Constant-Round Black-Box Construction of Composable Multi-Party Computation Protocol ⋆

Susumu Kiyoshima[1], Yoshifumi Manabe[2], and Tatsuaki Okamoto[1]

[1] NTT Secure Platform Laboratories, Japan.
{kiyoshima.susumu, okamoto.tatsuaki}@lab.ntt.co.jp
[2] Kogakuin University, Japan.
manabe@cc.kogakuin.ac.jp

Abstract. We present the first general MPC protocol that satisfies the following: (1) the construction is black-box, (2) the protocol is universally composable in the plain model, and (3) the number of rounds is constant. The security of our protocol is proven in angel-based UC security under the assumption of the existence of one-way functions that are secure against sub-exponential-time adversaries and constant-round semi-honest oblivious transfer protocols that are secure against quasi-polynomial-time adversaries. We obtain the MPC protocol by constructing a constant-round CCA-secure commitment scheme in a black-box way under the assumption of the existence of one-way functions that are secure against sub-exponential-time adversaries. To justify the use of such a sub-exponential hardness assumption in obtaining our constant-round CCA-secure commitment scheme, we show that if black-box reductions are used, there does not exist any constant-round CCA-secure commitment scheme under any falsifiable polynomial-time hardness assumptions.

1 Introduction

Protocols for *secure multi-party computation* (MPC) enable mutually distrustful parties to compute a functionality without compromising the correctness of the outputs and the privacy of the inputs. In the seminal work of Goldreich et al. [GMW87], a general MPC protocol was constructed in a model with malicious adversaries and a dishonest majority.[3] (By "a general MPC protocol," we mean a protocol that can be used to securely compute any functionality.)

Black-box constructions. A construction of a protocol is *black-box* if it uses the underlying cryptographic primitives only in a black-box way (that is, only through their input/output interfaces). In contrast, if a construction uses the codes of the underlying primitives, it is *non-black-box*.

Obtaining black-box constructions is an important step toward obtaining practical MPC protocols. This is because black-box constructions are typically more efficient than non-black-box ones. (Typical non-black-box constructions, such as that of [GMW87], use the codes of the primitives to compute NP reductions in general zero-knowledge proofs. Thus, they should be viewed as feasibility results.) Black-box constructions are also theoretically interesting, since

⋆ This is the full version of a paper that appears in TCC 2014 [KMO14].
[3] In the following, we consider only such a model.

understanding whether non-black-box use of primitives is necessary for a cryptographic task is of great theoretical interest.

Recently, a series of works showed black-box constructions of general MPC protocols. Ishai et al. [IKLP06] showed the first construction of a general MPC protocol that uses the underlying low-level primitives in a black-box way. Combined with the subsequent work of Haitner [Hai08], their work showed a black-box construction of a general MPC protocol based on a semi-honest oblivious transfer protocol [HIK$^+$11]. Subsequently, Wee [Wee10] showed an $O(\log^* n)$-round protocol under polynomial-time hardness assumptions and a constant-round protocol under sub-exponential-time hardness assumptions, and Goyal [Goy11] showed a constant-round protocol under polynomial-time hardness assumptions.

The security of these black-box protocols is considered in the *stand-alone setting*. That is, the protocols of [IKLP06, Wee10, Goy11] are secure in the setting where only a single instance of the protocol is executed at a time.

Composable security. The *concurrent setting*, in which many instances of protocols are executed concurrently in an arbitrary schedule, is a more general and realistic setting than the stand-alone one. In the concurrent setting, an adversary can perform a coordinated attack in which he chooses his messages in an instance based on the executions of the other instances.

As a strong and realistic security notion in the concurrent setting, Canetti [Can01] proposed *universally composable (UC) security*. The main advantage of UC security is *composability*, which guarantees that when we compose many UC-secure protocols, we can prove the security of the resultant protocol using the security of its components. Thus, UC security enables us to construct protocols in a modular way. Composability also guarantees that a protocol remains secure even when it is concurrently executed with any other protocols in any schedule. Canetti et al. [CLOS02] constructed a UC-secure general MPC protocol in the *common reference string (CRS) model* (i.e., in a model in which all parties are given a common public string that is chosen by a trusted third party).

UC security, however, turned out to be too strong to achieve in the *plain model* (i.e., in a model without any trusted setup except for authenticated communication channels). That is, we cannot construct UC-secure general MPC protocols in the plain model [CF01, CKL03].

To achieve composable security in the plain model, Prabhakaran and Sahai [PS04] proposed a variant of UC security called *angel-based UC security*. Roughly speaking, angel-based UC security is the same as UC security except that the adversary and the simulator have access to an additional entity—the *angel*—that allows some judicious use of super-polynomial-time resources. It was proven that, like UC security, angel-based UC security guarantees composability. Furthermore, as argued in [PS04], angel-based UC security guarantees meaningful security in many cases. (For example, angel-based UC security implies *super-polynomial-time simulation (SPS) security* [Pas03, BS05, GGJS12, PLV12]. In SPS security, we allow the simulator to run in super-polynomial time. Thus, SPS security guarantees that whatever an adversary can do in the real world can also be done in the ideal world in super-polynomial time.) Then, Prabhakaran and Sahai [PS04] presented a general MPC protocol that satisfies this security notion in the plain model, based on new (unstudied and non-standard)

assumptions. Subsequently, Malkin et al. [MMY06] constructed another general MPC protocol that satisfies this security notion in the plain model based on new number-theoretic assumption. In [BS05], Barak and Sahai remarked that their protocol (which is SPS secure under subexponential-time hardness assumptions) can be shown to be secure in angel-based UC security.

Recently, Canetti et al. constructed a polynomial-round general MPC protocol in angel-based UC security based on a standard assumption (the existence of enhanced trapdoor permutations). Subsequently, Lin [Lin11] and Goyal et al. [GLP+12] reduced the round complexity to $\widetilde{O}(\log n)$ under the same assumption. They also proposed constant-round protocols, where the security is based on a super-polynomial-time hardness assumption (the existence of enhanced trapdoor permutations that are secure against quasi-polynomial-time adversaries). These constructions, however, use the underlying primitives in a non-black-box way.

Black-box constructions of composable protocols. Lin and Pass [LP12] showed the first black-box construction of a general MPC protocol that guarantees composable security in the plain model. The security of their protocol is proven under angel-based UC security, and based on the minimum assumption of the existence of semi-honest oblivious transfer (OT) protocols.

The round complexity of their protocol is $O(n^\epsilon)$, where $\epsilon > 0$ is an arbitrary constant. In contrast, for non-black-box constructions of composable protocols, we have constant-round protocols in the plain model (under non-standard assumptions or super-polynomial-time hardness assumptions) [PS04, MMY06, Lin11, GLP+12]. Thus, a natural question is the following.

> *Does there exist a constant-round black-box construction of a general MPC protocol that guarantees composability in the plain model (possibly under super-polynomial-time hardness assumptions)?*

1.1 Our Result

In this paper, we answer the above question affirmatively.

Theorem (Informal). *Assume the existence of one-way functions that are secure against sub-exponential-time adversaries and constant-round semi-honest oblivious transfer protocols that are secure against quasi-polynomial-time adversaries. Then, there exists a constant-round black-box construction of a general MPC protocol that satisfies angel-based UC security in the plain model.*

The formal statement of this theorem is given in Section 7.

CCA-secure commitment schemes. We prove the above theorem by constructing a constant-round *CCA-secure commitment scheme* [CLP10, LP12] in a black-box way. Once

we obtain a CCA-secure commitment scheme, we can construct a general MPC protocol in essentially the same way as Lin and Pass do in [LP12].

Roughly speaking, a CCA-secure commitment scheme is a tag-based commitment scheme (i.e., a commitment scheme that takes an n-bit string, or *tag*, as an additional input) such that the committed value of a commitment with tag id remains hidden even if the receiver has access to a super-polynomial-time oracle—the *committed-value oracle*—that returns the committed value of any commitment with tag $\mathsf{id}' \neq \mathsf{id}$. Lin and Pass [LP12] showed an $O(n^\epsilon)$-round black-box construction of a CCA-secure commitment scheme for arbitrary $\epsilon > 0$ by assuming the minimum assumption of the existence of one-way functions.

Our main technical result is the following.

Theorem (Informal). *Assume the existence of one-way functions that are secure against sub-exponential-time adversaries. Then, there exists a constant-round black-box construction of a CCA-secure commitment scheme.*

The formal statement of this theorem is given in Section 7.

To obtain our CCA-secure commitment scheme, we use the idea of *non-malleability amplification* that was used in previous works on concurrent non-malleable (NM) commitment schemes [LP09, PW10]. That is, we construct a CCA commitment scheme in the following steps.

Step 1. We say that a commitment scheme is *one-one CCA secure* if it is CCA secure with respect to restricted classes of adversaries that receive only a single answer from the oracle. Then, we construct a constant-round one-one CCA-secure commitment for tags of length $O(\log\log\log n)$.

Step 2. We construct a transformation from the commitment scheme constructed in Step 1 to a CCA-secure commitment for tags of length $O(n)$ with a constant additive increase in round complexity. Toward this end, we construct the following two transformations:

- A transformation from any one-one CCA-secure commitment scheme for tags of length $t(n)$ to a CCA-secure commitment scheme for tags of length $t(n)$ with a constant additive increase in round complexity
- A transformation from any CCA-secure commitment scheme for tags of length $t(n)$ to a one-one CCA-secure commitment scheme for tags of length $2^{t(n)-1}$ with no increase in round complexity

(The latter transformation is essentially the same as the "DDN $\log n$ trick" [DDN00, LPV08].) By repeatedly composing these two transformations, we obtain the desired transformation.

On the use of super-polynomial-time hardness assumption. Although the round complexity of our CCA-secure commitment scheme is constant, it relies on a super-polynomial-time hardness assumption. (Recall that the $O(n^\epsilon)$-round CCA-secure commitment scheme of [LP12] relies on a polynomial-time hardness assumption.)

We show that the use of such a strong assumption is *inevitable*, as long as the security of a constant-round CCA-secure commitment scheme is proven under *falsifiable assumptions*

[Nao03, GW11] by using a *black-box reduction*. Roughly speaking, a falsifiable assumption is an assumption that is modeled as an interactive game between a challenger and an adversary such that the challenger can decide whether the adversary won the game in polynomial time. Then, we say that *the CCA security of a commitment scheme $\langle C, R\rangle$ is proven under a falsifiable assumption by using a black-box reduction* if the CCA security of $\langle C, R\rangle$ is proven by constructing a PPT Turing machine $\mathcal{R}$ such that for any adversary $\mathcal{A}$ that breaks the CCA security of $\langle C, R\rangle$, $\mathcal{R}$ can break the assumption by using $\mathcal{A}$ only in a black-box way. Then, we show the following theorem.

