Black-Box, Round-Efficient Secure Computation via Non-Malleability Amplification

Hoeteck Wee[⋆]

Queens College, CUNY
hoeteck@cs.qc.cuny.edu

Abstract. We present round-efficient protocols for secure multi-party computation with a dishonest majority that rely on *black-box* access to the underlying primitives. Our main contributions are as follows:

– a $O(\log^* n)$-round protocol that relies on black-box access to dense cryptosystems, homomorphic encryption schemes, or lossy encryption schemes. This improves upon the recent $O(1)^{\log^* n}$-round protocol of Lin, Pass and Venkitasubramaniam (STOC 2009) that relies on non-black-box access to a smaller class of primitives.

– a $O(1)$-round protocol requiring in addition, black-box access to a one-way function with sub-exponential hardness, improving upon the recent work of Pass and Wee (Eurocrypt 2010).

These are the first black-box constructions for secure computation with sublinear round complexity. Our constructions build on and improve upon the work of Lin and Pass (STOC 2009) on non-malleability amplification, as well as that of Ishai et al. (STOC 2006) on black-box secure computation.

In addition to the results on secure computation, we also obtain a simple construction of a $O(\log^* n)$-round non-malleable commitment scheme based on one-way functions, improving upon the recent $O(1)^{\log^* n}$-round protocol of Lin and Pass (STOC 2009). Our construction uses a novel transformation for handling arbitrary man-in-the-middle scheduling strategies which improves upon a previous construction of Barak (FOCS 2002).

Key words: secure multi-party computation, round complexity, black-box constructions, non-malleable commitments.

[⋆] Supported in part by NSF CAREER Award CNS-0953626 and PSC-CUNY Award # 63888-00 41.

1 Introduction

Secure multi-party computation (MPC) allows several mutually distrustful parties to perform a joint computation without compromising, to the greatest extent possible, the privacy of their inputs or the correctness of the outputs. The early work of Goldreich, Micali and Wigderson [19] showed that we may realize secure multi-party computation with a dishonest majority under general cryptographic assumptions. Over the last decade, substantial progress was made towards improving the round complexity and computational efficiency of these protocols in two separate lines of works, culminating in (1) *constant-round* protocols for secure computation [24, 34, 38, 29] as well as (2) *black-box* constructions that avoid the use of (typically expensive) general NP reductions [25, 12, 22, 20, 23]. However, simultaneously achieving both of these efficiency guarantees has so far remained quite elusive; the state-of-the-art for black-box constructions is a $O(n)$-round protocol where n is the number of parties.[1] This raises the following natural question:

> *Does there exist a black-box, $o(n)$-round protocol for secure multi-party computation, or is there an inherent trade-off between round complexity and computational efficiency?*

Before stating our results, we provide some additional context and motivation.

Round-efficient secure computation. In the GMW protocol for secure computation, each player takes turns to sequentially commit to its input (along with a "proof of knowledge"); any non-trivial improvement in round complexity will require interweaving these input commitments, which could potentially allow an adversary to violate input independence via a man-in-the-middle attack. For this reason, improvements in round complexity for secure computation has often paralleled results on *non-malleability* [13]. Constant-round MPC protocols were first obtained by Katz, Ostrosky and Smith [24] (relying on [2]) and by Pass [34] based on the existence of enhanced trapdoor permutations and in addition, collision-resistant hash functions. More recently, Lin, Pass and Venkitasubramaniam [29, 26] showed that the latter assumption can be eliminated while still maintaining almost constant – specifically, $O(1)^{\log^* n}$ – round complexity. In follow-up work, Pass and Wee [38] gave a constant-round protocol, assuming in addition one-way functions with sub-exponential hardness. An advantage of these latter two works is that they avoid the use of non-black-box simulation techniques [1] along with the sophisticated machinery (e.g. the PCP theorem) associated with them.

Black-box secure computation. The general question of whether we can securely realize cryptographic tasks via black-box access to a general primitive is of great theoretical and practical interest. In particular, black-box constructions (namely, those that refer only to the input/output behavior of the underlying primitive) are typically more efficient in terms of both computational and communication complexity, and also more suited for implementation as compared to non-black-box constructions. As such, non-black box constructions traditionally only serve as "feasibility" results, and indeed, a series of recent works on secure computation [12, 22, 31, 23] views black-box constructions as an important step towards making MPC more "practical". Upon closer examination, one notices that while the afore-mentioned black-box constructions of secure protocols do improve on the efficiency of previous non-black-box constructions as measured in terms of computational and communication complexity, except in the presence of an honest majority or an

[1] Throughout the introduction, we use n to denote the number of parties; in particular, the round complexity of all the protocols we discuss here depends only on the number of parties, and is independent of the security parameter.

ideal functionality. (Specifically, the round complexity grows linearly with the number of parties). We also point out here that the pursuit of black-box constructions has often yielded new techniques (e.g. the use of randomized encodings in [25, 9]), along with additional conceptual and technical insights into the original non-black-box constructions.

1.1 Our Results

In this work, we present the first black-box MPC protocols with a sub-linear number of rounds.

> **Theorem (informal).** There exists a $O(\log^* n)$-round protocol for securely computing any n-party functionality against a malicious adversary corrupting any number of parties that relies on black-box access to certifiable enhanced trapdoor permutations.

We point out here that this construction (and the next) may be extended to a larger class of assumptions, such as dense cryptosystems and lossy encryption schemes. This improves upon the recent $O(1)^{\log^* n}$-round protocol Lin, Pass and Venkitasubramaniam [29, 26], which relies on non-black-box access to a smaller class of assumptions. Next, we show that we can also obtain constant-round protocols by relying on an additional assumption, namely one-way functions secure against sub-exponential size circuits,[2]improving upon the non-black-box construction of Pass and Wee [38].

> **Theorem (informal).** There exists a $O(1)$-round protocol for securely computing any n-party functionality against a malicious adversary corrupting any number of parties that relies on black-box access to certifiable enhanced trapdoor permutations and a one-way function secure against sub-exponential size circuits.

1.2 Our Constructions and Techniques

Both of our constructions follow the same high-level framework, which we will describe in the context of our first result.

Basic Commitment Scheme. Our starting point is a $O(1)$-round non-malleable commitment scheme for a constant number of parties. We may rely on the trivial construction wherein each party *sequentially* commits to its input using an extractable commitment, as we assume a fully synchronized network for stand-alone MPC. To obtain our second result, we begin with a black-box variant of constant-round non-malleable commitments in [38] for $\log \log \log n + O(1)$ parties, based on one-way functions with sub-exponential hardness.

Non-Malleability Amplification. Next, we provide a black-box transformation of a commitment scheme that is (many-many) non-malleable for t parties into one for 2^{t-1} parties, while incurring only a constant *additive* blow-up in the number of rounds; applying this transformation $O(\log^* n)$ times to our basic scheme yields a $O(\log^* n)$-round non-malleable commitment scheme for n parties. Our transformation simplifies and improves upon the earlier construction of Lin and Pass [26] which is non-black-box and incurs a constant multiplicative blow-up in the number of rounds.[3]

[2] This is a relatively mild assumption since the best attacks on the standard candidates for one-way functions have sub-exponential complexity.

[3] Subsequent to our work, Lin and Pass observed that they may also achieve a constant additive overhead with a minor modification of their transformation (c.f. full version of [26]).

OT-Compiler. We show how to use our n-party non-malleable commitment scheme to realize an n-party oblivious transfer (OT) functionality, starting from a two-party OT protocol Π that is secure against a malicious sender and semi-honest receiver. As it turns out, many existing semi-honest OT protocols such as those where the sender encrypts both its inputs (c.f. [14, 16, 39, 10]) already has this property. In order to "boost" the security of Π to tolerate malicious receivers, we rely on a recent construction of Ishai et al. [22, 20, 11], which may in turn be viewed as a cut-and-choose variant of the "GMW compiler". However, this cut-and-choose compiler relies on a commitment scheme that is extractable and equivocable, and moreover, must remain non-malleable while simulating an equivocable commitment in the left interaction. Our main technical contribution for this step lies in eliminating the equivocability requirement.

MPC from OT. In the last step, we combine our round-efficient n-party OT protocol with the constant-round MPC protocol of Ishai, Prabhakaran and Sahai in the OT-hybrid model [23, Theorem 3]. Here, we rely on the composition theorem for the stand-alone model in [7].

Next, we provide an overview of the two novel building blocks in our construction, namely the non-malleability amplification protocol and the OT compiler.

Improved and simpler non-malleability amplification. Given a many-many non-malleable commitment scheme tagCom for identities of length $\log t + 1$, we construct a many-many non-malleable commitment scheme for identities of length t with a constant *additive* blow-up in the number of rounds. Our construction, roughly speaking, proceeds as follows: to commit to a string v with identity $\textsc{id} = (\textsc{id}_1, \ldots, \textsc{id}_t) \in \{0,1\}^t$:

- Commit to v using tagCom with identities $(1, \textsc{id}_1), \ldots, (t, \textsc{id}_t)$ a total of t times in *parallel.*
- Prove using a zero-knowledge argument of knowledge that all t committed values are equal.

We argue, informally, that the new scheme is many-many non-malleable. Consider for simplicity the stand-alone setting, where the adversary receives a single commitment to v on the left with identity $\textsc{id}$ and tries to commit to a related value $\tilde{v}$ with identity $\tilde{\textsc{id}} \neq \textsc{id}$. There must exist some i for which $\tilde{\textsc{id}}_i \neq \textsc{id}_i$ and thus $(i, \tilde{\textsc{id}}_i)$ is different from all of $(1, \textsc{id}_1), \ldots, (t, \textsc{id}_t)$. By many-many non-malleability of tagCom, the committed value for $(i, \tilde{\textsc{id}}_i)$ is independent of all the left commitments. Furthermore, by soundness of the argument of knowledge, this value determines $\tilde{v}$, and thus $\tilde{v}$ must be independent of v. This argument extends naturally to the setting where there are multiple commitments on the right, which in turn implies non-malleability with multiple commitments on both the left and on the right [35, 28].

We point out here that the overall approach of using multiple commitments to the same value and then providing a zero-knowledge proof of consistency is reminiscent of the constructions of CCA2-secure and non-malleable encryption schemes [13, 36, 9]. Our analysis considers explicitly an "alternative opening phase", which is inspired by the notion of an "alternative decryption oracle" in the literature on encryption.

To obtain a fully black-box construction that uses black-box access to a statistically binding commitment scheme Com (and thus any one-way function [33, 21]), we combine the previous construction with the message encoding technique from [9, 37]. Here, we rely crucially on the fact that in our construction, the zero-knowledge argument is used to enforce equality amongst committed values. Indeed, we do not know how to directly obtain a black-box variant of the Lin-Pass non-malleability amplification protocol [26] because the zero-knowledge arguments therein are used to enforce that committed values satisfy a more complex relation.

Protocol	Rounds	Assumptions		Black-Box	
		Beyond TDPs	Additional	Primitive	Simulation
GMW [19, 5, 30]	$O(n)$	yes	none	no	yes
IKLP [22, 23, 37]	$O(n)$	yes	none	yes	yes
KOS [24]	$O(\log n)$	no	none	no	yes
LP/LPV [26, 29]	$O(1)^{\log^* n}$	no	none	no	yes
KOS/Pass [24, 34]	$O(1)$	no	CRHFs	no	no
PW [38, 29]	$O(1)$	no	sub-exp OWFs	no	yes
this work	$O(\log^* n)$	yes	none	yes	yes
this work	$O(1)$	yes	sub-exp OWFs	yes	yes

Fig. 1. Summary of MPC Protocols.

OT compiler. We use the OT compiler in [22, 20, 11] to "boost" the security of a two-party OT protocol Π from tolerating semi-honest receivers to tolerating malicious receivers. The idea is to run multiple copies of Π and rely on cut-and-choose to guarantee that in most of these instantiations, the malicious OT receiver is behaving consistently with Π (see [11, Section 2] for an overview). The random n-bit challenge for the cut-and-choose phase is determined via a coin-tossing protocol as follows: (1) the sender first commits to a random n-bit string q_S (using our extractable non-malleable commitment); (2) the receiver then responds with a random n-bit string q_R; (3) the sender opens its commitment and the challenge is given by $q_S \oplus q_R$.

The OT compiler guarantees that there is at most one random challenge q^* that allows the malicious receiver to cheat in the cut-and-choose phase. If the sender's commitment is equivocable, then the probability of cheating is negligible since the probability that a random equivocation equals $q_R \oplus q^*$ is 2^{-n}. To eliminate the equivocability requirement while bounding the probability of cheating, we rely on the simulator from [11] which has the property that q^* is efficiently computable (this in turn relies on extractability of the receiver's commitment in an earlier stage of the protocol). Now suppose the simulator cheats with non-negligible probability; then, with roughly the same probability, the sender's committed value must equal $q_R \oplus q^*$, which contradicts the hiding guarantee of the commitment scheme (amidst extraction).

MPC from non-malleable commitments. We note that our approach for deriving round-efficient MPC protocols from non-malleable commitments is quite different from that used in previous protocols with sub-logarithmic round complexity [29, 24, 34]. One limitation of these approaches is that the ensuing constructions rely on enhanced trapdoor permutations, or similar primitives with an "oblivious sampling" requirement; in particular, we do not know how to extend these constructions to work with lossy trapdoor functions and lossy encryption schemes [40, 6]. This is in fact an inherent limitation in the techniques underlying previous constructions. Roughly speaking, these previous protocols all entail the use of a coin-tossing protocol to "obliviously sample" a random challenge,[4] whereas in the simulation, this challenge is generated in a non-oblivious manner along with some trapdoor.

[4] In [29, 34, 4], a random challenge is a random element in the range of a trapdoor permutation, and the trapdoor is its preimage, whereas in [24], a random challenge is the CRS in the CLOS protocol [8], and the trapdoor is that underlying a simulated CRS.

1.3 Additional Results

Starting from our improved non-malleability amplification, we obtain several new results on non-malleable commitments.

$O(\log^* n)$-round non-malleable commitments. The first is a simple, self-contained construction of non-malleable commitments from one-way functions with better round complexity:

> **Theorem (informal).** Suppose there exists one-way functions. Then, there exists a $O(\log^* n)$-round non-malleable commitment scheme with a black-box proof of security.

