information)
   (d) R and C run the **Proof Phase** of Protocol 1 with robustness parameter $\ell(n)$ to prove that the string r_1 sent by R in Step 5-(c) is indeed the value it committed to in Step 5-(a).

   The output of the coin-tossing phase is $\mathsf{ch} = r_1 \oplus r_2$. For $i \in [n]$, let ch_i denote the i-th bit of ch.
6. C sends to R the values $\{d_{i,\mathsf{ch}_i}\}_{i\in[n]}$. Note that these are the decommitments to $\{c_{i,\mathsf{ch}_i}\}_{i\in[n]}$ in Step 2. R rejects if any of these decommitments are invalid.

Reveal Phase:

1. C sends to R the values $\{d_{i,b}\}_{i\in[n],b\in\{0,1\}}$ (aka all the decommitments).
2. R rejects if any of the decommitments is invalid; otherwise, R computes the decommitted value as $\sigma = \oplus_{i,b} s_{i,b}$. (Note that $s_{i,b}$ is contained in $d_{i,b}$.)

It is straightforward to see the values $\{s_{i,b}\}$ defined above are identically distributed as in the real execution $\langle C(\sigma), R^*\rangle$, i.e. they constitute random secret shares whose exclusive-or equals σ. Also, we note that the value of g does not affect this hybrid at all, as it only introduces syntax changes. Therefore, we have $\mathcal{V}_0^{R^*}(n,\sigma) \overset{\text{id}}{=} \mathcal{V}^{R^*}(n,\sigma)$, which implies that $\forall\sigma \in \{0,1\}$:

$$\Pr[\mathcal{D}(\mathcal{V}^{R^*}(n,\sigma)) = 1] = \Pr[\mathcal{D}(\mathcal{V}_0^{R^*}(n,\sigma)) = 1] = \Pr[H_0(n,\sigma) = 1] \tag{2}$$

Hybrid $H_1(n,\sigma)$: this hybrid is identical to $H_0(n,\sigma)$ except that the n-th OT (in Step 4) are replaced with the ideal OT functionality $\mathcal{F}_{\mathsf{OT}}$. Concretely, in Step 4, the hybrid emulates $\mathcal{F}_{\mathsf{OT}}$ internally in the following way:

– On the committer side, it sets $\mathsf{Inp}_0^{(n)} = d_{n,0}$ and $\mathsf{Inp}_1^{(n)} = d_{n,1}$ (same as the honest committer).

– On the receiver side, it invokes the PPT ideal-world simulator $S^{\hat{R}^*}$ with oracle access to $\hat{R}^*$, which is the residual strategy of R^* with the view fixed up to the beginning of Step 4. Note that the existence of S is guaranteed by the security of OT against corrupted receiver.

– During the execution, $S^{\hat{R}^*}$ may send a bit b which is meant to the ideal-world receiver's input to $\mathcal{F}_{\mathsf{OT}}$ (the actually input of R^* "extracted" by S). In this case, the hybrid responds with $\mathsf{Inp}_b^{(n)} = d_{n,b}$.

– Once $S^{\hat{R}^*}$ stops and outputs the simulated view for $\hat{R}^*$, the hybrid continues to finish the execution of Step 5-6 in the same way as in $H_0(n,\sigma)$. (Note that simulated view for $\hat{R}^*$ contains necessary information to recover the status of R^* up to the end of Step 4. The hybrid can then use it to finish the remaining steps.)

Similar as in H_0, we use $\mathcal{V}_1^{R^*}(n,\sigma)$ to denote the view of R^* in this execution, and use $H_1(n,\sigma)$ to denote the output of this hybrid. By the security of OT against malicious R^*, the $\mathcal{V}_1^{R^*}(n,\sigma)$ should be computationally indistinguishable from $\mathcal{V}_0^{R^*}(n,\sigma)$. This implies that $\forall\sigma \in \{0,1\}$:

$$\Pr[H_1(n,\sigma) = 1] = \Pr[H_0(n,\sigma) = 1] \pm \mathsf{negl}(n) \tag{3}$$

Hybrid $H_2(n,\sigma)$: $H_2(n,\sigma)$ is identical to $H_1(n,\sigma)$ except that it aborts and outputs a special symbol $\perp$ if $b = 1 - \mathsf{g}$. Recall that b is the query of $S^{\hat{R}^*}$ in Step 4 of H_1 (and also H_2). Recall that the bit g is picked uniformly at random, independent of the view of R^*. Therefore, H_2 aborts with probability exactly $1/2$. This implies $\forall\sigma \in \{0,1\}$,

$$\Pr[H_2(n,\sigma)\big) = 1] = \frac{1}{2} \cdot \Pr[H_1(n,\sigma)\big) = 1] \tag{4}$$

Hybrid $H_3(n,\sigma)$: $H_3(n,\sigma)$ is identical to $H_2(n,\sigma)$ except that it aborts and outputs a special symbol $\perp$ if $\mathsf{ch}_n = 1 - \mathsf{g}$ (note that $1 - \mathsf{g} = b$ in this hybrid). recall that ch_n is the last bit of the result of the Step-5 coin-tossing in H_2 (and also H_3). Since the output of Step-5 coin-tossing should be pseudo-random, the probability $\Pr[\mathsf{ch}_n = b]$ is negligibly close to $1/2$. Thus, we have $\forall\sigma \in \{0,1\}$,

$$\Pr[H_3(n,\sigma)\big) = 1] = \frac{1}{2} \cdot \Pr[H_2(n,\sigma)\big) = 1] \pm \mathsf{negl}(n) \tag{5}$$

where the $\mathsf{negl}(n)$ term is due to the negligible possibility that R^* breaks the security of Step-5 coin-tossing.

We now finish the proof for computationally-hiding property by showing the following claim.

Claim 2.

$$\big|\Pr[H_3(n,1)\big) = 1] - \Pr[H_3(n,0) = 1]\big| \leq \mathsf{negl}(n) \tag{6}$$

Before presenting the proof for Claim 2, let us show why it closes the proof for computationally-hiding property of Protocol 2. First, note that Equation (2), (3), (4) and (5) imply that:

$$\begin{aligned}
& \big|\Pr[\mathcal{D}\big(\mathcal{V}^{R^*}(n,1)\big) = 1] - \Pr[\mathcal{D}\big(\mathcal{V}^{R^*}(n,0)\big) = 1]\big| \\
= \; & \big|\Pr[H_0(n,1) = 1] - \Pr[H_0(n,0) = 1]\big| \\
= \; & \big|\Pr[H_1(n,1) = 1] - \Pr[H_1(n,0) = 1] \pm \mathsf{negl}(n)\big| \\
= \; & 2 \cdot \big|\Pr[H_2(n,1) = 1] - \Pr[H_2(n,0) = 1] \pm \mathsf{negl}(n)\big| \\
= \; & 4 \cdot \big|\Pr[H_3(n,1) = 1] - \Pr[H_3(n,0) = 1] \pm \mathsf{negl}(n)\big| \\
\leq \; & 4 \cdot \big|\Pr[H_3(n,1) = 1] - \Pr[H_3(n,0) = 1]\big| \pm \mathsf{negl}(n)
\end{aligned} \tag{7}$$

Combining Inequality (7) with (6) proves Inequality (1), which finishes the proof for hiding property of Protocol 2.

In the following, we finish this part by presenting the proof for Claim 2.

Proof for Claim 2. First note that $\forall\sigma \in \{0,1\}$, $H_3(n,\sigma)$ does not need to know $d_{n,1-\mathsf{g}}$, the decommitment information for $c_{n,1-\mathsf{g}} = \mathsf{Com}(\eta \oplus \sigma)$. That is because once the query b of $S^{\hat{R}^*}$ or the value ch_n equals $1-\mathsf{g}$, the hybrid H_3 will simply abort. Therefore, if Inequality (6) does not hold, we can build a PPT machine $\mathcal{D}_{\mathsf{com}}$ that breaks the computationally-hiding property of Com. The distinguisher $\mathcal{D}_{\mathsf{com}}$ runs H_3 but define $c_{n,1-\mathsf{g}}$ in the following way:

- It forwards two values $m_0 \coloneqq \eta \oplus 0$ and $m_1 \coloneqq \eta \oplus 1$ to the outsider challenger for the hiding game of Com.
- Once it receives the commitment c^* from the challenger, it sets $c_{n,1-\mathsf{g}} = \mathsf{c}^*$.

Upon the halt of H_3, $\mathcal{D}_{\mathsf{com}}$ outputs whatever H_3 outputs. From the above description, it is easy to see that if the outside challenger commits to m_0, the view of R^* is identical to that in $H_3(n,0)$; if the outside challenger commits to m_0, the view of R^* is identical to that in $H_3(n,1)$. Therefore, if Equation (6) does not hold, $\mathcal{D}_{\mathsf{com}}$ breaks the computationally-hiding property of Com. This finishes the proof for Claim 2. □

Remark 6 (The position of the Step-4 OT.). If the Step-4 OT happens in parallel with Step 3, the hiding property can be proved using essentially the same hybrids as the above. The only place that requires extra attention is the ideal-world simulator $S^{\hat{R}^*}$, which is used in H_1, H_2 and H_3. Recall that the ideal-world simulator only interacts with $\hat{R}^*$ and does not take any other external message except for a single reply from $\mathcal{F}_{\mathsf{OT}}$ to its query. However, when Step-3 OT happens in parallel with Step 3, $S^{\hat{R}^*}$ needs to feed $\hat{R}^*$ other messages of the other $n-1$ OT instances (the original Step-3 OTs) to finish the simulation. So we need to modify S such that it forwards those messages between $\hat{R}^*$ and the running hybrid if $\hat{R}^*$ expects messages of the other OT instances. S may also need to rewind $\hat{R}^*$. In this case, S can simply reuse the same messages for the other OT instances in each rewinding. It can be easily verified that with this modification to $S^{\hat{R}^*}$, all the claims regarding the above hybrids still hold.

5.1.2 Straight-Line Extractability

At a high level, the extractor $\mathcal{E}$ works by biasing the outcome $\mathsf{ch} = r_1 \oplus r_2$ of the Step-5 coin-tossing, such that $\mathsf{ch}_i \oplus \tau_i = 1$ for all $i \in [n]$. In this case, $\mathcal{E}$ learns the decommitments to all the values $\{s_{i,b}\}_{i\in[n],b\in\{0,1\}}$ at the end of **Commit Phase**, thus being able to extract σ.

Extractor $\mathcal{E}$ works as follows:

1. $\mathcal{E}$ invokes C^* and interacts with it using the honest receiver strategy up to the beginning of Step 5.
2. In Step 5, $\mathcal{E}$ acts as follows:
   (a) $\mathcal{E}$ runs the VSS **Commit Phase** of Protocol 1 to generate $c_{r_1} = \mathsf{VSSCom}(0^n)$. R sends c_{r_1}.
   (b) $\mathcal{E}$ receives from C^* the value r_2.
   (c) $\mathcal{E}$ sends to C^* the value $r_1 \coloneqq r_2 \oplus (\overline{\tau}_1 \| \dots \| \overline{\tau}_n)$, where $\overline{\tau}_i = 1 \oplus \tau_i$ for $i \in [n]$.

(d) $\mathcal{E}$ invokes the (straight-line) simulator of Protocol 1, with the residual C^* as the verifier, for the (false) statement that c_{r_1} is a VSSCom commitment to the value r_1.

Note that the output of this (biased) coin-tossing is $\mathsf{ch} = r_1 \oplus r_2$, which equals $\overline{\tau}_1 \| \ldots \| \overline{\tau}_n$.

3. $\mathcal{E}$ receives from C^* the values $\{d_{i,\mathsf{ch}_i}\}$. It aborts if any of these decommitments are invalid; otherwise, it outputs $\sigma = \oplus_{i,b} s_{i,b}$. Note that if it does not abort, $\mathcal{E}$ learns all the $\{s_{i,b}\}$ values. Because it learns $\{s_{i,\tau_i}\}_{i\in[n]}$ from the OT executions in Step 3 and 4; it also learns $\{s_{i,\overline{\tau}_i}\}_{i\in[n]}$ from the Step-6 decommitments.

4. **Output:** $\mathcal{E}$ outputs C^*'s view of the above execution along with σ.

From the above description, it is clear that $\mathcal{E}$ runs in expected polynomial-time, because the simulator of Protocol 1 runs in expected polynomial time and all the remaining steps run in polynomial time. Also, if C^* does not abort, $\mathcal{E}$ will be able to extract the value σ. In the following, we show that C^*'s behavior (actually its view) will not change (up to negligible probability) between the real execution and its interaction with $\mathcal{E}$.

We now show that the view output by $\mathcal{E}$ is computationally indistinguishable from C^*'s view in a real execution through the following sequence of hybrids:

- **Hybrid** H_0**:** This hybrid runs the real execution $\langle C^*, R\rangle$ between the (malicious) committer C^* and the honest receiver R. At the end of the execution, H_1 outputs the view of C^*.
- **Hybrid** H_1**:** this hybrid is identical to the H_0 except that the zero-knowledge argument verified by C^* in Step 5-(d) is replaced by a simulated one using the simulator of Protocol 1.
- **Hybrid** H_2**:** this hybrid is identical to H_1 except that the commitment received by C^* in Step 5-(a) is to 0^n rather than to a random string r_1;
- **Hybrid** H_3**:** this hybrid is identical to H_2 except that the value r_1 in Step 5-(c) is set to $r_1 \coloneqq r_2 \oplus (\overline{\tau}_1 \| \ldots \| \overline{\tau}_n)$, where $\overline{\tau}_i = 1 \oplus \tau_i$. (Note that the output of this hybrid is identical to the view of C^* output by $\mathcal{E}$).

The computational indistinguishability between (the output of) H_0 and H_1 can be established by the $\mathcal{ZK}$ property of the proof stage of Protocol 1. The computational indistinguishability between H_1 and H_2 can be established by the hiding property of the committing stage of Protocol 1 (simply by forwarding the commitment on which these two hybrids differ to an outside challenger for the hiding property of VSSCom).

The computational indistinguishability between H_2 and H_3 can be established by standard hybrid arguments. More specifically, we consider the following intermediate hybrids:

- **Hybrids** H_3^i $(i \in [n])$**:** this hybrid is identical to H_3 except that the value r_1 in Step 5-(c) is set to $r_1 \coloneqq r_2 \oplus (\overline{\tau}_1 \| \ldots \| \overline{\tau}_i \| u_{i+1} \| \ldots \| u_n)$, where $\overline{\tau}_j = 1 \oplus \tau_j$ $(j \in [i])$ and u_k $(k \in \{i+1, \ldots, n\})$ is a random bit sampled independently.

Note that H_3^n is identical to H_3. We additional define $H_3^0 \coloneqq H_2$. Then the computational indistinguishability between H_3^{i-1} and H_3^i $(\forall i \in [n])$ follows from the security of OT against malicious senders. Concretely, the i-th OT execution is forwarded between C^* and an external OT challenger. If the view of C^* between H_3^{i-1} and H_3^i changes in a non-negligible way, the hybrid constitutes a PPT machine that tells the secret input of the challenger non-negligibly better than random guess. Thus, we have $H_0 \overset{c}{\approx} H_1 \overset{c}{\approx} H_2 \overset{c}{\approx} H_3$, which finishes the proof of extractability.

This finishes the proof of Theorem 4.

Remark 7 (Committing to Strings). One obvious approach to extend Protocol 2 to support multi-bit strings is to commit to each bit independently in parallel. A more efficient way is to replace the single-bit commitments in Step 2 and OTs in Step 3 and 4 with their multi-bit version. It is straightforward to see that same proof for correctness and security holds for this the multi-bit version.

6 Our Bounded-Concurrent OT Protocol

In this section, we prove the following theorem.

Theorem 5. *Assume the existence of constant-round semi-honest oblivious transfer protocols and collision-resistant hash functions. Let $\mathcal{F}_{\mathrm{OT}}$ be the ideal oblivious transfer functionality (Figure 1). Then, for every polynomial m, there exists a constant-round protocol that securely computes $\mathcal{F}_{\mathrm{OT}}$ under m-bounded concurrent composition, and it uses the underlying primitives in the black-box way.*

At a high-level, we obtain the OT claimed in Theorem 5 by replacing the statistically-binding commitment Com in the OT of [GKP18] (denoted as GKP-OT) with a new commitment based on Protocol 2. In the following, we will first describe the intuition behind our construction in Section 6.1, and then present our protocol in Section 6.2.

6.1 The High-Level Idea

As mentioned in the technical overview (Section 2.1), we want to employ the same simulation technique for GKP-OT, but with an (efficient) alternative way in which the simulator can "extract" the value committed in Com. Let us first recall the (only) two places where Com is used in GKP-OT:

1. In the very beginning (Stage 1), S (resp. R) uses Com to commit to a random set Γ_S (resp. Γ_R), which is used later for cut-and-chose.
2. Next, S (resp. R) uses Com to commit to a random string $\boldsymbol{a}^S$ (resp. $\boldsymbol{a}^R$) in the (Stage-2) coin tossing, which will later be used as inputs to the parallel execution of several random OT instances (which are in turned used for an OT-combiner stage later).

Since we now have the straight-line extractable commitment (Protocol 2) at our disposal, we may replace Com with Protocol 2. We notice that the GKP simulator $\mathcal{S}im_{\mathrm{OT}}$ can be extended to our setting by substituting the brute-forcing with the Protocol-2 extractor. However, this method requires us to insert many intermediate hybrids in carefully-chosen places as we need to ensure that the extractions happen in time, while not disturbing the adjacent hybrids. We thus take the following alternative approach.

Our Approach. We add a new step (called **Stage 0**) in the beginning of GKP-OT, where S (resp. R) commits using Protocol 2 to two random strings ϕ^S and ψ^S (resp. ϕ^R and ψ^R) of proper length. We then continue identically as in GKP-OT with the following modifications:

- In Stage 1, when S (resp. R) needs to commit to Γ_S (resp. Γ_R), he simply sends $\phi^S \oplus \Gamma_S$ (resp. $\phi^R \oplus \Gamma_R$);
- In Stage 2, when S (resp. R) needs to commit to $\boldsymbol{a}^S$ (resp. $\boldsymbol{a}^R$), he simply sends $\psi^S \oplus \boldsymbol{a}^S$ (resp. $\psi^R \oplus \boldsymbol{a}^R$).

Intuitively, we ask both parties commit to random strings which will later be used as one-time pads to "mask" the values they committed to by Com in the original GKP-OT. The hiding of Γ_S and $\boldsymbol{a}^S$ follows straightforwardly. To allow the simulator to extract them efficiently, it is sufficient to let $\mathcal{S}im_{\mathrm{OT}}$ use the extractor of Protocol 2 to extract ϕ^S and ψ^S. This can be done based on two important properties of the extractor for Protocol 2:

1. **Straight-line Extraction:** this guarantees that $\mathcal{S}im_{\mathrm{OT}}$ can finish the extraction efficiently, free of the exponential-time problem due to recursive rewinding (similar as that for concurrent zero-knowledges [DNS98]).
2. **Robustness:** since Protocol 2 is based on the ℓ-robust ZK (Section 4), its extractor inherits the ℓ-robustness. By setting the parameter ℓ carefully, we make sure that the simulator can switch from honest receiver's strategy to the extractor's strategy session by session, even in the presence of (bounded-ly) many other sessions.

Since we put the commitments to those masks in the very beginning, all the extractions can be done before further hybrids are defined. Similar arguments also apply when the receiver is corrupted. Therefore, we can make use of the GKP technique in a modular way to finish the proof of Theorem 5.

6.2 Protocol Description

In our protocol, we use the following building blocks.

- The robust-extractable commitment scheme defined in Protocol 2, to which we refer as RobCom. Note that the commitment protocol is based only on CRHFs and semi-honest OTs in a black-box manner.
- A four-round statistically-binding extractable commitment ExtCom, which can be constructed from one-way functions in the black-box way [Nao91, HILL99, PW09].
- A $O(1)$-round OT protocol mS-OT that is secure against malicious senders and semi-honest receivers.[13] As shown in [HIK+11], such a OT protocol can be obtained from any semi-honest one in the black-box way.
- A $O(1)$-round parallel non-malleable commitment NMCom that is parallel k-robust for sufficiently large constant k. (Concretely, we require that k is larger than the round complexity of the above three building blocks.) Such a non-malleable commitment scheme can be constructed from CRHFs in the black-box way [GKP18].

Our OT protocol Π_{OT} is described below. As explained in Section 2.1, (1) our protocol is based on the OT protocol of [GKP18], which roughly consists of coin-tossing, semi-honest OT, OT combiner, and cut-and-choose, and relies on non-malleable commitments to make sure adversary cannot setup the "trapdoor statement" to be true even in the bounded-concurrent setting; and (2) our protocol additionally uses a black-box "commit-and-prove" protocol that is ℓ-robust-$\mathcal{ZK}$ for a suitably large ℓ to commit a string and later prove in zero-knowledge that the opened value is indeed what was committed. Below, we give intuitive explanations in italic.

Parameters: The security parameter is n, and the bounded-concurrent composition parameter is $m := m(n)$.

[13] We only requires mS-OT to be secure under a game-based definition (which is preserved under parallel composition). For details, see the the proofs of Lemma 6 and Claim 8.

The ideal OT functionality $\mathcal{F}_{OT}$ interacts with a sender S and a receiver R.

- Upon receiving a message $(\mathsf{sid}, \mathsf{sender}, v_0, v_1)$ from S, where each $v_i \in \{0,1\}^n$, store (v_0, v_1).
- Upon receiving a message $(\mathsf{sid}, \mathsf{receiver}, u)$ from R, where $u \in \{0,1\}$, check if a $(\mathsf{sid}, \mathsf{sender}, \ldots)$ message was previously sent. If yes, send (sid, v_u) to R and (sid) to the adversary $\mathcal{S}im$ and halt. If not, send nothing to R.

Fig. 1. The oblivious transfer functionality $\mathcal{F}_{OT}$.

Inputs: The input to the sender S is $v_0, v_1 \in \{0,1\}^n$.The input to the receiver R is $u \in \{0,1\}$. The identities of S and R are $\mathsf{id}_S, \mathsf{id}_R$ respectively.

Stage 0: (Extractable Commitments to Randomness)

1. **Commitments to S's randomness.**
   (a) S samples independently two random strings ϕ^S and $\psi^S = \psi_1^S \| \ldots \| \psi_{11n}^S$ of proper length (see the comment at the end of this stage).
   (b) S and R involve in $11n+1$ executions of RobCom in parallel, where S commits to ϕ^S and $\psi_1^S, \ldots, \psi_{11n}^S$ respectively.

   Note that for the $\mathcal{ZK}$ argument in Step 5d of Protocol 2, we set the robustness parameter to be $\ell(n) = m \cdot \nu_{\text{OT}}(n)$ where ν_{OT} is defined towards the end. This **Proof Phase** includes the **long message** of Protocol 1. We call this message **receiver's long message**. (Note that although the sender commits in this step, the long message actually flows from the receiver to the sender. Thus, it is called the receiver's long message.)
2. **Commitments to R's randomness.**
   (a) R samples independently two random strings ϕ^R and $\psi^R = \psi_1^R \| \ldots \| \psi_{11n}^R$ of proper length (see the comment at the end of this stage).
   (b) S and R involve in $11n+1$ executions of RobCom in parallel, where R commits to ϕ^R and $\psi_1^R, \ldots, \| \psi_{11n}^R$ respectively.

   Note that for the $\mathcal{ZK}$ argument in Step 5d of Protocol 2, we set the robustness parameter to be $\ell(n) = m \cdot \nu_{\text{OT}}(n)$ where ν_{OT} is defined towards the end. This **Proof Phase** includes the **long message** of Protocol 1. We call this message **sender's long message**.

COMMENT: *In step 1 of this Stage, ϕ^S will be used in Stage 1-1 as a One-Time Pad to "mask" the sender's secrete Γ^S (which in turn is used as the sender's challenge for cut-and-choose). Similarly, ψ^S will be used in Stage 2-1 to mask the sender's secrete $\boldsymbol{a}^S$.*
Step 2 is just the symmetric execution of the same protocol where S and R exchange their role.

Stage 1: (Preprocess for cut-and-choose)

1. S samples a random subset $\Gamma_S := \{\gamma_1^S, ..., \gamma_n^S\} \subset [11n]$ of size n.[14] It then sends to R the value $\Gamma_S \oplus \phi^S$, i.e. the bit representation of Γ_S masked by the string ϕ^S using exclusive-or.
2. R samples a random subset $\Gamma_R := \{\gamma_1^R, ..., \gamma_n^R\} \subset [11n]$ of size n. It then sends to S the value $\Gamma_R \oplus \phi^R$.

[14] Note that Γ_S can be represented using a bit-string of length $11n$.