Theorem (Informal). *Let $\langle C, R\rangle$ be any constant-round commitment scheme. Then, the CCA security of $\langle C, R\rangle$ cannot be proven by using black-box reductions under any falsifiable polynomial-time hardness assumption.*

Roughly speaking, we obtain this theorem by using techniques of the negative result on concurrent zero-knowledge protocols [CKPR02].

Since all standard cryptographic assumptions are falsifiable, this theorem says that if we want to construct a constant-round CCA-secure commitment scheme based on standard assumptions, we must use either super-polynomial-time hardness assumptions (as this paper does) or non-black-box reductions.[4]

We note that this negative result holds *even for non-black-box constructions*. That is, we cannot construct constant-round CCA-secure commitment schemes even when we use primitives in a non-black-box way, as long as we use black-box reductions and polynomial-time hardness assumptions.

2 Overview of the Protocols

In this section, we give overviews of our main technical results: a one-one CCA-secure commitment scheme for short tags and a transformation from one-one CCA security to CCA security.

2.1 One-One CCA-Security for Short Tags

We obtain our one-one CCA-secure commitment scheme by observing that the non-black-box construction of a NM commitment scheme of [PW10] is one-one CCA secure and converting it into a black-box one.

First, we recall the scheme of [PW10].[5] The starting point of the scheme is "two-slot message length" technique [Pas04]. The basic idea of the technique is to let the receiver sequentially send two challenges—one "long" and one "short"—where the length of the challenges are determined by the tag of the commitment. The protocol is designed so that the response to a shorter challenge does not help a man-in-the-middle adversary to provide a

[4] We note that, although very recently Goyal [Goy13] showed how to use non-black-box techniques in the fully concurrent setting, Goyal's technique requires polynomially many rounds.

[5] In the following, some of the text is taken from [PW10].

response to a longer challenge. A key conceptual insight of [PW10] is to rely on the complexity leveraging technique [CGGM00] to construct these challenges: For one-way functions with sub-exponential hardness, an oracle for inverting challenges of length $n^{o(1)}$ (the "short" challenge) does not help invert random challenges of length n (the "long" challenge), since we can simulate such an oracle by brute force in time $2^{n^{o(1)}}$.

More precisely, the scheme of [PW10] is as follows. Let $d = O(\log \log n)$ be the number of tags, and let $n^{\omega(1)} = T_0(n) \ll T_1(n) \ll \cdots \ll T_{d+2}(n)$ be a hierarchy of running times. Then, to commit to $v \in \{0,1\}^n$ with tag $\mathsf{id} \in \{0, 1, \ldots, d-1\}$, the committer C does the following with the receiver R.

1. C commits to v by using a statistically binding commitment Com that is hiding against $T_{d+1}(n)$-time adversaries but is completely broken in time $T_{d+2}(n)$.
2. (Slot 1) C proves knowledge of v by using a zero-knowledge argument of knowledge that is computationally sound against $T_{\mathsf{id}+1}(n)$-time adversaries and can be simulated in straight line in time $o(T_{\mathsf{id}+2}(n))$, where the simulated view is indistinguishable from the real one in time $T_{d+2}(n)$.
3. (Slot 2) C proves knowledge of v by using a zero-knowledge argument of knowledge that is computationally sound against $T_{d-\mathsf{id}}(n)$-time adversaries and can be simulated in straight line in time $o(T_{d-\mathsf{id}+1}(n))$, where the simulated view is indistinguishable from the real one in time $T_{d+2}(n)$.

We can show that the scheme of [PW10] is one-one CCA secure as follows (by using essentially the same proof as the proof of its non-malleability). Recall that a commitment scheme is one-one CCA secure if it is hiding against adversaries that give a single query to the committed-value oracle $\mathcal{O}$. Let id be the tag used in the *left session* (a commitment from the committer to the adversary $\mathcal{A}$) and $\widetilde{\mathsf{id}}$ be the tag used in the *right session* (a commitment from $\mathcal{A}$ to $\mathcal{O}$). Then, let us consider a hybrid experiment in which the proofs in the second and third steps are replaced with the straight-line simulations in the left session. Since the running time of $\mathcal{O}$ is at most $T_{d+2}(n)$, the zero-knowledge property guarantees that the view of $\mathcal{A}$ in the hybrid experiment is indistinguishable from that of $\mathcal{A}$ in the real experiment even when $\mathcal{A}$ interacts with $\mathcal{O}$. Furthermore, in the right session of the hybrid experiment, the soundness of the zero-knowledge argument still holds either in the second step or in the third step. This follows from the following reasons. For simplicity, let us consider a synchronized adversary.[6] Then, since the simulation of the second step takes at most time $o(T_{\mathsf{id}+2}(n))$ and the soundness of the second step holds against $T_{\widetilde{\mathsf{id}}+1}(n)$-time adversaries, the soundness of the second step holds if $\mathsf{id} < \widetilde{\mathsf{id}}$; similarly, the soundness of the third step holds if $\mathsf{id} > \widetilde{\mathsf{id}}$. In the hybrid experiment, therefore, the committed value v can be extracted by using the knowledge extractor either in the second step or in the third step, and thus the committed value oracle $\mathcal{O}$ can be simulated in time $o(\max(T_{\mathsf{id}+2}(n), T_{d-\mathsf{id}+1}(n))) \cdot \mathsf{poly}(n) \ll T_{d+1}(n)$. Then, from the hiding property of Com in the first step, the view of $\mathcal{A}$ in the hybrid experiment is computationally independent of the value v. Thus, one-one CCA security follows.

[6] An synchronized adversary sends the i-th round message to $\mathcal{O}$ immediately after receiving the i-th round messages from the committer, and vise verse.

To convert the scheme of [PW10] into a black-box protocol, we use a black-box trapdoor commitment scheme $\mathsf{TrapCom}$ of [PW09]. We observe that $\mathsf{TrapCom}$ has similar properties to the zero-knowledge argument used in the scheme of [PW10]: $\mathsf{TrapCom}$ is extractable and a $\mathsf{TrapCom}$ commitment can be simulated in straight line in super-polynomial time. Then, we modify the scheme of [PW10] and let the committer commit to v instead of proving the knowledge of v. To ensure the "soundness," that is, to ensure that the committed value of $\mathsf{TrapCom}$ is v, we use the cut-and-choose technique and Shamir's secret sharing scheme in a similar manner to previous works on black-box protocols [CDSMW08, CDSMW09, Wee10, LP12]. That is, we let the committer commit to Shamir's secret sharing $\boldsymbol{s} = (s_1, \ldots, s_{10n})$ of value v in all steps, let the receiver choose a random subset $\Gamma \subset [10n]$ of size n, and let the committer reveal s_j and decommit the corresponding commitments for every $j \in \Gamma$. The resultant scheme uses the underlying primitives only in a black-box way, and can be proven to be one-one CCA secure from a similar argument to the scheme of [PW10]. (We note that the actual scheme is a little more complicated. For details, see Section 4.) We note that Lin and Pass [LP12] also use $\mathsf{TrapCom}$ to convert a non-black-box protocol into a black-box one. Unlike them, who mainly use the fact that $\mathsf{TrapCom}$ is extractable and is secure against selective opening attacks, we also use the fact that $\mathsf{TrapCom}$ commitments are straight-line simulatable.

2.2 CCA Security from One-one CCA Security

We give an overview of the transformation from any one-one CCA-secure commitment scheme to a CCA-secure commitment scheme. Let $n^{\omega(1)} = T_0(n) \ll T_1(n) \ll T_2(n) \ll T_3(n)$ be a hierarchy of running times. Then, we construct a CCA-secure commitment scheme CCACom_0 that is secure against $T_0(n)$-time adversaries from a one-one CCA-secure commitment scheme $\mathsf{CCACom}_3^{1:1}$ that is secure against $T_3(n)$-time adversaries. Let Com_1 be a 2-round statistically binding commitment scheme that is secure against $T_1(n)$-time adversaries but is completely broken in time $o(T_2(n))$, and CECom_2 be a constant-round commitment scheme that is hiding against $T_2(n)$-time adversaries and is concurrently extractable by rewinding the committer $\mathsf{poly}(n^{\log n})$ times [MOSV06, PV08]. Then, to commit to value v, the committer C does the following with the receiver R.

1. R commits to a random subset $\Gamma \subset [10n]$ of size n by using $\mathsf{CCACom}_3^{1:1}$.
2. C computes an $(n+1)$-out-of-$10n$ Shamir's secret sharing $\boldsymbol{s} = (s_1, \ldots, s_{10n})$ of value v and commits to s_j for each $j \in [10n]$ in parallel by using Com_1.
3. C commits to s_j for each $j \in [10n]$ in parallel by using CECom_2.
4. R decommits the commitment of the first step and reveal Γ.
5. For each $j \in \Gamma$, C decommits the Com_1 and CECom_2 commitments whose committed values are s_j.

The committed value of CCACom_0 is determined by the committed values of Com_1. Thus, the running time of $\mathcal{O}$ is at most $o(T_2(n)) \cdot \mathsf{poly}(n) \ll T_2(n)$.

To prove the CCA security of the scheme, we consider a series of hybrid experiments.

In the first hybrid, in the left interaction the committed value Γ of $\mathsf{CCACom}_3^{1:1}$ is extracted by brute force and the committed value of CECom_2 is switched from s_j to 0 for every $j \notin \Gamma$. Note that, during the CECom_2 commitments of the left, the combined running time of $\mathcal{A}$ and $\mathcal{O}$ is at most $T_2(n)$. Thus, from the hiding property of CECom_2, the view of $\mathcal{A}$ in the first hybrid is indistinguishable from that of $\mathcal{A}$ in the honest experiment.

The second hybrid is the same as the first one except for the following: in every right session of which the second step ends after the start of the second step of the left session, the committed values of the CECom commitments are extracted; then, the answer of $\mathcal{O}$ are computed from the extracted values (instead of the committed values of Com_1). We note that, since the second hybrid differs from the first one only in how the answers of $\mathcal{O}$ are computed, to show the indistinguishability it suffices to show that in the first hybrid the committed values of CECom_2 agree with those of Com_1 in "most" indexes in every right session. We first note that if we ignore the messages that $\mathcal{A}$ receives in the left session, we can prove that the committed values of CECom_2 agree with those of Com_1 in most indexes by using the property of the cut-and-choose technique. In the hybrid, however, $\mathcal{A}$ receives messages in the left session, in which Γ is extracted by brute force and the committed values of CECom_2 disagree with those of Com_1 in 90% of indexes. Thus, $\mathcal{A}$ may be able to use the messages in the left to break the hiding property of $\mathsf{CCACom}^{1:1}$ in the right. (Note that, if $\mathcal{A}$ can break the hiding property of $\mathsf{CCACom}^{1:1}$, we cannot use the property of the cut-and-choose technique.) We show that $\mathcal{A}$ cannot break the hiding property of $\mathsf{CCACom}^{1:1}$ even with the messages of the left session. A key is that given Γ, the left session can be simulated in polynomial time. Hence, one-one CCA security of $\mathsf{CCACom}^{1:1}$ guarantees that the messages of the left session are useless for breaking the hiding property of $\mathsf{CCACom}^{1:1}$. Thus, even with messages of the left session, the cut-and-choose guarantees that the committed values of CECom_2 agree with those of Com_1 in most indexes. The view of $\mathcal{A}$ in the second hybrid is therefore indistinguishable from that of $\mathcal{A}$ in the first one.