This improves upon the previous $O(1)^{\log^* n}$-round protocol of Lin and Pass [26]. The technical improvement comes from the fact that we accomplish non-malleability amplification with an additive instead of a multiplicative blow-up in the number of rounds. Our construction proceeds in three steps:

- We start with a $O(1)$-round many-many non-malleable commitment scheme for a constant number of parties and a synchronizing adversary. As noted earlier, the trivial construction wherein each party sequentially commits to its input using an extractable commitment suffices.
- Next, we apply non-malleability amplification a total of $O(\log^* n)$ times to obtain a $O(\log^* n)$-round many-many non-malleable commitment scheme for n-bit identities and a synchronizing adversary.
- Finally, we provide a simple and general transformation of non-malleable commitment schemes that are secure against synchronizing adversaries into one that are secure against arbitrary scheduling strategies, with an additive increase in round complexity. This construction improves upon a previous transformation of Barak [2, Theorem 6.1], which in turn requires constant-round perfectly hiding commitments. As with [2], our transformation proceeds by creating multiple rewinding opportunities; the difference is that we add rewinding slots to the sender (as with the transformation of stand-alone non-malleable commitments into many-many non-malleable commitments in [26]) as opposed to the receiver.

In addition, the construction is self-contained in that we never need to rely on the complex rewinding reschedules from [13, 28].

Black-box non-malleable commitments. The preceding construction can also be made black-box, which partially addresses in the affirmative an open problem posed by Pass (namely, whether the $O(1)^{\log^* n}$ protocol in [26] can be made black-box). However we only achieve a weaker notion of non-malleability w.r.t extraction, which nonetheless suffices for secure MPC and also implies the notion of non-malleability in [2, 13] (see [3, Definition 6.4.1]). Roughly speaking, the preceding construction and that in [26] guarantee the committed values in the right interactions must be computationally independent of those in the left interactions, whereas in this black-box construction, we only guarantee that the values output by the extractor (which is the same as the committed value whenever the commitment has a valid opening and may be arbitrary when the commitment opens to $\perp$) for the right interactions are independent of the committed values in the left interaction.

> **Theorem (informal).** There exists a (fully) black-box construction of a $O(\log^* n)$-round commitment scheme that is many-many non-malleable w.r.t extraction, starting from any one-way function.

We obtain this result by applying our black-box non-malleability amplification (for non-synchronizing adversaries) a total of $O(\log^* n)$ times to the constant-round many-many non-malleable commitment scheme for 4 parties in [37].

A note on "robustness". Our work clarifies the connection between the notion of robustness introduced in [26] –a commitment scheme is "robust" if remains non-malleable with respect to arbitrary constant-round protocols in the left interaction– and message synchronization. Unlike non-malleability amplification in [26] as well as the MPC protocol in [29], our basic non-malleability amplification and MPC protocol do not require that the underlying non-malleable commitment scheme be robust; this is because we only handle synchronizing adversaries in those constructions. On the other hand, we will require robustness in order to handle non-malleability amplification for non-synchronizing adversaries and for our general transformation for handling non-synchronizing adversaries.[5] This latter transformation also reinforces an observation used in [27], that robust commitments can be used in place of statistically-hiding commitments in many constructions of non-malleable protocols and possibly reducing the assumptions from collision-resistant hash functions to one-way functions.

1.4 Organization

In Section 3, we present our non-malleability amplification theorem for synchronizing adversaries and in Section A, we present a black-box variant of the construction. In Section B, we present a black-box variant of the constant-round non-malleable commitments from [38]. In Section C, we present our results for secure MPC. In Section D, we show how to transform a non-malleable commitment w.r.t. synchronizing adversaries into one for non-synchronizing adversaries. In Section E, we show how to achieve non-malleability amplification for non-synchronizing adversaries.

2 Preliminaries and Definitions

We use Com to denote a non-interactive statistically binding commitment scheme. Our constructions may be easily extended to handle the 2-message statistically binding commitment scheme based on one-way functions from [33, 21], where the first message can be fixed "once and for all". We also use WIPOK to denote 3-round witness-distinguishable proofs of knowledge for NP with special soundness (assuming Com) [17].

Non-malleable commitments. We recall the definition of many-many non-malleability from [28], which builds upon those in [13, 35]. Let $\mathsf{TagCom} = (\mathcal{C}, \mathcal{R})$ be a commitment scheme with identities, and 1^n be the security parameter. In the man-in-the-middle execution, the adversary $\mathcal{A}$ is participating m left interactions and m right interactions. In the left interactions, $\mathcal{A}$ interacts with $\mathcal{C}$ receiving a commitment to m values $v_1, \ldots, v_m$, using identities $\text{ID}^1, \ldots, \text{ID}^m$ of its choice. In the right interactions, $\mathcal{A}$ interacts with $\mathcal{R}$ attempting to commit to a sequence of m related values $\tilde{v}_1, \ldots, \tilde{v}_m$, again using identities $\tilde{\text{ID}}^1, \ldots, \tilde{\text{ID}}^m$ of its choice. $\mathcal{A}$ also receives an auxiliary z. In general, we allow $\mathcal{A}$ complete control over the scheduling of the messages, although we will also refer to an *synchronizing adversary* that always sends the i'th messages in each of the right sessions immediately after it receives the i'th messages in all of the left sessions and

[5] Indeed, our $O(\log^* n)$-round protocol for synchronizing adversaries is robust because it has a super-constant number of rewinding slots.

vice versa (i.e. it sends the jth messages in each of the left sessions immediately after it receives the j'th messages in all of the right sessions.) If any of the right commitments as determined by the transcript[6] are invalid or undefined, its value is set to $\perp$. For any i such that $\tilde{\text{ID}}^i \in \{\text{ID}^1, \ldots, \text{ID}^m\}$, the value $\tilde{v}_i$ is also set to $\perp$ (that is, any commitment where adversary uses the same identity as that in one of the left interactions is considered invalid). We write $\mathsf{mim}^{\mathsf{TagCom}}_{\mathcal{A}(z)}(\mathcal{C}(v_1, \ldots, v_m), \mathcal{R})$ to denote a random variable comprising the view of $\mathcal{A}$ along with the m-tuple of values $(\tilde{v}_1, \ldots, \tilde{v}_m)$. We abbreviate this as $\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}(v_1, \ldots, v_m), \mathcal{R})$ when the commitment scheme TagCom is clear from the context.

Definition 1. *A commitment scheme $(\mathcal{C}, \mathcal{R})$ is* many-many non-malleable (w.r.t. opening) *if for every* PPT *$\mathcal{A}$ and every polynomial $m = m(n)$ and every pair of m values $(v^0_1, \ldots, v^0_m), (v^1_1, \ldots, v^1_m)$ along with any $z \in \{0,1\}^*$, the distributions*

$$\left\{\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}(v^0_1, \ldots, v^0_m), \mathcal{R})\right\} \text{ and } \left\{\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}(v^1_1, \ldots, v^1_m), \mathcal{R})\right\}$$

are computationally indistinguishable.

We will also consider a restricted notion of many-many non-malleability where in the left and right interactions, the adversary $\mathcal{A}$ may only use identities of length at most d. In addition, we will refer to relaxed notions of many-many non-malleability: one-many and one-one non-malleability. In the former, the adversary participates in one interaction on the left and m interactions on the right, and in the latter, the adversary participates in one interaction on the left and one interaction on the right. As shown in [28], any commitment scheme that is one-many non-malleable is also many-many non-malleable.

Proposition 1 ([28]). *Let $(\mathcal{C}, \mathcal{R})$ be a one-many non-malleable commitment (resp. w.r.t. synchronizing adversaries). Then, $(\mathcal{C}, \mathcal{R})$ is also a many-many non-malleable commitment (resp. w.r.t. synchronizing adversaries).*

3 Improved Non-Malleability Amplification

We present our construction for non-malleability amplification in Fig 2.

Proposition 2 (Non-malleability amplification with synchronization). *For every $t = t(n) \geq 4$, if* tagCom *is one-many non-malleable for identities of length $\log t + 1$ w.r.t. synchronizing adversaries, then $(\mathcal{C}, \mathcal{R})$ as shown in Fig 2 is one-many non-malleable for identities of length t w.r.t synchronizing adversaries.*

3.1 Proof overview

Let ID denote the identity on the left, and fix some identity $\tilde{\text{ID}} \neq \text{ID}$ on the right. Following the informal argument given in Section 1.2, the key in the analysis is to examine the committed value in the tagCom commitment with identity $(i, \tilde{\text{ID}}_i)$ for which $\tilde{\text{ID}}_i \neq \text{ID}_i$. Towards formalizing this argument, it is helpful to consider an "alternative open phase" for the man-in-the-middle execution corresponding to a "receiver" $\mathcal{R}^*_0$ and $\mathcal{R}^*$.

[6] We stress here that the value is determined by enumerating over all possible openings in the "open phase" of the commitment scheme that are consistent with the transcript, and not by enumerating over all possible values and random tapes for the honest sender algorithm.

Common input : security parameter 1^n and an identity $\textsc{id} = (\textsc{id}_1, \ldots, \textsc{id}_t) \in \{0,1\}^t$.

Sender's input : a value $v \in \{0,1\}^{\mathrm{poly}(n)}$.

. . . .

COMMIT PHASE.

Stage 0: $\mathcal{R}$ sends a random $s = f(r)$.[a] $\mathcal{C}$ responds with a dummy message.[b]

Stage 1: $\mathcal{C}$ commits to v using tagCom with tags $(1, \textsc{id}_1), \ldots, (t, \textsc{id}_t)$. That is, $\mathcal{C}$ executes $\mathsf{tagCom}(\mathsf{id}_i, v)$ in parallel for $i = 1, 2, \ldots, t$, where $\mathsf{id}_i = (i, \textsc{id}_i)$.

Stage 2: $\mathcal{C}$ proves a WIPOK of the statement:

either all t commitments in Stage 1 are commitments to the same value or $s \in f(\{0,1\}^n)$

using as witness the value v along with the randomnesses it uses for the commitments in Stage 1.

. . . .

OPEN PHASE.

- $\mathcal{C}$ opens the first commitment to v in Stage 1 (the one using $(1, \textsc{id}_1)$).[c]

[a] Following [26], $\mathcal{R}$ should send a witness hiding proof that $s \in f(\{0,1\}^n)$ after Stage 1.

[b] The dummy message is essential for technical reasons, to ensure that $\mathcal{R}$'s first message in tagCom is always sent after it sends the random challenge $f(r)$.

[c] Note that if the WIPOK in Stage 2 is not accepting, then the committed value corresponds to $\perp$.

Fig. 2. Commitment scheme $\mathsf{TagCom} = (\mathcal{C}, \mathcal{R})$.

More formally, we first write $\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}(v), \mathcal{R}_0^*)$ to denote a random variable that is the same as $\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}(v), \mathcal{R})$, except the m-tuple of values $(\tilde{v}_1, \ldots, \tilde{v}_m)$ is defined as follows: for $\tilde{\textsc{id}} \in \{\tilde{\textsc{id}}^1, \ldots, \tilde{\textsc{id}}^m\}$, we set the corresponding committed value $\tilde{v}$ as follows:

- if $\tilde{\textsc{id}} = \textsc{id}$, then we set $\tilde{v}$ to $\perp$,
- if $\tilde{\textsc{id}} \neq \textsc{id}$, let $i \in [t]$ be the first index such that $\tilde{\textsc{id}}_i \neq \textsc{id}_i$ and set $\tilde{v}$ to be the committed value in Stage 1 corresponding to the tag $(i, \tilde{\textsc{id}}_i)$.

Next, we write $\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}(v), \mathcal{R}^*)$ to denote a random variable that is the same as $\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}(v), \mathcal{R}_0^*)$, except each committed value $\tilde{v}$ is set to $\perp$ whenever the corresponding WIPOK in Stage 2 is rejecting. We will argue that the committed values are essentially the same whether we refer to $\mathcal{R}$ or $\mathcal{R}^*$. Looking ahead, we highlight two properties of the intermediate $\mathcal{R}_0^*$ that will come in handy later:

- Property A: We can efficiently compute the output of $\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}(v), \mathcal{R}^*)$ given that of $\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}(v), \mathcal{R}_0^*)$; this is because we can check whether the WIPOK in Stage 2 is accepting given the transcript of the commit phase.
- Property B: The committed value according to $\mathcal{R}_0^*$ is completely determined upon the completion of Stage 1 on the right. (In contrast, the committed values according to $\mathcal{R}$ and $\mathcal{R}^*$ depend also on the outcome of Stage 2.)

3.2 The hybrid argument

STEP 1: SWITCHING TO $\mathcal{R}^*$. We claim that

$$\left\{\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}(v^0), \mathcal{R})\right\} \cong \left\{\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}(v^0), \mathcal{R}^*)\right\} \tag{3.1}$$

STEP 2: SWITCHING TO $\mathcal{C}^*(v^0)$. We change the WIPOK on the left to use the trapdoor witness r, i.e. we replace $\mathcal{C}(v^0)$ in the left execution with $\mathcal{C}^*(v^0)$ where $\mathcal{C}^*$ is the following (computationally unbounded) sender that on input v, behaves exactly like $\mathcal{C}(v)$ in Stages 0 and 1, and proceeds as follows in Stage 2:

- (Stage 2) Extract $r \in f^{-1}(s)$ via brute force (where s is the challenge sent by the $\mathcal{R}$ in Stage 0) and complete the WIPOK using witness r.

Here, we exploit security of the WIPOK and the fact that the adversary is synchronizing to argue that

$$\left\{\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}(v^0), \mathcal{R}^*)\right\} \cong \left\{\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}^*(v^0), \mathcal{R}^*)\right\} \tag{3.2}$$

STEP 3: SWITCHING TO $\mathcal{C}^*(v^1)$. We switch the left commitment in Stage 1 to v^1 (i.e. we replace $\mathcal{C}^*(v^0)$ on the left with $\mathcal{C}^*(v_1)$) and exploit many-many non-malleability of tagCom to argue that

$$\left\{\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}^*(v^0), \mathcal{R}^*)\right\} \cong \left\{\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}^*(v^1), \mathcal{R}^*)\right\} \tag{3.3}$$

STEP 4: SWITCHING TO $\mathcal{C}(v_1)$. This is analogous to Step 2.

$$\left\{\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}^*(v^1), \mathcal{R}^*)\right\} \cong \left\{\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}(v^1), \mathcal{R}^*)\right\}$$

STEP 5: SWITCHING BACK TO $\mathcal{R}$. This is analogous to Step 1.

$$\left\{\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}(v^1), \mathcal{R}^*)\right\} \cong \left\{\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}(v^1), \mathcal{R})\right\}$$

Looking ahead, for steps $2, 3$ and 4, it suffices to establish indistinguishability of the distributions where we replace every instance of $\mathcal{R}^*$ with $\mathcal{R}^*_0$. This is because we can efficiently compute the committed values according to $\mathcal{R}^*$ from that of $\mathcal{R}^*_0$ together with the transcript (c.f. Property A).