COMMENT: *As in the OT protocols of [LP12, GKP18], the subsets to for the cut-and-choose stages are committed in advance to prevent selective opening attacks.*

Stage 2: Coin-tossing for sub-protocols

1. **(Coin tossing for S)** S samples random strings $\boldsymbol{a}^S = (a_1^S, \ldots, a_{11n}^S)$. It then sends to R the values $z_i^S := a_i^S \oplus \psi_i^S$ for each $i \in [11n]$. Let d_i^S be the decommitments w.r.t. the Stage-0-1 RobCom of ϕ_i^S. R then sends random strings $\boldsymbol{b}^S = (b_1^S, \ldots, b_{11n}^S)$ to S. S then defines $\boldsymbol{r}^S = (r_1^S, \ldots, r_{11n}^S)$ by $r_i^S \stackrel{\text{def}}{=} a_i^S \oplus b_i^S$ for each $i \in [11n]$ and parses r_i^S as $s_{i,0} \,\|\, s_{i,1} \,\|\, \tau_i^S$ for each $i \in [11n]$.

2. **(Coin tossing for R)** R samples random strings $\boldsymbol{a}^R = (a_1^R, \ldots, a_{11n}^R)$. It then sends to S the values $z_i^R := a_i^R \oplus \psi_i^R$ for each $i \in [11n]$. Let d_i^R be the decommitments w.r.t. the Stage-0-2 RobCom of ϕ_i^R. S then sends random strings $\boldsymbol{b}^R = (b_1^R, \ldots, b_{11n}^R)$ to R. R then defines $\boldsymbol{r}^R = (r_1^R, \ldots, r_{11n}^R)$ by $r_i^R \stackrel{\text{def}}{=} a_i^R \oplus b_i^R$ for each $i \in [11n]$ and parses r_i^R as $c_i \,\|\, \tau_i^R$ for each $i \in [11n]$.

Stage 3: (mS-OTs with random inputs)

S and R execute $11n$ instances of mS-OT in parallel. In the i-th instance, S uses $(s_{i,0}, s_{i,1})$ as the input and τ_i^S as the randomness, and R uses c_i as the input and τ_i^R as the randomness, where $\{s_{i,0}, s_{i,1}, \tau_i^S\}_i$ and $\{c_i, \tau_i^R\}_i$ are the random coins that were obtained in Stage 2. The output to R is denoted by $\widetilde{s}_1, \ldots, \widetilde{s}_{11n}$, which are supposed to be equal to $s_{1,c_1}, \ldots, s_{11n,c_{11n}}$.

Stage 4: (NMCom and ExtCom for checking honesty of R)

1. R commits to $(a_1^R, d_1^R), \ldots (a_{11n}^R, d_{11n}^R)$ using NMCom and identity id_R. Let $e_1^R, \ldots, e_{11n}^R$ be the decommitments.

2. R commits to $(a_1^R, d_1^R, e_1^R), \ldots (a_{11n}^R, d_{11n}^R, e_{11n}^R)$ using ExtCom.

COMMENT: *Roughly, the commitments in this stage, along with the cut-and-choose in the next stage, will be used in the security proof to argue that even cheating R must behave honestly in most instances of* mS-OT *in Stage 3. A key point is that given the values that are committed to in* NMCom *or* ExtCom *in this stage, one can obtain the random coins that R obtained in Stage 2 and thus can check whether R behaved honestly in Stage 3.*

Stage 5: (Cut-and-choose against R)

1. S reveals Γ_S by sending ϕ^S and the decommitment information w.r.t. Stage-0-1 RobCom of ϕ^S.

2. For every $i \in \Gamma_S$, R reveals (a_i^R, d_i^R, e_i^R) by decommitting the i-th ExtCom commitment in Stage 4.

3. For every $i \in \Gamma_S$, S checks the following.

   (a) $((a_i^R, d_i^R), e_i^R)$ is a valid decommitment of the i-th NMCom commitment in Stage 4.

   (b) d_i^R is a valid decommitment of ψ_i^R w.r.t. Stage-0-1 RobCom, and $a_i^R \oplus \psi_i^R$ equals the value z_i^R it received in Stage-2-1.

   (c) R executed the i-th mS-OT in Stage 3 honestly using $c_i \,\|\, \tau_i^R$, which is obtained from $r_i^R = a_i^R \oplus b_i^R$ as specified by the protocol.

COMMENT: *In other words, for each index that it randomly selected in Stage 1, S checks whether R behaved honestly in Stages 3 and 4 on that index.*

Stage 6: (OT combiner) Let $\Delta := [11n] \setminus \Gamma_S$.

1. R sends $\alpha_i := u \oplus c_i$ to S for every $i \in \Delta$.
2. S computes a $(6n+1)$-out-of-$10n$ secret sharing of v_0, denoted by $\boldsymbol{\rho}_0 = (\rho_{0,i})_{i\in\Delta}$, and computes a $(6n+1)$-out-of-$10n$ secret sharing of v_1, denoted by $\boldsymbol{\rho}_1 = (\rho_{1,i})_{i\in\Delta}$. Then, S sends $\beta_{b,i} := \rho_{b,i} \oplus s_{i,b\oplus\alpha_i}$ to R for every $i \in \Delta$, $b \in \{0,1\}$.
3. R computes $\widetilde{\rho}_i := \beta_{u,i} \oplus \widetilde{s}_i$ for every $i \in \Delta$. Let $\widetilde{\boldsymbol{\rho}} := (\widetilde{\rho}_i)_{i\in\Delta}$.

COMMENT: *In this stage, S and R execute OT with their true inputs by using the outputs of* mS-OT *in Stage 3. Roughly speaking, this stage is secure as long as most instances of* mS-OT *in Stage 3 are correctly executed.*

Stage 7: (NMCom and ExtCom for checking honesty of S)

1. S commits to $(a_1^S, d_1^S), \ldots (a_{11n}^S, d_{11n}^S)$ using NMCom and identity id_S. Let $e_1^S, \ldots, e_{11n}^S$ be the decommitments.
2. S commits to $(a_1^S, d_1^S, e_1^S), \ldots (a_{11n}^S, d_{11n}^S, e_{11n}^S)$ using ExtCom.

Stage 8: (Cut-and-choose against S)

1. R reveals Γ_R by sending ϕ^R and the decommitment information w.r.t. Stage-0-2 RobCom of ϕ^R.
2. For every $i \in \Gamma_R$, S reveals (a_i^S, d_i^S, e_i^S) by decommitting the i-th ExtCom commitment in Stage 7.
3. For every $i \in \Gamma_R$, R checks the following.
   (a) $((a_i^S, d_i^S), e_i^S)$ is a valid decommitment of the i-th NMCom commitment in Stage 7.
   (b) d_i^S is a valid decommitment of ψ_i^S w.r.t. Stage-0-2 RobCom, and $a_i^S \oplus \psi_i^S$ equals the value z_i^S it received in Stage-2-2.
   (c) S executed the i-th mS-OT in Stage 3 honestly using $s_{i,0} \,\|\, s_{i,1} \,\|\, \tau_i^S$, which is obtained from $r_i^S = a_i^S \oplus b_i^S$ as specified by the protocol.

Parameter ν_{OT}: All messages of this OT protocol except the sender's and receiver's long messages are called **short messages**. Then, $\nu_{\mathrm{OT}}(n)$ denotes the total length of all short messages of this protocol.

Output: R outputs $\mathsf{Value}(\widetilde{\boldsymbol{\rho}}, \Gamma_R \cap \Delta)$, where $\mathsf{Value}(\cdot,\cdot)$ is the function that is defined in Fig. 2.
COMMENT: *As in the OT protocols of [LP12, GKP18], a carefully designed reconstruction procedure* $\mathsf{Value}(\cdot,\cdot)$ *is used here so that the simulator can extract correct implicit inputs from cheating S by obtaining sharing that is sufficiently "close" to $\widetilde{\boldsymbol{\rho}}$.*

6.3 Security Proof

The security proof for our OT protocol is similar to that of [GKP18], except that we substitute the "brute-force" extraction of the simulator with polynomial-time straight-line extractions to learn

Reconstruction Procedure $\mathsf{Value}(\cdot,\cdot)$**:** For a sharing $\boldsymbol{s} = (s_i)_{i\in\Delta}$ and a set $\Theta \subset \Delta$, the output of $\mathsf{Value}(\boldsymbol{s},\Theta)$ is computed as follows. If $\boldsymbol{s}$ is 0.9-close to a valid codeword $\boldsymbol{w} = (w_i)_{i\in\Delta}$ that satisfies $s_i = w_i$ for every $i \in \Theta$, then $\mathsf{Value}(\boldsymbol{s},\Theta)$ is the value decoded from $\boldsymbol{w}$; otherwise, $\mathsf{Value}(\boldsymbol{s},\Theta) = \perp$.

Fig. 2. The function $\mathsf{Value}(\cdot,\cdot)$.

the adversary's secrets. As mentioned in the technical overview part, our modification does not introduce new malleability issues, and the session-by-session substitution in the hybrids of [GKP18] will still apply (with careful modification). We give the full security proof in Section B.

7 Our Bounded-Concurrent MPC Protocol

In this section, we prove the following theorem.

Theorem 6. *Assume the existence of constant-round semi-honest oblivious transfer protocols and collision-resistant hash functions. Let $\mathcal{F}$ be any well-formed functionality. Then, for every polynomial m, there exists a constant-round protocol that securely computes $\mathcal{F}$ under m-bounded concurrent composition; furthermore, it uses the underlying primitives in the black-box way.*

The protocol and the proofs are identical to those in [GKP18] except that we use the bounded-concurrent secure OT protocol described in previous section. We now provide more details. We focus on the two-party case below (the MPC case is analogous).

Protocol Description. Roughly speaking, we obtain our bounded-concurrent 2PC protocol by composing our bounded-concurrent OT protocol in Section 6 with a UC-secure OT-hybrid 2PC protocol. Concretely, let Π_{OT} be our ℓ-bounded-concurrent OT protocol in Section 6, and $\Pi_{\mathrm{2PC}}^{\mathcal{F}_{\mathrm{OT}}}$ be a UC-secure OT-hybrid 2PC protocol with the following property: The two parties use the OT functionality $\mathcal{F}_{OT}$ only at the beginning of the protocol, and they send only randomly chosen inputs to $\mathcal{F}_{OT}$. Then, we obtain our bounded-concurrent 2PC protocol Π_{2PC} by replacing each invocation of $\mathcal{F}_{OT}$ in $\Pi_{\mathrm{2PC}}^{\mathcal{F}_{\mathrm{OT}}}$ with an execution of Π_{OT} (i.e., the two parties execute Π_{OT} instead of calling to $\mathcal{F}_{OT}$), where all the executions of Π_{OT} are carried out in a synchronous manner, i.e., in a manner that the first message of all the executions are sent before the second message of any execution is sent etc.; furthermore, the bounded-concurrency parameter for Π_{OT} is set to be m' defined as follows: let ν_{2PC} denote the length of all messages of the hybrid 2PC protocol $\Pi_{\mathrm{2PC}}^{\mathcal{F}_{\mathrm{OT}}}$ protocol (which does not include the length of messages corresponding to OT calls since we are in the hybrid model). Then, we set m' so that the length ℓ of long messages of Π_{OT} would be n bits longer than $\nu_{\mathrm{OT}} + \nu_{\mathrm{2PC}}$. This can be ensured by setting $m' = a \cdot m$ where a is the smallest integer that is bigger than $\max(\nu_{\mathrm{OT}}/\nu_{\mathrm{2PC}}, \nu_{\mathrm{2PC}}/\nu_{\mathrm{OT}})$.

As the UC-secure OT-hybrid 2PC protocol, we use the constant-round 2PC (actually, MPC) protocol of Ishai et al. [IPS08], which makes only black-box use of pseudorandom generators (which in turn can be obtained in the black-box way from any semi-honest OT protocol). (The protocol of Ishai et al. [IPS08] itself does not satisfy the above property, but as shown in [GKP18], it can be easily modified to satisfy it.) Since the OT-hybrid protocol of Ishai et al. [IPS08] (as well as its modification in [GKP18]) is a black-box construction and has only constant number of rounds, our protocol Π_{2PC} is also a black-box construction and has only constant number of rounds.

The security of this protocol can be proved in a similar way as our OT protocol. The formal proof is given in Section D.

References

ABG+20. Benny Applebaum, Zvika Brakerski, Sanjam Garg, Yuval Ishai, and Akshayaram Srinivasan. Separating two-round secure computation from oblivious transfer. In *11th Innovations in Theoretical Computer Science Conference (ITCS 2020)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2020.

Bar01. Boaz Barak. How to go beyond the black-box simulation barrier. In *42nd FOCS*, pages 106–115. IEEE Computer Society Press, October 2001.

Bar02. Boaz Barak. Constant-round coin-tossing with a man in the middle or realizing the shared random string model. In *43rd FOCS*, pages 345–355. IEEE Computer Society Press, November 2002.

BDH+17. Brandon Broadnax, Nico Döttling, Gunnar Hartung, Jörn Müller-Quade, and Matthias Nagel. Concurrently composable security with shielded super-polynomial simulators. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *EUROCRYPT 2017, Part I*, volume 10210 of *LNCS*, pages 351–381. Springer, Heidelberg, April / May 2017.

Bea91. Donald Beaver. Foundations of secure interactive computing. In *CRYPTO*, pages 377–391, 1991.

BGH+04. Eli Ben-Sasson, Oded Goldreich, Prahladh Harsha, Madhu Sudan, and Salil P. Vadhan. Robust PCPs of proximity, shorter PCPs and applications to coding. In László Babai, editor, *36th ACM STOC*, pages 1–10. ACM Press, June 2004.

BGJ+17. Saikrishna Badrinarayanan, Vipul Goyal, Abhishek Jain, Dakshita Khurana, and Amit Sahai. Round optimal concurrent MPC via strong simulation. In Yael Kalai and Leonid Reyzin, editors, *TCC 2017, Part I*, volume 10677 of *LNCS*, pages 743–775. Springer, Heidelberg, November 2017.

BL02. Boaz Barak and Yehuda Lindell. Strict polynomial-time in simulation and extraction. In *34th ACM STOC*, pages 484–493. ACM Press, May 2002.

BP12. Nir Bitansky and Omer Paneth. From the impossibility of obfuscation to a new non-black-box simulation technique. In *53rd FOCS*, pages 223–232. IEEE Computer Society Press, October 2012.

BP13. Nir Bitansky and Omer Paneth. On the impossibility of approximate obfuscation and applications to resettable cryptography. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *45th ACM STOC*, pages 241–250. ACM Press, June 2013.

BS05. Boaz Barak and Amit Sahai. How to play almost any mental game over the net - concurrent composition via super-polynomial simulation. In *46th FOCS*, pages 543–552. IEEE Computer Society Press, October 2005.

Can00. Ran Canetti. Security and composition of multiparty cryptographic protocols. *Journal of Cryptology*, 13(1):143–202, January 2000.

Can01. Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *42nd FOCS*, pages 136–145. IEEE Computer Society Press, October 2001.

CDMW09. Seung Geol Choi, Dana Dachman-Soled, Tal Malkin, and Hoeteck Wee. Simple, black-box constructions of adaptively secure protocols. In Omer Reingold, editor, *TCC 2009*, volume 5444 of *LNCS*, pages 387–402. Springer, Heidelberg, March 2009.

CDSMW17. Seung Geol Choi, Dana Dachman-Soled, Tal Malkin, and Hoeteck Wee. A black-box construction of non-malleable encryption from semantically secure encryption. *Journal of Cryptology*, Mar 2017.

CHH+07. Ronald Cramer, Goichiro Hanaoka, Dennis Hofheinz, Hideki Imai, Eike Kiltz, Rafael Pass, abhi shelat, and Vinod Vaikuntanathan. Bounded CCA2-secure encryption. In Kaoru Kurosawa, editor, *ASIACRYPT 2007*, volume 4833 of *LNCS*, pages 502–518. Springer, Heidelberg, December 2007.

CKL03. Ran Canetti, Eyal Kushilevitz, and Yehuda Lindell. On the limitations of universally composable two-party computation without set-up assumptions. In Eli Biham, editor, *EUROCRYPT 2003*, volume 2656 of *LNCS*, pages 68–86. Springer, Heidelberg, May 2003.

CLP10. Ran Canetti, Huijia Lin, and Rafael Pass. Adaptive hardness and composable security in the plain model from standard assumptions. In *51st FOCS*, pages 541–550. IEEE Computer Society Press, October 2010.

CLP20a. Rohit Chatterjee, Xiao Liang, and Omkant Pandey. Improved Black-Box Constructions of Composable Secure Computation. In Artur Czumaj, Anuj Dawar, and Emanuela Merelli, editors, *47th International Colloquium on Automata, Languages, and Programming (ICALP 2020)*, volume 168 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 28:1–28:20, Dagstuhl, Germany, 2020. Schloss Dagstuhl–Leibniz-Zentrum für Informatik.

CLP20b. Rohit Chatterjee, Xiao Liang, and Omkant Pandey. Improved black-box constructions of composable secure computation. Cryptology ePrint Archive, Report 2020/494, 2020. https://eprint.iacr.org/2020/494.

CPS13. Kai-Min Chung, Rafael Pass, and Karn Seth. Non-black-box simulation from one-way functions and applications to resettable security. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *45th ACM STOC*, pages 231–240. ACM Press, June 2013.

DMRV13. Dana Dachman-Soled, Tal Malkin, Mariana Raykova, and Muthuramakrishnan Venkitasubramaniam. Adaptive and concurrent secure computation from new adaptive, non-malleable commitments. In Kazue Sako and Palash Sarkar, editors, *ASIACRYPT 2013, Part I*, volume 8269 of *LNCS*, pages 316–336. Springer, Heidelberg, December 2013.

DNS98. Cynthia Dwork, Moni Naor, and Amit Sahai. Concurrent zero-knowledge. In *30th ACM STOC*, pages 409–418. ACM Press, May 1998.

FS90. Uriel Feige and Adi Shamir. Witness indistinguishable and witness hiding protocols. In *22nd ACM STOC*, pages 416–426. ACM Press, May 1990.

GGJS12. Sanjam Garg, Vipul Goyal, Abhishek Jain, and Amit Sahai. Concurrently secure computation in constant rounds. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 99–116. Springer, Heidelberg, April 2012.

GGS15. Vipul Goyal, Divya Gupta, and Amit Sahai. Concurrent secure computation via non-black box simulation. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *CRYPTO 2015, Part II*, volume 9216 of *LNCS*, pages 23–42. Springer, Heidelberg, August 2015.

GJ13. Vipul Goyal and Abhishek Jain. On concurrently secure computation in the multiple ideal query model. In Thomas Johansson and Phong Q. Nguyen, editors, *EUROCRYPT 2013*, volume 7881 of *LNCS*, pages 684–701. Springer, Heidelberg, May 2013.

GK90. Oded Goldreich and Hugo Krawczyk. On the composition of zero-knowledge proof systems. In *Automata, Languages and Programming, 17th International Colloquium, ICALP*, pages 268–282, 1990.

GKOV12. Sanjam Garg, Abishek Kumarasubramanian, Rafail Ostrovsky, and Ivan Visconti. Impossibility results for static input secure computation. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 424–442. Springer, Heidelberg, August 2012.

GKP18. Sanjam Garg, Susumu Kiyoshima, and Omkant Pandey. A new approach to black-box concurrent secure computation. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part II*, volume 10821 of *LNCS*, pages 566–599. Springer, Heidelberg, April / May 2018.

GL91. Shafi Goldwasser and Leonid A. Levin. Fair computation of general functions in presence of immoral majority. In Alfred J. Menezes and Scott A. Vanstone, editors, *CRYPTO'90*, volume 537 of *LNCS*, pages 77–93. Springer, Heidelberg, August 1991.

GL02. Shafi Goldwasser and Yehuda Lindell. Secure computation without agreement. In *DISC*, pages 17–32, 2002.

GLOV12. Vipul Goyal, Chen-Kuei Lee, Rafail Ostrovsky, and Ivan Visconti. Constructing non-malleable commitments: A black-box approach. In *53rd FOCS*, pages 51–60. IEEE Computer Society Press, October 2012.

GLP+15. Vipul Goyal, Huijia Lin, Omkant Pandey, Rafael Pass, and Amit Sahai. Round-efficient concurrently composable secure computation via a robust extraction lemma. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *TCC 2015, Part I*, volume 9014 of *LNCS*, pages 260–289. Springer, Heidelberg, March 2015.

GM00. Juan A. Garay and Philip D. MacKenzie. Concurrent oblivious transfer. In *41st FOCS*, pages 314–324. IEEE Computer Society Press, November 2000.

GMW87. Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or A completeness theorem for protocols with honest majority. In Alfred Aho, editor, *19th ACM STOC*, pages 218–229. ACM Press, May 1987.

GOSV14. Vipul Goyal, Rafail Ostrovsky, Alessandra Scafuro, and Ivan Visconti. Black-box non-black-box zero knowledge. In David B. Shmoys, editor, *46th ACM STOC*, pages 515–524. ACM Press, May / June 2014.

Goy11. Vipul Goyal. Constant round non-malleable protocols using one way functions. In Lance Fortnow and Salil P. Vadhan, editors, *43rd ACM STOC*, pages 695–704. ACM Press, June 2011.

Goy12. Vipul Goyal. Positive results for concurrently secure computation in the plain model. In *53rd FOCS*, pages 41–50. IEEE Computer Society Press, October 2012.

Hai08. Iftach Haitner. Semi-honest to malicious oblivious transfer - the black-box way. In Ran Canetti, editor, *TCC 2008*, volume 4948 of *LNCS*, pages 412–426. Springer, Heidelberg, March 2008.

HIK+11. Iftach Haitner, Yuval Ishai, Eyal Kushilevitz, Yehuda Lindell, and Erez Petrank. Black-box constructions of protocols for secure computation. *SIAM J. Comput.*, 40(2):225–266, 2011.

HILL99. Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28(4):1364–1396, 1999.

Hof11. Dennis Hofheinz. Possibility and impossibility results for selective decommitments. *Journal of Cryptology*, 24(3):470–516, 2011.

HV16. Carmit Hazay and Muthuramakrishnan Venkitasubramaniam. Composable adaptive secure protocols without setup under polytime assumptions. In Martin Hirt and Adam D. Smith, editors, *TCC 2016-B, Part I*, volume 9985 of *LNCS*, pages 400–432. Springer, Heidelberg, October / November 2016.

IKLP06. Yuval Ishai, Eyal Kushilevitz, Yehuda Lindell, and Erez Petrank. Black-box constructions for secure computation. In Jon M. Kleinberg, editor, *38th ACM STOC*, pages 99–108. ACM Press, May 2006.

IPS08. Yuval Ishai, Manoj Prabhakaran, and Amit Sahai. Founding cryptography on oblivious transfer - efficiently. In David Wagner, editor, *CRYPTO 2008*, volume 5157 of *LNCS*, pages 572–591. Springer, Heidelberg, August 2008.

Kiy14. Susumu Kiyoshima. Round-efficient black-box construction of composable multi-party computation. In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO 2014, Part II*, volume 8617 of *LNCS*, pages 351–368. Springer, Heidelberg, August 2014.

KMO14. Susumu Kiyoshima, Yoshifumi Manabe, and Tatsuaki Okamoto. Constant-round black-box construction of composable multi-party computation protocol. In Yehuda Lindell, editor, *TCC 2014*, volume 8349 of *LNCS*, pages 343–367. Springer, Heidelberg, February 2014.

KO04. Jonathan Katz and Rafail Ostrovsky. Round-optimal secure two-party computation. In Matthew Franklin, editor, *CRYPTO 2004*, volume 3152 of *LNCS*, pages 335–354. Springer, Heidelberg, August 2004.

Lin03. Yehuda Lindell. Bounded-concurrent secure two-party computation without setup assumptions. In *35th ACM STOC*, pages 683–692. ACM Press, June 2003.

Lin04. Yehuda Lindell. Lower bounds for concurrent self composition. In Moni Naor, editor, *TCC 2004*, volume 2951 of *LNCS*, pages 203–222. Springer, Heidelberg, February 2004.

Lin13. Yehuda Lindell. A note on constant-round zero-knowledge proofs of knowledge. *Journal of Cryptology*, 26(4):638–654, October 2013.

LP09. Huijia Lin and Rafael Pass. Non-malleability amplification. In Michael Mitzenmacher, editor, *41st ACM STOC*, pages 189–198. ACM Press, May / June 2009.

LP12. Huijia Lin and Rafael Pass. Black-box constructions of composable protocols without set-up. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 461–478. Springer, Heidelberg, August 2012.

LPV08. Huijia Lin, Rafael Pass, and Muthuramakrishnan Venkitasubramaniam. Concurrent non-malleable commitments from any one-way function. In Ran Canetti, editor, *TCC 2008*, volume 4948 of *LNCS*, pages 571–588. Springer, Heidelberg, March 2008.

LPV09. Huijia Lin, Rafael Pass, and Muthuramakrishnan Venkitasubramaniam. A unified framework for concurrent security: universal composability from stand-alone non-malleability. In Michael Mitzenmacher, editor, *41st ACM STOC*, pages 179–188. ACM Press, May / June 2009.

MMY06. Tal Malkin, Ryan Moriarty, and Nikolai Yakovenko. Generalized environmental security from number theoretic assumptions. In Shai Halevi and Tal Rabin, editors, *TCC 2006*, volume 3876 of *LNCS*, pages 343–359. Springer, Heidelberg, March 2006.