The third hybrid is the same as the first one except that in the left session, the committed value of Com_1 is switched from s_j to 0 for every $j \notin \Gamma$. Note that during the Com_1 commitments of the left, the combined running time of $\mathcal{A}$ and $\mathcal{O}$ is at most $T_0(n) \cdot \mathsf{poly}(n^{\log n}) \ll T_1(n)$. This is because

- for every right session in which $\mathcal{A}$ completes the second step before the start of the second step of the left session, the answer of $\mathcal{O}$ (i.e., the committed value of CCACom_0) can be computed before the start of Com_1 commitments of the left session, and
- for every right session in which $\mathcal{A}$ completes the second step after the start of the second step of the left session, the answer of $\mathcal{O}$ is computed by extracting the committed values of CECom_2, which requires rewinding $\mathcal{A}$ at most $\mathsf{poly}(n^{\log n})$ times.

Thus, from the hiding property of Com_1, the view of $\mathcal{A}$ in the third hybrid is indistinguishable from that of $\mathcal{A}$ in the second one.

Note that, since $\boldsymbol{s}$ is $(n+1)$-out-of-$10n$ secret sharing, $\mathcal{A}$ receives no information of v in the third hybrid. Thus, the view of $\mathcal{A}$ in the third hybrid is independent of v, and thus the CCA security follows.

3 Preliminaries

In this section, we explain the assumptions and the definitions that we use in this paper.

3.1 Assumptions

For our CCA-secure commitment scheme, we use a one-way function f that is secure against 2^{n^ϵ}-time adversaries, where $\epsilon < 1$ is a positive constant. Without loss of generality, we assume that f can be inverted in time 2^n. Let $T_i(n) \stackrel{\text{def}}{=} 2^{(\log n)^{(2/\epsilon)^{10i+1}}}$ for $i \in \mathbb{N}$. Then, by setting the security parameter of f to $\ell_i(n) = (\log n)^{(2/\epsilon)^{10i+2}}$, we obtain a one-way function f_i that is secure against $T_i(n)$-time adversaries but can be inverted in time less than $T_{i+0.5}(n)$. We note that when $i \leq O(\log \log n)$, we have $\ell_i(n) \leq \mathsf{poly}(n)$.

For our composable MPC protocol, we additionally use semi-honest oblivious transfer protocols that are secure against $2^{\mathsf{poly}(\log n)}$-time adversaries.

3.2 Shamir's Secret Sharing Scheme

In this paper, we use Shamir's $(n+1)$-out-of-$10n$ secret sharing scheme. For any positive real number $x \leq 1$ and any $\boldsymbol{s} = (s_1, \ldots, s_{10n})$ and $\boldsymbol{s'} = (s'_1, \ldots, s'_{10n})$, we say that $\boldsymbol{s}$ and $\boldsymbol{s'}$ are *x-close* if $|\{i \mid s_i = s'_i\}| \geq x \cdot 10n$. We note that Shamir's secret sharing is a codeword of Reed-Solomon code with minimum relative distance 0.9. Thus, for any $x > 0.55$ and any $\boldsymbol{s}$ that is x-close to a valid codeword $\boldsymbol{w}$, we can compute $\boldsymbol{w}$ from $\boldsymbol{s}$.

3.3 Commitment Schemes

Recall that commitment schemes are two-party protocols between the committer C and the receiver R. A transcript of the commit phase is *accepted* if R does not abort in the commit phase. A transcript of the commit phase is *valid* if there exists a valid decommitment of this transcript. We use a 2-round statistically binding commitment scheme Com based on one-way functions [Nao91].

Strong computational binding property. We say that a commitment scheme $\langle C, R \rangle$ satisfies a *strong computational binding property* if for any PPT committer $\mathcal{C}^*$ interacting with the honest receiver R, the probability that $\mathcal{C}^*$ generates a commitment that has more than one committed value is negligible.[7]

[7] The standard computational binding property guarantees only that for any PPT committer $\mathcal{C}^*$, the commitment that $\mathcal{C}^*$ generates cannot be decommitted to more than one value *in polynomial time*. Thus, this commitment may have more than one committed value.

3.4 Extractable Commitments

We recall the definition of *extractable commitments* from [PW09]. Roughly speaking, a commitment scheme is *extractable* if there exists an expected polynomial-time oracle machine (or *extractor*) E such that for any committer $\mathcal{C}^*$, $E^{\mathcal{C}^*}$ extracts the value that $\mathcal{C}^*$ commits to whenever the commitment is valid. We note that when the commitment is invalid, E may output a garbage value. (This is called *over-extraction*.)

Formally, let $\langle C, R\rangle$ be a commitment scheme that satisfies the strong computational binding property. Then, $\langle C, R\rangle$ is extractable if there exists an expected polynomial-time probabilistic oracle machine E such that for any PPT committer $\mathcal{C}^*$, extractor $E^{\mathcal{C}^*}$ outputs a pair (τ, σ) such that

- τ is identically distributed with the view of $\mathcal{C}^*$ of the commit phase in which $\mathcal{C}^*$ interacts with honest receiver R.
- If τ is accepted, then $\sigma \neq \perp$ except with negligible probability.
- If $\sigma \neq \perp$, then it is statistically impossible to decommit τ to any value other than σ.

Note that when τ is accepted but invalid, σ may be an arbitrary value.

There exists a 4-round extractable commitment scheme ExtCom based on one-way functions [PW09] (see Figure 1). The commit phase of ExtCom consists of three stages—`commit`, `challenge`, and `reply`—and given two accepted transcripts that have the same `commit` message but have different `challenge` messages, we can extract the committed value. Thus, we can extract the committed value by rewinding the committer and obtaining two such transcripts. In the following, we use *slot* to denote a pair of the `challenge` and `reply` messages in ExtCom.

As shown in [PW09], ExtCom is in fact *parallel extractable*. Thus, even when a committer commits to many values in parallel, we can extract all committed values.

3.5 Concurrently Extractable Commitments

Roughly speaking, a commitment scheme is *concurrently extractable* if there exists an expected polynomial-time extractor E such that for any committer $\mathcal{C}^*$ that concurrently commits to many values, $E^{\mathcal{C}^*}$ extracts the committed value of each commitment immediately after $\mathcal{C}^*$ generates each commitment.

Formally, an extractable commitment $\langle C, R\rangle$ is concurrently extractable if for every polynomial m, there exists an expected polynomial-time extractor E such that for any PPT committer $\mathcal{C}^*$ that concurrently commits to at most $m(n)$ values by using $\langle C, R\rangle$, the following hold:

- The output τ of $E^{\mathcal{C}^*}$ is identically distributed with the view of $\mathcal{C}^*$ of the commit phase in which $\mathcal{C}^*$ interacts with honest receivers.
- Except with negligible probability, whenever E makes an oracle query Q to $\mathcal{C}^*$ (where query Q is a partial transcript of an interaction between $\mathcal{C}^*$ and R), for each accepted commitment τ_i that is contained in Q, extractor E immediately outputs a value σ_i such that it is statistically impossible to decommit τ_i to any value other than σ_i.

Commit Phase

The committer C and the receiver R receive common inputs 1^n. To commit to $v \in \{0,1\}^n$, the committer C does the following with the receiver R.

`commit` **stage.** For each $i \in [n]$, the committer C chooses a pair of random n-bit strings (a_i^0, a_i^1) such that $a_i^0 \oplus a_i^1 = v$. Then, for each $i \in [n]$ in parallel, C commits to a_i^0 and a_i^1 by using Com. For each $i \in [n]$ and $b \in \{0,1\}$, let c_i^b be the commitment to a_i^b.
`challenge` **stage.** R sends random n-bit string $e = (e_1, \ldots, e_n)$ to C.
`reply` **stage.** For each $i \in [n]$, C decommits $c_i^{e_i}$ to $a_i^{e_i}$.

Decommit Phase

C sends v to R and decommits c_i^b for all $i \in [n]$ and $b \in \{0,1\}$. Then, R checks whether $a_1^0 \oplus a_1^1 = \cdots = a_n^0 \oplus a_n^1 = v$.

Fig. 1. Extractable commitment ExtCom [PW09].

Commit phase. The committer C and the receiver R receive common inputs 1^n and parameter r. To commit to $v \in \{0,1\}^n$, the committer C does the following.
Step 1. C and R execute `commit` stage of ExtCom r times in parallel.
Step $2i$ **(**$i \in [r]$**).** R sends the `challenge` message of ExtCom for the i-th session.
Step $2i+1$ **(**$i \in [r]$**).** C sends the `reply` message of ExtCom for the i-th session.

Decommit phase. C sends v to R and decommits all the ExtCom commitments in the commit phase.

Fig. 2. Concurrently extractable commitment CECom [MOSV06].

Micciancio et al. [MOSV06] showed a concurrently extractable commitment CECom (see Figure 2). In CECom with parameter r, the committer sends `commit` messages of ExtCom r times in parallel and then the committer and the receiver exchange `challenge` and `reply` messages r times in sequence (thus, CECom has r sequential slots). When $r = \omega(\log n)$, the committed values of CECom are concurrently extractable with the rewinding strategy of [PRS02]. We note that in the stand-alone setting, the committed value of CECom is extractable by rewinding any single slot.

Concurrently $T(n)$-Extractable Commitments

For any function $T(n)$, we consider a relaxed notion of concurrent extractability called *concurrent $T(n)$-extractability*, which is the same as concurrent extractability except that the expected running time of the extractor is $T(n)$.

By using the rewinding strategy of [PV08], we can show that CECom is concurrently $\mathsf{poly}(n^{\log n})$-extractable when $r \geq 3$. (Note that when r is a constant, the round complexity of CECom is constant.) In the rewinding strategy of [PV08], the extractor computes a sequence of "threads of execution"—the *main thread* and *look-ahead threads*—where each thread consists of the views of all the parties. For these threads, the following hold.

- Each thread is a perfect simulation of a prefix of an actual execution.
- Any two threads share a (possibly empty) prefix, but they are independent after the shared prefix (i.e., after the prefix, the extractor emulates the interaction independently in each thread).
- The main thread is a perfect simulation of a complete execution, and the extractor outputs the view of $\mathcal{C}^*$ in the main thread.

A little more precisely, the extractor does the following. First, the extractor begins to generate the main thread by simulating the interaction between $\mathcal{C}^*$ and a honest receiver. Then, whenever a slot ends in a session on the main thread and the slot contains only "small" number of other slots, the extractor repeatedly generates look-ahead threads—by rewinding the slot and starting new simulations with fresh randomness—until the committed value of the session is extracted. In the look-ahead thread, the extractor also rewinds the slot in the same manner as in the main thread (thus, the rewinding is performed recursively). It was guaranteed that, when $r \geq 3$, the extractor rewinds at least one slot in every accepted session and therefore the extraction succeeds with probability 1.