Switching to $\mathcal{R}^*$ (step 1). Here, we just need to argue that for each of the right sessions, if the Stage 2 WIPOK is accepting, then the t committed values in Stage 1 are equal (and whenever this holds, the committed values are the same whether we consider $\mathcal{R}$ or $\mathcal{R}^*$ and (3.1) follows). Suppose otherwise, that is, there exists a MIM adversary $\mathcal{A}$ that with non-negligible probability, produces an accepting right execution in which the t committed values in Stage 1 are not all equal. Now, we may incorporate the left execution into $\mathcal{A}$ (by honestly committing to v^0) to obtain a stand-alone cheating prover $\mathcal{P}^*$ for the WIPOK in that particular right execution. Then, rewinding and extracting from $\mathcal{P}^*$ must yield a witness for $s \in f(\{0,1\}^n)$, which contradicts one-wayness of f. We note that this is the only step of the hybrid argument (apart from the analogous Step 5) that requires rewinding or extraction.

Switching to $\mathcal{C}^*$ (step 2). As noted above, to prove (3.2), it suffices to establish the following claim:

Lemma 1 (exploiting WIPOK**).** $\left\{\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}(v^0), \mathcal{R}^*_0)\right\} \cong \left\{\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}^*(v^0), \mathcal{R}^*_0)\right\}$.

We begin with the observation that the only difference between these two distributions is the witness used in the WIPOK used in Stage 2 on the left.

Proof. Let $\Phi(\mathcal{A}, z)$ denote the distribution of all joint views τ of $\mathcal{A}$ and the receivers on the right up to the point before Stage 2 on the left begins (i.e., just after the completion of Stage 1 on the right). In addition, we add to $\Phi(\mathcal{A}, z)$ the following values: (1) v_0 and the randomness σ used for all of the Stage 1 commitments on the left in τ; (2) $r \in f^{-1}(s)$ where s is the Stage 0 challenge on the left in τ; and (3) the m committed values $(\tilde{v}_1, \ldots, \tilde{v}_m)$ in m executions on the right as determined by $\mathcal{R}_0^*$ (here, we use the fact that to determine the committed values according to $\mathcal{R}_0^*$, we only need to look at the transcript up to the completion of Stage 1, c.f. Property B). We do not require that these latter values $((v_0, \sigma), s, (\tilde{v}_1, \ldots, \tilde{v}_m))$ be efficiently computable.

Now, consider a WIPOK prover $\mathcal{P}$ for the statement

> either all t commitments in Stage 1 are commitments to the same value or $s \in f(\{0,1\}^n)$
> (the commitments and s refer to those for the left interaction embedded in the view τ).

against a cheating verifier $\mathcal{V}^*$ that receives as auxiliary input $\Phi(\mathcal{A}, z)$. It is straight-forward to construct $\mathcal{V}^*$ such that

- if $\mathcal{P}$ uses the witness (v_0, σ), then the output of $\mathcal{V}^*$ has the same distribution as $\left\{\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}(v^0), \mathcal{R}_0^*)\right\}$; and
- if $\mathcal{P}$ uses the witness $r \in f^{-1}(s)$, then the output of $\mathcal{V}^*$ has the same distribution as $\left\{\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}^*(v^0), \mathcal{R}_0^*)\right\}$.

Roughly speaking, $\mathcal{V}^*$ upon receiving the auxiliary input from $\Phi(\mathcal{A}, z)$ (i.e. the view τ together with the values $(v_0, \sigma), s, (\tilde{v}_1, \ldots, \tilde{v}_m)$), proceeds by simulating $\mathcal{A}(z)$ internally, using the messages from $\mathcal{P}$ for the messages from $\mathcal{C}$ or $\mathcal{C}^*$ in the left interaction and internally simulating the receiver in Stage 2 for the m right interactions; the committed values $(\tilde{v}_1, \ldots, \tilde{v}_m)$ for the m right interactions are provided as part of $\mathcal{V}^*$'s auxiliary input. The claim then follows from witness indistinguishability. □

Exploiting non-malleability of tagCom (step 3). Again, it suffices to show that $\left\{\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}^*(v^0), \mathcal{R}_0^*)\right\}$ and $\left\{\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}^*(v^1), \mathcal{R}_0^*)\right\}$ are indistinguishable. We begin with the observation that the only difference between these two distributions lies in Stage 1 on the left; in the former, they comprise t commitments to v^0 using tagCom and in the latter, they comprise t commitments to v^1 using tagCom. To carry out the reduction to the non-malleability of tagCom, we consider a "cut-off" point as in [28].

Let $\Phi(\mathcal{A}, z)$ denote the distribution of all joint views τ of $\mathcal{A}$ and the receivers on the right up to the point immediately after $\mathcal{A}$ sends the dummy messages in Stage 0 in the right interactions. In addition, we add to $\Phi(\mathcal{A}, z)$ the value $r \in f^{-1}(s)$ where s is the Stage 0 challenge on the left in τ.

Lemma 2 (reduction to tagCom). *For all ppt $\mathcal{A}$, there exists a ppt $\mathcal{B}$ and D such that for all z, v:*

$$\left\{\mathsf{mim}_{\mathcal{A}(z)}^{\mathsf{TagCom}}(\mathcal{C}^*(v), \mathcal{R}_0^*)\right\} \cong \left\{D(\mathsf{mim}_{\mathcal{B}(z^*)}^{\mathsf{tagCom}}(\mathcal{C}(\overbrace{v, \ldots, v}^{t\ times}), \mathcal{R})) : z^* \leftarrow \Phi(\mathcal{A}, z)\right\}$$

are statistically indistinguishable. Note that in the second distribution, there are t left interactions, all committing to v using tagCom.

Once we establish this lemma, (3.3) follows readily from the many-many non-malleability of tagCom, which guarantees that

$$\mathsf{mim}_{\mathcal{B}(z^*)}^{\mathsf{tagCom}}(\mathcal{C}(v^0, \ldots, v^0), \mathcal{R})) \cong \mathsf{mim}_{\mathcal{B}(z^*)}^{\mathsf{tagCom}}(\mathcal{C}(v^1, \ldots, v^1), \mathcal{R}))$$

Proof. The high-level idea is to construct a machine $\mathcal{B}$ that on input $z^* = z||\tau||r$ runs internally a copy of $\mathcal{A}$ and simulates the view of $\mathcal{A}$ in the experiment $\mathsf{mim}^{\mathsf{TagCom}}_{\mathcal{A}(z)}(\mathcal{C}^*(v), \mathcal{R}^*_0)$ while participating in $\mathsf{mim}^{\mathsf{tagCom}}_{\mathcal{B}(z^*)}(\mathcal{C}(v, \ldots, v), \mathcal{R})$. The machine D will essentially "post process" the committed values in the second distribution according to $\mathcal{R}^*_0$. Note that $\mathcal{B}$ will run t interactions of tagCom on the left, and tm interactions of tagCom on the right.

We first describe how to simulate the messages from $\mathcal{C}^*(v)$ in the left interaction in the view of $\mathcal{A}$:

Stage 0. Stage 0 is embedded in τ.

Stage 1. $\mathcal{B}$ chooses identities $(1, \text{ID}_1), \ldots, (t, \text{ID}_t)$ for the t left interactions (scheduled in parallel), and forwards the messages from the external $\mathcal{C}(v, \ldots, v)$ to $\mathcal{A}$ as if coming from $\mathcal{C}^*(v)$.

Stage 2. $\mathcal{B}$ computes the prover's messages in the WIPOK by using the witness r which is part of its auxiliary input z^*.

Next, we describe how $\mathcal{B}$ simulates the messages from $\mathcal{R}$ in the m executions of TagCom on the right and also how D computes the committed values. Again, let $\tilde{\text{ID}}^1, \ldots, \tilde{\text{ID}}^m$ denote the m identities on the right. For each $j = 1, \ldots, m$,

Stage 0. Stage 0 is embedded in τ.

Stage 1. $\mathcal{B}$ uses identities $(1, \tilde{\text{ID}}^j_1), \ldots, (t, \tilde{\text{ID}}^j_t)$ for the t executions of tagCom on the right. It forwards the messages from the t external copies of $\mathcal{R}$ to $\mathcal{A}$.

Stage 2. $\mathcal{B}$ simulates the verifier's messages in the WIPOK internally.

Committed value. If $\tilde{\text{ID}}^j = \text{ID}$, then D simply outputs $\perp$. Otherwise, D first computes the first index i for which $\text{ID}_i \neq \tilde{\text{ID}}^j_i$ and outputs as $\tilde{v}_j$ the committed value corresponding to the tag $(i, \tilde{\text{ID}}^j_i)$. ($D$ receives this value as part of the output of $\mathsf{mim}^{\mathsf{tagCom}}_{\mathcal{B}(z^*)}(\mathcal{C}(v, \ldots, v)), \mathcal{R})$.

This completes the reduction. □

Acknowledgments. I would like to thank Rafael Pass for numerous insightful discussions and his constant encouragement as well as the staff at Ninth Street Espresso for their hospitality.

References

1. B. Barak. How to go beyond the black-box simulation barrier. In *FOCS*, pages 106–115, 2001.
2. B. Barak. Constant-round coin-tossing with a man in the middle or realizing the shared random string model. In *FOCS*, pages 345–355, 2002.
3. B. Barak. *Non-Black-Box Techniques in Cryptography*. Ph.D., Weizmann Institute of Science, Jan. 2004.
4. B. Barak and Y. Lindell. Strict polynomial-time in simulation and extraction. *SIAM J. Comput.*, 33(4):738–818, 2004.
5. D. Beaver, S. Micali, and P. Rogaway. The round complexity of secure protocols. In *STOC*, pages 503–513, 1990.
6. M. Bellare and S. Yilek. Encryption schemes secure under selective opening attack. Cryptology ePrint Archive, Report 2009/101, 2009. http://eprint.iacr.org/.
7. R. Canetti. Security and composition of multiparty cryptographic protocols. *J. Cryptology*, 13(1):143–202, 2000.
8. R. Canetti, Y. Lindell, R. Ostrovsky, and A. Sahai. Universally composable two-party and multi-party secure computation. In *STOC*, pages 494–503, 2002.
9. S. G. Choi, D. Dachman-Soled, T. Malkin, and H. Wee. Black-box construction of a non-malleable encryption scheme from any semantically secure one. In *TCC*, pages 427–444, 2008.
10. S. G. Choi, D. Dachman-Soled, T. Malkin, and H. Wee. Improved non-committing encryption with applications to adaptively secure protocols. In *ASIACRYPT*, pages 287–302, 2009.
11. S. G. Choi, D. Dachman-Soled, T. Malkin, and H. Wee. Simple, black-box constructions of adaptively secure protocols. In *TCC*, pages 387–402, 2009.
12. I. Damgård and Y. Ishai. Constant-round multiparty computation using a black-box pseudorandom generator. In *CRYPTO*, pages 378–394, 2005.
13. D. Dolev, C. Dwork, and M. Naor. Nonmalleable cryptography. *SIAM J. Comput.*, 30(2):391–437, 2000.
14. S. Even, O. Goldreich, and A. Lempel. A randomized protocol for signing contracts. In *CRYPTO*, pages 205–210, 1982.
15. U. Feige and A. Shamir. Zero knowledge proofs of knowledge in two rounds. In *CRYPTO*, pages 526–544, 1989.
16. Y. Gertner, S. Kannan, T. Malkin, O. Reingold, and M. Viswanathan. The relationship between public key encryption and oblivious transfer. In *FOCS*, pages 325–335, 2000.
17. O. Goldreich. *Foundations of Cryptography: Basic Tools*. Cambridge University Press, 2001.
18. O. Goldreich. *Foundations of Cryptography: Volume II, Basic Applications*. Cambridge University Press, 2004.
19. O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game or a completeness theorem for protocols with honest majority. In *STOC*, pages 218–229, 1987.
20. I. Haitner. Semi-honest to malicious oblivious transfer - the black-box way. In *TCC*, pages 412–426, 2008.
21. J. Håstad, R. Impagliazzo, L. A. Levin, and M. Luby. A pseudorandom generator from any one-way function. *SIAM J. Comput.*, 28(4):1364–1396, 1999.
22. Y. Ishai, E. Kushilevitz, Y. Lindell, and E. Petrank. Black-box constructions for secure computation. In *STOC*, pages 99–108, 2006.
23. Y. Ishai, M. Prabhakaran, and A. Sahai. Founding cryptography on oblivious transfer - efficiently. In *CRYPTO*, pages 572–591, 2008.
24. J. Katz, R. Ostrovsky, and A. Smith. Round efficiency of multi-party computation with a dishonest majority. In *EUROCRYPT*, pages 578–595, 2003.
25. J. Kilian. Founding cryptography on oblivious transfer. In *STOC*, pages 20–31, 1988.
26. H. Lin and R. Pass. Non-malleability amplification. In *STOC*, pages 189–198, 2009.
27. H. Lin, R. Pass, W. D. Tseng, and M. Venkitasubramaniam. Concurrent non-malleable zero knowledge proofs. In *CRYPTO*, pages 429–446, 2010.
28. H. Lin, R. Pass, and M. Venkitasubramaniam. Concurrent non-malleable commitments from any one-way function. In *TCC*, pages 571–588, 2008.
29. H. Lin, R. Pass, and M. Venkitasubramaniam. A unified framework for concurrent security: universal composability from stand-alone non-malleability. In *STOC*, pages 179–188, 2009.
30. Y. Lindell. Parallel coin-tossing and constant-round secure two-party computation. *J. Cryptology*, 16(3):143–184, 2003.
31. Y. Lindell and B. Pinkas. An efficient protocol for secure two-party computation in the presence of malicious adversaries. In *EUROCRYPT*, pages 52–78, 2007.

32. D. Micciancio, S. J. Ong, A. Sahai, and S. P. Vadhan. Concurrent zero knowledge without complexity assumptions. In *TCC*, pages 1–20, 2006.
33. M. Naor. Bit commitment using pseudorandomness. *J. Cryptology*, 4(2):151–158, 1991.
34. R. Pass. Bounded-concurrent secure multi-party computation with a dishonest majority. In *STOC*, pages 232–241, 2004.
35. R. Pass and A. Rosen. Concurrent nonmalleable commitments. *SIAM J. Comput.*, 37(6):1891–1925, 2008.
36. R. Pass, A. Shelat, and V. Vaikuntanathan. Construction of a non-malleable encryption scheme from any semantically secure one. In *CRYPTO*, pages 271–289, 2006.
37. R. Pass and H. Wee. Black-box constructions of two-party protocols from one-way functions. In *TCC*, pages 403–418, 2009.
38. R. Pass and H. Wee. Constant-round non-malleable commitments from sub-exponential one-way functions. In *EUROCRYPT*, pages 638–655, 2010.
39. C. Peikert, V. Vaikuntanathan, and B. Waters. A framework for efficient and composable oblivious transfer. In *CRYPTO*, pages 554–571, 2008. Also, Cryptology ePrint Archive, Report 2007/118.
40. C. Peikert and B. Waters. Lossy trapdoor functions and their applications. In *STOC*, pages 187–196, 2008.
41. M. Prabhakaran, A. Rosen, and A. Sahai. Concurrent zero knowledge with logarithmic round-complexity. In *FOCS*, pages 366–375, 2002.
42. S. Wolf and J. Wullschleger. Oblivious transfer is symmetric. In *EUROCRYPT*, pages 222–232, 2006.