MPR06. Silvio Micali, Rafael Pass, and Alon Rosen. Input-indistinguishable computation. In *47th FOCS*, pages 367–378. IEEE Computer Society Press, October 2006.

MR92. Silvio Micali and Phillip Rogaway. Secure computation (abstract). In Joan Feigenbaum, editor, *CRYPTO'91*, volume 576 of *LNCS*, pages 392–404. Springer, Heidelberg, August 1992.

Nao91. Moni Naor. Bit commitment using pseudorandomness. *Journal of Cryptology*, 4(2):151–158, January 1991.

ORS15. Rafail Ostrovsky, Silas Richelson, and Alessandra Scafuro. Round-optimal black-box two-party computation. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *CRYPTO 2015, Part II*, volume 9216 of *LNCS*, pages 339–358. Springer, Heidelberg, August 2015.

OSV15. Rafail Ostrovsky, Alessandra Scafuro, and Muthuramakrishnan Venkitasubramaniam. Resettably sound zero-knowledge arguments from OWFs - the (semi) black-box way. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *TCC 2015, Part I*, volume 9014 of *LNCS*, pages 345–374. Springer, Heidelberg, March 2015.

Pas03. Rafael Pass. Simulation in quasi-polynomial time, and its application to protocol composition. In Eli Biham, editor, *EUROCRYPT 2003*, volume 2656 of *LNCS*, pages 160–176. Springer, Heidelberg, May 2003.

Pas04. Rafael Pass. Bounded-concurrent secure multi-party computation with a dishonest majority. In László Babai, editor, *36th ACM STOC*, pages 232–241. ACM Press, June 2004.

PLV12. Rafael Pass, Huijia Lin, and Muthuramakrishnan Venkitasubramaniam. A unified framework for UC from only OT. In Xiaoyun Wang and Kazue Sako, editors, *ASIACRYPT 2012*, volume 7658 of *LNCS*, pages 699–717. Springer, Heidelberg, December 2012.

PR03. Rafael Pass and Alon Rosen. Bounded-concurrent secure two-party computation in a constant number of rounds. In *44th FOCS*, pages 404–415. IEEE Computer Society Press, October 2003.

PS04. Manoj Prabhakaran and Amit Sahai. New notions of security: Achieving universal composability without trusted setup. In László Babai, editor, *36th ACM STOC*, pages 242–251. ACM Press, June 2004.

PW09. Rafael Pass and Hoeteck Wee. Black-box constructions of two-party protocols from one-way functions. In Omer Reingold, editor, *TCC 2009*, volume 5444 of *LNCS*, pages 403–418. Springer, Heidelberg, March 2009.

PW11. Chris Peikert and Brent Waters. Lossy trapdoor functions and their applications. *SIAM Journal on Computing*, 40(6):1803–1844, 2011.

Ven14. Muthuramakrishnan Venkitasubramaniam. On adaptively secure protocols. In Michel Abdalla and Roberto De Prisco, editors, *SCN 14*, volume 8642 of *LNCS*, pages 455–475. Springer, Heidelberg, September 2014.

Wee10. Hoeteck Wee. Black-box, round-efficient secure computation via non-malleability amplification. In *51st FOCS*, pages 531–540. IEEE Computer Society Press, October 2010.

Yao86. Andrew Chi-Chih Yao. How to generate and exchange secrets (extended abstract). In *27th FOCS*, pages 162–167. IEEE Computer Society Press, October 1986.

Appendix

A Additional Preliminaries

A.1 Shamir's Secret Sharing

We first recall Shamir's secret sharing scheme. (In this paper, we use only the $(6n+1)$-out-of-$10n$ version of it.) To compute a $(6n+1)$-out-of-$10n$ secret sharing $\boldsymbol{s} = (s_1, \ldots, s_{10n})$ of a value $v \in GF(2^n)$, we choose random $a_1, \ldots, a_{6n} \in GF(2^n)$, let $p(z) \stackrel{\text{def}}{=} v + a_1 z + \cdots + a_{6n} z^{6n}$, and set $s_i := p(i)$ for each $i \in [10n]$. Given $\boldsymbol{s}$, we can recover v by obtaining polynomial $p(\cdot)$ through interpolation and then computing $p(0)$. We use $\mathsf{Decode}(\cdot)$ to denote the function that recovers v from $\boldsymbol{s}$ as above.

For any positive real number $x \leq 1$ and any $\boldsymbol{s} = (s_1, \ldots, s_{10n})$ and $\boldsymbol{s'} = (s'_1, \ldots, s'_{10n})$, we say that $\boldsymbol{s}$ and $\boldsymbol{s'}$ are x-*close* if $|\{i \in [10n] \text{ s.t. } s_i = s'_i\}| \geq x \cdot 10n$. If $\boldsymbol{s}$ and $\boldsymbol{s'}$ are not x-close, we say that they are $(1-x)$-*far*. Since the shares generated by $(6n+1)$-out-of-$10n$ Shamir's secret sharing scheme are actually a codeword of the Reed-Solomon code with minimum relative distance 0.4, if a (possibly incorrectly generated) sharing $\boldsymbol{s}$ is 0.8-close to a valid codeword $\boldsymbol{w}$, we can recover $\boldsymbol{w}$ from $\boldsymbol{s}$ efficiently by using, for example, the Berlekamp-Welch algorithm.

A.2 Commitment Schemes

Recall that a commitment scheme is a two-party protocol between a *committer* and a *receiver*. We say that a commitment is *accepting* if the receiver does not abort in the commit phase, and *valid* if there exists a value to which the commitment can be decommitted (i.e., if there exists a decommitment that the verifier accepts in the decommit phase). The *committed value* of a commitment is the value to which the commitment can be decommitted. We define the committed value of an invalid commitment as $\perp$.

There exists a two-round statistically binding commitment scheme Com based on one-way functions [Nao91, HILL99], and it uses the underlying one-way function in a black-box way.

A.3 Extractable Commitment Schemes

We next recall the definition of *extractable commitment schemes* from [PW09]. Roughly speaking, a commitment scheme is *extractable* if there exists an expected polynomial-time oracle machine, an *extractor*, E such that for any adversarial committer C^* that gives a commitment to honest receiver, the extractor E^{C^*} extracts the committed value of the commitment from C^* as long as the commitment is valid. We note that when the commitment is invalid, E can output an arbitrary garbage value; this is called *over-extraction*.

Formally, extractable commitment schemes are defined as follows.

Definition 5 (Extractable Commitment). *A commitment scheme $\langle C, R \rangle$ is* extractable *if there exists an expected polynomial-time extractor E such that for any* PPT *committer C^*, the extractor E^{C^*} outputs a pair (τ, σ) that satisfies the following properties.*

- *τ is identically distributed with the view of C^* that interacts with an honest receiver R in the commit phase of $\langle C, R \rangle$. Let c_τ be the commitment that C^* gives in τ.*
- *If c_τ is accepting, then $\sigma \neq \perp$ except with negligible probability.*

– *If $\sigma \neq \perp$, then it is statistically impossible to decommit c_τ to any value other than σ.*

There exists a four-round extractable commitment scheme ExtCom based on one-way functions [PW09], and it uses the underlying one-way function in a black-box way. Furthermore, ExtCom satisfies extractability in a stronger sense: It is extractable even against adversarial committers that give polynomially many ExtCom commitments *in parallel*. (The extractor outputs $(\tau, \sigma_1, \sigma_2, \ldots)$ for such committers.)

B Security Proof For Our OT Protocol

B.1 Simulator $\mathcal{S}im_{\mathrm{OT}}$

To prove the security of Π_{OT}, we consider the following simulator $\mathcal{S}im_{\mathrm{OT}}$. Recall that our goal is to prove that Π_{OT} securely realizes $\mathcal{F}_{OT}$ (see Fig. 1) under m-bounded concurrent (self) composition. We therefore consider a simulator that works against adversaries that participate in at most (and w.l.o.g., exactly) m sessions of Π_{OT} both as senders and as receivers.

Let $\mathcal{A}$ be any adversary that participates in m sessions of Π_{OT}. Our simulator $\mathcal{S}im_{\mathrm{OT}}$ internally invokes $\mathcal{A}$ and simulates each of the sessions for $\mathcal{A}$ as follows.

When R is corrupted: In a session where the receiver R is corrupted, $\mathcal{S}im_{\mathrm{OT}}$ simulates the sender S for $\mathcal{A}$ by extracting the implicit input $u^* \in \{0,1\}$ from $\mathcal{A}$. During the simulation, $\mathcal{S}im_{\mathrm{OT}}$ extracts the values ϕ^R and ψ^R (in straight-line, using the code of $\mathcal{A}$) such that it can later extract the value Γ^R and $\boldsymbol{a}^R$ in Stages 1 and 2 ; the former extraction is needed to execute most instances $\mathsf{mS\text{-}OT}$ in Stage 3 with true randomness (which is crucial to use their security in the analysis), and the latter extraction is needed to infer what information $\mathcal{A}$ obtained in the $\mathsf{mS\text{-}OT}$ instances in Stage 3 (which is crucial to extract the implicit input $u^* \in \{0,1\}$ from $\mathcal{A}$).

Concretely, $\mathcal{S}im_{\mathrm{OT}}$ simulates all steps of Stage 0 in the same way as an honest S, except that in Stage 0-2, $\mathcal{S}im_{\mathrm{OT}}$ uses the strategy of the (straight-line) extractor for Protocol 2 in its interaction with $\mathcal{A}$. At the end of this Stage 0-2, it learns the value ϕ^R and $\psi^R = \psi_1^R \| \ldots \| \psi_{11n}^R$ committed by $\mathcal{A}$.

Remark 8. Note that $\mathcal{S}im_{\mathrm{OT}}$ can extract Γ_R using ϕ^R in Stage 1-2. Likewise, in Stage 2-2 it can extract $\boldsymbol{a}^R$ using relevant parts of ψ^R.

Next, from Stage 1 to Stage 5, $\mathcal{S}im_{\mathrm{OT}}$ interacts with $\mathcal{A}$ in the same way as an honest S except for the following.

– For the commitments from $\mathcal{A}$ in Stages 1-2 and 2-2, the committed subset Γ_R and the committed strings $\boldsymbol{a}^R = (a_1^R, \ldots, a_{11n}^R)$ are extracted by $\mathcal{S}im_{\mathrm{OT}}$ as described in Remark 8.

  $\mathcal{S}im_{\mathrm{OT}}$ then defines $\boldsymbol{r}^R = (r_1^R, \ldots, r_{11n}^R)$ by $r_i^R \stackrel{\text{def}}{=} a_i^R \oplus b_i^R$ for each $i \in [11n]$ and parses r_i^R as $c_i \| \tau_i^R$ for each $i \in [11n]$. (Notice that $\boldsymbol{r}^R$ is the outcome of the coin-tossing that $\mathcal{A}$ must have obtained.)

– In Stage 3, the i-th mS-OT is executed with a random input and true randomness rather than with $(s_{i,0}, s_{i,1})$ and τ_i^S for every $i \notin \Gamma_R$.

In Stage 6, $\mathcal{S}im_{\mathrm{OT}}$ interacts with $\mathcal{A}$ as follows.

1. Receive $\{\alpha_i\}_{i \in \Delta}$ from $\mathcal{A}$ in Stage 6-1.

2. Determine the implicit input u^* of $\mathcal{A}$ as follows. Let I_0, I_1 be the sets such that for $b \in \{0,1\}$ and $i \in \Delta$, we have $i \in I_b$ if and only if:
   - $i \in \Gamma_R$, or
   - $\mathcal{A}$ did not execute the i-th mS-OT in Stage 3 honestly using $c_i \parallel \tau_i^R$ as the input and randomness, or
   - $c_i = b \oplus \alpha_i$, and $\mathcal{A}$ executed the i-th mS-OT in Stage 3 honestly using $c_i \parallel \tau_i^R$ as the input and randomness.

   Abort the simulation if both of $|I_0| \geq 6n+1$ and $|I_1| \geq 6n+1$ hold. Otherwise, define u^* by $u^* \stackrel{\text{def}}{=} 0$ if $|I_0| \geq 6n+1$ and $u^* \stackrel{\text{def}}{=} 1$ otherwise. (Roughly, $|I_b|$ is the number of strings that $\mathcal{A}$ can obtain out of $\{s_{i,b\oplus\alpha_i}\}_{i\in\Delta}$ by requiring S to reveal them in Stage 8, by cheating in mS-OT, or by executing mS-OT honestly with input $b \oplus \alpha_i$. We remind the readers that $\{s_{i,b\oplus\alpha_i}\}_{i\in\Delta}$ are the strings that are used to mask $\boldsymbol{\rho}_b = (\rho_{b,i})_{i\in\Delta}$ in Stage 6.)
3. Send u^* to the ideal functionality and obtains v^*.
4. Subsequently, interact with $\mathcal{A}$ in the same way as an honest S assuming that the inputs to S are $v_{u^*} = v^*$ and random v_{1-u^*}.

From Stage 7 to Stage 8, $\mathcal{S}im_{\mathrm{OT}}$ interacts with $\mathcal{A}$ in the same way as an honest S except that in Stage 7, an all-zero string is committed in the i-th NMCom rather than (a_i^S, d_i^S) for every $i \notin \Gamma_R$, and an all-zero string is committed in the i-th ExtCom rather than (a_i^S, d_i^S, e_i^S) for every $i \notin \Gamma_R$.

When S is corrupted: In a session where the sender S is corrupted, $\mathcal{S}im_{\mathrm{OT}}$ simulates the receiver R for $\mathcal{A}$ by extracting the implicit input v_0^*, v_1^* from $\mathcal{A}$. During the simulation, $\mathcal{S}im_{\mathrm{OT}}$ extracts the values ϕ^S and ψ^S (in straight-line, using the code of $\mathcal{A}$) such that it can later extract the value Γ^S and $\boldsymbol{a}^S$ in Stages 1 and 2; the former extraction is needed to execute most instances $\mathsf{mS\text{-}OT}$ in Stage 3 with true randomness (which is crucial to use their security in the analysis), and the latter extraction is needed to learn what input $\mathcal{A}$ used in the $\mathsf{mS\text{-}OT}$ instances in Stage 3 (which is crucial to extract the implicit input v_0^*, v_1^* from $\mathcal{A}$).

Concretely, $\mathcal{S}im_{\mathrm{OT}}$ simulates all steps of Stage 0 in the same way as an honest R, except that in Stage 0-1, $\mathcal{S}im_{\mathrm{OT}}$ uses the strategy of the (straight-line) extractor for Protocol 2 in its interaction with $\mathcal{A}$. At the end of this Stage 0-1, it learns the value ϕ^S and $\psi^S = \psi_1^S \| \ldots \| \psi_{11n}^S$ committed by $\mathcal{A}$.

Remark 9. As in Remark 8, $\mathcal{S}im_{\mathrm{OT}}$ can extract Γ_S and $\boldsymbol{a}^S$ in Stage 1-1 and Stage 2-1 respectively.

Next, $\mathcal{S}im_{\mathrm{OT}}$ interacts with $\mathcal{A}$ in the same way as an honest R in all the stages except for the following.

- For the commitments from $\mathcal{A}$ in Stages 1-1, the committed subset Γ_S is extracted by $\mathcal{S}im_{\mathrm{OT}}$ as described in Remark 9.
- In Stage 3, the i-th mS-OT is executed with a random input and true randomness rather than with c_i and τ_i^R for every $i \notin \Gamma_S$.
- In Stage 4, an all-zero string is committed in the i-th NMCom rather than (a_i^S, d_i^S) for every $i \notin \Gamma_S$, and an all-zero string is committed in the i-th ExtCom rather than (a_i^S, d_i^S, e_i^S) for every $i \notin \Gamma_S$.

– In Stage 6, α_i is a random bit rather than $\alpha_i = u \oplus c_i$ for every $i \in \Delta$, and $\widetilde{\rho}_i$ is not computed for any $i \in \Delta$.

Then, $\mathcal{S}im_{\mathrm{OT}}$ determines the implicit inputs v_0^*, v_1^* of $\mathcal{A}$ as follows.

1. For the commitments from $\mathcal{A}$ in Stage 2-1, the committed strings $\boldsymbol{a}^S = (a_1^S, \ldots, a_{11n}^S)$ are extracted by $\mathcal{S}im_{\mathrm{OT}}$ as described in Remark 9.
2. Define $\boldsymbol{r}^S = (r_1^S, \ldots, r_{11n}^S)$ by $r_i^S \stackrel{\mathrm{def}}{=} a_i^S \oplus b_i^S$ for each $i \in [11n]$ and parse r_i^S as $s_{i,0} \,\|\, s_{i,1} \,\|\, \tau_i^S$ for each $i \in [11n]$. (Notice that $\boldsymbol{r}^S$ is the outcome of the coin-tossing that $\mathcal{A}$ must have obtained.)
3. Define $\boldsymbol{\rho}_b^{\mathsf{ext}} = (\rho_{b,i}^{\mathsf{ext}})_{i \in \Delta}$ for each $b \in \{0,1\}$ as follows: $\rho_{b,i}^{\mathsf{ext}} \stackrel{\mathrm{def}}{=} \beta_{b,i} \oplus s_{i,b \oplus \alpha_i}$ if $\mathcal{A}$ executed the i-th mS-OT in stage 3 honestly using $s_{i,0} \,\|\, s_{i,1} \,\|\, \tau_i^S$, and $\rho_{b,i}^{\mathsf{ext}} \stackrel{\mathrm{def}}{=} \perp$ otherwise.
4. For each $b \in \{0,1\}$, define $v_b^* \stackrel{\mathrm{def}}{=} \mathsf{Value}(\boldsymbol{\rho}_b^{\mathsf{ext}}, \Gamma_R \cap \Delta)$.

Then, $\mathcal{S}im_{\mathrm{OT}}$ sends v_0^*, v_1^* to the ideal functionality if both of the following hold for each $b \in \{0,1\}$:

1. $|\{i \in \Delta \text{ s.t. } \rho_{b,i}^{\mathsf{ext}} \neq \perp\}| < 0.1n$.
2. $\boldsymbol{\rho}_b^{\mathsf{ext}}$ is either 0.9-close to a valid codeword $\boldsymbol{w} = (w_i)_{i \in \Delta}$ that satisfies $w_i = \rho_{b,i}^{\mathsf{ext}}$ for every $i \in \Gamma_R$ or 0.14-far from any such valid codeword.

Otherwise (i.e. if there exists $b \in \{0,1\}$) such that one of the above does not holds), $\mathcal{S}im_{\mathrm{OT}}$ aborts the simulation.

B.2 Proof of Indistinguishability

We show the indistinguishability by using a hybrid argument. Before defining hybrid experiments, we define *special messages*, which we use in the definitions of the hybrid experiments. (Essentially, they are the messages on extracted by the simulator in straight-line using the code of $\mathcal{A}$.)

– first special message is the message sent by S in Stage 1-1 (which is supposed to be $\Gamma^S \oplus \phi^S$).
– second special message is the message sent by R in Stage 1-2 (which is supposed to be $\Gamma^R \oplus \phi^R$).
– third special message is the message sent by S in Stage 2-1 (which is supposed to be $\{z_i^S\}_{i \in [11n]}$).
– fourth special message is the message sent by R in Stage 2-2 (which is supposed to be $\{z_i^R\}_{i \in [11n]}$).

B.2.1 Hybrid experiments

Now, we define hybrid experiments. Let m be the bound on the number of the sessions that $\mathcal{A}$ starts. Note that the number of special messages among m sessions can be bounded by $4m$. We order those $4m$ special messages by the order of their appearances; we use SM_k to denote the k-th special message, and $s(k)$ to denote the session that SM_k belongs to.

We start by defining hybrids H_0, H_0^* and $H_{k:1}, \ldots, H_{k:7}$ for $k \in [4m]$.(For convenience, in what follows we occasionally denote H_0^* as $H_{0:7}$.)

Remark 10 (Rough idea of the hybrids). In the sequence of the hybrid experiments, we gradually modify the real-world experiment to the ideal-world one. We make sure that $H_{k:i}$ ($i \in [7]$) deviates from the previous hybrid only after SM_k. These properties help us prove the indistinguishability of

each neighboring hybrids by using the extracted commitment as non-uniform advice and rely on the non-uniform security of the underlying primitives to prove indistinguishability.[15] ◊

Hybrid H_0. H_0 is the same as the real experiment.

Hybrid H_0^*. Recall that Stage 0-1 (resp. Stage 0-2) contains $11n+1$ (independent) parallel executions of RobCom (Protocol 2), where S (resp. R) commits to $\phi^S, \psi_1^S, \ldots, \psi_{11n}^S$ (resp. $\phi^R, \psi_1^R, \ldots, \psi_{11n}^R$). Hybrid H_0^* is identical to H_0 except that for each session $i \in [m]$

- if S is corrupted, $\mathcal{S}im_{\mathrm{OT}}$ uses the (straight-line) extractor's strategy of Protocol 2 in all the $11n+1$ RobCom executions in Stage 0-1b;
- if R is corrupted, $\mathcal{S}im_{\mathrm{OT}}$ uses the (straight-line) extractor's strategy of Protocol 2 in all the $11n+1$ RobCom executions in Stage 0-2b.

Note that in hybrid H_0^*, $\mathcal{S}im_{\mathrm{OT}}$ extracts all the values $\phi^S, \psi_1^S, \| \ldots, \psi_{11n}^S$ for each session $i \in [m]$ where S is corrupted, and all the values $\phi^R, \psi_1^R, \| \ldots, \psi_{11n}^R$ for each session $i \in [m]$ where R is corrupted. With these values, the hybrid is able to

- *(if S is corrupted)* extract the values of Γ_S and $\boldsymbol{a}^S$ that S commits to in Stages 1-1 and 2-1 respectively, as described in Remark 9;
- *(if R is corrupted)* extracts the values of Γ_R and $\boldsymbol{a}^R$ that R commits to in Stages 1-2 and 2-2 respectively, as described in Remark 8;

For future use, these extracted values are stored in a global table $\mathcal{T}$ with the corresponding session number.

Hybrid $H_{k:1}$. $H_{k:1}$ is the same as $H_{k-1:7}$ except that in session $s(k)$, if S is corrupted and SM_k is first special message,

- Query table $\mathcal{T}$ to get the extracted value Γ_S corresponding to session $s(k)$,
- the value committed to in the i-th NMCom commitment in Stage 4 is switched to an all-zero string for every $i \notin \Gamma_S$,
- the value committed to in the i-th ExtCom commitment in Stage 4 is switched to an all-zero string for every $i \notin \Gamma_S$.

Hybrid $H_{k:2}$. $H_{k:2}$ is the same as $H_{k:1}$ except that in session $s(k)$, if S is corrupted and SM_k is first special message, the i-th mS-OT in Stage 3 is executed with a random input and true randomness for every $i \notin \Gamma_S$.

Hybrid $H_{k:3}$. $H_{k:3}$ is the same as $H_{k:2}$ except that in session $s(k)$, if S is corrupted and SM_k is third special message, the following modifications are made.

1. Query table $\mathcal{T}$ to get the extracted value $\boldsymbol{a}^S$ corresponding to session $s(k)$. Define $\boldsymbol{r}^S = (r_1^S, \ldots, r_{11n}^S)$ by $r_i^S \stackrel{\text{def}}{=} a_i^S \oplus b_i^S$ for each $i \in [11n]$, and parse r_i^S as $s_{i,0} \,\|\, s_{i,1} \,\|\, \tau_i^S$ for each $i \in [11n]$. Define $\boldsymbol{\rho}_b^{\mathsf{ext}} = (\rho_{b,i}^{\mathsf{ext}})_{i \in \Delta}$ for each $b \in \{0,1\}$ as follows: $\rho_{b,i}^{\mathsf{ext}} \stackrel{\text{def}}{=} \beta_{b,i} \oplus s_{i,b \oplus \alpha_i}$ if $\mathcal{A}$ executed the i-th mS-OT in stage 3 honestly using $s_{i,0} \,\|\, s_{i,1} \,\|\, \tau_i^S$, and $\rho_{b,i}^{\mathsf{ext}} = \perp$ otherwise.

[15] We remark that, unlike [GKP18], in our case it is possible to get rid of the non-uniform argument by using (a slightly more involved) averaging argument since in our case, the extraction procedure is polynomial time. The proof using non-uniform advice is simpler.

2. R outputs $\mathsf{Value}(\boldsymbol{\rho}_u^{\mathsf{ext}}, \Gamma_R \cap \Delta)$ rather than $\mathsf{Value}(\widetilde{\rho}, \Gamma_R \cap \Delta)$. (Recall that u is the real input to R.) if both of the following hold for each $b \in \{0,1\}$:

   (a) $|\{i \in \Delta \text{ s.t. } \rho_{b,i}^{\mathsf{ext}} \neq \bot\}| < 0.1n$.

   (b) $\boldsymbol{\rho}_b^{\mathsf{ext}}$ is either 0.9-close to a valid codeword $\boldsymbol{w} = (w_i)_{i\in\Delta}$ that satisfies $w_i = \rho_{b,i}^{\mathsf{ext}}$ for every $i \in \Gamma_R$ or 0.15-far from any such valid codeword.