3.6 Trapdoor Commitments

We recall the definition of *trapdoor commitments* from [PW09]. Roughly speaking, *trapdoor commitments* are ones such that there exists a simulator that can generate a simulated commitment and can later decommit it to any value.

Formally, a commitment scheme $\langle C, R \rangle$ is a trapdoor commitment if there exists an expected polynomial-time probabilistic oracle machine (or *simulator*) $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ such that for any PPT adversary $\mathcal{R}^*$ and all $v \in \{0,1\}^n$, the output (τ, ω) of the following two experiments are computationally indistinguishable:

Experiment 1. On input v, the honest committer C interacts with $\mathcal{R}^*$ to commit to v, and then decommits the commitment to v. Let τ be the view of $\mathcal{R}^*$ in the commit phase and ω be the message that C sends to $\mathcal{R}^*$ in the decommit phase.

Experiment 2. On input 1^n, the simulator $\mathcal{S}_1^{\mathcal{R}^*}$ generates a simulated view τ of the commit phase and a state STATE. Then, on inputs STATE and v, the simulator $\mathcal{S}_2$ generates ω.

Commit phase. To commit to $\sigma \in \{0,1\}$ on common input 1^n, the committer C does the following with the receiver R:

Step 1. R chooses a random n-bit string $e = (e_1, \ldots, e_n)$ and commits to e by using Com.

Step 2. For each $i \in [n]$, the committer C chooses a random $\eta_i \in \{0,1\}$ and sets

$$v_i := \begin{pmatrix} v_i^{00} & v_i^{01} \\ v_i^{10} & v_i^{11} \end{pmatrix} = \begin{pmatrix} \eta_i & \eta_i \\ \sigma \oplus \eta_i & \sigma \oplus \eta_i \end{pmatrix} \ .$$

Then, for each $i \in [n]$, $\alpha \in \{0,1\}$, and $\beta \in \{0,1\}$ in parallel, C commits to $v_i^{\alpha\beta}$ by using ExtCom; let $(v_i^{\alpha\beta}, d_i^{\alpha\beta})$ be the corresponding decommitment.

Step 3. R decommits the Step 1 commitment to e.

Step 4. For each $i \in [n]$, C sends $(v_i^{e_i 0}, d_i^{e_i 0})$ and $(v_i^{e_i 1}, d_i^{e_i 1})$ to R. Then, R checks whether these are valid decommitments and whether $v_i^{e_i 0} = v_i^{e_i 1}$.

Decommit phase. C sends σ and random $\gamma \in \{0,1\}$ to R. In addition, for every $i \in [n]$, C sends $(v_i^{0\gamma}, d_i^{0\gamma})$ and $(v_i^{1\gamma}, d_i^{1\gamma})$ to R. Then, R checks whether $(v_i^{0\gamma}, d_i^{0\gamma})$ and $(v_i^{1\gamma}, d_i^{1\gamma})$ are valid decommitments for every $i \in [n]$ and whether $v_0^{0\gamma} \oplus v_0^{1\gamma} = \cdots = v_n^{0\gamma} \oplus v_n^{1\gamma} = \sigma$.

Fig. 3. Black-box trapdoor bit commitment $\mathsf{TrapCom}$.

Pass and Wee [PW09] showed that the black-box protocol $\mathsf{TrapCom}$ in Figure 3 is a trapdoor bit commitment scheme. In fact, given the receiver's challenge e in advance, we can generate a simulated commitment and decommit it to both 0 and 1 in a straight-line manner (i.e., without rewinding the receiver) as follows. To generate a simulated commitment, the simulator internally simulates an interaction between C and $\mathcal{R}^*$ honestly except that in Step 2, the simulator chooses random $\gamma \in \{0,1\}$ and lets each v_i be a matrix such that the e_i-th row of v_i is (η_i, η_i) and the $(1-e_i)$-th row of v_i is $(\gamma \oplus \eta_i, (1-\gamma) \oplus \eta_i)$. To decommit the simulated commitment to $\sigma \in \{0,1\}$, the simulator decommits all the commitments in the $(\sigma \oplus \gamma)$-th column of each v_i.

From the extractability of ExtCom, we can show that $\mathsf{TrapCom}$ is extractable. In addition, by using the hiding property of Com, we can show that $\mathsf{TrapCom}$ satisfies the strong computational binding property. (Roughly speaking, if $\mathcal{C}^*$ generates a commitment that has more than one committed value, we can compute the committed value e of Com by extracting $v_1, \ldots, v_n$.)

Pass and Wee [PW09] showed that by running $\mathsf{TrapCom}$ in parallel, we obtain a black-box trapdoor commitment $\mathsf{PTrapCom}$ for multiple bits. $\mathsf{PTrapCom}$ also satisfies the strong computational binding property and extractability.

3.7 CCA-Secure Commitments

We recall the definition of CCA security and κ-robustness [CLP10,LP12]. *Tag-based commitment schemes* are ones such that both the committer and the receiver receive a string, or *tag*, as an additional input.

CCA security (w.r.t. the committed-value oracle). Roughly speaking, a tag-based commitment scheme $\langle C, R\rangle$ is *CCA-secure* if the hiding property of $\langle C, R\rangle$ holds even against adversary $\mathcal{A}$ that interacts with the *committed-value oracle* during the interaction with the committer. The committed-value oracle $\mathcal{O}$ interacts with $\mathcal{A}$ as an honest receiver in many concurrent sessions of the commit phase of $\langle C, R\rangle$ using tags chosen adaptively by $\mathcal{A}$. At the end of each session, if the commitment of this session is invalid or has multiple committed values, $\mathcal{O}$ returns $\perp$ to $\mathcal{A}$. Otherwise, $\mathcal{O}$ returns the unique committed value to $\mathcal{A}$.

More precisely, let us consider the following probabilistic experiment $\text{IND}_b(\langle C, R\rangle, \mathcal{A}, n, z)$ for each $b \in \{0,1\}$. On input 1^n and auxiliary input z, adversary $\mathcal{A}^{\mathcal{O}}$ adaptively chooses a pair of challenge values $v_0, v_1 \in \{0,1\}^n$ and an n-bit tag $\mathsf{id} \in \{0,1\}^n$. Then, $\mathcal{A}^{\mathcal{O}}$ receives a commitment to v_b with tag id, and $\mathcal{A}$ outputs y. The output of the experiment is $\perp$ if during the experiment, $\mathcal{A}$ sends $\mathcal{O}$ any commitment using tag id. Otherwise, the output of the experiment is y. Let $\mathsf{IND}_b(\langle C, R\rangle, \mathcal{A}, n, z)$ denote the output of experiment $\text{IND}_b(\langle C, R\rangle, \mathcal{A}, n, z)$.

Then, the CCA security of $\langle C, R\rangle$ is defined as follows.

Definition 1. *Let $\langle C, R\rangle$ be a tag-based commitment scheme and $\mathcal{O}$ be the committed-value oracle of $\langle C, R\rangle$. Then, $\langle C, R\rangle$ is* CCA-secure (w.r.t the committed-value oracle) *if for any* PPT *adversary $\mathcal{A}$, the following are computationally indistinguishable:*

- $\{\mathsf{IND}_0(\langle C, R\rangle, \mathcal{A}, n, z)\}_{n\in\mathbb{N}, z\in\{0,1\}^*}$
- $\{\mathsf{IND}_1(\langle C, R\rangle, \mathcal{A}, n, z)\}_{n\in\mathbb{N}, z\in\{0,1\}^*}$

If the length of the tags chosen by $\mathcal{A}$ is $t(n)$ instead of n, $\langle C, R\rangle$ is CCA-secure for tags of length $t(n)$. $\Diamond$

We also consider a relaxed notion of CCA security called *one-one CCA security*. In the definition of one-one CCA security, we consider adversaries that interact with $\mathcal{O}$ only in a single session of the commit phase.

In the following, we use *left session* to denote the session of the commit phase between the committer and $\mathcal{A}$, and use *right sessions* to denote the sessions between $\mathcal{A}$ and $\mathcal{O}$.

κ-robustness (w.r.t. the committed-value oracle). Roughly speaking, a tag-based commitment scheme is *κ-robust* if for any adversary $\mathcal{A}$ and any ITM B, the joint output of a κ-round interaction between $\mathcal{A}^{\mathcal{O}}$ and B can be simulated without $\mathcal{O}$ by a PPT simulator. Thus, the κ-robustness guarantees that the committed-value oracle is useless in attacking any κ-round protocol.

Formally, let $\langle C, R\rangle$ be a tag-based commitment scheme and $\mathcal{O}$ be the committed-value oracle of $\langle C, R\rangle$. For any constant $\kappa \in \mathbb{N}$, we say that $\langle C, R\rangle$ is κ-robust (w.r.t. the committed

value oracle) if there exists a PPT oracle machine (or simulator) $\mathcal{S}$ such that for any PPT adversary $\mathcal{A}$ and any κ-round PPT ITM B, the following are computationally indistinguishable:

- $\{\mathsf{output}_{B,\mathcal{A}^{\mathcal{O}}}[\langle B(y), \mathcal{A}^{\mathcal{O}}(z)\rangle(1^n, x)]\}_{n\in\mathbb{N},x,y,z\in\{0,1\}^n}$
- $\{\mathsf{output}_{B,\mathcal{S}^{\mathcal{A}}}[\langle B(y), \mathcal{S}^{\mathcal{A}}(z)\rangle(1^n, x)]\}_{n\in\mathbb{N},x,y,z\in\{0,1\}^n}$

Here, for any ITM A and B, we use $\mathsf{output}_{A,B}[\langle A(y), B(z)\rangle(x)]$ to denote the joint output of A and B in an interaction between them on inputs x, y to A and x, z to B respectively.

We also consider a relaxed notion of κ-robustness called κ-PQT-*robustness*. In the definition of κ-PQT-robustness, we allow the simulator to run in quasi-polynomial time.

Remark 1. In this paper, we assume that any adversary $\mathcal{A}$ (against CCA security or κ-robustness) interacts with $\mathcal{O}$ in at most $\mathsf{poly}(n)$ sessions even when the running time of $\mathcal{A}$ is super-polynomial. For our purpose, we can make this assumption without loss of generality. This is because when we use our CCA secure commitment scheme to obtain a general MPC protocol, the scheme need to be secure only for PPT adversaries.

4 One-One CCA Security for Short Tags

In this section, we construct a one-one CCA-secure commitment for tags of length $O(\log\log\log n)$. Since the length of the tags is $O(\log\log\log n)$, we can view each tag as a value in $\{0, 1 \ldots, d-1 = O(\log\log n)\}$.