A Non-malleability w.r.t. Extraction

A.1 Definitions

We begin with definitions for non-malleability w.r.t extraction.

Concurrently extractable commitments. Let $(\mathcal{C}, \mathcal{R})$ be a statistically binding commitment scheme. We say that $(\mathcal{C}, \mathcal{R})$ is a *concurrently extractable commitment scheme* if there exists an expected polynomial-time probabilistic oracle machine (the extractor) $\textsc{Ext}$ that given oracle access to any PPT cheating sender $\mathcal{C}^*$ participating in $m = m(n)$ concurrent executions of $(\mathcal{C}, \mathcal{R})$ outputs a view τ and a m-tuple of values $(v_1^*, \ldots, v_m^*)$ such that:

- (simulation) τ is identically distributed to the view of $\mathcal{C}^*$ at the end of interacting with m executions of an honest receiver $\mathcal{R}$ in commit phase.
- (extraction) Let $v_1, \ldots, v_m$ be the committed values as defined by the transcript τ. Then, $\Pr[\exists i : v_i \neq \perp$ and $v_i^* \neq v_i]$ is negligible.

The second condition means that except with negligible probability, if τ is a valid commitment to some value (not equal to $\perp$), then the extractor must output that value, and otherwise, there is no restriction on the output of the extractor.

Proposition 3 ([37, 32, 41]). *There exists a (fully) black-box construction of a 4-round public-coin concurrently extractable commitment* ExtCom *from one-way functions.*

Non-malleability w.r.t extraction. We consider a notion of non-malleability for concurrently extractable commitments. As before, consider a man-in-the-middle execution for a concurrently extractable commitment scheme $(\mathcal{C}, \mathcal{R})$. We use the random variable $\mathsf{extmim}_{\mathcal{A}(z)}(\mathcal{C}(v_1, \ldots, v_m), \mathcal{R})$ to denote the output $(\tau, \tilde{v}_1^*, \ldots, \tilde{v}_m^*)$ of $\textsc{Ext}$, treating $\mathcal{A}$ along with the m left interactions as a single cheating sender for the m right interactions. As before, for any i such that $\tilde{\text{ID}}^i \in \{\text{ID}^1, \ldots, \text{ID}^m\}$, we set $\tilde{v}_i^*$ to $\perp$.[7]

Definition 2. *A concurrently extractable commitment scheme $(\mathcal{C}, \mathcal{R})$ is* concurrent non-malleable w.r.t extraction *if for every* PPT *$\mathcal{A}$ and every polynomial $m = m(n)$ and every m values $(v_1, \ldots, v_m)$ along with any $z \in \{0,1\}^*$, the distributions*

$$\left\{\mathsf{extmim}_{\mathcal{A}(z)}(\mathcal{C}(v_1^0, \ldots, v_m^0), \mathcal{R})\right\} \text{ and } \left\{\mathsf{extmim}_{\mathcal{A}(z)}(\mathcal{C}(v_1^1, \ldots, v_m^1), \mathcal{R})\right\}$$

are computationally indistinguishable.

We have the following analogue of Proposition 1 for non-malleability w.r.t extraction:

Proposition 4. *Let $(\mathcal{C}, \mathcal{R})$ be a commitment scheme that is one-many concurrent non-malleable w.r.t extraction (resp. and w.r.t. synchronizing adversaries). Then, $(\mathcal{C}, \mathcal{R})$ is also concurrent non-malleable w.r.t. extraction (resp. and w.r.t. synchronizing adversaries).*

[7] If in addition we set $\tilde{v}_i^*$ to $\perp$ whenever the actual committed value $\tilde{v}_i$ is $\perp$, then we recover a distribution that is statistically close to $\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}(v_1, \ldots, v_m), \mathcal{R})$.

Common input : security parameter 1^n and an identity $\textsc{id} = (\textsc{id}_1, \ldots, \textsc{id}_t) \in \{0,1\}^t$.
Sender's input : a value $v \in \{0,1\}^{\mathrm{poly}(n)}$.

....

COMMIT PHASE.
Stage 0: $\mathcal{R}$ commits to a random subset $\Gamma \subset [10n]$ of size n. $\mathcal{C}$ responds with a dummy message.
Stage 1: $\mathcal{C}$ computes shares $\mathbf{s}$ of v using a n-out-of-$10n$ secret-sharing scheme and commits to the shares using tagCom with tags $(1, \textsc{id}_1), \ldots, (t, \textsc{id}_t)$.

- $\mathcal{C}$ picks a random degree n polynomial p over $\mathrm{GF}(2^{|v|})$ whose constant term is v, and computes $\mathbf{s} = (s_1, \ldots, s_{10n}) = (p(1), \ldots, p(10n))$. (Note that $\mathbf{s}$ is a Reed-Solomon encoding of v.)
- $\mathcal{C}$ executes $\mathsf{tagCom}(\mathsf{id}_i, s_1), \ldots, \mathsf{tagCom}(\mathsf{id}_i, s_{10n})$ in parallel, for $i = 1, 2, \ldots, t$ and where $\mathsf{id}_i = (i, \textsc{id}_i)$.

Stage 2: $\mathcal{C}$ proves consistency of the commitments by opening to the shares indexed by Γ.

- $\mathcal{R}$ opens the commitment to Γ.
- $\mathcal{C}$ opens all t commitments to s_j in Stage 1 for all $j \in \Gamma$.
- $\mathcal{R}$ checks that all t commitments to s_j are consistent for all $j \in \Gamma$.

....

OPEN PHASE.

- $\mathcal{C}$ sends v and opens the commitment to $\mathbf{s}$ corresponding to the tag $(1, \textsc{id}_1)$ in Stage 1.
- $\mathcal{R}$ computes the codeword $\mathbf{w}$ that is 0.9-close to $\mathbf{s}$.
- $\mathcal{R}$ checks that $\mathbf{w}$ is a codeword corresponding to v and that $\mathbf{w}$ and $\mathbf{s}$ agree on all positions in Γ.

Fig. 3.

A.2 The construction and analysis

We present our black-box construction for non-malleability amplification in Figure 3. The key step in our analysis lies in establishing the following analogue of Prop 2:

Proposition 5 (Black-box amplification with synchronization). *For every $t = t(n) \geq 4$, if* tagCom *is one-many non-malleable w.r.t extraction and synchronizing adversaries for identities of length* $\log t + 1$*, then* $(\mathcal{C}, \mathcal{R})$ *is one-many non-malleable w.r.t. extraction and synchronizing adversaries for identities of length t.*

As before, we first present an alternative open phase for the commitment scheme TagCom.

An alternative opening $\mathcal{R}^*$ for TagCom. As before, we consider an "alternative open phase" for the man-in-the-middle execution corresponding to a "receiver" $\mathcal{R}_0^*$ and $\mathcal{R}^*$. More formally, we first write $\mathsf{extmim}_{\mathcal{A}(z)}(\mathcal{C}(v), \mathcal{R}_0^*)$ to denote a random variable that is the same as $\mathsf{extmim}_{\mathcal{A}(z)}(\mathcal{C}(v), \mathcal{R})$, except the m-tuple of values $(\tilde{v}_1, \ldots, \tilde{v}_m)$ is defined as follows: for $\tilde{\textsc{id}} \in \{\tilde{\textsc{id}}^1, \ldots, \tilde{\textsc{id}}^m\}$, we set the corresponding committed value $\tilde{v}$ as follows:

- if $\tilde{\textsc{id}} = \textsc{id}$, then we set $\tilde{v}$ to $\perp$,

– if $\tilde{\text{ID}} \neq \text{ID}$, let $i \in [t]$ be the first index such that $\tilde{\text{ID}}_i \neq \text{ID}_i$. Then we determine $\tilde{v}$ as follows (analogous to $\mathsf{NMDec}^*_{\mathsf{SK}}$ in [9, Section 4.3]):

- We define the vector $\tilde{\mathbf{s}}' = (\tilde{s}'_1, \ldots, \tilde{s}'_{10n})$ to be the extracted values[8] in the i'th execution of tagCom in Stage 1 on the right corresponding to the tag $(i, \tilde{\text{ID}}_i)$.
- Compute a codeword $\tilde{\mathbf{w}}$ that is 0.8-close to $\tilde{\mathbf{s}}'$.
- If $\tilde{\mathbf{w}}$ and $\tilde{\mathbf{s}}'$ agree on all positions on Γ, output the value $\tilde{v}$ corresponding to the codeword $\tilde{\mathbf{w}}$, else output $\bot$.

Next, we write $\mathsf{extmim}_{\mathcal{A}(z)}(\mathcal{C}(v), \mathcal{R}^*)$ to denote a random variable that is the same as $\mathsf{extmim}_{\mathcal{A}(z)}(\mathcal{C}(v), \mathcal{R}^*_0)$, except each committed value $\tilde{v}$ is set to $\bot$ whenever the cut-and-choose phase in Stage 2 is rejecting.

A.3 The hybrid argument

STEP 1: SWITCHING TO $\mathcal{R}^*$. We claim that

$$\left\{\mathsf{extmim}_{\mathcal{A}(z)}(\mathcal{C}(v^0), \mathcal{R})\right\} \cong \left\{\mathsf{extmim}_{\mathcal{A}(z)}(\mathcal{C}(v^0), \mathcal{R}^*)\right\}$$

STEP 2: SWITCHING TO $\mathcal{C}(v_1)$. We switch the left commitment in Stage 1 to v^1, i.e., we replace $\mathcal{C}(v^0)$ in the left execution with $\mathcal{C}(v^1)$. As before, we will use a reduction to the many-many non-malleability of tagCom to argue that:

$$\left\{\mathsf{extmim}_{\mathcal{A}(z)}(\mathcal{C}(v^0), \mathcal{R}^*)\right\} \cong \left\{\mathsf{extmim}_{\mathcal{A}(z)}(\mathcal{C}(v^1), \mathcal{R}^*)\right\}$$

STEP 3: SWITCHING BACK TO $\mathcal{R}$.

$$\left\{\mathsf{extmim}_{\mathcal{A}(z)}(\mathcal{C}(v^1), \mathcal{R}^*)\right\} \cong \left\{\mathsf{extmim}_{\mathcal{A}(z)}(\mathcal{C}(v^1), \mathcal{R})\right\}$$

Switching to $\mathcal{R}^*$ (step 1, also step 3). Let $\mathbf{s}$ denote the extracted values in the first execution of tagCom in stage 1 on the right corresponding to the tag $(1, \tilde{\text{ID}}_1)$, whereas $\mathbf{s}'$ is that corresponding to the tag $(i, \tilde{\text{ID}}_i)$. Note that $\mathbf{s}$ and $\mathbf{s}'$ are used to determine $\tilde{v}$ in $\mathsf{extmim}_{\mathcal{A}(z)}(\mathcal{C}(v^0), \mathcal{R})$ and $\mathsf{extmim}_{\mathcal{A}(z)}(\mathcal{C}(v^0), \mathcal{R}^*)$ respectively. We consider two cases, almost exactly as in [9, 37]:

YES *instance* — $\tilde{\mathbf{s}}$ is 0.9-close to some codeword $\tilde{\mathbf{w}}$ and $\tilde{\mathbf{s}}'$ is 0.9-close to $\tilde{\mathbf{s}}$.

This means that $\tilde{\mathbf{s}}'$ is 0.8-close to $\tilde{\mathbf{w}}$. Therefore, both $\mathcal{R}$ and $\mathcal{R}^*$ will recover the same codeword $\tilde{\mathbf{w}}$ and therefore the same value $\tilde{v}$.

NO *instance* — either $\tilde{\mathbf{s}}$ is 0.1-far from every codeword or $\tilde{\mathbf{s}}'$ is 0.1-far from $\tilde{\mathbf{s}}$.

We claim here that (with high probability) both $\mathcal{R}$ and $\mathcal{R}^*$ will output $\bot$. Suppose $\mathcal{R}$ commited to a dummy value instead of Γ in Stage 0; then this follows from a simple statistical argument as in [9, Section 4.2]. To switch from a dummy commitment to a commitment to Γ, we follow exactly the same argument as in [37] (the idea is to exploit the fact that the commitment to Γ is hiding and that we can extract the values $\tilde{\mathbf{s}}$ and $\tilde{\mathbf{s}}'$).

[8] By "extracted values", we refer to the values output by the extractor for tagCom.

Exploiting non-malleability of tagCom (Step 2). We provide a sketch of the analysis for Step 2, which relies on the many-many non-malleability of tagCom. It suffices to show that:

$$\left\{\mathsf{extmim}_{\mathcal{A}(z)}(\mathcal{C}(v^0), \mathcal{R}_0^*)\right\} \cong \left\{\mathsf{extmim}_{\mathcal{A}(z)}(\mathcal{C}(v^1), \mathcal{R}_0^*)\right\}$$

As before, we work with $\mathcal{R}_0^*$ instead of $\mathcal{R}^*$ for simplicity as it allows us to essentially ignore the outcome of Stage 2. We define Φ and Ξ as in Section 3, except Ξ now extracts (Γ, σ) from $\mathsf{Com}(\Gamma; \sigma)$ instead of r from s. The following lemma is an analogue of Lemma 2.

Lemma 3 (reduction to tagCom**).** *For all ppt $\mathcal{A}$, there exists a ppt $\mathcal{B}$ and D such that for all z, v^0, v^1, there exists $(s_1^0, \ldots, s_{9n}^0), (s_1^1, \ldots, s_{9n}^1)$ such that for all $b \in \{0,1\}$:*

$$\left\{\mathsf{extmim}_{\mathcal{A}(z)}^{\mathsf{TagCom}}(\mathcal{C}_{0,b}^*(v^0, v^1), \mathcal{R}_0^*)\right\} \qquad \textit{and}$$

$$\left\{D(\mathsf{extmim}_{\mathcal{B}(z^*)}^{\mathsf{tagCom}}(\mathcal{C}(s_1^b, \ldots, s_{9n}^b, \ldots, s_1^b \ldots, s_{9n}^b), \mathcal{R})) : z^* \leftarrow \Xi(\Phi(\mathcal{A}, z))\right\}$$

are statistically indistinguishable.