   Otherwise (i.e. if there exists $b \in \{0,1\}$) such that one of the above does not holds), the execution of the hybrid is aborted.

Hybrid $H_{k:4}$. $H_{k:4}$ is the same as $H_{k:3}$ except that in session $s(k)$, if S is corrupted and SM_k is third special message, α_i is a random bit rather than $\alpha_i = u \oplus c_i$ for every $i \in \Delta$ in Stage 6-1 and $\widetilde{\rho}_i$ is no longer computed for any $i \in \Delta$ in Stage 6-3.

Hybrid $H_{k:5}$. $H_{k:5}$ is the same as $H_{k:4}$ except that in session $s(k)$, if R is corrupted and SM_k is second special message,

- Query table $\mathcal{T}$ to get the extracted value Γ_R corresponding to session $s(k)$,
- the value committed in the i-th NMCom commitment in Stage 7 is switched to an all-zero string for every $i \notin \Gamma_R$,
- the value committed in the i-th ExtCom commitment in Stage 7 is switched to an all-zero string for every $i \notin \Gamma_R$.

Hybrid $H_{k:6}$. $H_{k:6}$ is the same as $H_{k:5}$ except that in session $s(k)$, if R is corrupted and SM_k is second special message, the i-th mS-OT in Stage 3 is executed with a random input and true randomness for every $i \notin \Gamma_R$.

Hybrid $H_{k:7}$. $H_{k:7}$ is the same as $H_{k:6}$ except that in session $s(k)$, if R is corrupted and SM_k is fourth special message, the following modifications are made.

1. Query table $\mathcal{T}$ to get the extracted value $\boldsymbol{a}^R$ corresponding to session $s(k)$. Define $\boldsymbol{r}^R = (r_1^R, \ldots, r_{11n}^R)$ by $r_i^R \stackrel{\text{def}}{=} a_i^R \oplus b_i^R$ for each $i \in [11n]$, and parse r_i^R as $c_i \,\|\, \tau_i^R$ for each $i \in [11n]$. Define u^* as follows. Let I_0 and I_1 be the set such that for $b \in \{0,1\}$ and $i \in \Delta$, we have $i \in I_b$ if and only if:

   - $i \in \Gamma_R$, or
   - $\mathcal{A}$ did not execute the i-th mS-OT in Stage 3 honestly using $c_i \,\|\, \tau_i^R$ as the input and randomness, or
   - $c_i = b \oplus \alpha_i$, and $\mathcal{A}$ executed the i-th mS-OT in Stage 3 honestly using $c_i \,\|\, \tau_i^R$ as the input and randomness.

   Abort the execution if both of $|I_0| \geq 6n+1$ and $|I_1| \geq 6n+1$ hold. Otherwise, define u^* by $u^* \stackrel{\text{def}}{=} 0$ if $|I_0| \geq 6n+1$ and $u^* \stackrel{\text{def}}{=} 1$.

2. In Stage 6, $\boldsymbol{\rho}_{1-u^*}$ is a secret sharing of a random bit rather than that of v_{1-u^*}.

We remark that in $H_{4m:7}$, all the messages from the honest parties and their output are computed as in $\mathcal{S}im_{\mathrm{OT}}$.

B.2.2 Indistinguishability of each neighboring hybrids

Below, we show that each neighboring hybrids are indistinguishable, and additionally show, for technical reasons, that an invariant condition holds in each session of every hybrid.

First, we define the invariant condition.

Definition 6 (Invariant Condition (when R is corrupted)). *For any session in which R is corrupted, we say that the invariant condition holds in that session if the following holds when the cut-and-choose in Stage 5 is accepted.*

1. *Let $(\hat{a}_1^R, \hat{d}_1^R), \ldots (\hat{a}_{11n}^R, \hat{d}_{11n}^R)$ be the values that are committed in* NMCom *in Stage 4. Let $I_{\mathrm{bad}} \subset [11n]$ be the set such that $i \in I_{\mathrm{bad}}$ if and only if*
   (a) *$(\hat{a}_i^R, \hat{d}_i^R)$ is not valid in terms of the check in Stage 5-3b, i.e. $\hat{d}_i^R$ is not a valid decommitment of ψ_i^R w.r.t. Stage-0-2* RobCom*, or $\hat{a}_i^R \oplus \psi_i^R$ does not equal the value z_i^R it received in Stage-2-2;* **or**
   (b) *R does not execute the i-th mS-OT in Stage 3 honestly using $\hat{c}_i \,\|\, \hat{\tau}_i^R$ as the input and randomness, where $\hat{c}_i \,\|\, \hat{\tau}_i^R$ is obtained from $\hat{r}_i^R = \hat{a}_i^R \oplus b_i^R$.*

   Then, it holds that $|I_{\mathrm{bad}}| < n$.

Remark 11. Roughly speaking, this condition guarantees that most of the mS-OTs in Stage 3 are honestly executed using the outcome of the coin tossing, which in turn guarantees that the cheating receiver's input can be extracted by extracting the outcome of the coin tossing. ◊

Remark 12. When Stage 5 is accepted, we also have $I_{\mathrm{bad}} \cap \Gamma_S = \emptyset$ from the definition of I_{bad}. ◊

Definition 7 (Invariant Condition (when S is corrupted)). *For any session in which S is corrupted, we say that the invariant condition holds in that session if the following hold when the cut-and-choose in Stage 8 is accepted.*

1. *Let $(\hat{a}_1^S, \hat{d}_1^S), \ldots (\hat{a}_{11n}^S, \hat{d}_{11n}^S)$ be the values that are committed in* NMCom *in Stage 7. Let $I_{\mathrm{bad}} \subset [11n]$ be the set such that $i \in I_{\mathrm{bad}}$ if and only if*
   (a) *$(\hat{a}_i^S, \hat{d}_i^S)$ is not valid in terms of the check in Stage 8-3b, i.e. $\hat{d}_i^S$ is not a valid decommitment of ψ_i^S w.r.t. Stage-0-1* RobCom*, or $\hat{a}_i^S \oplus \psi_i^S$ does not equal the value z_i^S it received in Stage-2-1;* **or**
   (b) *S does not execute the i-th mS-OT in Stage 3 honestly using $\hat{s}_{i,0} \,\|\, \hat{s}_{i,1} \,\|\, \hat{\tau}_i^S$ as the input and randomness, where $\hat{s}_{i,0} \,\|\, \hat{s}_{i,1} \,\|\, \hat{\tau}_i^S$ is obtained from $\hat{r}_i^S = \hat{a}_i^S \oplus b_i^S$.*

   Then, it holds that $|I_{\mathrm{bad}}| < 0.1n$.
2. *For each $b \in \{0,1\}$, define $\boldsymbol{\rho}_b^{\mathsf{nm}} = (\rho_{b,i}^{\mathsf{nm}})_{i \in \Delta}$ as follows: $\rho_{b,i}^{\mathsf{nm}} \stackrel{\mathrm{def}}{=} \beta_{b,i} \oplus \hat{s}_{i,b \oplus \alpha_i}$ if $i \notin I_{\mathrm{bad}}$ and $\rho_{b,i}^{\mathsf{nm}} \stackrel{\mathrm{def}}{=} \perp$ otherwise. Then, for each $b \in \{0,1\}$, ρ_b^{nm} is either 0.9-close to a valid codeword $\boldsymbol{w} = (w_i)_{i \in \Delta}$ that satisfies $w_i = \rho_{b,i}^{\mathsf{nm}}$ for every $i \in \Gamma_R$ or 0.15-far from any such valid codeword.*

Remark 13. Roughly speaking, this condition guarantees that the cheating sender's input can be extracted from the outcome of the coin tossing. In particular, it guarantees that the sharing that is computed from the outcome of mS-OTs (i.e., the sharing that is computed by the honest receiver) and the sharing that is computed from the outcome of the coin tossing (i.e., the sharing that is computed by the simulator) are very "close" (see Claim 6 below). ◊

Remark 14. When Stage 8 is accepted, we also have $I_{\mathrm{bad}} \cap \Gamma_R = \emptyset$ from the definition of I_{bad}. $\diamondsuit$

Next, we show that the invariant condition holds in every session in H_0 (i.e., the real experiment).

Definition 8. *We say that $\mathcal{A}$ **cheats** in a session if the invariant condition does not hold in that session.*

Next, we establish the computational indistinguishability among hybrids by a sequence of lemmata. We start with the following lemma:

Lemma 1. *In H_0, $\mathcal{A}$ does not cheat in every session except with negligible probability.*

Proof. The proof of this lemma is identical to the proof in [GKP18]. We include it here for completeness.

Assume for contradiction that in H_0, $\mathcal{A}$ cheats in a session with non-negligible probability. Since the number of the sessions is bounded by a polynomial, there exists a function $i^*(\cdot)$ and a polynomial $p(\cdot)$ such that for infinitely many n, $\mathcal{A}$ cheats in the $i^*(n)$-th session with probability at least $1/p(n)$; furthermore, since $\mathcal{A}$ cheats only when either R or S is corrupted, in the $i^*(n)$-th session either R is corrupted for infinitely many such n or S is corrupted for infinitely many such n. In both cases, we derive contradiction by using $\mathcal{A}$ to break the hiding property of RobCom.

Case 1. R is corrupted in the $i^*(n)$-th session. We show that when $\mathcal{A}$ cheats, we can break the hiding property of the $\mathsf{RobCom}(\phi^S)$ commitment in Stage 0-1 (i.e., the commitment by which ϕ^S is committed to). From the definition of the invariant condition (Definition 6), when $\mathcal{A}$ cheats, we have $|I_{\mathrm{bad}}| \geq n$ even though the cut-and-choose in Stage 5 is accepting (and hence $I_{\mathrm{bad}} \cap \Gamma_S = \emptyset$ as remarked in Remark 12), where $I_{\mathrm{bad}} \subseteq [11n]$ is the set defined from the committed values of the NMCom commitments in Stage 4. If we can compute I_{bad} efficiently, we can use it to distinguish Γ_S from a random subset of size n (this is because a random subset Γ of size n satisfies $I_{\mathrm{bad}} \cap \Gamma = \emptyset$ only with negligible probability when $|I_{\mathrm{bad}}| \geq n$), so we can use it to break the hiding property of the RobCom commitment to ϕ^S, which is used to mask Γ_S. However, I_{bad} is not efficiently computable since the committed values of the NMCom commitments are not efficiently computable. We thus first show that we can "approximate" I_{bad} by extracting the committed values of the ExtCom commitments in Stage 4. Details are given below.

First, we observe that if we extract the committed values of the ExtCom commitments in Stage 4 of the $i^*(n)$-th session, the extracted values $(\hat{a}_1^R, \hat{d}_1^R, \hat{e}_1^R), \ldots, (\hat{a}_{11n}^R, \hat{d}_{11n}^R, \hat{e}_{11n}^R)$, satisfy the following condition.

- Let $\hat{I}_{\mathrm{bad}} \subset [11n]$ be a set such that $i \in \hat{I}_{\mathrm{bad}}$ if and only if
  1. $((\hat{a}_i^R, \hat{d}_i^R), \hat{e}_i^R)$ is not a valid decommitment of the i-th NMCom commitment in Stage 4; **or**
  2. $(\hat{a}_i^R, \hat{d}_i^R)$ is not valid in terms of the check in Stage 5-3b, i.e. $\hat{d}_i^R$ is not a valid decommitment of ψ_i^R w.r.t. Stage-0-2 RobCom, or $\hat{a}_i^R \oplus \psi_i^R$ does not equal the value z_i^R it received in Stage-2-2; **or**
  3. R does not execute the i-th mS-OT in Stage 3 honestly using $\hat{c}_i \parallel \hat{\tau}_i^R$ as the input and randomness, where $\hat{c}_i \| \hat{\tau}_i^R$ is obtained from $\hat{r}_i^R = \hat{a}_i^R \oplus b_i^R$.

  Then, $|\hat{I}_{\mathrm{bad}}| \geq n$ and $\hat{I}_{\mathrm{bad}} \cap \Gamma_S = \emptyset$ with probability at least $1/2p(n)$.

The extracted values satisfy this condition because when $\mathcal{A}$ cheats, we have $|\hat{I}_{\mathrm{bad}}| \geq n$ and $\hat{I}_{\mathrm{bad}} \cap \Gamma_S = \emptyset$ except with negligible probability. (We have $|\hat{I}_{\mathrm{bad}}| \geq n$ since we have $I_{\mathrm{bad}} \subset \hat{I}_{\mathrm{bad}}$ from the definitions of $I_{\mathrm{bad}}, \hat{I}_{\mathrm{bad}}$ and the binding property of NMCom. We have $\hat{I}_{\mathrm{bad}} \cap \Gamma_S = \emptyset$ since when the cut-and-choose in Stage 5 is accepting, for every $i \in \Gamma_S$ the i-th ExtCom commitment is a valid decommitment of the i-th NMCom commitment, and $I_{\mathrm{bad}} \cap \Gamma_S = \emptyset$.)

Based on this observation, we derive contradiction by considering the following adversary $\mathcal{A}_{\mathsf{RobCom}}$ against the hiding property of RobCom.

> $\mathcal{A}_{\mathsf{RobCom}}$ receives a RobCom commitment c^* in which either ϕ_S^0 or ϕ_S^1 is committed. Then, $\mathcal{A}_{\mathsf{RobCom}}$ internally executes the experiment H_0 honestly except that in the $i^*(n)$-th session, $\mathcal{A}_{\mathsf{RobCom}}$ uses c^* as the commitment in Stage 0-1 (i.e., as the RobCom commitment in which S commits to string which will be used to mask Γ_S in Stage 1-1). In Stage 1-1, $\mathcal{A}_{\mathsf{RobCom}}$ always use ϕ_S^1 to mask Γ_S. When the experiment H_0 reaches Stage 4 of the $i^*(n)$-th session, $\mathcal{A}_{\mathsf{RobCom}}$ extracts the committed values of the ExtCom commitments in this stage by using its extractability.[16] Let $\hat{I}_{\mathrm{bad}} \subset [11n]$ be the set that is defined as above from the extracted values. Then, $\mathcal{A}_{\mathsf{RobCom}}$ outputs 1 if and only if $|\hat{I}_{\mathrm{bad}}| \geq n$ and $\hat{I}_{\mathrm{bad}} \cap \Gamma_S = \emptyset$.

If $\mathcal{A}_{\mathsf{RobCom}}$ receives a commitment to ϕ_S^1, $\mathcal{A}_{\mathsf{RobCom}}$ outputs 1 with probability at least $1/2p(n)$ (this follows from the above observation). In contrast, if $\mathcal{A}_{\mathsf{RobCom}}$ receives a commitment to ϕ_S^0, $\mathcal{A}_{\mathsf{RobCom}}$ outputs 1 with exponentially small probability (this is because $\phi_1^S \oplus \Gamma_S$ is a pure random string now, so the probability that $|\hat{I}_{\mathrm{bad}}| \geq n$ but $\hat{I}_{\mathrm{bad}} \cap \Gamma_S^1 = \emptyset$ is exponentially small). Hence, $\mathcal{A}_{\mathsf{RobCom}}$ breaks the hiding property of RobCom.

Case 2. S is corrupted in the $i^*(n)$-th session. The proof for this case is similar to (but a little more complex than) the one for Case 1. Specifically, we show that if the invariant condition does not hold, we can break the hiding property of $\mathsf{RobCom}(\phi^R)$ in Stage 0-2 by approximating I_{bad} using the extractability of ExtCom. We give a formal proof for this case in Section C.1. (A somewhat similar proof is given as the proof of Claim 5 later.) □

Next, we show the indistinguishability between each neighboring hybrids.

Lemma 2. *Hybrids H_0 and H_0^* are indistinguishable, and in H_0^*, $\mathcal{A}$ does not cheat in every sessions except with negligible probability.*

Proof. We first prove the indistinguishability by a sequence of intermediate hybrids where the RobCom is replaced one-by-one. This relies on the robust extractability of RobCom. Then, using the established indistinguishability and the robust non-malleability of NMCom, we show that $\mathcal{A}$ does not cheat in H_0^*.

We first set $\hat{H}_0^0 = H_0$. Then, we define the following sequence of m intermediate hybrids:

Hybrid $\hat{H}_0^i$ ($i \in [m]$). This hybrid is identical to $\hat{H}_0^{i-1}$ except that in session i,

- if S is corrupted, $\mathcal{S}im_{\mathrm{OT}}$ uses the (straight-line) extractor's strategy of Protocol 2 in all the $11n+1$ RobCom executions in Stage 0-1b (in session i);
- if R is corrupted, $\mathcal{S}im_{\mathrm{OT}}$ uses the (straight-line) extractor's strategy of Protocol 2 in all the $11n+1$ RobCom executions in Stage 0-2b (in session i);

[16] This extraction involves rewinding the execution of the whole experiment, i.e., the adversary as well as all the other parties.

It is easy to see that $\hat{H}_0^m = H_0^*$. Thus, to prove Lemma 2, we only need to show that each adjacent $\hat{H}_0^i$ and $\hat{H}_0^{i+1}$ is indistinguishable and $\mathcal{A}$ does not cheat in $\hat{H}_0^i$ for all $i \in [m]$. For this purpose, we provide a proof for $\hat{H}_0^0$ (i.e. H_0) and $\hat{H}_0^1$ in the following Claim 3. The same argument extends straightforwardly to other adjacent $\hat{H}_0^i$ and $\hat{H}_0^{i+1}$.

Claim 3. *Hybrids $\hat{H}_0^0$ and $\hat{H}_0^1$ are indistinguishable, and in $\hat{H}_0^1$, $\mathcal{A}$ does not cheat in every sessions except with negligible probability.*

Proof. Note that our OT protocol contains $11n + 1$ RobCom executions in Stage 0-1b (and in Stage 0-2b). To conduct the proof, we actually need a finer-grained sequence of hybrids, which is listed in the following. To provide an intuitive explanation, the following hybrids are obtained by inserting $11n+1$ hybrids between $\hat{H}_0^0$ and $\hat{H}_0^1$, where the $11n+1$ RobCom instances are substituted one-by-one.

Hybrid $\hat{H}_0^{0:i}$ ($i \in [11n + 1]$). this hybrid is the same as $\hat{H}_0^0$ except that

- if S is corrupted in session 1, $\mathcal{S}im_{\mathrm{OT}}$ uses the (straight-line) extractor's strategy of Protocol 2 in *the first* i of the $11n + 1$ RobCom executions in Stage 0-1b;
- if R is corrupted in session 1, $\mathcal{S}im_{\mathrm{OT}}$ uses the (straight-line) extractor's strategy of Protocol 2 in *the first* i of the $11n + 1$ RobCom executions in Stage 0-2b;

It is easy to see that $\hat{H}_0^{0:11n+1} = \hat{H}_0^1$. To prove Claim 3, we need to show that each adjacent $\hat{H}_0^{0:i}$ and $\hat{H}_0^{0:i+1}$ is indistinguishable and $\mathcal{A}$ does not cheat in $\hat{H}_0^{0:i}$ for all $i \in [11n + 1]$. In the following, we focus on the switch between $\hat{H}_0^{0:0}$ and $\hat{H}_0^{0:1}$ (the same argument extends to the switch from $\hat{H}_0^{0:i-1}$ to $\hat{H}_0^{0:i}$ for all $i \in [11n + 1]$). Concretely, in the following we prove that:

- $\hat{H}_0^{0:0}$ and $\hat{H}_0^{0:1}$ are indistinguishable, and $\mathcal{A}$ does not cheat in $\hat{H}_0^{0:1}$.

Let us stress that the only difference between $\hat{H}_0^{0:0}$ and $\hat{H}_0^{0:1}$ lies in the *first* RobCom in Stage 0-1b of session 1 (if $\mathcal{A}$ corrupts S in session 1), **or** in Stage 0-2b of session 1 (if $\mathcal{A}$ corrupts R in session 1).

Indistinguishability. First note that, in session 1, if no party is corrupted, $\hat{H}_0^{0:0}$ and $\hat{H}_0^{0:1}$ are identical. So the statement holds trivially.

When one party is corrupted (S or R), we prove the indistinguishability based on the robust extractability of RobCom.

The argument is actually identical to the proof of the extractability for Protocol 2 (Section 5.1.2), with the only difference that there are other sessions running besides the RobCom under our consideration. To deal with this, we note that Step 5d in Protocol 2 is instantiated with the commit-and-proof scheme shown in Protocol 1, whose **Proof Phase** gives us ℓ-robustness (as we proved in Theorem 3). In the design of our OT, we set the robustness to be $\ell(n) = m \cdot \nu_{\mathrm{OT}}(n)$, which is large enough to encompass all the messages from the remaining part of the execution. With this modification, the same sequence of hybrids for the extractability of Protocol 2 also works for proving indistinguishability among $\hat{H}_0^{0:0}$ to $\hat{H}_0^{0:1}$. More specifically, we only need to modify the switch from H_0 to H_1 (where we switch from the real prover to the robust-ZK simulator in Step 5d) in Section 5.1.2, by relying additional on the ℓ-robustness. For completeness, we provide the full proof for this switch in the following.

Assume for contradiction that the indistinguishability does not hold due to the switch from the real prover to the robust-ZK simulator (from H_0 to H_1 as mentioned above). We can then

construct a robust-$\mathcal{ZK}$ verifier V^* and a machine B, who interact with either a prover or the simulator $S_{\mathcal{ZK}}(\mathsf{code}[\mathcal{A}])$ and violate the robust-$\mathcal{ZK}$ property, as follows.

Machine B: incorporates all honest parties, including the honest party for session 1. B performs all steps honestly for each party except the corresponding Step 5d in session 1; the messages of this phase are expected to come from an external machine (either the prover or the simulator). The messages of B are sent to the network which delivers them appropriately to the cheating verifier (specified below). We note that, by definition of robust-$\mathcal{ZK}$ B receives a copy of all messages that V^* receives.

Verifier V^*: this algorithm is just the adversary $\mathcal{A}$, with the understanding that all messages that do not belong to the corresponding Step 5d in session 1 are viewed as external messages sent to (or received from) machine B.

Observe that B is polynomial time, and since it receives a copy of all messages sent to V^* (by the prover or the simulator), it can indeed function correctly even though it runs the simulator algorithm for the $\mathcal{ZK}$-proof stage in all sessions different from session 1. (This is not necessary in the switch from $\hat{H}_0^0$ to $\hat{H}_0^1$, as the only session 1 contains simulated RobCom. But it will be important in later hybrids as more sessions contain simulated (Step 5d of) RobCom.)

Notice that if V^* interacts with honest prover of the robust-$\mathcal{ZK}$, then this experiment is identical to the aforementioned H_0. On the other hand, if it interacts with $S_{\mathcal{ZK}}$ then the experiment is identical to the aforementioned H_1;[17] furthermore, since $S_{\mathcal{ZK}}$ receives a copy of all messages that V^* receives from B, it can indeed run in polynomial time.

It follows that if the output of hybrids are not indistinguishable, we violate the robust-$\mathcal{ZK}$ property.

Remark 15 (On the Bound $\ell(n)$). The above argument relies on the assumption that the size of messages coming from B to V^* is bounded by $\ell(n)$ (the definition of $\ell(n)$-robust ZK). As a vigilant reader may have realized already, this is not quite true for our definition of V^*. More specifically, consider all the messages coming from B to V^*. There must be one **long message** for each session where V^* plays the role of RobCom sender (note that the other **long message** in the same session comes from V^* to B, so we do not need to worry)[18]. Recall that we set $\ell(n) = m \cdot \nu_{\text{OT}}(n)$. This is enough to capture all the short messages, but not these long messages (from B to V^*). This can be resolved in the following way. Note that the long messages coming from B to V^* are nothing more than random strings. Thus, we can extend V^* to incorporate the subroutine in B that samples these random strings. Let us denote this extended adversary as $\widetilde{V}^*$. We modify the above argument by passing the code of $\widetilde{V}^*$ to the ZK simulator. Now everything works as the size of messages flowing from B to $\widetilde{V}^*$ is bounded by $\ell(n)$.

Invariant Condition. We next show that in $\hat{H}_0^{0:1}$, $\mathcal{A}$ does not cheat. Assume for contradiction that $\mathcal{A}$ cheats in some session $i^*(n) \in [m]$ with non-negligible probability. Note that the only difference between $\hat{H}_0^{0:0}$ and $\hat{H}_0^{0:1}$ is that the adversary in session 1 sees a real proof in RobCom of $\hat{H}_0^{0:0}$, but a simulated one (from the straight-line extractor) in RobCom of $\hat{H}_0^{0:1}$. Then, by expecting this stage

[17] We stress that the H_0 and H_1 here are the ones as defined in Section 5.1.2, but in our current context of the concurrent OT setting. They should not be confused with other hybrids defined in this section

[18] More accurately, using our definition for B and V^*, in any session, if S is corrupted, then V^* will incorporate S and the **receiver's long massages** flows from B to V^*; if R is corrupted, the **sender's long massages** flows from B to V^*.

in session 1 from an external prover or simulator (RobCom extractor), we can break the robust non-malleability of NMCom *in session* $i^*(n)$. We elaborate on this argument in the following.