4.1 Building Blocks

Let $T_i(n) \stackrel{\text{def}}{=} 2^{(\log n)^{(2/\epsilon)^{10i+1}}}$ for $i \in \mathbb{N}$. Then, for constants $a, b \in \mathbb{N}$, $\mathsf{PTrapCom}^b_a$ is a commitment scheme such that

- the hiding property holds against any $T_a(n)$-time adversary but is completely broken in time $T_{a+0.5}(n)$,
- the strong computational binding property holds against any $T_b(n)$-time adversary, and
- there exists a $T_{b+0.5}(n)$-time straight-line simulator (of the trapdoor property) such that the simulated commitment is indistinguishable from the actual commitment in time $T_a(n)$. (This holds even when $T_{b+0.5}(n) \gg T_a(n)$.)

We can construct $\mathsf{PTrapCom}^b_a$ by appropriately setting the security parameters of Com and ExtCom in $\mathsf{PTrapCom}$. (Recall that in $\mathsf{PTrapCom}$, many $\mathsf{TrapCom}$ are executed in parallel.) More precisely, we scale down the security parameter of ExtCom so that the parallel execution of ExtCom is hiding against $T_a(n)$-time adversaries but is completely broken in time less than $T_{a+0.5}(n)$, and scale down the security parameter of Com so that the parallel execution of Com is hiding against $T_b(n)$-time adversaries but is completely broken in time less than $T_{b+0.5}(n)$.

$\mathsf{PCETrapCom}^b_a$ is the same as $\mathsf{PTrapCom}^b_a$ except that we use CECom in Step 2 instead of ExtCom.

Commit phase. The committer C and the receiver R receive common inputs 1^n and $\mathsf{id} \in \{0, 1, \ldots, d-1 = O(\log\log n)\}$. To commit to $v \in \{0,1\}^n$, the committer C does the following with the receiver R.

Stage 1. C computes an $(n+1)$-out-of-$10n$ Shamir's secret sharing $\boldsymbol{s} = (s_1, \ldots, s_{10n})$ of value v. Then, for each $j \in [10n]$ in parallel, C commits to s_j by using $\mathsf{PTrapCom}_{i+d+1}^{i+d+1}$. Let (s_j, d_j) be the decommitment of the j-th commitment.

Stage 2. For each $j \in [10n]$ in parallel, C commits to (s_j, d_j) by using $\mathsf{PCETrapCom}_{i+d+2}^{i+\mathsf{id}+1}$. Here, the number of slots in $\mathsf{PCETrapCom}_{i+d+2}^{i+\mathsf{id}+1}$ is $\max(3, r+1)$, where r is the round complexity of $\mathsf{PTrapCom}_{i+d+1}^{i+d+1}$ in Stage 1.

Stage 3. For each $j \in [10n]$ in parallel, C commits to (s_j, d_j) by using $\mathsf{PCETrapCom}_{i+d+2}^{i+d-\mathsf{id}}$. Here, the number of slots in $\mathsf{PCETrapCom}_{i+d+2}^{i+d-\mathsf{id}}$ is $\max(3, r+1)$.

Stage 4. R sends a random subset $\varGamma \subseteq [10n]$ of size n to C.

Stage 5. For each $j \in \varGamma$, C decommits the j-th Stage 2 commitment and the j-th Stage 3 commitment to (s_j, d_j). Then, R checks whether (s_j, d_j) is a valid decommitment of the j-th Stage 1 commitment.

Decommit phase. C sends v, $\boldsymbol{s} = (s_1, \ldots, s_{10n})$, and $\boldsymbol{d} = (d_1, \ldots, d_{10n})$ to R. Then, R checks whether (s_j, d_j) is a valid decommitment of the j-th Stage 1 commitment for every $j \in [10n]$. Furthermore, R checks whether (1) $\boldsymbol{s}$ is 0.9-close to a valid codeword $\boldsymbol{w} = (w_1, \ldots, w_{10n})$ and (2) for each $j \in \varGamma$, w_j is equal to the share that was revealed in Stage 5. Finally, R checks whether $\boldsymbol{w}$ is a codeword corresponding to v.

Fig. 4. One-one CCA-secure commitment $\mathsf{CCACom}_i^{1:1}$.

4.2 One-One CCA Security for Tags of Length $O(\log\log\log n)$

Lemma 1. *Let $\epsilon < 1$ be a positive constant, and for any $i \in \mathbb{N}$, let $T_i(n) \stackrel{\text{def}}{=} 2^{(\log n)^{(2/\epsilon)^{10i+1}}}$. Assume the existence of one-way functions that are secure against 2^{n^ϵ}-time adversaries. Then, for any $i \in \mathbb{N}$, there exists a constant-round commitment scheme $\mathsf{CCACom}_i^{1:1}$ that satisfies the following for any $T_i(n)$-time adversary.*

- *Strong computational binding property, and*
- *One-one CCA security for tags of length $O(\log\log\log n)$.*

Furthermore, $\mathsf{CCACom}_i^{1:1}$ uses the underlying one-way function only in a black-box way.

Proof. $\mathsf{CCACom}_i^{1:1}$ is shown in Figure 4. The binding property follows from that of $\mathsf{PTrapCom}_{i+d+1}^{i+d+1}$. Thus, it remains to show that $\mathsf{CCACom}_i^{1:1}$ is one-one CCA secure for tags of length $O(\log\log\log n)$.

To show that $\mathsf{CCACom}_1^{1:1}$ is one-one CCA secure, we show that for any $T_i(n)$-time adversary $\mathcal{A}$ that interacts with $\mathcal{O}$ only in a single session, the following are computationally indistinguishable:

– $\{\mathsf{IND}_0(\mathsf{CCACom}_i^{1:1}, \mathcal{A}, n, z)\}_{n\in\mathbb{N}, z\in\{0,1\}^*}$
– $\{\mathsf{IND}_1(\mathsf{CCACom}_i^{1:1}, \mathcal{A}, n, z)\}_{n\in\mathbb{N}, z\in\{0,1\}^*}$

At the end of the right session, the committed-value oracle $\mathcal{O}$ does the following. First, $\mathcal{O}$ computes the committed values $\boldsymbol{s} = (s_1, \ldots, s_{10n})$ of the Stage 1 commitments by brute force. (If the committed value of the j-th commitment is not uniquely determined, s_j is defined to be $\perp$.) Then, $\mathcal{O}$ checks whether the following conditions hold: (1) $\boldsymbol{s}$ is 0.9-close to a valid codeword $\boldsymbol{w} = (w_1, \ldots, w_{10n})$ and (2) for every $j \in \mathit{\Gamma}$ (where $\mathit{\Gamma}$ is the subset that $\mathcal{O}$ sends to $\mathcal{A}$ in Stage 4), w_j is equal to the share that was revealed in Stage 5. If both conditions hold, $\mathcal{O}$ recovers v from $\boldsymbol{w}$ and returns v to $\mathcal{A}$. Otherwise, $\mathcal{O}$ returns $v := \perp$ to $\mathcal{A}$. We note that the running time of $\mathcal{O}$ is at most $\mathsf{poly}(n) \cdot T_{i+d+1.5}(n)$.

To show the indistinguishability, we consider hybrid experiments $G_a^b(n, z)$ for $a \in \{0, 1, 2, 3\}$ and $b \in \{0, 1\}$.

Hybrid $G_0^b(n, z)$ is the same as experiment $\mathrm{IND}_b(\mathsf{CCACom}_i^{1:1}, \mathcal{A}, n, z)$.

Hybrid $G_1^b(n, z)$ is the same as $G_0^b(n, z)$ except for the following:

– In Stage 2 (resp., Stage 3) on the left, the left committer simulates the $10n$ commitments of $\mathsf{PCETrapCom}_{i+d+2}^{i+\mathsf{id}+1}$ (resp., $\mathsf{PCETrapCom}_{i+d+2}^{i+d-\mathsf{id}}$) by using the straight-line simulator.
– In Stage 5 on the left, for each $j \in \mathit{\Gamma}$, the left committer decommits the simulated commitment of $\mathsf{PCETrapCom}_{i+d+2}^{i+\mathsf{id}+1}$ (resp., $\mathsf{PCETrapCom}_{i+d+2}^{i+d-\mathsf{id}}$) to (s_j, d_j) by using the simulator.

We note that the running time of $G_1^b(n, z)$ is at most $\mathsf{poly}(n) \cdot T_{i+d+1.5}(n)$.

Hybrid $G_2^b(n, z)$ is the same as $G_1^b(n, z)$ except for the following:

– Let $\widetilde{\mathsf{id}}$ be the tag of the right session. In Stage 2 (resp., Stage 3) of the right session, the committed values of the $\mathsf{PCETrapCom}_{i+d+2}^{i+\widetilde{\mathsf{id}}+1}$ (resp., $\mathsf{PCETrapCom}_{i+d+2}^{i+d-\widetilde{\mathsf{id}}}$) commitments are extracted *without rewinding Stage 1 on the left* by using the technique of [LP09, CLP10]. (That is, in Step 2 of each $\mathsf{PCETrapCom}_{i+d+2}^{i+\widetilde{\mathsf{id}}+1}$ (resp. $\mathsf{PCETrapCom}_{i+d+2}^{i+d-\widetilde{\mathsf{id}}}$) commitment, the committed values of CECom are extracted by rewinding a single slot that does not contain any Stage 1 messages of the left session. Such a slot must exist, since the number of slots in CECom is $\max(3, r+1)$.) Then, $\widehat{\boldsymbol{s}} = (\widehat{s}_1, \ldots, \widehat{s}_{10n})$ is defined as follows: if there exists $a \in \{2, 3\}$ such that the extracted value $(\widehat{s}_j^{(a)}, \widehat{d}_j^{(a)})$ of the j-th commitment in Stage a is a valid decommitment of the j-th commitment in Stage 1, let $\widehat{s}_j \stackrel{\text{def}}{=} \widehat{s}_j^{(a)}$ (if both $(\widehat{s}_j^{(2)}, \widehat{d}_j^{(2)})$ and $(\widehat{s}_j^{(3)}, \widehat{d}_j^{(3)})$ are valid decommitments but $\widehat{s}_j^{(2)} \neq \widehat{s}_j^{(3)}$, let $\widehat{s}_j \stackrel{\text{def}}{=} \perp$); otherwise, let $\widehat{s}_j \stackrel{\text{def}}{=} \perp$.
– At the end of the right session, $\mathcal{O}$ checks whether the following conditions hold: (1) $\widehat{\boldsymbol{s}}$ is *0.8-close* to a valid codeword $\widehat{\boldsymbol{w}} = (\widehat{w}_1, \ldots, \widehat{w}_{10n})$ and (2) for every $j \in \mathit{\Gamma}$, $\widehat{w}_j$ is equal to the share that was revealed in Stage 5. If both conditions hold, $\mathcal{O}$ recovers $\widehat{v}$ from $\widehat{\boldsymbol{w}}$ and returns $\widehat{v}$ to $\mathcal{A}$. Otherwise, $\mathcal{O}$ returns $\widehat{v} := \perp$ to $\mathcal{A}$. We note that $\mathcal{O}$ does not extract the committed values of the Stage 1 commitments.