Proof. We construct $\mathcal{B}$ essentially as in the proof of Lemma 2 and also the proof of the third claim in [9, Section 4.3]. For notational simplicity and WLOG, we assume $\Gamma = \{9n+1, \ldots, 10n\}$. We may then sample random shares $\mathbf{s}^0 = (s_0^0, \ldots, s_{10n}^0)$ and $\mathbf{s}^1 = (s_1^1, \ldots, s_{10n}^1)$ of the messages v^0 and v^1 that agree on all of the positions in Γ, that is, $s_{9n+1}^0 = s_{9n+1}^1, \ldots, s_{10n}^0 = s_{10n}^1$.

Next, we describe how to simulate the messages from $\mathcal{C}(v^b)$ in the left interaction in the view of $\mathcal{A}$:

Stage 0. Stage 0 is embedded in τ.

Stage 1. $\mathcal{B}$ internally simulates tagCom for the indices in Γ in parallel with the messages from the external sender $\mathcal{C}$.

- The choice of the $9nt$ external executions of tagCom on the left will depend on the set Γ, which we will assume WLOG to be $\{9n+1, \ldots, 10n\}$ for notational simplicity.
- The $9nt$ external executions of tagCom will correspond to commitments to $s_1^b, \ldots, s_{9n}^b$ with identities $\mathsf{id}_1, \ldots, \mathsf{id}_t$, i.e., $\mathsf{tagCom}(\mathsf{id}_1, s_1^b), \ldots, \mathsf{tagCom}(\mathsf{id}_1, s_{9n}^b), \ldots, \mathsf{tagCom}(\mathsf{id}_t, s_1^b), \ldots, \mathsf{tagCom}(\mathsf{id}_t, s_{9n}^b)$
- $\mathcal{B}$ will internally simulate nt executions of tagCom corresponding to commitments of $s_{9n+1}^b, \ldots, s_{10n}^b$ (these n values do not depend on b) with identities $\mathsf{id}_1, \ldots, \mathsf{id}_t$, i.e. $\mathsf{tagCom}(\mathsf{id}_1, s_{9n+1}^b), \ldots, \mathsf{tagCom}(\mathsf{id}_1, s_{10n}^b), \ldots,$ $\mathsf{tagCom}(\mathsf{id}_t, s_{9n+1}^b), \ldots, \mathsf{tagCom}(\mathsf{id}_t, s_{10n}^b)$

Stage 2. Internally simulate the openings for the indices in Γ.

- If $\mathcal{A}$ opens the Stage 0 commitment to a value different from Γ, abort. Otherwise, open the commitments nt to the values $s_{9n+1}^b, \ldots, s_{10n}^b$.

Next, we describe how $\mathcal{B}$ handles the m executions of TagCom on the right, with identities $\tilde{\mathrm{ID}}^1, \ldots, \tilde{\mathrm{ID}}^m$. For each $j = 1, \ldots, m$:

Stage 0. Stage 0 is embedded in τ.

Stage 1. $\mathcal{B}$ initiates $10nt$ external executions of tagCom on the right, with $10n$ executions for each of the identities $\mathsf{id}_1^j, \ldots, \mathsf{id}_t^j$.

Stage 2. $\mathcal{B}$ internally simulates the opening to the challenges in Stage 0 (it gets the randomness used for the commmitments $\mathsf{Com}(\Gamma)$ from z^*).

Committed value. If $\tilde{\text{ID}}^j = \text{ID}$,then D simply outputs $\perp$. Otherwise, D first computes the first index i for which $\text{ID}_i \neq \tilde{\text{ID}}^j_i$ and computes $\tilde{v}_j$ from the extracted values corresponding to the tag $(i, \tilde{\text{ID}}^j_i)$ as in $\mathcal{R}^*_0$.

B Constant-Round Non-Malleable Commitments from Sub-Exponential OWFs

Suppose there exists a one-way function secure against sub-exponential size circuits. Then, we may construct a hierarchy of commitment schemes $\mathsf{Com}_0, \ldots, \mathsf{Com}_{d+1}$ with $d = \log\log\log n + O(1)$ such that:

- For each $i = 0, 1, \ldots, d-1$: Com_i (and ExtCom_i) is T_i-hiding but can be broken in time $T_{i+1}^{1/2}$.

Proposition 6 (Non-malleability from sub-exponential hardness). *Suppose there exists one-way functions secure against sub-exponential sized circuits. Then, the commitment scheme shown in Fig 4 is one-many non-malleable w.r.t. extraction for identities of length* $\log\log\log n + O(1)$.

Common input : security parameter 1^n and an identity $\mathsf{id} \in \{0, 1, \ldots, d-1\}$.

Sender's input : a value $v \in \{0,1\}^{\mathrm{poly}(n)}$.

. . . .

COMMIT PHASE.

Stage 0:

$\mathcal{C} \to \mathcal{R}$: $\mathcal{C}$ computes shares $\mathbf{s} = (s_1, \ldots, s_{10n})$ of v using Shamir's n-out-of-$10n$ secret-sharing scheme and commits to the shares using Com_d.

Stage 1:

- $\mathcal{R}$ commits to a random string γ_1^{R} using $\mathsf{ExtCom}_{\mathsf{id}}$.
- $\mathcal{C}$ commits to $\mathbf{s}$ using ExtCom_{d+1}, and in addition, sends a random string γ_1^{S}.
- $\mathcal{R}$ opens the commitment to γ_1^{R} and both parties use $\gamma_1^{\mathsf{R}} \oplus \gamma_1^{\mathsf{S}}$ to select a subset $\Gamma_1 \subset [10n]$ of size $n/2$.
- $\mathcal{C}$ opens the commitments to s_j in Stages 0 and 1 for all $j \in \Gamma_1$.
- $\mathcal{R}$ checks that the commitments are consistent for all $j \in \Gamma_1$.

Stage 2:

- $\mathcal{R}$ commits to a random string γ_2^{R} using $\mathsf{ExtCom}_{d-\mathsf{id}}$.
- $\mathcal{C}$ commits to $\mathbf{s}$ using ExtCom_{d+1}, and in addition, sends a random string γ_2^{S}.
- $\mathcal{R}$ opens the commitment to γ_2^{R} and both parties use $\gamma_2^{\mathsf{R}} \oplus \gamma_2^{\mathsf{S}}$ to select a subset $\Gamma_2 \subset [10n]$ of size $n/2$.
- $\mathcal{C}$ opens the commitments to s_j in Stages 0 and 2 for all $j \in \Gamma_2$.
- $\mathcal{R}$ checks that the commitments are consistent for all $j \in \Gamma_2$.

. . . .

OPEN PHASE.

- $\mathcal{C}$ sends v and opens the commitment to $\mathbf{s}$ in Stage 0.
- $\mathcal{R}$ computes the codeword $\mathbf{w}$ that is 0.9-close to $\mathbf{s}$.
- $\mathcal{R}$ checks that $\mathbf{w}$ is a codeword corresponding to v and that $\mathbf{w}$ and $\mathbf{s}$ agree on all positions in $\Gamma_1 \cup \Gamma_2$.

Fig. 4. The commitment scheme $\mathsf{tagCom} = (\mathcal{C}, \mathcal{R})$ for short identities.

C Secure MPC

We consider the standard stand-alone model for secure computation [7, 18]. Specifically, we consider a fully synchronized network with a global clock, a rushing adversary, and where all n players have unique identities. For simplicity, we also assume pairwise private channels, an assumption that can be removed with the use of public-key encryption.

C.1 Building blocks and the n-party OT protocol

We require two building blocks to instantiate our compiler Comp shown in Figure 5: a constant-round OT protocol Π that is the input to Comp, and a commitment scheme that Comp relies on.

OT tolerating malicious senders. Our first building block is a constant-round OT protocol Π that is secure against a malicious sender and a semi-honest receiver. As it turns out, many existing semi-honest OT protocols already have this property (c.f. [39, 10]):

Proposition 7. *There exists a (fully) black-box construction of a constant-round OT protocol secure against a malicious sender and a semi-honest receiver, starting from lossy encryption schemes, homomorphic encryption schemes, dense cryptosystems or certifiable enhanced trapdoor permutations. Moreover, the OT protocol admits a straight-line simulator.*

We sketch the proofs for the OT protocols based on lossy encryption schemes or dense cryptosystems. These are 2-message protocols where the receiver first sends two public keys, such that it can decrypt for exactly one of the two keys (the other being either lossy or oblivously sampled), and the sender sends encryptions of both secrets using the respectively public key. To handle a malicious sender, the simulator simply generates both public keys along with the corresponding secret keys so that it can decrypt messages encrypted either key and thus extract both of the sender's inputs.[9] For the semi-honest OT protocol based on homomorphic encryption schemes from [22], the same idea allows us to handle a malicious receiver but not a malicious sender; we will handle this by first applying OT reversal [42] to the protocol.

The n-party OT protocol. In the n-party OT protocol, every pair of parties P_i, P_j run two executions $\mathsf{Comp}(\Pi)$ in parallel, one with P_i as the sender and the other with P_j as the sender. Whenever the parties in $\mathsf{Comp}(\Pi)$ run a commitment scheme, we execute our non-malleable commitment scheme tagCom in parallel. The adversary $\mathcal{A}$ in the n-party OT protocol then corresponds to a man-in-the-middle adversary for the commitment scheme. Looking ahead, we handle the simultaneous commitments and extraction in the simulation by running $\textsc{Ext}$.

C.2 Proof of security

Lemma 4. *Suppose* Π *is a OT protocol that is secure against a malicious sender and a semi-honest receiver with a straight-line simulator and that* tagCom *is a many-many non-malleable commitment scheme w.r.t. extraction and synchronizing adversaries for* n *identities. Then,* $\mathsf{Comp}(\Pi)$ *realizes the* n*-party OT functionality against a malicious adversary that may corrupt any number of parties. Moreover, if* Π *is constant-round and* tagCom *has* c *rounds, then* $\mathsf{Comp}(\Pi)$ *has* $O(c)$ *rounds.*

[9] Here, we require perfect correctness of the decryption algorithm to hold w.h.p. over the public key.

Common input : security parameter 1^n
Sender's input : $(\mathsf{sender}, \mathit{sid}, s_0, s_1)$ where $s_0, s_1 \in \{0,1\}^\ell$.
Receiver's input : $(\mathsf{receiver}, \mathit{sid}, r)$ where $r \in \{0,1\}$.

. . . .

PHASE I: RANDOM TAPE GENERATION.

1. $\mathbf{R}$ chooses $2n$ random strings $(r_1^{\mathsf{R}}, \tau_1^{\mathsf{R}}), \ldots, (r_{2n}^{\mathsf{R}}, \tau_{2n}^{\mathsf{R}})$ and commits to $(r_i^{\mathsf{R}}, \tau_i^{\mathsf{R}})$, for $i = 1, 2, \ldots, 2n$.
2. $\mathbf{S}$ sends $2n$ random strings $(r_1^{\mathsf{S}}, \tau_1^{\mathsf{S}}), \ldots, (r_{2n}^{\mathsf{S}}, \tau_{2n}^{\mathsf{S}})$.
3. $\mathbf{R}$ sets $r_i = r_i^{\mathsf{R}} \oplus r_i^{\mathsf{S}}$ and $\tau_i = \tau_i^{\mathsf{R}} \oplus \tau_i^{\mathsf{S}}$, for $i = 1, 2, \ldots, 2n$.

PHASE II: BASIC EXECUTION.

1. $\mathbf{S}$ chooses $2n$ pairs of random inputs $(s_1^0, s_1^1), \ldots, (s_{2n}^0, s_{2n}^1)$.
2. $\mathbf{S}$ and $\mathbf{R}$ engages in $2n$ parallel executions of the protocol Π. In the ith execution, $\mathbf{S}$ inputs (s_i^0, s_i^1) and $\mathbf{R}$ inputs r_i with randomness τ_i and obtains output $s_i^{r_i}$.

PHASE III: CUT-AND-CHOOSE.

1. $\mathbf{S}$ commits to a random string $q_{\mathsf{S}} \in \{0,1\}^n$.
2. $\mathbf{R}$ sends a random string $q_{\mathsf{R}} \in \{0,1\}^n$.
3. $\mathbf{S}$ opens the commitment to q_{S} and both parties compute $q = (q_1, \ldots, q_n) = q_{\mathsf{R}} \oplus q_{\mathsf{S}}$. The string q is used to define a set of indices $Q \subset \{1, 2, \ldots, 2n\}$ of size n in the following way: $Q = \{2i - q_i\}_{i=1}^n$.
4. For every $i \in Q$, $\mathbf{R}$ opens the commitment to $(r_i^{\mathsf{R}}, \tau_i^{\mathsf{R}})$, and $\mathbf{S}$ computes (r_i, τ_i).
5. $\mathbf{S}$ checks that for all $i \in Q$, (r_i, τ_i) is consistent with $\mathbf{R}$'s messages in the i'th execution of Π. If not, $\mathbf{S}$ aborts and halts.

PHASE IV: COMBINER.

1. For every $j \notin Q$, $\mathbf{R}$ computes $\alpha_j = r \oplus r_j$ and sends $\{\alpha_j\}_{j \notin Q}$ to $\mathbf{S}$.
2. $\mathbf{S}$ computes $\sigma_0 = s_0 \oplus (\bigoplus_{j \notin Q} s_j^{\alpha_j})$ and $\sigma_1 = s_1 \oplus (\bigoplus_{j \notin Q} s_j^{1-\alpha_j})$ and sends (σ_0, σ_1).
3. $\mathbf{R}$ computes and outputs $s_r = \sigma_r \oplus (\bigoplus_{j \notin Q} s_j^{r_j})$.

Fig. 5. OT protocol $\mathsf{Comp}(\Pi)$

The simulator and the analysis are similar to that in [11, 22]; the key difference lies in bounding the failure probability of the simulator, which is related to the "soundness" of the cut-and-choose phase against a cheating receiver. (A cheating receiver breaks soundness if for all $i \notin Q$, it did not behave consistently in the i'th execution of Π.) The previous analysis relies on the fact that the sender's commitment in Phase III is equivocable. Our analysis only relies on the fact that the commitment scheme is hiding (even amidst extraction) and exploits the observation from [11] that the cut-and-choose phase has an efficiently recoverable "easy challenge" (c.f. [37]). (This is because by extracting from the cheating receiver's commitments in Phase I, we may efficiently check whether it is behaving consistently for each execution of Π in Phase II.)