> The man-in-the-middle adversary $\mathcal{A}_{\mathsf{NMCom}}$ internally executes $\hat{H}_0^{0:0}$. If S (resp. R) is corrupted in session 1, then in Stage 0-1b (resp. Stage 0-2b) of session 1, $\mathcal{A}_{\mathsf{NMCom}}$ forward the message between the external party, which is either the honest RobCom receiver or the straihgt-line simulator. Also, in session $i^*(n)$, $\mathcal{A}_{\mathsf{NMCom}}$ forwards the NMCom commitments from $\mathcal{A}$ to the external receiver (specifically, the NMCom commitments in Stage 4 if R is corrupted in session $i^*(n)$, and in Stage 7 if S is corrupted in session $i^*(n)$). After the execution of $\hat{H}_0^{0:0}$ finishes, $\mathcal{A}_{\mathsf{NMCom}}$ outputs its view.
> The distinguisher $\mathcal{D}_{\mathsf{NMCom}}$ takes as input the view of $\mathcal{A}_{\mathsf{NMCom}}$ and the values committed by $\mathcal{A}_{\mathsf{NMCom}}$ (which are equal to the values committed to by $\mathcal{A}$ in session $i^*(n)$ in the internally executed experiment). $\mathcal{D}_{\mathsf{NMCom}}$ then outputs 1 if and only if $\mathcal{A}$ cheated in session $i^*(n)$. (Notice that given the committed values of the NMCom commitments, $\mathcal{D}_{\mathsf{NMCom}}$ can check whether $\mathcal{A}$ cheated or not in polynomial time.)

When the external party mentioned above is the honest RobCom receiver, the view of $\mathcal{A}$ is identical to that in $\hat{H}_0^{0:0}$; whereas when the external party mentioned above is the extractor for RobCom, the view of $\mathcal{A}$ is identical to that in $\hat{H}_0^{0:1}$. Hence, from the assumption that $\mathcal{A}$ cheats in session $i^*(n)$ with negligible probability in $\hat{H}_0^{0:0}$ but with non-negligible probability in $\hat{H}_0^{0:1}$, $\mathcal{A}_{\mathsf{NMCom}}$ breaks the robust non-malleability of NMCom.

This finishes the proof for Claim 3. □

This finishes the proof for Lemma 2. □

From here onwards, the proof of indistinguishability of hybrids is very similar to the proofs in [GKP18] (with minor notational changes) except that we do not require brute-force extraction of inputs; instead they are accessed directly from table $\mathcal{T}$. We provide the proofs here for completeness.

Lemma 3. *Assume that in $H_{k-1:7}$ ($k \in [4m]$), $\mathcal{A}$ does not cheat in sessions $s(k), \ldots, s(4m)$ except with negligible probability. Then,*

- *$H_{k-1:7}$ and $H_{k:1}$ are indistinguishable, and*
- *in $H_{k:1}$, $\mathcal{A}$ does not cheat in sessions $s(k), \ldots, s(4m)$ except with negligible probability.*

Proof. This proof for this lemma is identical to that in [GKP18], except that the hybrid uses the table $\mathcal{T}$ to get the extracted values it needs, instead of extracting by brute force. We present the full proof here for completeness.

We prove the lemma by using a hybrid argument. Specifically, we consider the following intermediate hybrid $H'_{k-1:7}$.

Hybrid $H'_{k-1:7}$. $H'_{k-1:7}$ is the same as $H_{k-1:7}$ except that in session $s(k)$, if S is corrupted and SM_k is first special message,

- the committed subset Γ_S is extracted by querying $\mathcal{T}$ in Stage 1-1, and
- the value committed to in the i-th ExtCom commitment in Stage 4 is switched to an all-zero string for every $i \notin \Gamma_S$.

Claim 4. *Assume that in $H_{k-1:7}$, $\mathcal{A}$ does not cheat in sessions $s(k), \ldots, s(4m)$ except with negligible probability. Then,*

– $H_{k-1:7}$ *and* $H'_{k-1:7}$ *are indistinguishable, and*

– *in* $H'_{k-1:7}$, $\mathcal{A}$ *does not cheat in sessions* $s(k), \ldots, s(4m)$ *except with negligible probability.*

Proof. We first show the indistinguishability between $H_{k-1:7}$ and $H'_{k-1:7}$. Assume for contradiction that $H_{k-1:7}$ and $H'_{k-1:7}$ are distinguishable. From an average argument, we can fix the execution of the experiment up until SM_k (inclusive) in such a way that even after being fixed, $H_{k-1:7}$ and $H'_{k-1:7}$ are still distinguishable. By considering the transcript (including the inputs and randomness of all the parties) up until SM_k and the table $\mathcal{T}$ as non-uniform advice, we can break the hiding property of ExtCom as follows.

> The adversary $\mathcal{A}_{\mathsf{ExtCom}}$ internally executes $H_{k-1:7}$ from SM_k using the non-uniform advice. In Stage 4 of session $s(k)$, $\mathcal{A}_{\mathsf{ExtCom}}$ sends $(a_i^R, d_i^R, e_i^R)_{i \notin \Gamma_S}$ and $(0,0,0)_{i \notin \Gamma_S}$ to the external committer, receives back ExtCom commitments (in which either $(a_i^R, d_i^R, e_i^R)_{i \notin \Gamma_S}$ or $(0,0,0)_{i \notin \Gamma_S}$ are committed to), and feeds them into $H_{k-1:7}$. After the execution of $H_{k-1:7}$ finishes, $\mathcal{A}_{\mathsf{ExtCom}}$ outputs whatever $\mathcal{Z}$ outputs in the experiment.

When $\mathcal{A}_{\mathsf{ExtCom}}$ receives commitments to $(a_i^R, d_i^R, e_i^R)_{i \notin \Gamma_S}$, the internally executed experiment is identical with $H_{k-1:7}$, whereas when $\mathcal{A}_{\mathsf{ExtCom}}$ receives a commitments to $(0,0,0)_{i \notin \Gamma_S}$, the internally executed experiment is identical with $H'_{k-1:7}$. Hence, from the assumption that $H_{k-1:7}$ and $H'_{k-1:7}$ are distinguishable (even after being fixed up until SM_k), $\mathcal{A}_{\mathsf{ExtCom}}$ distinguishes ExtCom commitments.

We next show that in $H'_{k-1:7}$, $\mathcal{A}$ does not cheat in sessions $s(k), \ldots, s(4m)$. Assume for contradiction that in $H'_{k-1:7}$, $\mathcal{A}$ cheats in one of those sessions, say, session $s(j)$, with non-negligible probability. Then, from an average argument, we can fix the execution of the experiment up until SM_k (inclusive) in such a way that even after being fixed, $\mathcal{A}$ cheats in session $s(j)$ only with negligible probability in $H_{k-1:7}$ but with non-negligible probability in $H'_{k-1:7}$.

Then, by considering the transcript up until SM_k and the table $\mathcal{T}$ as non-uniform advice, we can break the robust non-malleability of NMCom as follows.

> The man-in-the-middle adversary $\mathcal{A}_{\mathsf{NMCom}}$ internally executes $H_{k-1:7}$ from SM_k using the non-uniform advice. In Stage 4 of session $s(k)$, $\mathcal{A}_{\mathsf{NMCom}}$ sends $(a_i^R, d_i^R, e_i^R)_{i \notin \Gamma_S}$ and $(0,0,0)_{i \notin \Gamma_S}$ to the external committer, receives back ExtCom commitments (in which either $(a_i^R, d_i^R, e_i^R)_{i \notin \Gamma_S}$ or $(0,0,0)_{i \notin \Gamma_S}$ are committed to), and feeds them into $H_{k-1:7}$. Also, in session $s(j)$, $\mathcal{A}_{\mathsf{NMCom}}$ forwards the NMCom commitments from $\mathcal{A}$ to the external receiver (specifically, the NMCom commitments in Stage 4 if R is corrupted and in Stage 7 if S is corrupted). After the execution of $H_{k-1:7}$ finishes, $\mathcal{A}_{\mathsf{NMCom}}$ outputs its view.
> The distinguisher $\mathcal{D}_{\mathsf{NMCom}}$ takes as input the view of $\mathcal{A}_{\mathsf{NMCom}}$ and the values committed by $\mathcal{A}_{\mathsf{NMCom}}$ (which are equal to the values committed to by $\mathcal{A}$ in session $s(j)$ in the internally executed experiment). $\mathcal{D}_{\mathsf{NMCom}}$ then outputs 1 if and only if $\mathcal{A}$ cheated in session $s(j)$. (Notice that given the committed values of the NMCom commitments, $\mathcal{D}_{\mathsf{NMCom}}$ can check whether $\mathcal{A}$ cheated or not in polynomial time.)

When $\mathcal{A}_{\mathsf{NMCom}}$ receives commitments to $(a_i^R, d_i^R, e_i^R)_{i \notin \Gamma_S}$, the internally executed experiment is identical with $H_{k-1:7}$, whereas when $\mathcal{A}_{\mathsf{NMCom}}$ receives a commitments to $(0,0,0)_{i \notin \Gamma_S}$, the internally executed experiment is identical with $H'_{k-1:7}$. Hence, from the assumption that $\mathcal{A}$ cheats in session $s(j)$ with negligible probability in $H_{k-1:7}$ but with non-negligible probability in $H'_{k-1:7}$, $\mathcal{A}_{\mathsf{NMCom}}$ breaks the robust non-malleability of NMCom.

This completes the proof of Claim 4. □

Claim 5. *Assume that in $H'_{k-1:7}$, $\mathcal{A}$ does not cheat in sessions $s(k), \ldots, s(4m)$ except with negligible probability. Then,*

- *$H'_{k-1:7}$ and $H_{k:1}$ are indistinguishable, and*
- *in $H_{k:1}$, $\mathcal{A}$ does not cheat in sessions $s(k), \ldots, s(4m)$ except with negligible probability.*

This claim can be proven very similarly to Claim 4 (the only difference is that we use the hiding property of NMCom rather than that of ExtCom in the first part, and use the non-malleability of NMCom rather than its robust non-malleability in the second part). We thus omit the proof.

This completes the proof of Lemma 3. □

Lemma 4. *Assume that in $H_{k:1}$ ($k \in [4m]$), $\mathcal{A}$ does not cheat in sessions $s(k), \ldots, s(4m)$ except with negligible probability. Then,*

- *$H_{k:1}$ and $H_{k:2}$ are indistinguishable, and*
- *in $H_{k:2}$, $\mathcal{A}$ does not cheat in sessions $s(k), \ldots, s(4m)$ except with negligible probability.*

Recall that hybrids $H_{k:1}, H_{k:2}$ differ only in the input and the randomness that are used in some of the mS-OTs in Stage 3, where those that are derived from the outcomes of the coin tossing is used in $H_{k:1}$ and random inputs and true randomness are used in $H_{k:2}$. Intuitively, we prove this lemma by using the security of the Stage-2-2 coin tossing (which is guaranteed by the hiding property of $\mathsf{RobCom}(\psi_i^R)$'s) because it guarantees that the outcome of the coin tossing is pseudorandom. The proof is quite similar to the proof of Claim 4 (we use the hiding of $\mathsf{RobCom}(\psi_i^R)$'s rather than that of ExtCom), and given in Section C.2.

Lemma 5. *Assume that in $H_{k:2}$ ($k \in [4m]$), $\mathcal{A}$ does not cheat in sessions $s(k), \ldots, s(4m)$ except with negligible probability. Then,*

- *$H_{k:2}$ and $H_{k:3}$ are indistinguishable, and*
- *in $H_{k:3}$, $\mathcal{A}$ does not cheat in sessions $s(k), \ldots, s(4m)$ except with negligible probability.*

Proof. Recall that $H_{k:2}$ and $H_{k:3}$ differ only in that in session $s(k)$ of $H_{k:3}$, if S is corrupted and SM_k is third special message, either R outputs $\mathsf{Value}(\boldsymbol{\rho}_u^{\mathsf{ext}}, \Gamma_R \cap \Delta)$ rather than $\mathsf{Value}(\widetilde{\boldsymbol{\rho}}, \Gamma_R \cap \Delta)$ or the hybrid is aborted.

For proving the lemma, it suffices to show that in session $s(k)$ of $H_{k:3}$,

1. the hybrid is not aborted except with negligible probability, and
2. if the hybrid is not aborted we have $\mathsf{Value}(\boldsymbol{\rho}_u^{\mathsf{ext}}, \Gamma_R \cap \Delta) = \mathsf{Value}(\widetilde{\boldsymbol{\rho}}, \Gamma_R \cap \Delta)$

To see that showing these two is indeed sufficient for proving the lemma, observe the following. First, these two imply that in session $s(k)$ of $H_{k:3}$, the probability that the hybrid is aborted or we have $\mathsf{Value}(\boldsymbol{\rho}_u^{\mathsf{ext}}, \Gamma_R \cap \Delta) = \mathsf{Value}(\widetilde{\boldsymbol{\rho}}, \Gamma_R \cap \Delta)$ is negligible, so $H_{k:2}$ and $H_{k:3}$ are statistically close. Second, since $H_{k:2}$ and $H_{k:3}$ proceed identically until the end of session $s(k)$, and

1. if the experiment is not aborted in session $s(k)$, $H_{k:2}$ and $H_{k:3}$ continue to proceed identically after the end of session $s(k)$, and
2. if the hybrid is aborted in session $s(k)$, $\mathcal{A}$ clearly does not cheat in any session after the end of session $s(k)$

the probability that $\mathcal{A}$ cheats in sessions $s(k), ..., s(4m)$ is not increased in $H_{k:3}$.

Now, we first show that in session $s(k)$ of $H_{k:3}$, the hybrid is not aborted except with negligible probability. Since $H_{k:2}$ and $H_{k:3}$ proceed identically until the end of session $s(k)$, we have that in $H_{k:3}$, $\mathcal{A}$ does not cheat in session $s(k)$ except with negligible probability. So, it suffices to show that when session $s(k)$ is accepting and $\mathcal{A}$ does not cheat in session $s(k)$, the hybrid is not aborted in session $s(k)$. Recall that if $\mathcal{A}$ does not cheat in an accepting session (in which S is corrupted), we have the following.

1. Let $(\hat{a}_1^S, \hat{d}_1^S), \ldots (\hat{a}_{11n}^S, \hat{d}_{11n}^S)$ be the values that are committed in NMCom in Stage 7. Let $I_{\mathrm{bad}} \subset [11n]$ be the set such that $i \in I_{\mathrm{bad}}$ if and only if
   (a) $(\hat{a}_i^S, \hat{d}_i^S)$ is not valid in terms of the check in Stage 8-3b, i.e. $\hat{d}_i^S$ is not a valid decommitment of ψ_i^S w.r.t. Stage-0-1 RobCom, or $\hat{a}_i^S \oplus \psi_i^S$ does not equal the value z_i^S it received in Stage-2-1; **or**
   (b) S does not execute the i-th mS-OT in Stage 3 honestly using $\hat{s}_{i,0} \| \hat{s}_{i,1} \| \hat{\tau}_i^S$ as the input and randomness, where $\hat{s}_{i,0} \| \hat{s}_{i,1} \| \hat{\tau}_i^S$ is obtained from $\hat{r}_i^S = \hat{a}_i^S \oplus b_i^S$.

   Then, it holds that $|I_{\mathrm{bad}}| < 0.1n$.
2. For each $b \in \{0,1\}$, define $\boldsymbol{\rho}_b^{\mathsf{nm}} = (\rho_{b,i}^{\mathsf{nm}})_{i \in \Delta}$ as follows: $\rho_{b,i}^{\mathsf{nm}} \stackrel{\mathrm{def}}{=} \beta_{b,i} \oplus \hat{s}_{i,b \oplus \alpha_i}$ if $i \notin I_{\mathrm{bad}}$ and $\rho_{b,i}^{\mathsf{nm}} \stackrel{\mathrm{def}}{=} \perp$ otherwise. Then, for each $b \in \{0,1\}$, ρ_b^{nm} is either 0.9-close to a valid codeword $\boldsymbol{w} = (w_i)_{i \in \Delta}$ that satisfies $w_i = \rho_{b,i}^{\mathsf{nm}}$ for every $i \in \Gamma_R$ or 0.15-far from any such valid codeword.

We show that the above two imply that the hybrid is not aborted at the end of the session, i.e. that both of the following hold for each $b \in \{0,1\}$.

1. $|\{i \in \Delta \text{ s.t. } \rho_{b,i}^{\mathsf{ext}} \neq \perp\}| < 0.1n$.
2. $\boldsymbol{\rho}_b^{\mathsf{ext}}$ is either 0.9-close to a valid codeword $\boldsymbol{w} = (w_i)_{i \in \Delta}$ that satisfies $w_i = \rho_{b,i}^{\mathsf{ext}}$ for every $i \in \Gamma_R$ or 0.14-far from any such valid codeword.

Fix any $b \in \{0,1\}$. First, we notice that we can obtain $|\{i \in \Delta \text{ s.t. } \rho_{b,i}^{\mathsf{ext}} \neq \perp\}| < 0.1n$ from $|I_{\mathrm{bad}}| < 0.1n$ since we have $\{i \in \Delta \text{ s.t. } \rho_{b,i}^{\mathsf{ext}} \neq \perp\} \subseteq I_{\mathrm{bad}}$ from the definition of $\boldsymbol{\rho}_b^{\mathsf{ext}}$ and I_{bad}. Next, we observe that $\boldsymbol{\rho}_b^{\mathsf{ext}}$ is either 0.9-close to a valid codeword $\boldsymbol{w} = (w_i)_{i \in \Delta}$ that satisfies $w_i = \rho_{b,i}^{\mathsf{ext}}$ for every $i \in \Gamma_R$ or 0.14-far from any such valid codeword. From the assumption that $\mathcal{A}$ does not cheat, it suffices to consider the following two cases.

Case 1. $\boldsymbol{\rho}_b^{\mathsf{nm}}$ is 0.9-close to a valid codeword $\boldsymbol{w} = (w_i)_{i \in \Delta}$ that satisfies $w_i = \rho_{b,i}^{\mathsf{nm}}$ for every $i \in \Gamma_R \cap \Delta$: In this case, $\boldsymbol{\rho}_b^{\mathsf{ext}}$ is 0.9-close to $\boldsymbol{w}$, and $w_i = \rho_{b,i}^{\mathsf{ext}}$ holds for every $i \in \Gamma_R$. This is because for every i such that $\rho_{b,i}^{\mathsf{nm}} = w_i$, we have $\rho_{b,i}^{\mathsf{nm}} \neq \perp$ and thus we have $\rho_{b,i}^{\mathsf{nm}} = \rho_{b,i}^{\mathsf{ext}}$ from the definition of $\boldsymbol{\rho}_b^{\mathsf{nm}}$.

Case 2. $\boldsymbol{\rho}_b^{\mathsf{nm}}$ is 0.15-far from any valid codeword $\boldsymbol{w} = (w_i)_{i \in \Delta}$ that satisfies $w_i = \rho_{b,i}^{\mathsf{nm}}$ for every $i \in \Gamma_R \cap \Delta$: In this case, $\boldsymbol{\rho}_b^{\mathsf{ext}}$ is 0.14-far from any valid codeword $\boldsymbol{w'}$ that satisfies $w'_i = \rho_{b,i}^{\mathsf{ext}}$ for every $i \in \Gamma_R \cap \Delta$. This can be seen by observing the following: (1) for every $i \in \Gamma_R \cap \Delta$, we have $i \notin I_{\mathrm{bad}}$ (this is because the session is accepting) and hence $\rho_{b,i}^{\mathsf{ext}} = \rho_{b,i}^{\mathsf{nm}}$; (2) therefore, for any valid codeword $\boldsymbol{w'}$ that satisfies $w'_i = \rho_{b,i}^{\mathsf{ext}}$ for every $i \in \Gamma_R \cap \Delta$, we have that $\boldsymbol{w'}$ also satisfies $w'_i = \rho_{b,i}^{\mathsf{nm}}$ for every $i \in \Gamma_R \cap \Delta$; (3) then, from the assumption of this case, $\boldsymbol{\rho}_b^{\mathsf{nm}}$ is 0.15-far from $\boldsymbol{w'}$; (4) now, since $\boldsymbol{\rho}_b^{\mathsf{nm}}$ and $\boldsymbol{\rho}_b^{\mathsf{ext}}$ are 0.99-close (this follows from $|I_{\mathrm{bad}}| < 0.1n$), $\boldsymbol{\rho}_b^{\mathsf{ext}}$ is 0.14-far from $\boldsymbol{w'}$.

We therefore conclude that when session $s(k)$ is accepting and $\mathcal{A}$ does not cheat in session $s(k)$, the hybrid is not aborted in session $s(k)$.

Next, we show that in session $s(k)$ of $H_{k:3}$ if the hybrid is not aborted, we have $\mathsf{Value}(\boldsymbol{\rho}_u^{\mathsf{ext}}, \Gamma_R \cap \Delta) = \mathsf{Value}(\widetilde{\boldsymbol{\rho}}, \Gamma_R \cap \Delta)$. To show this, it suffices to show the following two claims.

Claim 6. *For any $\boldsymbol{x} = (x_i)_{i\in\Delta}, \boldsymbol{y} = (y_i)_{i\in\Delta}$ and a set Θ, we have $\mathsf{Value}(\boldsymbol{x}, \Theta) = \mathsf{Value}(\boldsymbol{y}, \Theta)$ if the following conditions hold.*

1. *$\boldsymbol{x}$ and $\boldsymbol{y}$ are 0.99-close, and $x_i = y_i$ holds for every $i \in \Theta$.*
2. *If $x_i \neq \bot$, then $x_i = y_i$.*
3. *$\boldsymbol{x}$ is either 0.9-close to a valid codeword $\boldsymbol{w} = (w_i)_{i\in\Delta}$ that satisfies $w_i = x_i$ for every $i \in \Theta$ or 0.14-far from any such valid codeword.*

Claim 7. *In session s(k) of $H_{k:3}$, if the sender S is corrupted, the session is accepting, and the session is not aborted the following hold.*

1. *$\boldsymbol{\rho}_u^{\mathsf{ext}}$ and $\widetilde{\boldsymbol{\rho}}$ are 0.99-close, and $\rho_{u,i}^{\mathsf{ext}} = \tilde{\rho}_i$ holds for every $i \in \Gamma_R \cap \Delta$.*
2. *If $\rho_{u,i}^{\mathsf{ext}} \neq \bot$, then $\rho_{u,i}^{\mathsf{ext}} = \tilde{\rho}_i$.*
3. *$\boldsymbol{\rho}_u^{\mathsf{ext}}$ is either 0.9-close to a valid codeword $\boldsymbol{w} = (w_i)_{i\in\Delta}$ that satisfies $w_i = \rho_{u,i}^{\mathsf{ext}}$ for every $i \in \Gamma_R \cap \Delta$ or 0.14-far from any such valid codeword.*

We prove each of the claims below.

Proof (of Claim 6). We consider the following two cases.

Case 1. $\boldsymbol{x}$ is 0.9-close to a valid codeword $\boldsymbol{w} = (w_i)_{i\in\Delta}$ that satisfies $w_i = x_i$ for every $i \in \Theta$: First, we observe that $\boldsymbol{y}$ is 0.9-close to $\boldsymbol{w}$. Since $\boldsymbol{w}$ is a valid codeword, we have $w_i \neq \bot$ for every $i \in \Delta$; thus, for every i such that $x_i = w_i$, we have $x_i \neq \bot$. Recall that from the assumed conditions, for every i such that $x_i \neq \bot$, we have $x_i = y_i$. Therefore, for every i such that $x_i = w_i$, we have $y_i = w_i$, which implies that $\boldsymbol{y}$ is 0.9-close to $\boldsymbol{w}$.

Next, we observe that $\boldsymbol{w}$ satisfies $w_i = y_i$ for every $i \in \Theta$. From the assumed conditions, we have $x_i = y_i$ for every $i \in \Theta$. Also, from the condition of this case, $\boldsymbol{w}$ satisfies $w_i = x_i$ for every $i \in \Theta$. From these two, we have that $\boldsymbol{w}$ satisfies $w_i = y_i$ for every $i \in \Theta$.

Now, from the definition of $\mathsf{Value}(\cdot,\cdot)$, we have $\mathsf{Value}(\boldsymbol{x}, \Theta) = \mathsf{Value}(\boldsymbol{y}, \Theta) = \mathsf{Decode}(\boldsymbol{w})$.

Case 2. $\boldsymbol{x}$ is 0.14-far from any valid codeword $\boldsymbol{w} = (w_i)_{i\in\Delta}$ that satisfies $w_i = x_i$ for every $i \in \Theta$: For any valid codeword $\boldsymbol{w'} = (w'_i)_{i\in\Delta}$ that satisfies $w'_i = y_i$ for every $i \in \Theta$, we observe that $\boldsymbol{y}$ is 0.1-far from $\boldsymbol{w'}$. Since we assume that $x_i = y_i$ holds for every $i \in \Theta$, we have $w'_i = x_i$ for every $i \in \Theta$. Therefore, from the assumption of this case, $\boldsymbol{x}$ is 0.14-far from $\boldsymbol{w'}$. Now, since we assume that $\boldsymbol{x}$ and $\boldsymbol{y}$ are 0.99-close, $\boldsymbol{y}$ is 0.1-far from $\boldsymbol{w'}$.