We note that the expected running time of $G_2^b(n, z)$ is $\mathsf{poly}(n) \cdot T_{i+d+0.5}(n)$.

Hybrid $G_3^b(n,z)$ is the same as $G_2^b(n,z)$ except that on the left, the Stage 1 commitments are simulated by the straight-line simulator of $\mathsf{PTrapCom}_{i+d+1}^{i+d+1}$.

Since $\mathcal{A}$ receives no information about $\{s_j\}_{j\notin\Gamma}$ in $G_3^0(n,z)$ and $G_3^1(n,z)$, the output of $G_3^0(n,z)$ and that of $G_3^1(n,z)$ are identically distributed. Then, we consider the following claims. In what follows, we use $\mathsf{G}_i^b(n,z)$ to denote the output of experiment $G_i^b(n,z)$.

Claim 1. *For each $b\in\{0,1\}$, $\{\mathsf{G}_0^b(n,z)\}_{n\in\mathbb{N},z\in\{0,1\}^*}$ and $\{\mathsf{G}_1^b(n,z)\}_{n\in\mathbb{N},z\in\{0,1\}^*}$ are computationally indistinguishable.*

Claim 2. *For each $b\in\{0,1\}$, $\{\mathsf{G}_1^b(n,z)\}_{n\in\mathbb{N},z\in\{0,1\}^*}$ and $\{\mathsf{G}_2^b(n,z)\}_{n\in\mathbb{N},z\in\{0,1\}^*}$ are statistically indistinguishable.*

Claim 3. *For each $b\in\{0,1\}$, $\{\mathsf{G}_2^b(n,z)\}_{n\in\mathbb{N},z\in\{0,1\}^*}$ and $\{\mathsf{G}_3^b(n,z)\}_{n\in\mathbb{N},z\in\{0,1\}^*}$ are computationally indistinguishable.*

The lemma follows from these claims. □

Proof (of Claim 1). $G_1^b(n,z)$ differs from $G_0^b(n,z)$ only in that the Stage 2 commitments and the Stage 3 commitments on the left are simulated by the simulator of $\mathsf{PCETrapCom}_{i+d+2}^{i+\mathsf{id}+1}$ and that of $\mathsf{PCETrapCom}_{i+d+2}^{i+d-\mathsf{id}}$. Then, since the running time of $G_0^b(n,z)$ and that of $G_1^b(n,z)$ are at most $\mathsf{poly}(n)\cdot T_{i+d+1.5}(n)\ll T_{i+d+2}(n)$, the claim follows from the trapdoor property of $\mathsf{PCETrapCom}_{i+d+2}^{i+\mathsf{id}+1}$ and that of $\mathsf{PCETrapCom}_{i+d+2}^{i+d-\mathsf{id}}$. □

Next, we consider Claim 2. Note that $G_2^b(n,z)$ differs from $G_1^b(n,z)$ in that $\mathcal{O}$ computes the committed value of the right session from the extracted values of the Stage 2 commitments and those of the Stage 3 commitments instead of from those of Stage 1 commitments. We prove Claim 2 by showing that the value $\widehat{v}$ that $\mathcal{O}$ computes in $G_2^b(n,z)$ is the same as the value v that $\mathcal{O}$ computes in $G_1^b(n,z)$. Toward this end, we first show that in the right session of $G_1^b(n,z)$, the strong computational binding property holds in Stage 1 and either in Stage 2 or in Stage 3. Note that from the property of the cut-and-choose technique, this implies that the committed values of either the Stage 2 commitments or the Sage 3 commitments are 0.9-close to those of the Stage 1 commitments except with negligible probability. Let us say that $\mathcal{A}$ *cheats* in Stage 1 if at least one of $10n$ $\mathsf{PTrapCom}$ commitments in Stage 1 on the right has more than one committed value. We define cheating in Stage 2 and cheating in Stage 3 similarly. Then, we prove two subclaims.

Subclaim 1. *In $G_1^b(n,z)$, the probability that $\mathcal{A}$ cheats in Stage 1 is negligible.*

Proof. This subclaim follows directly from the strong computational binding property of $\mathsf{PTrapCom}_{i+d+1}^{i+d+1}$, since the running time of $G_1^b(n,z)$ is at most $\mathsf{poly}(n)\cdot T_{i+d+0.5}(n)\ll T_{i+d+1}(n)$ when $\mathcal{A}$ completes Stage 1 on the right.

Formally, assume for contradiction that in $G_1^b(n,z)$, $\mathcal{A}$ cheats in Stage 1 with non-negligible probability.

Let us consider the following adversary $\mathcal{B}$ against the strong computational binding property of $\mathsf{PTrapCom}^{i+d+1}_{i+d+1}$. $\mathcal{B}$ internally invokes $\mathcal{A}$ and simulates $G^b_1(n,z)$ for $\mathcal{A}$ as follows until $\mathcal{A}$ completes Stage 1 on the right. On the left, $\mathcal{B}$ perfectly simulates $G^b_1(n,z)$. On the right, $\mathcal{B}$ perfectly simulates $G^b_1(n,z)$ except that in Stage 1, $\mathcal{B}$ chooses random $j^* \in [10n]$ and forwards the j^*-th $\mathsf{PTrapCom}^{i+d+1}_{i+d+1}$ commitment from $\mathcal{A}$ to the external receiver.

Since $\mathcal{B}$ does not simulate $G^b_1(n,z)$ after $\mathcal{A}$ completes Stage 1 on the right (and thus, $\mathcal{B}$ does not need to simulate the value that $\mathcal{O}$ returns to $\mathcal{A}$ at the end of the right session), the running time of $\mathcal{B}$ is at most $\mathsf{poly}(n) \cdot T_{i+d+0.5}(n) \ll T_{i+d+1}(n)$. In addition, since $\mathcal{B}$ perfectly simulates $G^b_1(n,z)$, the j^*-th $\mathsf{PTrapCom}^{i+d+1}_{i+d+1}$ commitment has more than one committed value with non-negligible probability. Thus, we reach a contradiction. □

Subclaim 2. *In $G^b_1(n,z)$, the probability that $\mathcal{A}$ cheats in Stage 2 and Stage 3 simultaneously is negligible.*

Proof. To prove this subclaim, we need to show that even though the left committer "cheats," $\mathcal{A}$ cannot use the messages received on the left to cheat on the right. This can be proven by following the proof of the scheme of [PW10]. Roughly speaking, we show that there always exists $a^* \in \{2,3\}$ such that during Stage a^* on the right, the left session can be simulated in "short" time (i.e., the left session can be simulated without breaking the strong computational binding property of $\mathsf{PCETrapCom}$ in Stage a^*). A little more precisely, we show the following. Recall that the commitment of $\mathsf{PCETrapCom}$ can be simulated in polynomial time if we know the committed value of the Step 1 commitment of $\mathsf{PCETrapCom}$. Then, we show that in the left session, either this committed value can be extracted in "short" time (during Stage a^* of the right session) or it can be extracted before $\mathcal{A}$ starts Stage a^* on the right (and thus can be considered as an auxiliary input). Once we show that $\mathcal{A}$ cannot use the messages received on the left to cheat on the right, the subclaim follows from the strong computational binding property of $\mathsf{PCETrapCom}$ on the right.

Formally, assume for contradiction that with non-negligible probability, $\mathcal{A}$ cheats in Stage 2 and Stage 3 simultaneously. Let $\mathsf{Cheat}_{2,3}$ be the event that $\mathcal{A}$ cheats in Stage 2 and Stage 3 simultaneously, and let $\mathsf{Tag}_{\mathsf{id},\widetilde{\mathsf{id}}}$ be the event that $\mathcal{A}$ uses id on the left and uses $\widetilde{\mathsf{id}}$ on the right for any tags id and $\widetilde{\mathsf{id}}$. Then, since the number of tags is $d = O(\log \log n)$, there exists two distinct tags id and $\widetilde{\mathsf{id}}$ such that $\mathsf{Cheat}_{2,3}$ and $\mathsf{Tag}_{\mathsf{id},\widetilde{\mathsf{id}}}$ occur at the same time with non-negligible probability. In the following, we fix any such id and $\widetilde{\mathsf{id}}$. Then, to reach a contradiction, we consider the following three message schedules of $G^b_1(n,z)$ (see Figure 5).

Schedule 1. $\mathcal{A}$ completes Step 1 of $\mathsf{PCETrapCom}^{i+\mathsf{id}+1}_{i+d+2}$ in Stage 2 on the left after $\mathcal{A}$ completes Stage 2 on the right.

Schedule 2. $\mathcal{A}$ completes Step 1 of $\mathsf{PCETrapCom}^{i+\mathsf{id}+1}_{i+d+2}$ in Stage 2 on the left before $\mathcal{A}$ completes Stage 2 on the right, and $\mathcal{A}$ completes Step 1 of $\mathsf{PCETrapCom}^{i+d-\mathsf{id}}_{i+d+2}$ in Stage 3 on the left after $\mathcal{A}$ completes Stage 2 on the right.

Schedule 3. $\mathcal{A}$ completes Step 1 of $\mathsf{PCETrapCom}^{i+d-\mathsf{id}}_{i+d+2}$ in Stage 3 on the left before $\mathcal{A}$ completes Stage 2 on the right.

For any $k \in \{1,2,3\}$, let $\mathsf{Schedule}_k$ be the event that $\mathcal{A}$ chooses Schedule k. Since $\mathcal{A}$ must choose one of these schedules, there exists $k^* \in \{1,2,3\}$ such that the probability that all of $\mathsf{Schedule}_{k^*}$, $\mathsf{Cheat}_{2,3}$, and $\mathsf{Tag}_{\mathsf{id},\widetilde{\mathsf{id}}}$ occur simultaneously is non-negligible. Below, we reach a contradiction for each $k^* \in \{1,2,3\}$.

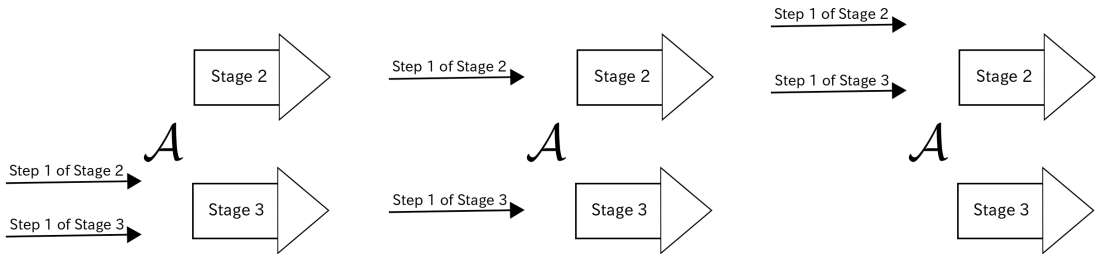

Fig. 5. Schedule 1 (left), Schedule 2 (center), and Schedule 3 (right).