Our simulator. We construct a simulator $\mathcal{S}$ in the ideal execution that given access to a n-party OT functionality, simulates the view of an adversary $\mathcal{A}$ in the real execution. As usual, the simulator $\mathcal{S}$ works by invoking a copy of $\mathcal{A}$ and simulating its interaction with the honest parties. In particular, $\mathcal{S}$ does the following in parallel for each pair of parties running $\mathsf{Comp}(\Pi)$:

– *Simulating the case when only the sender is corrupted:* $\mathcal{S}$ begins by picking a random $q \in \{0,1\}^n$ and computing the corresponding $Q \subset \{1, 2, \ldots, 2n\}$, then proceeds as follows:

  PHASE I. For each $i \in Q$, $\mathcal{S}$ follows the strategy of the honest $\mathbf{R}$ to commit to random $(r_i^{\mathrm{R}}, \tau_i^{\mathrm{R}})$ and to compute (r_i, τ_i). For each $i \notin Q$, $\mathcal{S}$ commits to $(0, 0^{\mathrm{poly}(n)})$ as $\mathbf{R}$.

  PHASE II. For each $i \in Q$, $\mathcal{S}$ continues to follow the strategy of the honest $\mathbf{R}$ and executes Π as the receiver with input r_i and randomness τ_i. For each $i \notin Q$, $\mathcal{S}$ runs the simulator for Π to simulate the view of $\mathcal{A}$ and in addition, extract its input (s_i^0, s_i^1).

  PHASE III. $\mathcal{S}$ extracts the value q_{S} that $\mathcal{A}$ committed to as $\mathbf{S}$, and sends $q_{\mathrm{R}} = q_{\mathrm{S}} \oplus q$ to $\mathcal{A}$ as if sent from $\mathbf{R}$. When $\mathcal{A}$ opens its commitment to q_{S}, $\mathcal{S}$ opens its commitment to $(r_i^{\mathrm{R}}, \tau_i^{\mathrm{R}})$ for all $i \in Q$.

  PHASE IV. $\mathcal{S}$ sends random $\{\alpha_j\}_{j \notin Q}$ to $\mathcal{A}$ as if sent from $\mathbf{R}$. When $\mathcal{A}$ sends (σ_0, σ_1) as $\mathbf{S}$, $\mathcal{S}$ computes $s_0 = \sigma_0 \oplus (\bigoplus_{j \notin Q} s_j^{\alpha_j})$ and $s_1 = \sigma_1 \oplus (\bigoplus_{j \notin Q} s_j^{1-\alpha_j})$. Next, $\mathcal{S}$ sends $(\mathsf{sender}, s_0, s_1)$ to $\mathcal{F}_{\mathrm{OT}}$ as if sent from $\mathbf{S}$.

– *Simulating the case when only the receiver is corrupted:*

  PHASE I. For each $i = 1, 2, \ldots, 2n$, $\mathcal{S}$ extracts the value $(r_i^{\mathrm{S}}, \tau_i^{\mathrm{S}})$ that $\mathcal{A}$ committed to as $\mathbf{R}$, picks a random string $(r_i^{\mathrm{R}}, \tau_i^{\mathrm{R}})$, sets $r_i^{\mathrm{S}} = r_i \oplus r_i^{\mathrm{R}}$ and $\tau_i^{\mathrm{S}} = \tau_i \oplus \tau_i^{\mathrm{R}}$ and internally passes $(r_1^{\mathrm{S}}, \tau_1^{\mathrm{S}}), \ldots, (r_{2n}^{\mathrm{S}}, \tau_{2n}^{\mathrm{S}})$ to $\mathcal{A}$ as if sent by $\mathbf{S}$ to $\mathbf{R}$.

  PHASE II AND III. $\mathcal{S}$ runs the honest sender algorithm.

  PHASE IV. $\mathcal{S}$ computes $j^* \notin Q$ such that the values (r_{j^*}, τ_{j^*}) are consistent with the messages $\mathcal{A}$ sent as $\mathbf{R}$ in the j^*'th execution of Π in Phase II, and output $\mathsf{failure}$ if such a j^* does not exist. When $\mathcal{A}$ sends $\{\alpha_j\}_{j \notin Q}$ as $\mathbf{R}$, $\mathcal{S}$ computes $r = \alpha_{j^*} \oplus r_{j^*}$ and sends $(\mathsf{receiver}, \mathit{sid}, r)$ to $\mathcal{F}_{\mathrm{OT}}$. Upon receiving (sid, s_r) from $\mathcal{F}_{\mathrm{OT}}$, compute (σ_0, σ_1) so that σ_r is consistent with s_r as follows:
  - If $r = 0$, then $\sigma_0 = s_0 \oplus (\bigoplus_{j \notin Q} s_j^{\alpha_j})$ and σ_1 is a random string in $\{0,1\}^\ell$.
  - If $r = 1$, then σ_0 is a random string in $\{0,1\}^\ell$ and $\sigma_1 = s_1 \oplus (\bigoplus_{j \notin Q} s_j^{1-\alpha_j})$.

  $\mathcal{S}$ then sends (σ_0, σ_1) to $\mathcal{A}$ as if sent by $\mathbf{S}$ to $\mathbf{R}$.

Analyzing $\mathcal{S}$. By an analogous analysis to that in [11, 22], it follows that if $\mathcal{S}$ does not output $\mathsf{failure}$, the output of $\mathcal{S}$ in the ideal execution is indistinguishable from that of $\mathcal{A}$ in the real execution. Specifically, we will use a distinguisher for this two distributions to violate the security of Π. Roughly speaking, the reduction works by randomly embedding an execution of Π into $\mathsf{Comp}(\Pi)$ (we also need to pick a random $\mathsf{Comp}(\Pi)$ for a random pair of parties), and whenever $\mathcal{S}$ does not output $\mathsf{failure}$, we have that with probability at least $1/2n$, $\mathcal{A}$ behaves like a semi-honest receiver in the embedded execution of Π.

Bounding failure probability of $\mathcal{S}$. It remains to show that probability $1 - \mathrm{neg}(n)$, $\mathcal{S}$ does not output $\mathsf{failure}$ in any execution of $\mathsf{Comp}(\Pi)$. This is where our analysis differs significantly from previous works. Suppose otherwise, that is, $\mathcal{S}$ outputs $\mathsf{failure}$ in some execution of $\mathsf{Comp}(\Pi)$ with non-negligible probability ϵ. Fix a $\mathsf{Comp}(\Pi)$ execution for which this happens, and observe that $\mathcal{S}$ only outputs $\mathsf{failure}$ if only the receiver is corrupted and at the start of Phase IV, all of the following conditions hold:

- For each of the n pairs of executions of Π corresponding indices $(1, 2), \ldots, (2n - 1, 2n)$, $\mathcal{A}$ (as the corrupted receiver) behaved consistently with (r_i, τ_i) in exactly one execution in each pair. This is because if there is a pair where both are inconsistent, then the sender will abort at the end of Phase III; and if there is a pair where both are consistent, then there exists a $j^* \notin Q$ in that pair.
- Let q^* denotes the unique challenge that allows $\mathcal{A}$ to cheat. If $\mathcal{S}$ outputs $\mathsf{failure}$, it must be because after the honest commits to q_{S}, $\mathcal{A}$ sends q_{R} such that $q_{\mathrm{R}} \oplus q_{\mathrm{S}} = q^*$. Note that the simulator can compute q^* efficiently since it knows all the (r_i, τ_i) (by extraction for Phase I).

Now, suppose we pick a random execution π of $\mathsf{Comp}(\Pi)$ and run $\mathcal{S}$ – in particular, committing to a random q_{S} – up to the point that the embedded $\mathcal{A}$ sends q_{R}. Then with probability $\epsilon/2n^2$, we have that $q_{\mathrm{R}} = q_{\mathrm{S}} \oplus q^*$. Note that while committing to a random q_{S}, $\mathcal{S}$ is possibly simultaneously extracting a commitment to q_{S} from $\mathcal{A}$ in a different execution π' of $\mathsf{Comp}(\Pi)$ where $\mathcal{A}$ corrupted the sender, in order to compute q_{R} in π'. Moreover, $\mathcal{A}$'s choice of q_{R} in π may in turn depend on that in π'. Informally, non-malleability w.r.t. extraction guarantees that changing the commitment to q_{S} in π to a commitment to 0^n does not affect q_{R} in π' and therefore will not affect q_{R} in π. This is because we may compute q_{R} in π' and π from the output of $\textsc{Ext}$.

More precisely, suppose we still pick a random q_{S} while running $\mathcal{S}$ but commit to 0^n instead of q_{S} in π. Then, non-malleability guarantees that with probability $\epsilon/2n^2 - \mathrm{neg}(n)$, we still have that $q_{\mathrm{R}} = q_{\mathrm{S}} \oplus q^*$ in π (again because we may compute q_{R} from the output of $\textsc{Ext}$). This is impossible because the view of $\mathcal{A}$ is statistically independent of q_{S}.

D Handling Non-Synchronizing Adversaries

We present our construction for handling non-synchronizing adversaries in Fig 6.

Proposition 8 (Handling non-synchronization). *If* tagCom *is non-malleable w.r.t. synchronizing adversaries and non-malleable w.r.t* 4*-round protocols and non-synchronizing adversaries, then* $(\mathcal{C}, \mathcal{R})$ *as shown in Fig 6 is non-malleable w.r.t. non-synchronizing adversaries.*[10]

This improves upon [2, Theorem 6.1] which requires in addition, the existence of constant-round, perfectly-hiding commitments. In the construction, we use the 4-round public-coin witness-indistinguishable proof of knowledge WIPOK based on the Feige-Shamir protocol from [15], which satisfies the following properties:

- The first two messages depend only on the length of the instance and the security parameter and can be pre-computed efficiently without knowing the instance or the witness.
- The third message can be computed efficiently given the instance, the witness, and the randomness used to generate the first message.
- The protocol is *special-sound*—namely, given any two accepting proofs of x, $(\alpha, \beta, \gamma), (\alpha, \beta', \gamma')$ such that $\beta \neq \beta'$. a witness to x can be efficiently recovered.

D.1 An informal argument

We start with an informal argument that the new commitment scheme is stand-alone non-malleable.

Two representative schedulings. We regard the interaction on the right as comprising $c + 1$ slots, where the first slot comprises all messages sent before round 2, and for $i = 1, \ldots, c + 1$, the $i + 1$'th slot comprise all messages sent after round $2i + 1$ and before round $2i + 2$ (i.e., between the second and third messages of the i'th WIPOK). We will examine how $c + 2$ sender's messages sent in the left interaction in rounds $2, 4, \ldots, 2c + 2, 2c + 4$ are distributed across the $c + 2$ slots.

Synchronizing: all slots are non-empty. This means that the executions of tagCom in the left and right interactions are fully synchronized. As such, we may exploit the fact that tagCom is non-malleable w.r.t. synchronizing adversaries.

Non-Synchronizing: some slot is empty. Here, we consider two cases:

- the first slot is empty: This means that the sender's first message $\mathsf{Com}(\tilde{v})$ on the right is sent before the sender's first message $\mathsf{Com}(v)$ on the left; therefore, $\tilde{v}$ must be statistically independent of v.
- the $i + 1$'th slot is empty, for some $i \in \{1, \ldots, c + 1\}$: Following [28], we refer to the first such i as a *safe point*. Here, we will simply rewind and extract $\tilde{v}$ from the i'th WIPOK on the right.

We will refer to the three cases as Type I, Type II and Type III schedulings respectively.

[10] The statement also holds if we replace non-malleable with one-many non-malleable.

Common input : security parameter 1^n and an identity $\mathrm{ID} \in \{0,1\}^t$.

Sender's input : a value $v \in \{0,1\}^{\mathrm{poly}(n)}$.

COMMIT PHASE.

Stage 0: In round 1, $\mathcal{R}$ sends a random $s = f(r)$.[a]

Stage 1: In round 2, $\mathcal{C}$ commits to v using Com. In addition, $\mathcal{C}$ pre-computes and sends the first messages of $c+2$ WIPOK to be used in Stages 3 and 4.

Stage 2: In rounds $3, \ldots, 2c+2$, $\mathcal{C}$ commits to v using tagCom with identity ID.

Stage 3: In rounds $2c+3, 2c+4$, $\mathcal{C}$ sends a WIPOK of the statement.

either the commitments in Stages 1 and 2 are commitments to the same value or $s \in f(\{0,1\}^n)$

using as witness the value v along with the randomnesses it used for Stages 1 and 2. (The first message of the WIPOK was sent in round 2.)

Stage 4: $\mathcal{C}$ sends $c+1$ WIPOK of the statement:

either the Stage 1 commitment is a valid commitment or $s \in f(\{0,1\}^n)$

using as witness the value v along with the randomnesses it used for Stage 1. The $c+1$ WIPOK are scheduled as follows: the first messages are all sent in round 2, and for $i = 1, \ldots, c+1$, the second and third messages of the i'th WIPOK are sent in rounds $2i+1, 2i+2$.

OPEN PHASE.

- $\mathcal{C}$ opens the commitment to v in Stage 1.

[a] Following [28, 26], $\mathcal{R}$ should send a witness hiding proof that $s \in f(\{0,1\}^n)$ after Stage 1.

Fig. 6. Commitment scheme $\mathsf{TagCom} = (\mathcal{C}, \mathcal{R})$.

An alternative opening $\mathcal{R}^*$ for TagCom. As before, we consider an "alternative open phase" for the man-in-the-middle execution corresponding to a "receiver" $\mathcal{R}^*$, where the committed value for the transcript of the right interactions depends on the scheduling.

More formally, we first write $\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}(v), \mathcal{R}_0^*)$ to denote a random variable that is the same as $\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}(v), \mathcal{R})$, except the committed value $\tilde{v}$ is defined as follows:

- Type I scheduling (i.e. all slots are non-empty): set $\tilde{v}$ to the committed value in Stage 2 on the right;
- Type II scheduling (i.e. first slot is empty): set $\tilde{v}$ to be the committed value in Stage 1 on the right if $\tilde{\mathrm{ID}} \neq \mathrm{ID}$ and $\perp$ otherwise;
- Type III scheduling (i.e. $i+1$'th slot is empty for some $i \in \{1, \ldots, c+1\}$): set $\tilde{v}$ to be value obtained by rewinding and extracting from the i'th WIPOK on the right (if there is more than one such i, pick the smallest one), and we refer to i as the safe point. Following [28, Lemma 2], we keep rewinding after the first message of the i'th WIPOK on the right until we get a transcript with the same safe point. The expected number of rewindings is bounded by c.

Next, we write $\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}(v), \mathcal{R}^*)$ to denote a random variable that is the same as $\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}(v), \mathcal{R}_0^*)$, except the committed value $\tilde{v}$ is set to $\perp$ whenever any of the WIPOK in Stage 3 or 4 is rejecting.

D.2 The hybrid argument

STEP 1: SWITCHING TO $\mathcal{R}^*$. We claim that

$$\left\{\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}(v^0), \mathcal{R})\right\} \cong \left\{\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}(v^0), \mathcal{R}^*)\right\} \tag{D.1}$$

STEP 2: SWITCHING TO TRAPDOOR IN STAGE 3. We change the WIPOK in Stage 3 on the left to use the trapdoor witness $r \in f^{-1}(s)$, which we compute via brute force.