Now, from the definition of $\mathsf{Value}(\cdot,\cdot)$, we conclude that:

$$\mathsf{Value}(\boldsymbol{x}, \Theta) = \mathsf{Value}(\boldsymbol{y}, \Theta) = \bot.$$

Notice that from the assumed conditions, either Case 1 or Case 2 is true. This concludes the proof of Claim 6. □

Proof (of Claim 7). Recall that if the hybrid is not aborted in an accepting session in which S is corrupted, we have the following for each $b \in \{0,1\}$ in that session.

1. $|\{i \in \Delta \text{ s.t. } \rho^{\mathsf{ext}}_{b,i} \neq \bot\}| < 0.1n$.
2. $\boldsymbol{\rho}^{\mathsf{ext}}_b$ is either 0.9-close to a valid codeword $\boldsymbol{w} = (w_i)_{i\in\Delta}$ that satisfies $w_i = \rho^{\mathsf{ext}}_{b,i}$ for every $i \in \Gamma_R$ or 0.14-far from any such valid codeword.

Thus, it suffices to show that the above two imply the first condition in the claim statement.

First, we show that $\boldsymbol{\rho}^{\mathsf{ext}}_u$ and $\widetilde{\boldsymbol{\rho}}$ are 0.99-close and that $\rho^{\mathsf{ext}}_{u,i} = \tilde{\rho}_i$ holds for every $i \in \Gamma_R \cap \Delta$. From the definition of $\boldsymbol{\rho}^{\mathsf{ext}}_u$, we have $\rho^{\mathsf{ext}}_{u,i} = \tilde{\rho}_i$ for every i such that $\rho^{\mathsf{ext}}_{b,i} \neq \bot$ (this is because for every such i, $\mathcal{A}$ executed the i-th mS-OT in Stage 3 honestly using the coin obtained in Stage 2-1, which implies that the value $\widetilde{s}_i$ that was obtained from the i-th mS-OT is equal to the value s_{i,c_i} that was obtained by extracting the coin in Stage 2-1 by brute-force). Then, since $|\{i \in \Delta \text{ s.t. } \rho^{\mathsf{ext}}_{b,i} \neq \bot\}| < 0.1n$ and $\{i \in \Delta \text{ s.t. } \rho^{\mathsf{ext}}_{b,i} \neq \bot\} \cap \Gamma_R = \emptyset$ (the latter holds since the session would be rejected otherwise), we have that $\boldsymbol{\rho}^{\mathsf{ext}}_u$ and $\widetilde{\boldsymbol{\rho}}$ are 0.99-close and that $\rho^{\mathsf{ext}}_{u,i} = \tilde{\rho}_i$ holds for every $i \in \Gamma_R \cap \Delta$.

Next, we show that if $\rho^{\mathsf{ext}}_{u,i} \neq \bot$ then $\rho^{\mathsf{ext}}_{u,i} = \tilde{\rho}_i$. From the definition of $\boldsymbol{\rho}^{\mathsf{ext}}_u$, if $\rho^{\mathsf{ext}}_{u,i} \neq \bot$, $\mathcal{A}$ executed the i-th mS-OT in Stage 3 honestly using the coin obtained in Stage 2-1, so we have $\rho^{\mathsf{ext}}_{u,i} = \tilde{\rho}_i$ from the argument same as above.

This concludes the proof of Claim 7. □

This concludes the proof of Lemma 5. □

Lemma 6. *Assume that in $H_{k:3}$ ($k \in [4m]$), $\mathcal{A}$ does not cheat in sessions $s(k), \ldots, s(4m)$ except with negligible probability. Then,*

- *$H_{k:3}$ and $H_{k:4}$ are indistinguishable, and*
- *in $H_{k:4}$, $\mathcal{A}$ does not cheat in sessions $s(k), \ldots, s(4m)$ except with negligible probability.*

Recall that $H_{k:3}$ and $H_{k:4}$ differ only in that in session $s(k)$ of $H_{k:4}$, if S is corrupted and SM_k is third special message, α_i is a random bit rather than $\alpha_i = u \oplus c_i$ for every $i \in \Delta$ in Stage 6-1. Intuitively, we can prove this lemma by using the security of mS-OT: For every $i \notin \Gamma_S$, the choice bit c_i of the i-th mS-OT in Stage 3 is hidden from $\mathcal{A}$ and hence $\alpha_i = u \oplus c_i$ in $H_{k:3}$ is indistinguishable from a random bit. Formally, we prove this Lemma in the same way as we do for Claim 4 (we use the security of mS-OT rather than the hiding of ExtCom); the proof is given in Section C.3.

The next lemma is the counterpart of Lemma 3 when R is corrupted.

Lemma 7. *Assume that in $H_{k:4}$ ($k \in [4m]$), $\mathcal{A}$ does not cheat in sessions $s(k), \ldots, s(4m)$ except with negligible probability. Then,*

- *$H_{k:4}$ and $H_{k:5}$ are indistinguishable, and*
- *in $H_{k:5}$, $\mathcal{A}$ does not cheat in sessions $s(k), \ldots, s(4m)$ except with negligible probability.*

Note that hybrids $H_{k:4}$ and $H_{k:5}$ differ only in the values committed to in NMCom and ExtCom for the indices outside of Γ_R, in session $s(k)$, when R is corrupted. This lemma can be proven identically with Lemma 3. For completeness, we give a formal proof in Section C.4.

Lemma 8. *Assume that in $H_{k:5}$ ($k \in [4m]$), $\mathcal{A}$ does not cheat in sessions $s(k), \ldots, s(4m)$ except with negligible probability. Then,*

- *$H_{k:5}$ and $H_{k:6}$ are indistinguishable, and*
- *in $H_{k:6}$, $\mathcal{A}$ does not cheat in sessions $s(k), \ldots, s(4m)$ except with negligible probability.*

Since hybrids $H_{k:5}, H_{k:6}$ differ only in the inputs and the randomness that are used in some of the mS-OTs in Stage 3, this lemma can be proven identically with Lemma 4 (which in turn can be proven quite similarly to Lemma 3). For completeness, we give a formal proof in Section C.5.

Lemma 9. *Assume that in $H_{k:6}$ ($k \in [4m]$), $\mathcal{A}$ does not cheat in sessions $s(k), \ldots, s(4m)$ except with negligible probability. Then,*

- *$H_{k:6}$ and $H_{k:7}$ are indistinguishable, and*
- *in $H_{k:7}$, $\mathcal{A}$ does not cheat in sessions $s(k), \ldots, s(4m)$ except with negligible probability.*

Proof. We prove the lemma by considering the following intermediate hybrids $H'_{k:6}$, $H''_{k:6}$, and $H'''_{k:6}$.

Hybrid $H'_{k:6}$. $H'_{k:6}$ is the same as $H_{k:6}$ except that in session $s(k)$, if R is corrupted and SM_k is fourth special message, the following modifications are made.

1. As in $H_{k:7}$, the committed strings $\boldsymbol{a}^R = (a_1^R, \ldots, a_{11n}^R)$ are extracted by querying table $\mathcal{T}$, $\boldsymbol{r}^R = (r_1^R, \ldots, r_{11n}^R)$ is defined by $r_i^R \stackrel{\text{def}}{=} a_i^R \oplus b_i^R$ for each $i \in [11n]$, and r_i^R is parsed as $c_i \,\|\, \tau_i^R$ for each $i \in [11n]$. Also, I_0, I_1, and u^* are defined as in $H_{k:7}$.
2. In Stage 6, $\beta_{b,i}$ is a random bit rather than $\beta_{b,i} = \rho_{b,i} \oplus s_{i,b\oplus\alpha_i}$ for every $b \in \{0,1\}$ and $i \in \Delta \setminus I_b$. (Recall that, roughly, $I_b \subset \Delta$ is the set of indices on which $\mathcal{A}$ could have obtained $s_{i,b\oplus\alpha_i}$.)

Hybrid $H''_{k:6}$. $H''_{k:6}$ is the same as $H'_{k:6}$ except that in session $s(k)$, if R is corrupted and SM_k is fourth special message, the following modification is made.

1. The execution of the hybrid is aborted if both of $|I_0| \geq 6n+1$ and $|I_1| \geq 6n+1$ holds.
2. In Stage 6, $\boldsymbol{\rho}_{1-u^*} = \{\rho_{1-u^*,i}\}_{i\in\Delta}$ is a secret sharing of a random bit rather than that of v_{1-u^*}.

Hybrid $H'''_{k:6}$. $H'''_{k:6}$ is the same as $H''_{k:6}$ except that in session $s(k)$, if R is corrupted and SM_k is fourth special message, the following modification is made.

1. In Stage 6, $\beta_{b,i}$ is $\beta_{b,i} = \rho_{b,i} \oplus s_{i,b\oplus\alpha_i}$ rather than a random bit for every $b \in \{0,1\}$ and $i \in \Delta \setminus I_b$.

Notice that $H'''_{k:6}$ is identical with $H_{k:7}$.

Claim 8. *Assume that in $H_{k:6}$, $\mathcal{A}$ does not cheat in sessions $s(k), \ldots, s(4m)$ except with negligible probability. Then,*

- *$H_{k:6}$ and $H'_{k:6}$ are indistinguishable, and*
- *in $H'_{k:6}$, $\mathcal{A}$ does not cheat in sessions $s(k), \ldots, s(4m)$ except with negligible probability.*

Recall that $H_{k:6}$ and $H'_{k:6}$ differ only in that in session $s(k)$ of $H'_{k:6}$, if R is corrupted and SM_k is fourth special message, $\beta_{b,i}$ is a random bit rather than $\beta_{b,i} = \rho_{b,i} \oplus s_{i,b\oplus\alpha_i}$ for every $b \in \{0,1\}$ and $i \in \Delta \setminus I_b$. Intuitively, we can prove this claim by using the security of mS-OT: For every $i \in \Delta \setminus I_b$, $\mathcal{A}$ executed the i-th mS-OT honestly with choice bit $(1-b) \oplus \alpha_i$, and the sender's input and randomness of this mS-OT are not revealed in Stage 8; therefore, the value of $s_{i,b\oplus\alpha_i}$ is hidden from $\mathcal{A}$ and thus $\beta_{b,i} = \rho_{b,i} \oplus s_{i,b\oplus\alpha_i}$ is indistinguishable from a random bit. Formally, we prove this claim in the same way as we do for Claim 4 (we use the security of mS-OT rather than the hiding of ExtCom); a formal proof is given in Section C.6.

Claim 9. *Assume that in $H'_{k:6}$, $\mathcal{A}$ does not cheat in sessions $s(k), \ldots, s(4m)$ except with negligible probability. Then,*

– $H'_{k:6}$ *and* $H''_{k:6}$ *are indistinguishable, and*

– *in* $H''_{k:6}$, $\mathcal{A}$ *does not cheat in sessions* $s(k), \ldots, s(4m)$ *except with negligible probability.*

Proof. Recall that hybrid $H''_{k:6}$ differs from $H'_{k:6}$ in that in Stage 6 of session $s(k)$, either the hybrid is aborted or $\boldsymbol{\rho}_{1-u^*} = \{\rho_{1-u^*,i}\}_{i\in\Delta}$ is a secret sharing of a random bit rather than that of v_{1-u^*}.

For proving the lemma, it suffices to show that in session $s(k)$ of $H''_{k:6}$, the hybrid is not aborted (i.e. we have $|I_0| \leq 6n$ or $|I_1| \leq 6n$) except with negligible probability. To see that showing this is indeed sufficient for proving the lemma, observe the following: First, if the hybrid is not aborted, we have $|I_{1-u^*}| \leq 6n$, so $\beta_{1-u^*,i}$ is a random bit on at least $4n$ indices and thus $\rho_{1-u^*,i}$ is hidden on at least $4n$ indices, which implies that $H'_{k:6}$ and $H''_{k:6}$ are statistically indistinguishable. Second, since $H'_{k:6}$ and $H''_{k:6}$ proceed indentically until the beginning of Stage 6-2 of session $s(k)$, and

1. if the experiment is not aborted in session $s(k)$, $H'_{k:6}$ and $H''_{k:6}$ continue to proceed identically after Stage 6-2 of session $s(k)$, and
2. if the hybrid is aborted in session $s(k)$, $\mathcal{A}$ clearly does not cheat in any session after Stage 6-2 of session $s(k)$,

the probability that $\mathcal{A}$ cheat in sessions $s(k), ..., s(4m)$ is not increased in $H''_{k:6}$.

Hence, we show that in sessoion $s(k)$ of $H''_{k:7}$, the hybrid is not aborted in except with negligible probability, or equivalently, that we have $|I_0| \leq 6n$ or $|I_1| \leq 6n$ except with negligible probability. Since $H''_{k:7}$ proceeds identically with $H'_{k:7}$ until Stage 6-2 of session $s(k)$, we have that $\mathcal{A}$ does not cheat in session $s(k)$ of $H''_{k:7}$ except with negligible probability, so it suffices to show that in session $s(k)$ of $H''_{k:7}$, we have either $|I_0| \leq 6n$ or $|I_1| \leq 6n$ whenever $\mathcal{A}$ does not cheat. Assume that $\mathcal{A}$ does not cheat in session $s(k)$ of $H''_{k:7}$. Then, since $|\Gamma_R| = n$ and that the number of indices on which $\mathcal{A}$ does not execute mS-OT using the outcome of coin-tossing is at most n, we have $|I_0 \cap I_1| \leq 2n$. Now, since $I_0, I_1 \subset \Delta$ and thus $|I_0 \cup I_1| \leq |\Delta| = 10n$, we have $|I_0| + |I_1| \leq 12n$, and hence, we have either $|I_0| \leq 6n$ or $|I_1| \leq 6n$. □

Claim 10. *Assume that in* $H''_{k:6}$, $\mathcal{A}$ *does not cheat in sessions* $s(k), \ldots, s(4m)$ *except with negligible probability. Then,*

– $H''_{k:6}$ *and* $H'''_{k:6}$ *are indistinguishable, and*

– *in* $H'''_{k:6}$, $\mathcal{A}$ *does not cheat in sessions* $s(k), \ldots, s(4m)$ *except with negligible probability.*

Proof. This claim can be proven identically with Claim 8. □

This completes the proof of Lemma 9. □

From Lemmas 3 to 9, we conclude that the output of H_0 and that of $H_{4m:7}$ are indistinguishable, i.e., the output of the real world and that of the ideal world are indistinguishable. This concludes the proof of Theorem 5.

C Omitted Proofs

C.1 The Second Half of Proof of Lemma 1

Case 2. S **is corrupted in the** $i^*(n)$**-th session.** We show that when $\mathcal{A}$ cheats, we can break the hiding property of the $\mathsf{RobCom}(\phi^S)$ commitment in Stage 0-2 (i.e., the commitment by which

ϕ^R is committed to). From the definition of the invariant condition (Definition 7), when $\mathcal{A}$ cheats, we have $I_{\mathrm{bad}} \cap \Gamma_R = \emptyset$ and either $|I_{\mathrm{bad}}| \geq 0.1n$ or $\exists b \in \{0,1\}$ s.t. $\boldsymbol{\rho}_b^{\mathsf{nm}}$ is 0.85-close to but 0.1-far from a valid codeword $\boldsymbol{w} = (w_i)_{i \in \Delta}$ that satisfies $w_i = \rho_{b,i}^{\mathsf{nm}}$ for every $i \in \Gamma_R$, where I_{bad} and $\boldsymbol{\rho}_b^{\mathsf{nm}}$ are defined from the committed values of the NMCom commitments in Stage 7. Similar to Case 1, we first show that we can "approximate" I_{bad} and $\boldsymbol{\rho}_b^{\mathsf{nm}}$ by extracting the committed values of the ExtCom commitments in Stage 7 using its extractability.

First, we observe that if we extract the committed values of the ExtCom commitments in Stage 7 of the $i^*(n)$-th session, the extracted values, $(\hat{a}_1^S, \hat{d}_1^S, \hat{e}_1^S), \ldots, (\hat{a}_{11n}^S, \hat{d}_{11n}^S, \hat{e}_{11n}^S)$, satisfy the following.

- Let $\hat{I}_{\mathrm{bad}} \subset [11n]$ be a set such that $i \in \hat{I}_{\mathrm{bad}}$ if and only if
  1. $((\hat{a}_1^S, \hat{d}_1^S), \hat{e}_1^S)$ is not a valid decommitment of the i-th NMCom commitment in Stage 7, **or**
  2. $(\hat{a}_i^S, \hat{d}_i^S)$ is not valid in terms of the check in Stage 8-3b, i.e. $\hat{d}_i^S$ is not a valid decommitment of ψ_i^S w.r.t. Stage-0-1 RobCom, or $\hat{a}_i^S \oplus \psi_i^S$ does not equal the value z_i^S it received in Stage-2-1; **or**
  3. S does not execute the i-th mS-OT in Stage 3 honestly using $\hat{s}_{i,0} \,\|\, \hat{s}_{i,1} \,\|\, \hat{\tau}_i^S$ as the input and randomness, where $\hat{s}_{i,0} \,\|\, \hat{s}_{i,1} \,\|\, \hat{\tau}_i^S$ is obtained from $\hat{r}_i^S = \hat{a}_i^S \oplus b_i^S$.

  Also, for each $b \in \{0,1\}$, let $\hat{\boldsymbol{\rho}}_b = (\hat{\rho}_{b,i})_{i \in \Delta}$ be defined as follows: $\hat{\rho}_{b,i} \stackrel{\mathrm{def}}{=} \beta_{b,i} \oplus \hat{s}_{i,b \oplus \alpha_i}$ if $i \notin \hat{I}_{\mathrm{bad}}$ and $\hat{\rho}_{b,i} \stackrel{\mathrm{def}}{=} \perp$ otherwise. Then, we have
  - $\hat{I}_{\mathrm{bad}} \cap \Gamma_R = \emptyset$, and
  - either $|\hat{I}_{\mathrm{bad}}| \geq 0.1n$ or there exists $b \in \{0,1\}$ such that $\hat{\rho}_b$ is 0.8-close to but 0.1-far from a valid codeword $\boldsymbol{w} = (w_i)_{i \in \Delta}$ that satisfies $w_i = \hat{\rho}_{b,i}$ for every $i \in \Gamma_R$

  with probability at least $1/2p(n)$.

More precisely, we observe that when $\mathcal{A}$ cheats in the $i^*(n)$-th session, the extracted values satisfied the above condition except with negligible probability. Recall that when $\mathcal{A}$ cheats, the cut-and-choose in Stage 8 is accepting but we have

- $|I_{\mathrm{bad}}| \geq 0.1n$, or
- $\exists b \in \{0,1\}$ s.t. ρ_b^{nm} is 0.85-close to but 0.1-far from a valid codeword $\boldsymbol{w} = (w_i)_{i \in \Delta}$ that satisfies $w_i = \rho_{b,i}^{\mathsf{nm}}$ for every $i \in \Gamma_R$.

Also, notice that we have $\hat{I}_{\mathrm{bad}} \cap \Gamma_R = \emptyset$ when the cut-and-choose in Stage 8 is accepting, and have $|\hat{I}_{\mathrm{bad}}| \geq 0.1n$ when $|I_{\mathrm{bad}}| \geq 0.1n$ (this is because we have $I_{\mathrm{bad}} \subseteq \hat{I}_{\mathrm{bad}}$ from the definitions of $I_{\mathrm{bad}}, \hat{I}_{\mathrm{bad}}$). Hence, to show that the extracted values satisfy the above condition when $\mathcal{A}$ cheats, it suffices to show that when $\exists b^* \in \{0,1\}$ s.t. $\boldsymbol{\rho}_{b^*}^{\mathsf{nm}}$ is 0.85-close to but 0.1-far from a valid codeword $\boldsymbol{w} = (w_i)_{i \in \Delta}$ that satisfies $w_i = \rho_{b^*,i}^{\mathsf{nm}}$ for every $i \in \Gamma_R$, we have either $|\hat{I}_{\mathrm{bad}}| \geq 0.1n$ or $\hat{\boldsymbol{\rho}}_{b^*}$ is 0.8-close to but 0.1-far from $\boldsymbol{w}$ and satisfies $w_i = \hat{\rho}_{b^*,i}$ for every $i \in \Gamma_R$. This can be shown as follows.

- If $|\hat{I}_{\mathrm{bad}}| \geq 0.1n$, we are done.
- If $|\hat{I}_{\mathrm{bad}}| < 0.1n$, we have that $\hat{\rho}_{b^*}$ is 0.8-close to but 0.1-far from $\boldsymbol{w}$ and satisfies $w_i = \hat{\rho}_{b^*,i}$ for every $i \in \Gamma_R$. This is because if $|\hat{I}_{\mathrm{bad}}| < 0.1n$,
  1. $\hat{\boldsymbol{\rho}}_{b^*}$ is 0.8-close to $\boldsymbol{w}$ since it is 0.99-close to $\boldsymbol{\rho}_{b^*}^{\mathsf{nm}}$ when $|\hat{I}_{\mathrm{bad}}| < 0.1n$, and $\boldsymbol{\rho}_{b^*}^{\mathsf{nm}}$ is 0.85-close to $\boldsymbol{w}$,

2. $\hat{\boldsymbol{\rho}}_{b^*}$ is 0.1-far from $\boldsymbol{w}$ since for every i such that $\rho^{\mathsf{nm}}_{b^*,i} \neq w_i$, we have $\hat{\rho}_{b^*,i} \neq w_i$ from the definition of $\hat{\boldsymbol{\rho}}$, and
3. $\hat{\boldsymbol{\rho}}_{b^*}$ satisfies $w_i = \hat{\rho}_{b^*,i}$ for every $i \in \Gamma_R$ since we have $\hat{\rho}_{b^*,i} = \rho^{\mathsf{nm}}_{b^*,i}$ for every $i \in \Gamma_R$ when the cut-and-choose in Stage 8 is accepting, and $\boldsymbol{\rho}^{\mathsf{nm}}_{b^*}$ satisfies $w_i = \rho^{\mathsf{nm}}_{b^*,i}$ for every $i \in \Gamma_R$.

Based on this observation, we derive contradiction by considering the following adversary $\mathcal{A}_{\mathsf{RobCom}}$ against the hiding property of RobCom.

$\mathcal{A}_{\mathsf{RobCom}}$ receives a RobCom commitment c^* in which either ϕ^0_R or ϕ^1_R is committed. Then, $\mathcal{A}_{\mathsf{RobCom}}$ internally executes the experiment H_0 honestly except that in the $i^*(n)$-th session, $\mathcal{A}_{\mathsf{RobCom}}$ uses c^* as the commitment in Stage 0-2 (i.e., as the RobCom commitment in which R commits to string which will be used to mask Γ_R in Stage 1-2). In Stage 1-2, $\mathcal{A}_{\mathsf{RobCom}}$ always use ϕ^1_R to mask Γ_R. When the experiment H_0 reaches Stage 7 of the $i^*(n)$-th session, $\mathcal{A}_{\mathsf{RobCom}}$ extracts the committed values of the ExtCom commitments in this stage by using its extractability. Let $\hat{I}_{\mathrm{bad}}$ and $\hat{\boldsymbol{\rho}}_b$ ($b \in \{0,1\}$) be defined as above from the extracted values. Then, $\mathcal{A}_{\mathsf{RobCom}}$ outputs 1 if and only if

- $\hat{I}_{\mathrm{bad}} \cap \Gamma_R = \emptyset$, and
- either $|\hat{I}_{\mathrm{bad}}| \geq 0.1n$ or there exists $b \in \{0,1\}$ such that $\hat{\boldsymbol{\rho}}_b$ is 0.8-close to but 0.1-far from a valid codeword $\boldsymbol{w} = (w_i)_{i \in \Delta}$ that satisfies $w_i = \hat{\rho}_{b,i}$ for every $i \in \Gamma_R$.

When $\mathcal{A}_{\mathsf{RobCom}}$ receives a commitment to ϕ^1_R, $\mathcal{A}_{\mathsf{RobCom}}$ outputs 1 with probability $1/2p(n)$ (this follows from the above observation). It thus suffices to see that when $\mathcal{A}_{\mathsf{RobCom}}$ receives a commitment to ϕ^0_R, $\mathcal{A}_{\mathsf{RobCom}}$ outputs 1 with exponentially small probability. This can be seen by noting that $\phi^R_1 \oplus \Gamma_R$ is a pure random string now, and thus the following probabilities are exponentially small.

1. the probability that $|\hat{I}_{\mathrm{bad}}| \geq 0.1n$ but $\hat{I}_{\mathrm{bad}} \cap \Gamma_R = \emptyset$
2. the probability that there exists $b \in \{0,1\}$ such that $\hat{\boldsymbol{\rho}}_b$ is 0.8-close to but 0.1-far from a valid codeword $\boldsymbol{w} = (w_i)_{i \in \Delta}$ that satisfies $w_i = \hat{\rho}_{b,i}$ for every $i \in \Gamma_R$

Hence, $\mathcal{A}_{\mathsf{RobCom}}$ breaks the hiding property of RobCom.