When $k^* = 1$, let us consider the following adversary $\mathcal{B}_1$ against the strong computational binding property of $\mathsf{PCETrapCom}^{i+\widetilde{\mathsf{id}}+1}_{i+d+2}$. $\mathcal{B}_1$ internally invokes $\mathcal{A}$ and simulates $G^b_1(n,z)$ for $\mathcal{A}$ as follows until $\mathcal{A}$ completes Stage 2 on the right. On the left, $\mathcal{B}_1$ honestly simulates $G^b_1(n,z)$. On the right, $\mathcal{B}_1$ honestly simulates $G^b_1(n,z)$ except that in Stage 2, $\mathcal{B}_1$ chooses a uniformly random $j^* \in [10n]$ and forwards the j^*-th $\mathsf{PCETrapCom}^{i+\mathsf{id}+1}_{i+d+2}$ commitment from $\mathcal{A}$ to the external receiver. If $\mathsf{Schedule}_1$ or $\mathsf{Tag}_{\mathsf{id},\widetilde{\mathsf{id}}}$ does not occur, $\mathcal{B}_1$ terminates and outputs fail.

Since $\mathcal{B}_1$ does not simulate the left session after $\mathcal{A}$ completes Step 1 of Stage 2, and since $\mathcal{B}_1$ does not simulate the right session after $\mathcal{A}$ completes Stage 2, the running time of $\mathcal{B}_1$ is $\mathsf{poly}(n) + T_i(n) \ll T_{i+\widetilde{\mathsf{id}}+1}(n)$. In addition, from our assumption, $\mathcal{B}_1$ breaks the strong computational binding property with non-negligible probability. Thus, we reach a contradiction.

When $k^* = 2$ and $\mathsf{id} < \widetilde{\mathsf{id}}$, let us consider the following adversary $\mathcal{B}_2$ against the strong computational binding property of $\mathsf{PCETrapCom}^{i+\widetilde{\mathsf{id}}+1}_{i+d+2}$. $\mathcal{B}_2$ is the same as $\mathcal{B}_1$ except that $\mathcal{B}_2$ terminates and outputs fail if $\mathsf{Schedule}_2$ or $\mathsf{Tag}_{\mathsf{id},\widetilde{\mathsf{id}}}$ does not occur.

Since $\mathcal{B}_2$ does not simulate the left session after $\mathcal{A}$ completes Step 1 of Stage 3, and since $\mathcal{B}_1$ does not simulate the right interaction after $\mathcal{A}$ completes Stage 2, the running time of $\mathcal{B}_2$ is $\mathsf{poly}(n) \cdot T_{i+\mathsf{id}+1.5}(n) \ll T_{i+\widetilde{\mathsf{id}}+1}(n)$. In addition, from our assumption, $\mathcal{B}_2$ breaks the strong computational binding property with non-negligible probability. Thus, we reach a contradiction.

When $k^* = 2$ and $\mathsf{id} > \widetilde{\mathsf{id}}$, from an average argument, there exists a partial joint view $\mathcal{V}$ of all the parties (the left committer, $\mathcal{A}$, and $\mathcal{O}$) such that after $\mathcal{V}$, (1) $\mathcal{A}$ immediately starts Stage 3 on the right and (2) the probability that all of $\mathsf{Schedule}_2$, $\mathsf{Cheat}_{2,3}$, and $\mathsf{Tag}_{\mathsf{id},\widetilde{\mathsf{id}}}$ occur is non-negligible. Then, since $\mathsf{Schedule}_2$ and $\mathsf{Tag}_{\mathsf{id},\widetilde{\mathsf{id}}}$ can occur, $\mathcal{V}$ contains Step 1 of

$\mathsf{PCETrapCom}^{i+\mathsf{id}+1}_{i+d+2}$ in Stage 2 on the left. Let $\boldsymbol{e}$ be the sequence of the committed values of Com commitments in Step 1 of these $\mathsf{PCETrapCom}^{i+\mathsf{id}+1}_{i+d+2}$ commitments.
Then, let us consider the following adversary $\mathcal{B}'_2$ against the strong computational binding property of $\mathsf{PCETrapCom}^{i+d-\widetilde{\mathsf{id}}}_{i+d+2}$. Upon receiving non-uniform advice $\mathcal{V}$ and $\boldsymbol{e}$, $\mathcal{B}'_2$ internally invokes $\mathcal{A}$ and simulates $G^b_1(n,z)$ for $\mathcal{A}$ as follows until $\mathcal{A}$ completes Stage 3 on the right. First, $\mathcal{B}'_2$ feeds $\mathcal{V}$ to $\mathcal{A}$. Then, on the left, $\mathcal{B}'_2$ honestly simulates $G^b_1(n,z)$ by using $\boldsymbol{e}$ to simulate Stage 2. On the right, $\mathcal{B}'_2$ honestly simulates $G^b_1(n,z)$ except that in Stage 3, $\mathcal{B}'_2$ chooses uniformly random $j^* \in [10n]$ and forwards the j^*-th $\mathsf{PCETrapCom}^{i+d-\widetilde{\mathsf{id}}}_{i+d+2}$ commitment from $\mathcal{A}$ to the external receiver.
Since $\mathcal{B}'_2$ can simulate Stage 2 in the left session in polynomial time by using $\boldsymbol{e}$, the running time of $\mathcal{B}'_2$ is at most $\mathsf{poly}(n) \cdot T_{i+d-\mathsf{id}+0.5}(n) \ll T_{i+d-\widetilde{\mathsf{id}}}(n)$. In addition, from our assumption, $\mathcal{B}'_2$ breaks the strong computational binding property of $\mathsf{PCETrapCom}^{i+d-\widetilde{\mathsf{id}}}_{i+d+2}$ with non-negligible probability. Thus, we reach a contradiction.

When $k^* = 3$, from an average argument, there exists a partial joint view $\mathcal{V}'$ of all the parties (the left committer, $\mathcal{A}$, and $\mathcal{O}$) such that after $\mathcal{V}'$, (1) $\mathcal{A}$ immediately starts Stage 3 on the right and (2) the probability that all of $\mathsf{Schedule}_3$, $\mathsf{Cheat}_{2,3}$, and $\mathsf{Tag}_{\mathsf{id},\widetilde{\mathsf{id}}}$ occur is non-negligible. Then, since $\mathsf{Schedule}_3$ and $\mathsf{Tag}_{\mathsf{id},\widetilde{\mathsf{id}}}$ can occur, $\mathcal{V}'$ contains Step 1 of $\mathsf{PCETrapCom}^{i+d-\mathsf{id}}_{i+d+2}$ in Stage 3 on the left. Let $\boldsymbol{e}$ be the sequence of the committed values of Com commitments in Step 1 of these $\mathsf{PCETrapCom}^{i+d-\mathsf{id}}_{i+d+2}$.
Then, let us consider the following adversary $\mathcal{B}_3$ against the strong computational binding property of $\mathsf{PCETrapCom}^{i+d-\widetilde{\mathsf{id}}}_{i+d+2}$. $\mathcal{B}_3$ is the same as $\mathcal{B}'_2$ except that $\mathcal{B}_3$ receives $\mathcal{V}'$ and $\boldsymbol{e}$ as non-uniform advice, feeds $\mathcal{V}'$ to $\mathcal{A}$, and simulates Stage 3 on the left by using $\boldsymbol{e}$. (Since $\mathcal{V}'$ contains Stage 2 on the left, $\mathcal{B}_3$ does not need to simulate Stage 2 on the left.)
The running time of $\mathcal{B}_3$ is at most $\mathsf{poly}(n) + T_i(n) \ll T_{i+d-\widetilde{\mathsf{id}}}(n)$, and $\mathcal{B}_3$ breaks the strong computational binding property of $\mathsf{PCETrapCom}^{i+d-\widetilde{\mathsf{id}}}_{i+d+2}$ with non-negligible probability. Thus, we reach a contradiction.

Thus, for any case, we reach a contradiction. □

Now, we are ready to prove Claim 2.

Proof (of Claim 2). As noted above, we prove Claim 2 by showing that the value computed by $\mathcal{O}$ in $G^b_2(n,z)$ is equal to the value computed by $\mathcal{O}$ in $G^b_1(n,z)$. From Subclaim 2, there exists $a \in \{2,3\}$ such that the committed values of the Stage a commitments are uniquely determined. Then, since the committed values of the Stage 1 commitments and those of Stage a commitments are uniquely determined before Γ is chosen, the committed values of the Stage 1 commitments and those of Stage a commitments are 0.9-close except with negligible probability. Then, since we have carefully defined the behavior of $\mathcal{O}$ in $G^b_2(n,z)$ (in particular, since $\mathcal{O}$ checks whether the share is *0.8-close* to a valid codeword in $G^b_2(n,z)$), we can show that the value computed by $\mathcal{O}$ from the extracted values of Stage 2 and 3 is the same as the one computed from the committed values of Stage 1 in a similar manner to the previous works on black-box constructions [CDSMW08, CDSMW09, Wee10, LP12].

Formally, to show the indistinguishability, let us consider the following hybrid experiment:

Hybrid $G^b_{1.5}(n,z)$ is the same as $G^b_2(n,z)$ except that (1) on the right, $\mathcal{O}$ computes both v (as in $G^b_1(n,z)$) and $\widehat{v}$ (as in $G^b_2(n,z)$) and (2) if $v \neq \widehat{v}$, then $G^b_{1.5}(n,z)$ terminates and outputs fail.

In the following, we show that $G^b_{1.5}(n,z)$ outputs fail with at most negligible probability. Since the view of $\mathcal{A}$ in $G^b_1(n,z)$ and that of $\mathcal{A}$ in $G^b_2(n,z)$ differ only in the value that $\mathcal{O}$ returns to $\mathcal{A}$, Claim 2 follows.