STEP 3: SWITCHING TO TRAPDOOR IN STAGE 4. We change the $c+1$ WIPOKs in Stage 4 on the left – one by one – to use the trapdoor witness $r \in f^{-1}(s)$.

STEP 4: SWITCHING TO $\mathsf{Com}(v_1)$ IN STAGE 1. We switch the left commitment in Stage 1 from v^0 to v^1.

STEP 5: SWITCHING TO $\mathsf{tagCom}(v_1)$ IN STAGE 2. We switch the left commitment in Stage 2 from v^0 to v^1.

STEP 6: REVERSING STEPS 3, 2, 1. At this point, we just need to reverse steps 3, 2, and 1.

Switching to $\mathcal{R}^*$ (step 1). Here, we just need to argue that the committed values according to $\mathcal{R}$ and $\mathcal{R}^*$ are the same w.h.p. We establish this via two separate claims:

- First, the probability that in the right interaction, the committed values in Stages 1 and 2 are different and the WIPOK in Stage 3 is accepting, is negligible. Suppose otherwise; then we may as before incorporate the left execution into $\mathcal{A}$ to obtain a stand-alone cheating prover $\mathcal{P}^*$ for the WIPOK in Stage 3 on the right. Then, rewinding and extracting from $\mathcal{P}^*$ must yield a witness for $s \in f(\{0,1\}^n)$, which contradicts the one-wayness of f.
- Next, the probability that we have a Type III scheduling and extraction from the safe point does not yield the committed value $\tilde{v}$ in Stage 1 on the right is negligible. Suppose otherwise; then, extraction must have yielded a witness for $s \in f(\{0,1\}^n)$, which again contradicts the one-wayness of f. (Here, we also use the fact that the probability that rewinding does not yield accepting responses to two separate challenges is $2^{-\Omega(n)}$.)

Switching to trapdoor in Stage 3 (step 2). Informally, we consider two cases based on the schedulings:

- Type I or II scheduling: Stage 3 on the left takes place after Stages 1 and 2 on the right, so it does not affect the distribution of committed values in either Stages 1 or 2.
- Type III scheduling: when we rewind and extract from the i'th WIPOK on the right, there is no message on the left, and therefore this case follows immediately from the indistinguishability of the WIPOK in Stage 3.

More formally, suppose there exists a MIM adversary $\mathcal{A}$ and a distinguisher D that breaks indistinguishability with advantage ϵ. Then, for one of the three schedulings, the adversary achieves advantage at least $\epsilon/3$, and we can construct an adversary that breaks the security of the WIPOK (the reduction itself depends on the type of scheduling).

Switching to trapdoor in Stage 4 (step 3). Again, we consider two cases based on the schedulings:

- Type I scheduling: Here, we use the fact that tagCom is non-malleable w.r.t. the 4-round protocols WIPOK.
- Type II or III scheduling: This can be handled via the same arguments as that for Step 2.

More formally, suppose there exists a MIM adversary $\mathcal{A}$ and a distinguisher D that breaks indistinguishability with advantage ϵ. Then, for one of the three schedulings, the adversary achieves advantage at least $\epsilon/3$, and we can construct an adversary that breaks the security of the WIPOK (for Type II or III scheduling) or an adversary that breaks the non-malleability of tagCom w.r.t. the 4-round protocols WIPOK.

Switching to $\mathsf{Com}(v_1)$ in Stage 1 (step 4). As before, we consider two cases based on the schedulings:

- Type I scheduling: Here, we use the fact that tagCom is non-malleable w.r.t. the 2-round protocol Com.
- Type II or III scheduling: This can be handled via the same arguments as that for Step 2.

Switching to $\mathsf{tagCom}(v_1)$ in Stage 2 (step 5). As before, we consider two cases based on the schedulings:

- Type I scheduling: Here, we use the fact that tagCom is non-malleable w.r.t. synchronizing adversaries.
- Type II or III scheduling: This can be handled via the same arguments as that for Step 2.

E Non-Malleability Amplification for Non-Synchronizing Adversaries

We present our construction for non-malleability amplification with non-synchronizing adversaries in Fig 7.

Lemma 5. *For every $t = t(n) \geq 4$, if tagCom is one-many non-malleable for identities of length $\log t + 1$ and one-many non-malleable w.r.t 4-round protocols, then $(\mathcal{C}, \mathcal{R})$ as shown in Fig 7 is one-many non-malleable for identities of length t.*

E.1 An informal argument

We start with an informal argument that the new commitment scheme is stand-alone non-malleable.

Two representative schedulings. Let α and $\tilde{\alpha}$ denote the Stage 1 commitment in the left and right interaction respectively, and let s denote the Stage 0 challenge in the left interaction. We consider two representative schedulings, depending on whether $\tilde{\alpha}$ is sent before or after s (see Fig 8). The intuition is as follows:

Type I: $\tilde{\alpha}$ is sent before s. The distribution of the messages up to s in the left interaction are independent of the value v, so this means that the value $\tilde{\alpha}$ and thus $\tilde{v}$ must be independent of the value v.

Type II: $\tilde{\alpha}$ is sent after s. It must be the case that the first message in TagCom (which comes from the receiver) in the right interaction is sent after $\mathcal{A}$ sends s. We may then rely on the one-many non-malleability of tagCom with respect to both Com and tagCom to argue that the value committed to in Stage 2 on the right is independent of the values committed to in both Stages 1 and 2 on the left. Moreover, this holds even if $\mathcal{A}$ is given additional auxiliary information about s (specifically, $f^{-1}(s)$).

Common input : security parameter 1^n and an identity $\text{ID} = (\text{ID}_1, \ldots, \text{ID}_t) \in \{0,1\}^t$.

Sender's input : a value $v \in \{0,1\}^{\text{poly}(n)}$.

. . . .

COMMIT PHASE.

Stage 0: $\mathcal{R}$ sends a random $s = f(r)$.[a]

Stage 1: $\mathcal{C}$ commits to v using Com.

Stage 2: $\mathcal{C}$ commits to v using tagCom with tags $(1, \text{ID}_1), \ldots, (t, \text{ID}_t)$. That is, $\mathcal{C}$ executes $\mathsf{tagCom}(\mathsf{id}_i, v), \ldots, \mathsf{tagCom}(\mathsf{id}_i, v)$ in parallel, for $i = 1, 2, \ldots, t$ and where $\mathsf{id}_i = (i, \text{ID}_i)$.

Stage 3: $\mathcal{C}$ proves a WIPOK of the statement:

either all $t+1$ commitments in Stages 1 and 2 are commitments to the same value
or $s \in f(\{0,1\}^n)$

using as witness the value v along with the randomnesses it uses for the commitments in Stages 1 and 2.

. . . .

OPEN PHASE.

- $\mathcal{C}$ opens the commitment to v in Stage 1.[b]

[a] Following [26], $\mathcal{R}$ should send a witness hiding proof that $s \in f(\{0,1\}^n)$ after Stage 1.

[b] It is important in our proof of security that $\mathcal{C}$ does not open to the commitments in both Stages 1 and 2.

Fig. 7. Commitment scheme $\mathsf{TagCom} = (\mathcal{C}, \mathcal{R})$.

Remark 1. It may seem apriori that we can eliminate Stage 1, and consider the same scheduling, with α and $\tilde{\alpha}$ being the respective sender's first messages in the commitment scheme tagCom. This does not work because $\mathcal{A}$ may initiate Stage 0 on the left immediately after receiving the receiver's first message for tagCom on the right (since the receiver goes first in tagCom) and still sends $\tilde{\alpha}$ after receiving α. This falls into a Type II scheduling, but we cannot exploit the many-many non-malleability of tagCom because tagCom in the right interaction started before the cut-off point.

An alternative opening $\mathcal{R}^*$ for TagCom. By our construction of $\mathsf{TagCom} = (\mathcal{C}, \mathcal{R})$, the committed value in a transcript is determined by the commitment in Stage 1 (assuming for the rest of this informal discussion that the entire commit phase transcript is accepting). In the preceding informal argument, we reason about the committed value in Stage 1 on the right for a Type I scheduling, and the committed value in Stage 2 on the right for a Type II scheduling. Towards formalizing this argument, it is helpful to consider an "alternative open phase" for the man-in-the-middle execution corresponding to a "receiver" $\mathcal{R}^*$, where the committed value for the transcript of the right interactions depends on whether we have a Type I or Type II scheduling.

More formally, we write $\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}(v), \mathcal{R}^*)$ to denote a random variable that is the same as $\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}(v), \mathcal{R})$, except the m-tuple of values $(\tilde{v}_1, \ldots, \tilde{v}_m)$ is defined as follows: for each $j = 1, 2, \ldots, m$:

- if $\tilde{\text{ID}}^j = \text{ID}$, then we set $\tilde{v}_j$ to $\perp$ as before.

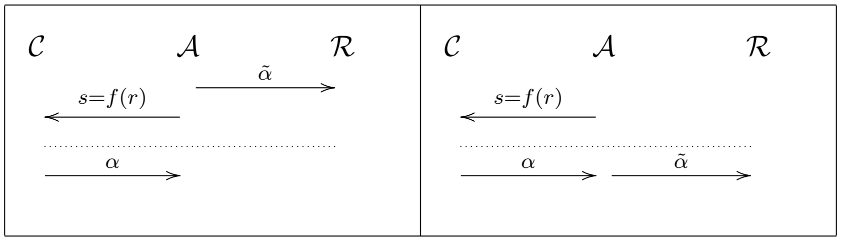

Fig. 8. Two representative schedulings

- if $\tilde{\text{ID}}^j \neq \text{ID}$ and interaction j is a Type I scheduling, then we set $\tilde{v}_j$ to be committed value in Stage 1 if entire commit phase for that interaction is accepting, and $\perp$ otherwise.
- if $\tilde{\text{ID}}^j \neq \text{ID}$ and interaction j is a Type II scheduling, let $i \in [t]$ be the first index such that $\tilde{\text{ID}}_i \neq \text{ID}_i$ and therefore, $\tilde{\mathsf{id}}_i \notin \{\mathsf{id}_1, \ldots, \mathsf{id}_t\}$. Then we set $\tilde{v}_j$ to be committed value in Stage 2 if entire commit phase for that interaction is accepting, and $\perp$ otherwise.

In the first two cases, the definition of $\tilde{v}_j$ is exactly the same as that $\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}(v), \mathcal{R})$. In the third case, soundness of the WIPOK and one-wayness of f essentially guarantees the commited values in both Stages 1 and 2 are the same if the entire commit phase is accepting.

E.2 The hybrid argument

STEP 1: SWITCHING TO $\mathcal{R}^*$. We claim that

$$\left\{\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}(v^0), \mathcal{R})\right\} \cong \left\{\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}(v^0), \mathcal{R}^*)\right\} \tag{E.1}$$

STEP 2: SWITCHING TO $\mathcal{C}^*_{0,0}$. We change the WIPOK on the left to use the trapdoor witness r, i.e. we replace $\mathcal{C}(v_0)$ in the left execution with $\mathcal{C}^*_{0,0}$ where $\mathcal{C}^*_{b_0,b_1}$ for any $b_0, b_1 \in \{0,1\}$ is the following (computationally unbounded) sender that on input (v^0, v^1):

- (Stage 0) After the receiver sends s, extract $r \in f^{-1}(s)$ via brute force.
- (Stage 1) Commit to v^{b_0} using Com.
- (Stage 2) Commit to v^{b_1} using tagCom with tags $(1, \text{ID}_1), \ldots, (t, \text{ID}_t)$.
- (Stage 3) Provide a WIPOK of the statement:

  either all $t+1$ commitments in Stages 1 and 2 are commitments to the same value
  or $s \in f(\{0,1\}^n)$

  using as witness r.

Here, we exploit one-many non-malleability w.r.t. the WIPOK to argue that

$$\left\{\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}(v^0), \mathcal{R}^*)\right\} \cong \left\{\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}^*_{0,0}(v^0, v^1), \mathcal{R}^*)\right\} \tag{E.2}$$

STEP 3: SWITCHING TO $\mathcal{C}^*_{0,1}$. We switch the left commitment in Stage 2 to v^1 and exploit many-many non-malleability of tagCom to argue that

$$\left\{\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}^*_{0,0}(v^0, v^1), \mathcal{R}^*)\right\} \cong \left\{\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}^*_{0,1}(v^0, v^1), \mathcal{R}^*)\right\} \tag{E.3}$$

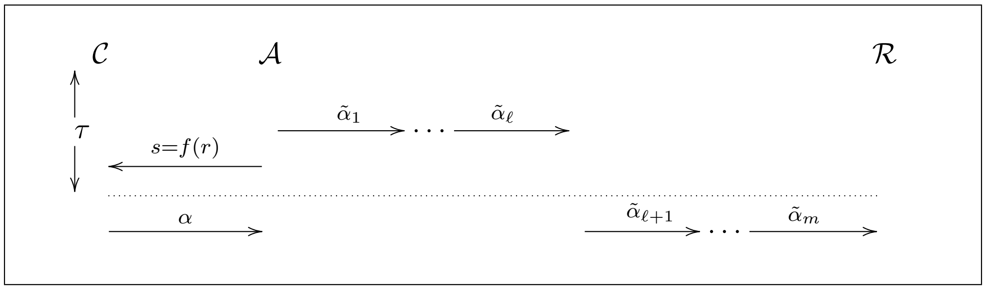

Fig. 9. cut-off point

STEP 4: SWITCHING TO $\mathcal{C}^*_{1,1}$. We switch the left commitment in Stage 1 to v^1 and exploit one-many non-malleability w.r.t. Com to argue that

$$\left\{\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}^*_{0,1}(v^0,v^1),\mathcal{R}^*)\right\} \cong \left\{\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}^*_{1,1}(v^0,v^1),\mathcal{R}^*)\right\} \tag{E.4}$$

STEP 5: SWITCHING TO $\mathcal{C}(v_1)$. This is analogous to Step 2.

$$\left\{\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}^*_{1,1}(v^0,v^1),\mathcal{R}^*)\right\} \cong \left\{\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}(v^1),\mathcal{R}^*)\right\}$$

STEP 6: SWITCHING BACK TO $\mathcal{R}$. This is analogous to Step 1.

$$\left\{\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}(v^1),\mathcal{R}^*)\right\} \cong \left\{\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}(v^1),\mathcal{R})\right\}$$

Switching to $\mathcal{R}^*$ (step 1). Here, we just need to argue that reasoning about the committed values in Stages 1 or 2 yields the same values for the right sessions $\ell+1,\ldots,m$. This follows readily from an argument used in [28, 26]. Suppose otherwise, that is, with non-negligible probability, there exists a right execution, for which the committed value in Stages 1 is different from that for one of the committed values in Stage 2. Now, we may incorporate the left execution into $\mathcal{A}$ (by honestly committing to v^0) to obtain a stand-alone cheating prover $\mathcal{P}^*$ for the WIPOK in that particular right execution. Then, rewinding and extracting $\mathcal{P}^*$ must yield a witness for $s \in f(\{0,1\}^n)$, which contradicts one-wayness of f. We note that this is the only step of the hybrid argument (apart from the analogous Step 6) that requires rewinding or extraction.