C.2 Proof of Lemma 4

Proof. Recall that hybrids $H_{k:1}, H_{k:2}$ differ only in the input and the randomness that are used in some of the mS-OTs in Stage 3, where those that are derived from the outcomes of the coin tossing is used in $H_{k:1}$ and random inputs and true randomness are used in $H_{k:2}$. We first show the indistinguishability between $H_{k:1}$ and $H_{k:2}$, relying on the hiding property of RobCom.

Assume for contradiction that $H_{k:1}$ and $H_{k:2}$ are distinguishable. We build an efficient adversary $\mathcal{A}_{\mathsf{RobCom}}$ that breaks the hiding property of RobCom.

The adversary $\mathcal{A}_{\mathsf{RobCom}}$ internally executes $H_{k:1}$ with the following modification: in Stage 0-2 of session $s(k)$, it picks two random strings $\psi^R = \psi^R_1 \| \ldots, \psi^R_{11n}$ and $\tilde{\psi}^R = \tilde{\psi}^R_1 \| \ldots, \tilde{\psi}^R_{11n}$ and sends $\{\psi^R_i\}_{i \notin \Gamma_S}$ and $\{\tilde{\psi}^R_i\}_{i \notin \Gamma_S}$ to the external committer and receives back RobCom^{f_R} commitments (in which either $\{\psi^R_i\}_{i \notin \Gamma_S}$ or $\{\tilde{\psi}^R_i\}_{i \notin \Gamma_S}$ are committed in parallel). Then in Stage 2-2 of session $s(k)$, $\mathcal{A}_{\mathsf{RobCom}}$ always use ψ^R_i's to mask a^R_i (i.e. $z^R_i := a^R_i \oplus \psi^R_i$ for all $i \in [11n]$). in the subsequent stages, $\mathcal{A}$ proceeds the experiment as in $H_{k:1}$. After the execution of $H_{k:1}$ finishes, $\mathcal{A}_{\mathsf{RobCom}}$ outputs whatever $\mathcal{Z}$ outputs in the experiment.

When $\mathcal{A}_{\mathsf{RobCom}}$ receives commitments to $\{\psi_i^R\}_{i\notin\Gamma_S}$, the internally executed experiment is identical with $H_{k:1}$, whereas when $\mathcal{A}_{\mathsf{RobCom}}$ receives commitments to $\{\tilde{\psi}_i^R\}_{i\notin\Gamma_S}$, the internally executed experiment is identical with $H_{k:2}$ (this is because when $\mathcal{A}_{\mathsf{RobCom}}$ receives commitments to $(\tilde{\psi}_i^R)_{i\notin\Gamma_S}$, the values $z_i^R = \psi_i^R \oplus a_i^R$ (thus the values $r_i^R = a_i^R \oplus b_i^R$) for each $i \notin \Gamma_S$ are uniformly random for $\mathcal{A}$. Hence the mS-OT for each $i \notin \Gamma_S$ is executed with a random input and true randomness). Hence, from the assumption that $H_{k:1}$ and $H_{k:2}$ are distinguishable, $\mathcal{A}_{\mathsf{RobCom}}$ distinguishes RobCom commitments.

We next show that in $H_{k:2}$, $\mathcal{A}$ does not cheat in sessions $s(k), \ldots, s(4m)$. Assume for contradiction that in $H_{k:2}$, $\mathcal{A}$ cheats in one of those sessions, say, session $s(j)$, with non-negligible probability. Then, from an average argument, we can fix the execution of the experiment up until SM_k (inclusive) in such a way that even after being fixed, $\mathcal{A}$ cheats in session $s(j)$ only with negligible probability in $H_{k:1}$ but with non-negligible probability in $H_{k:2}$. Then, we can break the robust non-malleability of NMCom as follows.

> The adversary $\mathcal{A}_{\mathsf{NMCom}}$, who interacts with a committer of RobCom and a receiver of NMCom, internally executes $H_{k:1}$ from SM_k using the non-uniform advice. In Stage 0-2 of session $s(k)$, $\mathcal{A}_{\mathsf{NMCom}}$ chooses random strings $\tilde{\psi}^R = \tilde{\psi}_1^R \| \ldots \| \tilde{\psi}_{11n}^R$ in addition to $\psi^R = \psi_1^R \| \ldots \| \psi_{11n}^R$, sends $\{\psi_i^R\}_{i\notin\Gamma_S}$ and $\{\tilde{\psi}_i^R\}_{i\notin\Gamma_S}$ to the external committer and receives back parallel RobCom commitments (in which either $\{\psi_i^R\}_{i\notin\Gamma_S}$ or $\{\tilde{\psi}_i^R\}_{i\notin\Gamma_S}$ are committed to), and feeds them into $H_{k:1}$. Then in Stage 2-2 of session $s(k)$, $\mathcal{A}_{\mathsf{NMCom}}$ always use ψ_i^R's to mask a_i^R (i.e. $z_i^R := a_i^R \oplus \psi_i^R$ for all $i \in [11n]$). Also, in session $s(j)$, $\mathcal{A}_{\mathsf{NMCom}}$ forwards the NMCom commitments from $\mathcal{A}$ to the external receiver. After the execution of $H_{k:1}$ finishes, $\mathcal{A}_{\mathsf{NMCom}}$ outputs its view.
> The distinguisher $\mathcal{D}_{\mathsf{NMCom}}$ takes as input the view of $\mathcal{A}_{\mathsf{NMCom}}$ and the values committed by $\mathcal{A}_{\mathsf{NMCom}}$ (which are equal to the values committed to by $\mathcal{A}$ in session $s(j)$ in the internally executed experiment). $\mathcal{D}_{\mathsf{NMCom}}$ then outputs 1 if and only if $\mathcal{A}$ cheated in session $s(j)$.

When $\mathcal{A}_{\mathsf{NMCom}}$ receives commitments to $\{\psi_i^R\}_{i\notin\Gamma_S}$, the internally executed experiment is identical with $H_{k:1}$, whereas when $\mathcal{A}_{\mathsf{Com}}$ receives commitments to $\{\tilde{\psi}_i^R\}_{i\notin\Gamma_S}$, the internally executed experiment is identical with $H_{k:2}$. Hence, from the assumption that $\mathcal{A}$ cheats in session $s(j)$ with negligible probability in $H_{k:1}$ but with non-negligible probability in $H_{k:2}$, $\mathcal{A}_{\mathsf{NMCom}}$ breaks the robust non-malleability of NMCom.

This completes the proof of Lemma 4. □

C.3 Proof of Lemma 6

Proof. Recall that $H_{k:3}$ and $H_{k:4}$ differ only in that in session $s(k)$ of $H_{k:4}$, if S is corrupted and SM_k is third special message, α_i is a random bit rather than $\alpha_i = u \oplus c_i$ for every $i \in \Delta$ in Stage 6-1.

We first show the indistinguishability between $H_{k:3}$ and $H_{k:4}$. Intuitively, the indistinguishability follows from the security of mS-OT: For every $i \notin \Gamma_S$, the choice bit c_i of the i-th mS-OT in Stage 3 is hidden from $\mathcal{A}$ and hence $\alpha_i = u \oplus c_i$ in $H_{k:3}$ is indistinguishable from a random bit. Formally, we consider the following security game against cheating sender S^* of mS-OT.

> The cheating sender S^* first participates in $10n$ instances of mS-OTs in parallel with an honest receiver R, who uses a random input $c_i \in \{0,1\}$ in the i-th instance. After the execution with R, S^* receives either the choice bits $\{c_i\}$ or random bits and then guesses which is the case. If S^* guesses correctly, we say that S^* wins the game.

From the security of mS-OT against malicious senders, any cheating S^* wins the game with probability at most $1/2+\mathsf{negl}(n)$. Now, we assume for contradiction that $H_{k:3}$ and $H_{k:4}$ are distinguishable, and we derive a contradiction by constructing an adversary who wins the above game with probability non-negligibly higher than $1/2$. From an average argument, we can fix the execution of the experiment up until SM_k (inclusive) in such a way that even after being fixed, $H_{k:3}$ and $H_{k:4}$ are still distinguishable. Then, by considering the transcript up until SM_k and the table $\mathcal{T}$ as non-uniform advice, we can obtain an adversary who wins the above game with probability non-negligibly higher than $1/2$ as follows.

> The adversary $\mathcal{A}_{\mathsf{OT}}$ internally executes $H_{k:3}$ from SM_k using the non-uniform advice. In Stage 3 of session $s(k)$, $\mathcal{A}_{\mathsf{OT}}$ executes the i-th mS-OT by itself for every $i \in \mathit{\Gamma}_S$ but obtains the other $10n$ instances of mS-OT from the external receiver. (Recall that in $H_{k:3}$, the subset $\mathit{\Gamma}_S$ is extracted in Stage 1-1.) Then, in Stage 6 of session $s(k)$, $\mathcal{A}_{\mathsf{OT}}$ receives bits $\{c_i^*\}_{i\in\mathit{\Delta}}$ from the external receiver and uses them to compute $\{\alpha\}_{i\in\mathit{\Delta}}$, i.e., $\alpha_i \stackrel{\text{def}}{=} u \oplus c_i^*$. After the execution of $H_{k:3}$ finishes, $\mathcal{A}_{\mathsf{OT}}$ outputs whatever $\mathcal{Z}$ outputs in the experiment.

When $\mathcal{A}_{\mathsf{OT}}$ receives the choice bits of the mS-OTs as $\{c_i^*\}_{i\in\mathit{\Delta}}$, the internally executed experiment is identical with $H_{k:3}$, whereas when $\mathcal{A}_{\mathsf{OT}}$ receives random bits as $\{c_i^*\}_{i\in\mathit{\Delta}}$, the internally executed experiment is identical with $H_{k:4}$. Hence, from the assumption that $H_{k:3}$ and $H_{k:4}$ are distinguishable, $\mathcal{A}_{\mathsf{OT}}$ wins the game with probability non-negligibly higher than $1/2$.

We next show that in $H_{k:4}$, $\mathcal{A}$ does not cheat in sessions $s(k),\ldots,s(4m)$. (The argument below is similar to the one in the proof of Lemma 3.) Assume for contradiction that in $H_{k:4}$, $\mathcal{A}$ cheats in one of those sessions, say, session $s(j)$, with non-negligible probability. Then, from an average argument, we can fix the execution of the experiment up until SM_k (inclusive) in such a way that even after being fixed, $\mathcal{A}$ cheats in session $s(j)$ only with negligible probability in $H_{k:3}$ but with non-negligible probability in $H_{k:4}$. Then, by considering the transcript up until SM_k and the table $\mathcal{T}$ as non-uniform advice, we can break the robust non-malleability of NMCom as follows.

> The adversary $\mathcal{A}_{\mathsf{NMCom}}$, who participates in the above game of mS-OT while interacting with a receiver of NMCom, internally executes $H_{k:3}$ from SM_k using the non-uniform advice. In Stage 3 of session $s(k)$, $\mathcal{A}_{\mathsf{OT}}$ executes the i-th mS-OT by itself for every $i \in \mathit{\Gamma}_S$ but obtains the other $10n$ instances of mS-OT from the external receiver. Then, in Stage 6 of session $s(k)$, $\mathcal{A}_{\mathsf{OT}}$ receives bits $\{c_i^*\}_{i\in\mathit{\Delta}}$ from the external receiver and uses them to compute $\{\alpha\}_{i\in\mathit{\Delta}}$, i.e., $\alpha_i \stackrel{\text{def}}{=} u \oplus c_i^*$. Also, in session $s(j)$, $\mathcal{A}_{\mathsf{NMCom}}$ forwards the NMCom commitments from $\mathcal{A}$ to the external receiver. After the execution of $H_{k:3}$ finishes, $\mathcal{A}_{\mathsf{NMCom}}$ outputs its view.
>
> The distinguisher $\mathcal{D}_{\mathsf{NMCom}}$ takes as input the view of $\mathcal{A}_{\mathsf{NMCom}}$ and the values committed by $\mathcal{A}_{\mathsf{NMCom}}$ (which are equal to the values committed to by $\mathcal{A}$ in session $s(j)$ in the internally executed experiment). $\mathcal{D}_{\mathsf{NMCom}}$ then outputs 1 if and only if $\mathcal{A}$ cheated in session $s(j)$.

When $\mathcal{A}_{\mathsf{OT}}$ receives the choice bits of the mS-OTs as $\{c_i^*\}_{i\in\mathit{\Delta}}$, the internally executed experiment is identical with $H_{k:3}$, whereas when $\mathcal{A}_{\mathsf{OT}}$ receives random bits as $\{c_i^*\}_{i\in\mathit{\Delta}}$, the internally executed experiment is identical with $H_{k:4}$. Hence, from the assumption that $\mathcal{A}$ cheats in session $s(j)$ with negligible probability in $H_{k:3}$ but with non-negligible probability in $H_{k:4}$, $\mathcal{A}_{\mathsf{NMCom}}$ breaks the robust non-malleability of NMCom.

This completes the proof of Lemma 6. □

C.4 Proof of Lemma 7

Proof. Recall that hybrids $H_{k:4}, H_{k:5}$ differ only in the values committed to in NMCom and ExtCom for the indices outside of Γ_R. Since the binding property of RobCom guarantees that the subset opened in Stage 7 is equal to Γ_R, those commitments are never opened, and the check in Stage 8 does not fail in both hybrids.

We prove the lemma by using a hybrid argument. Specifically, we consider the following intermediate hybrid $H'_{k:5}$.

- $H'_{k:5}$ is the same as $H_{k:4}$ except that in session $s(k)$, if R is corrupted and SM_k is second special message,
  - the committed subset Γ_R is extracted by querying the table $\mathcal{T}$, and
  - the value committed in the i-th ExtCom commitment in Stage 7 is switched to an all-zero string for every $i \notin \Gamma_R$.

Claim 11. *Assume that in $H_{k:4}$, $\mathcal{A}$ does not cheat in sessions $s(k), \ldots, s(4m)$ except with negligible probability. Then,*

- *$H_{k:4}$ and $H'_{k:5}$ are indistinguishable, and*
- *in $H'_{k:5}$, $\mathcal{A}$ does not cheat in sessions $s(k), \ldots, s(4m)$ except with negligible probability.*

Proof. We first show the indistinguishability between $H_{k:4}$ and $H'_{k:5}$. Assume for contradiction that $H_{k:4}$ and $H'_{k:5}$ are distinguishable. From an average argument, we can fix the execution of the experiment up until SM_k (inclusive) in such a way that even after being fixed, $H_{k:4}$ and $H'_{k:5}$ are still distinguishable. Then, by considering the transcript up until SM_k and the table $\mathcal{T}$ as non-uniform advice, we can break the hiding property of ExtCom as follows.

> The adversary $\mathcal{A}_{\mathsf{ExtCom}}$ internally executes $H_{k:4}$ from SM_k using the non-uniform advice. In Stage 7 of session $s(k)$, $\mathcal{A}_{\mathsf{ExtCom}}$ sends $(a_i^S, d_i^S, e_i^S)_{i \notin \Gamma_R}$ and $(0,0,0)_{i \notin \Gamma_R}$ to the external committer, receives back ExtCom commitments (in which either $(a_i^S, d_i^S, e_i^S)_{i \notin \Gamma_R}$ or $(0,0,0)_{i \notin \Gamma_R}$ are committed to), and feeds them into $H_{k:4}$. After the execution of $H_{k:4}$ finishes, $\mathcal{A}_{\mathsf{ExtCom}}$ outputs whatever $\mathcal{Z}$ outputs in the experiment.

When $\mathcal{A}_{\mathsf{ExtCom}}$ receives commitments to $(a_i^S, d_i^S, e_i^S)_{i \notin \Gamma_R}$, the internally executed experiment is identical with $H_{k:4}$, whereas when $\mathcal{A}_{\mathsf{ExtCom}}$ receives a commitments to $(0,0,0)_{i \notin \Gamma_R}$, the internally executed experiment is identical with $H'_{k:5}$. Hence, from the assumption that $H_{k:4}$ and $H'_{k:5}$ are distinguishable (even after being fixed up until SM_k), $\mathcal{A}_{\mathsf{ExtCom}}$ distinguishes ExtCom commitments.

We next show that in $H'_{k:5}$, $\mathcal{A}$ does not cheat in sessions $s(k), \ldots, s(4m)$. Assume for contradiction that in $H'_{k:5}$, $\mathcal{A}$ cheats in one of those sessions, say, session $s(j)$, with non-negligible probability. Then, from an average argument, we can fix the execution of the experiment up until SM_k (inclusive) in such a way that even after being fixed, $\mathcal{A}$ cheats in session $s(j)$ only with negligible probability in $H_{k:4}$ but with non-negligible probability in $H'_{k:5}$. Then, by considering the transcript up until SM_k and the table $\mathcal{T}$ as non-uniform advice, we can break the robust non-malleability of NMCom as follows.

> The man-in-the-meddle adversary $\mathcal{A}_{\mathsf{NMCom}}$ internally executes $H_{k:4}$ from SM_k using the non-uniform advice. In Stage 7 of session $s(k)$, $\mathcal{A}_{\mathsf{NMCom}}$ sends $(a_i^S, d_i^S, e_i^S)_{i \notin \Gamma_R}$ and $(0,0,0)_{i \notin \Gamma_R}$ to the external committer, receives back ExtCom commitments (in which either $(a_i^S, d_i^S, e_i^S)_{i \notin \Gamma_R}$

or $(0,0,0)_{i\notin\Gamma_R}$ are committed to), and feeds them into $H_{k:4}$. Also, in session $s(j)$, $\mathcal{A}_{\mathsf{NMCom}}$ forwards the NMCom commitments from $\mathcal{A}$ to the external receiver. After the execution of $H_{k:4}$ finishes, $\mathcal{A}_{\mathsf{NMCom}}$ outputs its view.
The distinguisher $\mathcal{D}_{\mathsf{NMCom}}$ takes as input the view of $\mathcal{A}_{\mathsf{NMCom}}$ and the values committed by $\mathcal{A}_{\mathsf{NMCom}}$ (which are equal to the values committed to by $\mathcal{A}$ in session $s(j)$ in the internally executed experiment). $\mathcal{D}_{\mathsf{NMCom}}$ then outputs 1 if and only if $\mathcal{A}$ cheated in session $s(j)$.

When $\mathcal{A}_{\mathsf{NMCom}}$ receives commitments to $(a_i^S, d_i^S, e_i^S)_{i\notin\Gamma_R}$, the internally executed experiment is identical with $H_{k:4}$, whereas when $\mathcal{A}_{\mathsf{NMCom}}$ receives a commitments to $(0,0,0)_{i\notin\Gamma_R}$, the internally executed experiment is identical with $H'_{k:5}$. Hence, from the assumption that $\mathcal{A}$ cheats in session $s(j)$ with negligible probability in $H_{k:4}$ and $H'_{k:5}$, $\mathcal{A}_{\mathsf{NMCom}}$ breaks the non-malleability of NMCom. □

Claim 12. *Assume that in $H'_{k:5}$, $\mathcal{A}$ does not cheat in sessions $s(k),\ldots,s(4m)$ except with negligible probability. Then,*

- *$H'_{k:5}$ and $H_{k:5}$ are indistinguishable, and*
- *in $H_{k:5}$, $\mathcal{A}$ does not cheat in sessions $s(k),\ldots,s(4m)$ except with negligible probability.*

Proof. We first notice that the indistinguishability between $H'_{k:5}$ and $H_{k:5}$ can be shown as in the proof of Claim 11. (The only difference is that we use the hiding property of NMCom rather than that of ExtCom.)

We next show that in $H_{k:5}$, $\mathcal{A}$ does not cheat in sessions $s(k),\ldots,s(4m)$. Assume for contradiction that in $H_{k:5}$, $\mathcal{A}$ cheats in one of those sessions, say, session $s(j)$, with non-negligible probability. Then, from an average argument, we can fix the execution of the experiment up until SM_k (inclusive) in such a way that even after being fixed, $\mathcal{A}$ cheats in session $s(j)$ only with negligible probability in $H'_{k:5}$ but with non-negligible probability in $H_{k:5}$. Then, by considering the transcript up until SM_k and the table $\mathcal{T}$ as non-uniform advice, we can break the non-malleability of NMCom as follows.

The man-in-the-meddle adversary $\mathcal{A}_{\mathsf{NMCom}}$ internally executes $H'_{k:5}$ from SM_k using the non-uniform advice. In Stage 7 of session $s(k)$, $\mathcal{A}_{\mathsf{NMCom}}$ sends $(a_i^S, d_i^S)_{i\notin\Gamma_R}$ and $(0,0)_{i\notin\Gamma_R}$ to the external committer, receives back NMCom commitments (in which either $(a_i^S, d_i^S)_{i\notin\Gamma_R}$ or $(0,0)_{i\notin\Gamma_R}$ are committed to), and feeds them into $H'_{k:5}$. Also, in session $s(j)$, $\mathcal{A}_{\mathsf{NMCom}}$ forwards the NMCom commitments from $\mathcal{A}$ to the external receiver. After the execution of $H'_{k:5}$ finishes, $\mathcal{A}_{\mathsf{NMCom}}$ outputs its view.
The distinguisher $\mathcal{D}_{\mathsf{NMCom}}$ takes as input the view of $\mathcal{A}_{\mathsf{NMCom}}$ and the values committed by $\mathcal{A}_{\mathsf{NMCom}}$ (which are equal to the values committed to by $\mathcal{A}$ in session $s(j)$ in the internally executed experiment). $\mathcal{D}_{\mathsf{NMCom}}$ then outputs 1 if and only if $\mathcal{A}$ cheated in session $s(j)$.

When $\mathcal{A}_{\mathsf{NMCom}}$ receives commitments to $(a_i^S, d_i^S)_{i\notin\Gamma_R}$, the internally executed experiment is identical with $H'_{k:5}$, whereas when $\mathcal{A}_{\mathsf{NMCom}}$ receives a commitments to $(0,0)_{i\notin\Gamma_R}$, the internally executed experiment is identical with $H_{k:5}$. Hence, from the assumption that $\mathcal{A}$ cheats in session $s(j)$ with negligible probability in $H'_{k:5}$ but with non-negligible probability in $H_{k:5}$, $\mathcal{A}_{\mathsf{NMCom}}$ breaks the non-malleability of NMCom. □

This completes the proof of Lemma 7. □

C.5 Proof of Lemma 8

Proof. Recall that hybrids $H_{k:5}, H_{k:6}$ differ only in the inputs and the randomness that are used in some of the mS-OTs in Stage 3, where those that are derived from the outcomes of the coin tossing is used in $H_{k:5}$ and random inputs and true randomness are used in $H_{k:6}$.

First, we show the indistinguishability. Assume for contradiction that $H_{k:5}$ and $H_{k:6}$ are computationally distinguishable. We build an efficient adversary $\mathcal{A}_{\mathsf{RobCom}}$ that breaks the hiding property of RobCom.

> The adversary $\mathcal{A}_{\mathsf{RobCom}}$ internally executes $H_{k:5}$ with the following modification: in Stage 0-1 of session $s(k)$, it picks two random strings $\psi^S = \psi_1^S \| \dots, \psi_{11n}^S$ and $\tilde{\psi}^S = \tilde{\psi}_1^S \| \dots, \tilde{\psi}_{11n}^S$ and sends $\{\psi_i^S\}_{i \notin \Gamma_R}$ and $\{\tilde{\psi}_i^S\}_{i \notin \Gamma_R}$ to the external committer and receives back RobCom commitments (in which either $\{\psi_i^S\}_{i \notin \Gamma_R}$ or $\{\tilde{\psi}_i^S\}_{i \notin \Gamma_R}$ are committed in parallel). Then in Stage 2-1 of session $s(k)$, $\mathcal{A}_{\mathsf{RobCom}}$ always use ψ_i^S's to mask a_i^S (i.e. $z_i^S := a_i^S \oplus \psi_i^S$ for all $i \in [11n]$). in the subsequent stages, $\mathcal{A}$ proceeds the experiment as in $H_{k:1}$. After the execution of $H_{k:1}$ finishes, $\mathcal{A}_{\mathsf{RobCom}}$ outputs whatever $\mathcal{Z}$ outputs in the experiment.

When $\mathcal{A}_{\mathsf{RobCom}}$ receives commitments to $\{\psi_i^S\}_{i \notin \Gamma_R}$, the internally executed experiment is identical with $H_{k:5}$, whereas when $\mathcal{A}_{\mathsf{RobCom}}$ receives commitments to $\{\tilde{\psi}_i^S\}_{i \notin \Gamma_R}$, the internally executed experiment is identical with $H_{k:6}$ (this is because when $\mathcal{A}_{\mathsf{RobCom}}$ receives commitments to $(\tilde{\psi}_i^S)_{i \notin \Gamma_R}$, the values $z_i^S = \psi_i^S \oplus a_i^S$ (thus the values $r_i^S = a_i^S \oplus b_i^S$) for each $i \notin \Gamma_R$ are uniformly random for $\mathcal{A}$. Hence the mS-OT for each $i \notin \Gamma_R$ is executed with a random input and true randomness). Hence, from the assumption that $H_{k:5}$ and $H_{k:6}$ are distinguishable, $\mathcal{A}_{\mathsf{RobCom}}$ distinguishes RobCom commitments.