Exploiting many-many non-malleability of tagCom. We skip ahead to Step 3 (switching to $\mathcal{C}^*_{1,1}$), which relies on the many-many non-malleability of tagCom. Steps 2 and 4 may be handled with an entirely analogous (and also simpler) reduction.

We begin with the observation that the only difference between the distributions $\left\{\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}^*_{0,0}(v^0,v^1),\mathcal{R}^*)\right\}$ and $\left\{\mathsf{mim}_{\mathcal{A}(z)}(\mathcal{C}^*_{0,1}(v^0,v^1),\mathcal{R}^*)\right\}$ lies in the Stage 2 commitment on the left. In the first distribution, the Stage 2 commitment is a commitment to v^0 using tagCom, and in the second distribution, it is a commitment to v^1 using tagCom. Following [28, 26], we consider a "cut-off point" in the analysis.

Let $\Phi(\mathcal{A},z)$ denote the distribution of all joint views τ of $\mathcal{A}$ and the receivers on the right up to the point immediately after $\mathcal{A}$ sends the message s in Stage 0 in the left interaction (see Fig 9); we also incorporate z into τ. We assume here that the first ℓ interactions on the right have a type I scheduling, and the remaining $m-\ell$ interactions have a type II scheduling. Let the function $\Xi : \{0,1\}^* \to \{0,1\}^*$ be such that $\Xi(\tau) =$

$z||\tau||r||\tilde{v}_1||\cdots||\tilde{v}_\ell$ where $r \in f^{-1}(s)$ (corresponding to the s embedded in τ) and $\tilde{v}_1, \ldots, \tilde{v}_\ell$ are the values committed to by $\mathcal{A}(z)$ in τ in Stages 1 of the first ℓ interactions with type I scheduling.

Lemma 6 (reduction to tagCom**).** *For all ppt $\mathcal{A}$, there exists a ppt $\mathcal{B}$ and D such that for all z, v^0, v^1 and all $b \in \{0,1\}$:*

$$\left\{\mathsf{mim}^{\mathsf{TagCom}}_{\mathcal{A}(z)}(\mathcal{C}^*_{0,b}(v^0, v^1), \mathcal{R}^*)\right\} \cong \left\{D(\mathsf{mim}^{\mathsf{tagCom}}_{\mathcal{B}(z^*)}(\mathcal{C}(v^b, \ldots, v^b), \mathcal{R})) : z^* \leftarrow \Xi(\Phi(\mathcal{A}, z))\right\}$$

are statistically indistinguishable. Note that in the second distribution, there are t left interactions, all committing to v^b using tagCom. [11]

Proof. The high-level idea is to construct a machine $\mathcal{B}$ that on input z^* runs internally a copy of $\mathcal{A}$ and simulates the view of $\mathcal{A}$ in the experiment $\mathsf{mim}^{\mathsf{TagCom}}_{\mathcal{A}(z)}(\mathcal{C}^*_{0,b}(v^0, v^1), \mathcal{R}^*)$ while participating in $\mathsf{mim}^{\mathsf{tagCom}}_{\mathcal{B}(z^*)}(\mathcal{C}(v^b, \ldots, v^b), \mathcal{R})$. The machine D will essentially "post process" the committed values in the second distribution according to $\mathcal{R}^*$.

We first describe how to simulate the left interaction with $\mathcal{C}^*_{0,b}(v^0, v^1)$ in the view of $\mathcal{A}$:

Stage 0. Stage 0 is embedded in τ.

Stage 1. $\mathcal{B}$ commits to v^0 using TagCom.

Stage 2. $\mathcal{B}$ chooses identities $(1, \text{ID}_1), \ldots, (t, \text{ID}_t)$ for the t left interactions (scheduled in parallel), and forwards the messages from the external $\mathcal{C}(v^b, \ldots, v^b)$ to $\mathcal{A}$.

Stage 3. $\mathcal{B}$ simulates the WIPOK here by using the witness r which is part of its auxiliary input z^*.

Next, we describe how $\mathcal{B}$ simulates the m executions of TagCom on the right, with identities $\tilde{\text{ID}}^1, \ldots, \tilde{\text{ID}}^m$. For each of these interactions, we consider two cases depending on whether the scheduling is Type I or II. Again, we assume that the first ℓ have Type I scheduling and the last $m - \ell$ have Type II scheduling. For each $j = 1, \ldots, m$, assuming $\tilde{\text{ID}}^j \neq \text{ID}$,

Type I (i.e. $j \leq \ell$): $\mathcal{B}$ will internally simulate the receiver for this execution, using the state of the corresponding receiver embedded in z^*. D can then figure out the committed value for this interaction of TagCom by looking at the value $\tilde{v}_j$ in its auxiliary input z^*, setting it to $\perp$ if the overall commit phase transcript is not accepting.

Type II (i.e. $j > \ell$): $\mathcal{B}$ will internally simulate Stages 0, 1 and 3. For Stage 1, $\mathcal{B}$ will rely on the t external interactions with $\mathcal{R}$ in tagCom, with identities $(1, \tilde{\text{ID}}^j_1), \ldots, (t, \tilde{\text{ID}}^j_t)$. D will compute $\tilde{v}_j$ by looking at the committed value on the right corresponding to the $(i, \tilde{\text{ID}}^j_i)$ where i is the first index for which $\text{ID}_i \neq \tilde{\text{ID}}^j_i$. □

E.3 Black-Box Non-Malleability Amplification.

Lemma 7. *For every $t = t(n) \geq 4$, if* tagCom *is one-many non-malleable w.r.t extraction for identities of length $\log t + 1$ and one-many non-malleable w.r.t extraction for 4-round protocols, then $(\mathcal{C}, \mathcal{R})$ as shown in Fig 10 is one-many non-malleable w.r.t. extraction for identities of length t.*

As before, we first present an alternative open phase for the commitment scheme TagCom.

[11] In the first distribution, we are referring to the commitment scheme TagCom and in the second, we are referring to tagCom.

Common input : security parameter 1^n and an identity $\text{ID} = (\text{ID}_1, \ldots, \text{ID}_t) \in \{0,1\}^t$.
Sender's input : a value $v \in \{0,1\}^{\text{poly}(n)}$.

COMMIT PHASE.
Stage 0: $\mathcal{R}$ commits to a random subset $\Gamma \subset [10n]$ of size n.
Stage 1: $\mathcal{C}$ computes shares **s** of v using a n-out-of-$10n$ secret-sharing scheme and commits to the shares using ExtCom.

- $\mathcal{C}$ picks a random degree n polynomial p over $\text{GF}(2^{|v|})$ whose constant term is v, and computes $\mathbf{s} = (s_1, \ldots, s_{10n}) = (p(1), \ldots, p(10n))$. (Note that **s** is a Reed-Solomon encoding of v.)
- $\mathcal{C}$ executes $\mathsf{ExtCom}(s_1), \cdots, \mathsf{ExtCom}(s_{10n})$ in parallel.

Stage 2: $\mathcal{C}$ commits to **s** using tagCom with tags $(1, \text{ID}_1), \ldots, (t, \text{ID}_t)$.

- $\mathcal{C}$ executes $\mathsf{tagCom}(\mathsf{id}_i, s_1), \ldots, \mathsf{tagCom}(\mathsf{id}_i, s_{10n})$ in parallel, for $i = 1, 2, \ldots, t$ and where $\mathsf{id}_i = (i, \text{ID}_i)$.

Stage 3: $\mathcal{C}$ proves consistency of the commitments by opening to the shares indexed by Γ.

- $\mathcal{R}$ opens the commitment to Γ.
- $\mathcal{C}$ opens all $t+1$ commitments to s_j in Stages 1 and 2 for all $j \in \Gamma$.
- $\mathcal{R}$ checks that all $t+1$ commitments to s_j are consistent for all $j \in \Gamma$.

OPEN PHASE.

- $\mathcal{C}$ sends v and opens the commitment to **s** in Stage 1.
- $\mathcal{R}$ computes the codeword **w** that is 0.9-close to **s**.
- $\mathcal{R}$ checks that **w** is a codeword corresponding to v and that **w** and **s** agree on all positions in Γ.

Fig. 10.

An alternative opening $\mathcal{R}^*$ for TagCom. As before, we consider two representative schedulings, except the cut-off is based on the message $\mathsf{Com}(\Gamma)$ instead of s. Also, we write $\mathsf{extmim}_{\mathcal{A}(z)}(\mathcal{C}(v), \mathcal{R}^*)$ to denote a random variable that is the same as $\mathsf{extmim}_{\mathcal{A}(z)}(\mathcal{C}(v), \mathcal{R})$, except the m-tuple of values $(\tilde{v}_1, \ldots, \tilde{v}_m)$ is defined as follows: for each $j = 1, 2, \ldots, m$:

- if $\tilde{\text{ID}}^j = \text{ID}$, then we set $\tilde{v}_j$ to $\perp$ as before.
- if $\tilde{\text{ID}}^j \neq \text{ID}$ and interaction j is a Type I scheduling, then we set $\tilde{v}_j$ according to the committed value in Stage 1 as in $\mathcal{R}$ and whether the entire commit phase for that interaction is accepting.
- if $\tilde{\text{ID}}^j \neq \text{ID}$ and interaction j is a Type II scheduling, let $i \in [t]$ be the first index such that $\tilde{\text{ID}}_i \neq \text{ID}_i$ and therefore, $\tilde{\mathsf{id}}_i \notin \{\mathsf{id}_1, \ldots, \mathsf{id}_t\}$. Then we determine $\tilde{v}_j$ as follows:
  - We define the vector $\tilde{\mathbf{s}}' = (\tilde{s}'_1, \ldots, \tilde{s}'_{10n})$ to be the extracted values in the i'th execution of tagCom in Stage 2 on the right.
  - Compute a codeword $\tilde{\mathbf{w}}$ that is 0.8-close to $\tilde{\mathbf{s}}'$.
  - If $\tilde{\mathbf{w}}$ and $\tilde{\mathbf{s}}'$ agrees on all positions on Γ, output the value $\tilde{v}_j$ corresponding to $\tilde{\mathbf{w}}$, else output $\perp$.

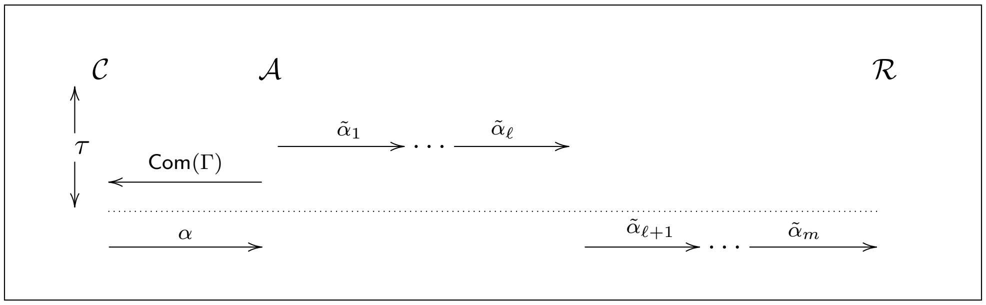

Fig. 11. cut-off point

The hybrid argument.

STEP 1: SWITCHING TO $\mathcal{R}^*$. We claim that

$$\left\{\mathsf{extmim}_{\mathcal{A}(z)}(\mathcal{C}(v^0), \mathcal{R})\right\} \cong \left\{\mathsf{extmim}_{\mathcal{A}(z)}(\mathcal{C}(v^0), \mathcal{R}^*)\right\}$$

STEP 2: SWITCHING TO $\mathcal{C}^*_{0,1}$. We switch the left commitment in Stage 2 to v^1, i.e., we replace $\mathcal{C}$ in the left execution with $\mathcal{C}^*_{0,1}$ where $\mathcal{C}^*_{b_0,b_1}$ for any $b_0, b_1 \in \{0,1\}$ is the following (computationally unbounded) sender that on input (v^0, v^1):

- (Stage 0) After the receiver commits to Γ in Stage 0, extract the value Γ via brute-force, and compute random shares $\mathbf{s}^0$ and $\mathbf{s}^1$ of v^{b_0} and v^{b_1} that agree on the positions in Γ. Specifically, we first pick n random values for the positions in Γ, then do polynomial interpolation with the point $(0, v^0)$ to compute $\mathbf{s}^0$ and with the point $(0, v^1)$ to compute $\mathbf{s}^1$.
- (Stage 1) Commit to $\mathbf{s}^0$ using ExtCom.
- (Stage 2) Commit to $\mathbf{s}^1$ using tagCom with tags $(1, \textsc{id}_1), \ldots, (t, \textsc{id}_t)$.
- (Stage 3) Open the commitments to s_j for all $j \in \Gamma$.

Note that $\mathcal{C}(v^0)$ and $\mathcal{C}^*_{0,0}(v^0, v^1)$ are identically distributed. We claim that

$$\left\{\mathsf{extmim}_{\mathcal{A}(z)}(\mathcal{C}^*_{0,0}(v^0, v^1), \mathcal{R}^*)\right\} \cong \left\{\mathsf{extmim}_{\mathcal{A}(z)}(\mathcal{C}^*_{0,1}(v^0, v^1), \mathcal{R}^*)\right\}$$

STEP 3: SWITCHING BACK TO $\mathcal{C}$. We switch the left commitment in Stage 1 to v^1, i.e.,

$$\left\{\mathsf{extmim}_{\mathcal{A}(z)}(\mathcal{C}^*_{0,1}(v^0, v^1), \mathcal{R}^*)\right\} \cong \left\{\mathsf{extmim}_{\mathcal{A}(z)}(\mathcal{C}^*_{1,1}(v^0, v^1), \mathcal{R}^*)\right\}$$

STEP 4: SWITCHING BACK TO $\mathcal{R}$.

$$\left\{\mathsf{extmim}_{\mathcal{A}(z)}(\mathcal{C}(v^1), \mathcal{R}^*)\right\} \cong \left\{\mathsf{extmim}_{\mathcal{A}(z)}(\mathcal{C}(v^1), \mathcal{R})\right\}$$

We omit the details of the analysis since they are almost exactly identical to that in Sections A.3 and E.2.