We next show that in $H_{k:6}$, $\mathcal{A}$ does not cheat in sessions $s(k), \dots, s(4m)$. Assume for contradiction that in $H_{k:6}$, $\mathcal{A}$ cheats in one of those sessions, say, session $s(j)$, with non-negligible probability. Then, from an average argument, we can fix the execution of the experiment up until SM_k (inclusive) in such a way that even after being fixed, $\mathcal{A}$ cheats in session $s(j)$ only with negligible probability in $H_{k:5}$ but with non-negligible probability in $H_{k:6}$. Then, we can break the robust non-malleability of NMCom as follows.

> The adversary $\mathcal{A}_{\mathsf{NMCom}}$, who interacts with a committer of RobCom and a receiver of NMCom, internally executes $H_{k:5}$ from SM_k using the non-uniform advice. In Stage 0-1 of session $s(k)$, $\mathcal{A}_{\mathsf{NMCom}}$ chooses random strings $\tilde{\psi}^S = \tilde{\psi}_1^S \| \dots \| \tilde{\psi}_{11n}^S$ in addition to $\psi^S = \psi_1^S \| \dots \| \psi_{11n}^S$, sends $\{\psi_i^S\}_{i \notin \Gamma_R}$ and $\{\tilde{\psi}_i^S\}_{i \notin \Gamma_R}$ to the external committer and receives back parallel RobCom commitments (in which either $\{\psi_i^S\}_{i \notin \Gamma_R}$ or $\{\tilde{\psi}_i^S\}_{i \notin \Gamma_R}$ are committed to), and feeds them into $H_{k:5}$. Then in Stage 2-1 of session $s(k)$, $\mathcal{A}_{\mathsf{NMCom}}$ always use ψ_i^S's to mask a_i^S (i.e. $z_i^S := a_i^S \oplus \psi_i^S$ for all $i \in [11n]$). Also, in session $s(j)$, $\mathcal{A}_{\mathsf{NMCom}}$ forwards the NMCom commitments from $\mathcal{A}$ to the external receiver. After the execution of $H_{k:5}$ finishes, $\mathcal{A}_{\mathsf{NMCom}}$ outputs its view.
>
> The distinguisher $\mathcal{D}_{\mathsf{NMCom}}$ takes as input the view of $\mathcal{A}_{\mathsf{NMCom}}$ and the values committed by $\mathcal{A}_{\mathsf{NMCom}}$ (which are equal to the values committed to by $\mathcal{A}$ in session $s(j)$ in the internally executed experiment). $\mathcal{D}_{\mathsf{NMCom}}$ then outputs 1 if and only if $\mathcal{A}$ cheated in session $s(j)$.

When $\mathcal{A}_{\mathsf{NMCom}}$ receives commitments to $\{\psi_i^S\}_{i \notin \Gamma_R}$, the internally executed experiment is identical with $H_{k:1}$, whereas when $\mathcal{A}_{\mathsf{Com}}$ receives commitments to $\{\tilde{\psi}_i^S\}_{i \notin \Gamma_R}$, the internally executed experiment is identical with $H_{k:6}$. Hence, from the assumption that $\mathcal{A}$ cheats in session $s(j)$ with

negligible probability in $H_{k:5}$ but with non-negligible probability in $H_{k:6}$, $\mathcal{A}_{\mathsf{NMCom}}$ breaks the robust non-malleability of NMCom.

This completes the proof of Lemma 8. □

C.6 Proof of Claim 8

Proof. Recall that $H_{k:6}$ and $H'_{k:6}$ differ only in that in session $s(k)$ of $H'_{k:6}$, if R is corrupted and SM_k is fourth special message, $\beta_{b,i}$ is a random bit rather than $\beta_{b,i} = \rho_{b,i} \oplus s_{i,b\oplus\alpha_i}$ for every $b \in \{0,1\}$ and $i \in \Delta \setminus I_b$.

First, we show the indistinguishability between $H_{k:6}$ and $H'_{k:6}$. Roughly, we prove the indistinguishability using the security of mS-OT: For every $i \in \Delta \setminus I_b$, $\mathcal{A}$ executed the i-th mS-OT honestly with choice bit $(1-b)\oplus\alpha_i$, and the sender's input and randomness of this mS-OT are not revealed in Stage 8; therefore, the value of $s_{i,b\oplus\alpha_i}$ is hidden from $\mathcal{A}$ and thus $\beta_{b,i} = \rho_{b,i} \oplus s_{i,b\oplus\alpha_i}$ is indistinguishable from a random bit. Formally, we consider the following security game against cheating receiver R^* of mS-OT.

> The cheating receiver R^* gets random input-randomness pairs $(c_i, \tau_i^R)_i$ of mS-OT instances as input. R^* then participates in $9n$ instances of mS-OTs in parallel with an honest sender S, who uses a random input $(s_{i,0}, s_{i,1})$ in the i-th instance. After the execution with S, R^* receives bits $(s^*_{i,0}, s^*_{i,1})_i$ that are defined as follows: Let $b^* \in \{0,1\}$ be a randomly chosen bit; if $b^* = 0$, then for every i, $s^*_{i,0} \stackrel{\text{def}}{=} s_{i,0}$ and $s^*_{i,1} \stackrel{\text{def}}{=} s_{i,1}$; if $b^* = 1$, then for every i such that R^* behaved honestly in the i-th mS-OT using (c_i, τ_i^R) as input and randomness, $s^*_{i,c_i} \stackrel{\text{def}}{=} s_{i,c_i}$ but $s^*_{i,1-c_i}$ is a random bit, and for every other i, $s^*_{i,0} \stackrel{\text{def}}{=} s_{i,0}$ and $s^*_{i,1} \stackrel{\text{def}}{=} s_{i,1}$. Then, R^* guesses the value of b^*, and if the guess is correct, we say that R^* wins the game.

From the security of mS-OT against semi-honest receivers, any cheating R^* wins the game with probability at most $1/2 + \mathsf{negl}(n)$. Now, we assume for contradiction that $H_{k:6}$ and $H'_{k:6}$ are distinguishable, and we derive a contradiction by constructing an adversary who wins the above game with probability non-negligibly higher than $1/2$. From an average argument, we can fix the execution of the experiment up until SM_k (inclusive) in such a way that even after being fixed, $H_{k:6}$ and $H'_{k:6}$ are still distinguishable. Then, by considering the transcript up until SM_k and the table $\mathcal{T}$ as non-uniform advice, we can obtain an adversary who wins the above game with probability non-negligibly higher than $1/2$ as follows.

> The adversary R^* gets random input-randomness pairs $(c_i, \tau_i^R)_{i\in\Delta\setminus\Gamma_R}$ of mS-OT instances as its input, and internally executes $H'_{k:6}$ from SM_k using the non-uniform advice. In Stage 2-2, R^* chooses $\boldsymbol{b}^R = (b_1^R, \ldots, b_{11n}^R)$ in such a way that $\boldsymbol{r}^R = (r_1^R, \ldots, r_{11n}^R)$ satisfies $r_i^R = c_i \,\|\, \tau_i^R$ for every $i \in \Delta \setminus \Gamma_R$, namely, chooses $\boldsymbol{b}^R$ such that $b_i^R = a_i^R \oplus (c_i \,\|\, \tau_i^R)$ for every $i \in \Delta \setminus \Gamma_R$. (Recall that in $H'_{k:6}$, the subset Γ_R and the strings $\boldsymbol{a}^R = (a_1^R, \ldots, a_{11n}^R)$ are extracted by brute force and they are included in the non-uniform advice.) In Stage 3 of session $s(k)$, $\mathcal{A}_{\mathsf{NMCom}}$ obtains the i-th mS-OT from the external sender for every $i \in \Delta \setminus \Gamma_R$ and executes other instances of mS-OT by itself. Then, in Stage 6 of session $s(k)$, R^* receives bits $(s^*_{i,0}, s^*_{i,1})_{i\in\Delta\setminus\Gamma_R}$ from the external sender and uses them to compute $\beta_{b,i}$ for every $i \in \Delta \setminus \Gamma_R$, i.e., $\beta_{b,i} := \rho_{b,i} \oplus s^*_{i,b\oplus\alpha_i}$. After the execution of $H'_{k:6}$ finishes, R^* outputs whatever $\mathcal{Z}$ outputs in the experiment.

When $b^* = 0$ in the security game (and hence $s^*_{i,b\oplus\alpha_i} = s_{i,b\oplus\alpha_i}$ for every i and b), the internally executed experiment is identical with $H_{k:6}$, whereas when $b^* = 1$ (and hence $s^*_{i,b\oplus\alpha_i}$ is a random bit if $i \in \Delta \setminus I_b$ and $s^*_{i,b\oplus\alpha_i} = s_{i,b\oplus\alpha_i}$ otherwise), the internally executed experiment is identical with $H'_{k:6}$. Hence, from the assumption that $H_{k:6}$ and $H'_{k:6}$ are distinguishable, R^* wins the game with probability non-negligibly higher than 1/2.

Next, we show that in $H'_{k:6}$, $\mathcal{A}$ does not cheat in sessions $s(k), \ldots, s(4m)$. (The argument below is similar to the one in the proof of Lemma 3.) Assume for contradiction that in $H'_{k:6}$, $\mathcal{A}$ cheats in one of those sessions, say, session $s(j)$, with non-negligible probability. Then, from an average argument, we can fix the execution of the experiment up until SM_k (inclusive) in such a way that after being fixed, $\mathcal{A}$ cheats in session $s(j)$ only with negligible probability in $H_{k:6}$ but with non-negligible probability in $H'_{k:6}$. Then, by considering the transcript up until SM_k and the table $\mathcal{T}$ as non-uniform advice, we can break the robust non-malleability of NMCom as follows.

> The adversary $\mathcal{A}_{\mathsf{NMCom}}$, who participates in the above game while interacting with a receiver of NMCom, gets random input-randomness pairs $(c_i, \tau_i^R)_{i\in\Delta\setminus\Gamma_R}$ of mS-OT instances as its input, and internally executes $H'_{k:6}$ from SM_k using the non-uniform advice. In Stage 2-2, $\mathcal{A}_{\mathsf{NMCom}}$ chooses $\boldsymbol{b}^R = (b_1^R, \ldots, b_{11n}^R)$ in such a way that $\boldsymbol{r}^R = (r_1^R, \ldots, r_{11n}^R)$ satisfies $r_i^R = c_i \,\|\, \tau_i^R$ for every $i \in \Delta \setminus \Gamma_R$, namely, chooses $\boldsymbol{b}^R$ such that $b_i^R = a_i^R \oplus (c_i \,\|\, \tau_i^R)$ for every $i \in \Delta \setminus \Gamma_R$. In Stage 3 of session $s(k)$, $\mathcal{A}_{\mathsf{NMCom}}$ obtains the i-th mS-OT from the external sender for every $i \in \Delta \setminus \Gamma_R$ and executes other instances of mS-OT by itself. Then, in Stage 6 of session $s(k)$, $\mathcal{A}_{\mathsf{NMCom}}$ receives bits $(s^*_{i,0}, s^*_{i,1})_{i\in\Delta\setminus\Gamma_R}$ from the external sender and uses them to compute $\beta_{b,i}$ for every $i \in \Delta \setminus \Gamma_R$, i.e., $\beta_{b,i} := \rho_{b,i} \oplus s^*_{i,b\oplus\alpha_i}$. Also, in session $s(j)$, $\mathcal{A}_{\mathsf{NMCom}}$ forwards the NMCom commitments from $\mathcal{A}$ to the external receiver. After the execution of $H'_{k:6}$ finishes, $\mathcal{A}_{\mathsf{NMCom}}$ outputs its view.
> The distinguisher $\mathcal{D}_{\mathsf{NMCom}}$ takes as input the view of $\mathcal{A}_{\mathsf{NMCom}}$ and the values committed by $\mathcal{A}_{\mathsf{NMCom}}$ (which are equal to the values committed to by $\mathcal{A}$ in session $s(j)$ in the internally executed experiment). $\mathcal{D}_{\mathsf{NMCom}}$ then outputs 1 if and only if $\mathcal{A}$ cheated in session $s(j)$.

When $b^* = 0$ in the security game (and hence $s^*_{i,b\oplus\alpha_i} = s_{i,b\oplus\alpha_i}$ for every i and b), the internally executed experiment is identical with $H_{k:6}$, whereas when $b^* = 1$ (and hence $s^*_{i,b\oplus\alpha_i}$ is a random bit if $i \in \Delta \setminus I_b$ and $s^*_{i,b\oplus\alpha_i} = s_{i,b\oplus\alpha_i}$ otherwise), the internally executed experiment is identical with $H'_{k:6}$. Hence, from the assumption that $\mathcal{A}$ cheats in session $s(j)$ with negligible probability in $H_{k:6}$ but with non-negligible probability in $H'_{k:6}$, $\mathcal{A}_{\mathsf{NMCom}}$ breaks the robust non-malleability of NMCom.

This completes the proof. □

D Security Proof for Our MPC Protocol

Simulator $\mathcal{S}im$. As in Section B.1, we consider a simulator that works against any adversary, say $\mathcal{A}$, that participates in m sessions of Π_{2PC}. Our simulator $\mathcal{S}im$ internally invokes the adversary $\mathcal{A}$, and simulates each of the sessions by using the simulator of Π_{OT} (Section B.1) and that of $\Pi_{\mathrm{2PC}}^{\mathcal{F}_{\mathrm{OT}}}$ as follows.

1. In each execution of Π_{OT} at the beginning of Π_{2PC}, $\mathcal{S}im$ simulates the honest party's messages for $\mathcal{A}$ in the same way as $\mathcal{S}im_{\mathrm{OT}}$.
   Recall that $\mathcal{S}im_{\mathrm{OT}}$ makes a query to $\mathcal{F}_{OT}$ during the simulation. When $\mathcal{S}im_{\mathrm{OT}}$ makes a query to $\mathcal{F}_{OT}$, $\mathcal{S}im$ sends those queries to the simulator of $\Pi_{\mathrm{2PC}}^{\mathcal{F}_{\mathrm{OT}}}$ in order to simulate the answer from $\mathcal{F}_{OT}$. (Recall that the simulator of $\Pi_{\mathrm{2PC}}^{\mathcal{F}_{\mathrm{OT}}}$ simulates $\mathcal{F}_{OT}$ for the adversary.)

2. In the execution of $\Pi_{\mathrm{2PC}}^{\mathcal{F}_{\mathrm{OT}}}$ during Π_{2PC}, $\mathcal{S}im$ simulates the honest party's messages for $\mathcal{A}$ by using the simulator of $\Pi_{\mathrm{2PC}}^{\mathcal{F}_{\mathrm{OT}}}$, who obtained the queries to $\mathcal{F}_{OT}$ as above.

We remark that here we use the simulator of $\Pi_{\mathrm{2PC}}^{\mathcal{F}_{\mathrm{OT}}}$ in the setting where multiple sessions of $\Pi_{\mathrm{2PC}}^{\mathcal{F}_{\mathrm{OT}}}$ are concurrently executed. However, the use of it in this setting does not cause any problem because it runs in the black-box straight-line manner.

D.1 Proof of Indistinguishability.

We show that the view of the adversary in the real world and the view output by the simulator in the ideal world are indistinguishable. The proof proceeds very similarly to the proof for our bounded concurrent OT protocol (Section 6). To simplify the exposition, below we assume that $\Pi_{\mathrm{2PC}}^{\mathcal{F}_{\mathrm{OT}}}$ makes only a single call to $\mathcal{F}_{OT}$. (The proof can be modified straightforwardly when $\Pi_{\mathrm{2PC}}^{\mathcal{F}_{\mathrm{OT}}}$ makes multiple calls to $\mathcal{F}_{OT}$.)

Recall that Π_{2PC} is obtained by composing our OT protocol Π_{OT} with an OT-hybrid 2PC protocol $\Pi_{\mathrm{2PC}}^{\mathcal{F}_{\mathrm{OT}}}$. Roughly, we consider a sequence of hybrid experiments in which:

- Each execution of Π_{OT} is gradually changed to simulation as in the sequence of hybrid experiments that we considered in the proof of Π_{OT} (Section B.2.1).
- Once the execution of Π_{OT} in a session of Π_{2PC} is changed to simulation completely, the execution of $\Pi_{\mathrm{2PC}}^{\mathcal{F}_{\mathrm{OT}}}$ in that session is changed to simulation.

More concretely, we consider hybrids H_0, H_0^* and $H_{k:1}, \ldots, H_{k:9}$ for $k \in [4m]$, where hybrids $H_{k:8}$ and $H_{k:9}$ are defined in the following, and the others are defined as in Section B.2.1;.

Hybrid $H_{k:8}$. $H_{k:8}$ is the same as $H_{k:7}$ except that in session $s(k)$, if S is corrupted and SM_k is third special message, all the messages of $\Pi_{\mathrm{2PC}}^{\mathcal{F}_{\mathrm{OT}}}$ from R are generated by the simulator of $\Pi_{\mathrm{2PC}}^{\mathcal{F}_{\mathrm{OT}}}$. More concretely, the messages of $\Pi_{\mathrm{2PC}}^{\mathcal{F}_{\mathrm{OT}}}$ from R are generated as follows. Recall that from the definition of Hybrid $H_{k:3}$, the implicit input $v_b^* \stackrel{\mathrm{def}}{=} \mathsf{Value}(\boldsymbol{\rho}_b^{\mathsf{ext}}, \Gamma_R \cap \Delta)$ $(b \in \{0,1\})$ to Π_{OT} is extracted from the adversary in session $s(k)$ (as $\boldsymbol{\rho}_b^{\mathsf{ext}}$ are computed for both $b \in \{0,1\}$). Now, the messages of $\Pi_{\mathrm{2PC}}^{\mathcal{F}_{\mathrm{OT}}}$ from R are simulated by feeding those extracted implicit input and the subsequent messages to the simulator of $\Pi_{\mathrm{2PC}}^{\mathcal{F}_{\mathrm{OT}}}$.

Hybrid $H_{k:9}$. $H_{k:9}$ is the same as $H_{k:8}$ except that in session $s(k)$, if R is corrupted and SM_k is fourth special message, all the messages of $\Pi_{\mathrm{2PC}}^{\mathcal{F}_{\mathrm{OT}}}$ from S are generated by the simulator of $\Pi_{\mathrm{2PC}}^{\mathcal{F}_{\mathrm{OT}}}$.

Lemma 10. *Assume that in $H_{k:7}$ $(k \in [4m])$, $\mathcal{A}$ does not cheat in sessions $s(k), \ldots, s(4m)$ except with negligible probability. Then,*

- *$H_{k:7}$ and $H_{k:8}$ are indistinguishable, and*
- *in $H_{k:8}$, $\mathcal{A}$ does not cheat in sessions $s(k), \ldots, s(4m)$ except with negligible probability.*

Lemma 11. *Assume that in $H_{k:8}$ $(k \in [4m])$, $\mathcal{A}$ does not cheat in sessions $s(k), \ldots, s(4m)$ except with negligible probability. Then,*

- *$H_{k:8}$ and $H_{k:9}$ are indistinguishable, and*
- *in $H_{k:9}$, $\mathcal{A}$ does not cheat in sessions $s(k), \ldots, s(4m)$ except with negligible probability.*

Lemma 11 can be proven identically with Lemma 10, and Lemma 10 can be proven quite similarly to Claim 4 (Section B.2); the only difference is that we use the security of $\Pi_{\mathrm{2PC}}^{\mathcal{F}_{\mathrm{OT}}}$ rather than the hiding of ExtCom. We give a proof of Lemma 10 in Section D.2.

By combining Lemmas 10 and 11 with Lemma 3 to 9 in Section B.2, we conclude that the output of H_0 and that of $H_{4m:9}$ are indistinguishable, i.e., the output of the real world and that of the ideal world are indistinguishable. This concludes the proof of Theorem 6.

D.2 Proof of Lemma 10

Proof (of Lemma 10). We first show the indistinguishability between $H_{k:7}$ and $H_{k:8}$. Assume for contradiction that $H_{k:7}$ and $H_{k:8}$ are distinguishable. From an average argument, we can fix the execution of the experiment up until SM_k (inclusive) in such a way that even after being fixed, $H_{k:7}$ and $H_{k:8}$ are still distinguishable. Then, by considering the transcript up until SM_k and the table $\mathcal{T}$ as non-uniform advice, we can break the UC security of $\Pi_{\mathrm{2PC}}^{\mathcal{F}_{\mathrm{OT}}}$ as follows.

> The environment $\mathcal{Z}$ internally executes $H_{k:7}$ from SM_k using the non-uniform advice while externally participating in a single session of $\Pi_{\mathrm{2PC}}^{\mathcal{F}_{\mathrm{OT}}}$ via the dummy adversary that corrupts S. In session $s(k)$, $\mathcal{Z}$ forwards all the messages of $\Pi_{\mathrm{2PC}}^{\mathcal{F}_{\mathrm{OT}}}$ from the internal $\mathcal{A}$ to the external dummy adversary (including the query to $\mathcal{F}_{OT}$),[19] and those from the external dummy adversary to the internal $\mathcal{A}$. After the execution of $H_{k:7}$ finishes, $\mathcal{Z}$ outputs the output of the internally emulated experiment.

When $\mathcal{Z}$ interacts with the dummy adversary, the internally executed experiment is identical with $H_{k:7}$, whereas when $\mathcal{Z}$ interacts with the simulator of $\Pi_{\mathrm{2PC}}^{\mathcal{F}_{\mathrm{OT}}}$, the internally executed experiment is identical with $H_{k:8}$. Hence, from the assumption that $H_{k:7}$ and $H_{k:8}$ are distinguishable, $\mathcal{Z}$ breaks the security of $\Pi_{\mathrm{2PC}}^{\mathcal{F}_{\mathrm{OT}}}$

We next show that in $H_{k:8}$, $\mathcal{A}$ does not cheat in sessions $s(k), \ldots, s(4m)$. Assume for contradiction that in $H_{k:8}$, $\mathcal{A}$ cheats in one of those sessions, say, session $s(j)$, with non-negligible probability. Then, from an average argument, we can fix the execution of the experiment up until SM_k (inclusive) in such a way that even after being fixed, $\mathcal{A}$ cheats in session $s(j)$ only with negligible probability in $H_{k:7}$ but with non-negligible probability in $H_{k:8}$. Then, by considering the transcript up until SM_k and the table $\mathcal{T}$ as non-uniform advice, we can break the robust non-malleability of NMCom as follows.

> The adversary $\mathcal{A}_{\mathsf{NMCom}}$, who participates in an execution of $\Pi_{\mathrm{2PC}}^{\mathcal{F}_{\mathrm{OT}}}$ as the environment (where the dummy adversary corrupts S) while interacting with a receiver of NMCom, internally executes $H_{k:7}$ from SM_k using the non-uniform advice. In session $s(k)$, $\mathcal{A}_{\mathsf{NMCom}}$ forwards all the messages of $\Pi_{\mathrm{2PC}}^{\mathcal{F}_{\mathrm{OT}}}$ from the internal $\mathcal{A}$ to the external dummy adversary (including the query to $\mathcal{F}_{\mathrm{OT}}$), and those from the external dummy adversary to the internal $\mathcal{A}$. Also, in session $s(j)$, $\mathcal{A}_{\mathsf{NMCom}}$ forwards the NMCom commitments from $\mathcal{A}$ to the external receiver. After the execution of $H_{k:7}$ finishes, $\mathcal{A}_{\mathsf{NMCom}}$ outputs the output of the internally emulated experiment.
>
> The distinguisher $\mathcal{D}_{\mathsf{NMCom}}$ takes as input the view of $\mathcal{A}_{\mathsf{NMCom}}$ and the values committed by $\mathcal{A}_{\mathsf{NMCom}}$ (which are equal to the values committed to by $\mathcal{A}$ in session $s(j)$ in the internally executed experiment). $\mathcal{D}_{\mathsf{NMCom}}$ then outputs 1 if and only if $\mathcal{A}$ cheated in session $s(j)$.

[19] Note that these messages appear after SM_k

When $\mathcal{A}_{\mathsf{NMCom}}$ interacts with the dummy adversary in the execution of $\mathit{\Pi}_{\mathrm{2PC}}^{\mathcal{F}_{\mathrm{OT}}}$, the internally executed experiment is identical with $H_{k:7}$, whereas when $\mathcal{A}_{\mathsf{NMCom}}$ interacts with the simulator there, the internally executed experiment is identical with $H_{k:8}$. Hence, from the assumption that $\mathcal{A}$ cheats in session $s(j)$ with negligible probability in $H_{k:7}$ but with non-negligible probability in $H_{k:8}$, $\mathcal{A}_{\mathsf{NMCom}}$ breaks the robust non-malleability of NMCom.

This completes the proof of Lemma 10. □