If the right session is not accepted, we have $v = \widehat{v} = \bot$, and thus $G^b_{1.5}(n,z)$ does not output fail. Thus, in the following, we assume that the right session is accepted. Since the view of $\mathcal{A}$ in $G^b_{1.5}(n,z)$ is identical with that of $\mathcal{A}$ in $G^b_1(n,z)$ until $\mathcal{A}$ completes the right session, Subclaims 1 and 2 imply that except with negligible probability, there exists $a \in \{2,3\}$ such that each commitment in Stage 1 and Stage a on the right has at most one committed value. Let $\boldsymbol{s}^{(1)} = (s^{(1)}_1, \dots, s^{(1)}_{10n})$ be the committed values of the Stage 1 commitment. Let $((s^{(a)}_1, d^{(a)}_1), \dots, (s^{(a)}_{10n}, d^{(a)}_{10n}))$ be the committed values of the Stage a commitments, and let $\boldsymbol{s}^{(a)} = (s^{(a)}_1, \dots, s^{(a)}_{10n})$ and $\boldsymbol{d}^{(a)} = (d^{(a)}_1, \dots, d^{(a)}_{10n})$. For $j \in [10n]$, we say that *the j-th column is bad* if $(s^{(a)}_j, d^{(a)}_j)$ is not a valid decommitment of the j-th Stage 1 commitment. Then, from the property of the cut-and-choose technique, the number of bad columns is less than n except with exponentially small probability. Thus, when the committed values of the Stage a commitments are extracted as in $G^b_2(n,z)$, the valid decommitments of the Stage 1 commitments are extracted in at least $9n$ columns. Then, since each Stage 1 commitment has at most one committed value, the shares $\widehat{\boldsymbol{s}}$, which is computed as in $G^b_2(n,z)$, is also 0.9-close to $\boldsymbol{s}^{(1)}$ except with negligible probability. Then, let us consider the following two cases:

- In Case 1, $\boldsymbol{s}^{(1)}$ is 0.9-close to a valid codeword $\boldsymbol{w} = (w_1, \dots, w_{10n})$. In this case, except with negligible probability, $\widehat{\boldsymbol{s}}$ is 0.8-close to $\boldsymbol{w}$. Thus, the codeword $\widehat{\boldsymbol{w}}$, which $\mathcal{O}$ computed from $\widehat{\boldsymbol{s}}$ as in $G^b_2(n,z)$, is identical with $\boldsymbol{w}$. Thus, we have $\widehat{v} = v$ except with negligible probability.
- In Case 2, $\boldsymbol{s}^{(1)}$ is not 0.9-close to any valid codeword. In this case, we have $v = \bot$. If $\widehat{\boldsymbol{s}}$ is not 0.8-close to any valid codeword, we have $\widehat{v} = \bot$; thus we have $\widehat{v} = v$. If $\widehat{\boldsymbol{s}}$ is 0.8-close to a valid codeword $\widehat{\boldsymbol{w}} = (\widehat{w}_1, \dots, \widehat{w}_{10n})$, since $\boldsymbol{s}^{(1)}$ and $\widehat{\boldsymbol{w}}$ are not 0.9-close, except with exponentially small probability there exists $j \in \mathit{\Gamma}$ such that $\widehat{w}_j \neq s^{(1)}_j$. Thus, except with negligible probability, we have $\widehat{v} = v = \bot$.

Thus, $G^b_{1.5}(n,z)$ outputs fail with at most negligible probability. □

Finally, we prove Claim 3.

Proof (of Claim 3). $G^b_3(n,z)$ differs from $G^b_2(n,z)$ only in that on the left, the Stage 1 commitments and their decommitments are generated by the simulator of $\mathsf{PTrapCom}^{i+d+1}_{i+d+1}$. Then, since the running time of $G^b_2(n,z)$ and that of $G^b_3(n,z)$ are at most $\mathsf{poly}(n) \cdot T_{i+d+0.5}(n) \ll T_{i+d+1}(n)$ except for Stage 1 on the left, and since Stage 1 on the left is not rewound in $G^b_2(n,z)$ and in $G^b_3(n,z)$, the claim follows from the trapdoor property of $\mathsf{PTrapCom}^{i+d+1}_{i+d+1}$. □

5 CCA Security from One-One CCA Security

In this section, we show a transformation from any one-one CCA-secure commitment scheme to a CCA-secure commitment scheme. To use this transformation to obtain a general MPC protocol, we also show that the resultant CCA-secure commitment satisfies κ-PQT-robustness for any $\kappa \in \mathbb{N}$.

Lemma 2. *Let $\epsilon < 1$ be a positive constant, and assume the existence of one-way functions that are secure against 2^{n^ϵ}-time adversaries. Let $r(\cdot)$ and $t(\cdot)$ be arbitrary functions, let $T_i(n) \stackrel{\text{def}}{=} 2^{(\log n)^{(2/\epsilon)^{10i+1}}}$ for any $i \in \mathbb{N}$, and let $\mathsf{CCACom}^{1:1}_{i+3}$ be an $r(n)$-round commitment scheme that satisfies the following for any $T_{i+3}(n)$-time adversary.*

- *Strong computational binding property, and*
- *One-one CCA security for tags of length $t(n)$.*

Then, for any $\kappa \in \mathbb{N}$, there exists an $(r(n)+O(1))$-round commitment scheme CCACom_i that satisfies the following for any $T_i(n)$-time adversary.

- *Statistical binding property,*
- *CCA security for tags of length $t(n)$, and*
- *κ-PQT-robustness.*

If $\mathsf{CCACom}^{1:1}_{i+3}$ uses the underlying one-way function only in a black-box way, then CCACom_i uses the underlying one-way function only in a black-box way.

In the proof of Lemma 2, we use the following building blocks, which we can obtain by appropriately setting the security parameters of known protocols [Nao91, MOSV06, PV08].

- A 2-round statistically binding commitment Com_{i+1} that is secure against $T_{i+1}(n)$-time adversaries but is completely broken in time $T_{i+1.5}(n)$.
- A constant-round concurrently $\mathsf{poly}(n^{\log n})$-extractable commitment CECom_{i+2} that is secure against $T_{i+2}(n)$-time adversaries but is completely broken in time $T_{i+2.5}(n)$. The number of slots in CECom_{i+2} is $\kappa + 3$, and the extractor uses the rewinding strategy of [PV08].

We note that both Com_{i+1} and CECom_{i+2} use the underlying one-way function in a black-box way.

Proof (of Lemma 2). CCACom_i is shown in Figure 6. The statistical binding property of CCACom_i follows from that of Com_{i+1}. Then, we consider the following propositions.

Proposition 1. *For any $T_i(n)$-time adversary, CCACom_i is CCA secure for tags of length $t(n)$.*

Proposition 2. *For any $T_i(n)$-time adversary, CCACom_i is κ-PQT-robust.*

The lemma follows from these propositions. □

Commit phase. The committer C and the receiver R receive common inputs 1^n and $\mathsf{id} \in \{0,1\}^{t(n)}$. To commit to $v \in \{0,1\}^n$, the committer C does the following with the receiver R.

Stage 1. R chooses a random subset $\Gamma \subseteq [10n]$ of size n. Then, R commits to Γ by using $\mathsf{CCACom}^{1:1}_{i+3}$ with tag id.

Stage 2. C computes an $(n+1)$-out-of-$10n$ Shamir's secret sharing $\boldsymbol{s} = (s_1, \ldots, s_{10n})$ of value v. Then, for each $j \in [10n]$ in parallel, C commits to s_j by using Com_{i+1}.

Stage 3. For each $j \in [10n]$ in parallel, C commits to s_j by using CECom_{i+2}.

Stage 4. R decommits the Stage 1 commitment to Γ.

Stage 5. For every $j \in [10n]$, let the *j-th column* denote the j-th commitment in Stage 2 and the j-th one in Stage 3 (that is, the commitments whose committed value is s_j). Then, for each $j \in \Gamma$, C decommits the commitments of the j-th column to s_j.

Decommit phase. C sends v to R and decommits the Stage 2 commitments to $\boldsymbol{s}$. Then, R checks whether all of these decommitments are valid. Furthermore, R checks whether (1) $\boldsymbol{s}$ is 0.9-close to a valid codeword $\boldsymbol{w} = (w_1, \ldots, w_{10n})$ and (2) for every $j \in \Gamma$, w_j is equal to the share that was revealed in Stage 5. Finally, R checks whether $\boldsymbol{w}$ is a codeword corresponding to v.

Fig. 6. CCA-secure commitment CCACom_i.

5.1 Proof of Proposition 1

Proof (of Proposition 1). We show that for any $T_i(n)$-time adversary $\mathcal{A}$, the following are computationally indistinguishable:

- $\{\mathsf{IND}_0(\mathsf{CCACom}_i, \mathcal{A}, n, z)\}_{n\in\mathbb{N}, z\in\{0,1\}^*}$
- $\{\mathsf{IND}_1(\mathsf{CCACom}_i, \mathcal{A}, n, z)\}_{n\in\mathbb{N}, z\in\{0,1\}^*}$

Note that $\mathcal{O}$ does the following in each right session. First, $\mathcal{O}$ extracts the committed values $\boldsymbol{s} = (s_1, \ldots, s_{10n})$ of the Stage 2 commitments by brute force. (If the committed value of the j-th commitment is not uniquely determined, s_j is defined to be $\bot$.) Then, at the end of the session, $\mathcal{O}$ checks whether the following conditions hold: (1) $\boldsymbol{s}$ is 0.9-close to a valid codeword $\boldsymbol{w} = (w_1, \ldots, w_{10n})$, and (2) for every $j \in \Gamma$ (where Γ is the value that $\mathcal{O}$ sends to $\mathcal{A}$ in Stage 4), w_j is equal to the share that was revealed in Stage 5. If both conditions hold, $\mathcal{O}$ recovers v from $\boldsymbol{w}$ and returns v to A. Otherwise, $\mathcal{O}$ returns $v := \bot$ to $\mathcal{A}$. We note that the running time of $\mathcal{O}$ is at most $\mathsf{poly}(n) \cdot T_{i+1.5}(n)$.

To show the indistinguishability, we consider hybrid experiments $H^b_a(n, z)$ for $a \in \{0, 1, 2, 3\}$ and $b \in \{0, 1\}$.

Hybrid $H^b_0(n, z)$ is the same as experiment $\mathrm{IND}_b(\mathsf{CCACom}_i, \mathcal{A}, n, z)$.
Hybrid $H^b_1(n, z)$ is the same as $H^b_0(n, z)$ except for the following:

– In Stage 1 of the left session, the committed value Γ is extracted by brute force. If the commitment is invalid or has multiple committed values, Γ is defined to be a random subset.[8]
– In Stage 3 of the left session, the left committer commits to 0 instead of s_j for each $j \notin \Gamma$.

The running time of $H_1^b(n,z)$ is at most $\mathsf{poly}(n) \cdot T_{i+1.5}(n)$ except for the brute-force extraction of the Stage 1 commitment on the left.

Hybrid $H_2^b(n,z)$ is the same as $H_1^b(n,z)$ except for the following:
– In every right session of which Stage 2 ends after $\mathcal{A}$ starts Stage 2 on the left, the committed values of the Stage 3 commitments are extracted by using the concurrent $\mathsf{poly}(n^{\log n})$-extractability of CECom_{i+2}. Let $\widehat{\boldsymbol{s}} = (\widehat{s}_1, \ldots, \widehat{s}_{10n})$ be the extracted values.
– At the end of each right session in which $\widehat{\boldsymbol{s}} = (\widehat{s}_1, \ldots, \widehat{s}_{10n})$ is extracted, $\mathcal{O}$ does the following. First, $\mathcal{O}$ checks whether the following conditions hold: (1) $\widehat{\boldsymbol{s}}$ is *0.8-close* to a valid codeword $\widehat{\boldsymbol{w}} = (\widehat{w}_1, \ldots, \widehat{w}_{10n})$ and (2) for every $j \in \widetilde{\Gamma}$ (where $\widetilde{\Gamma}$ is the value that $\mathcal{O}$ sends to $\mathcal{A}$ in this session), $\widehat{w}_j$ is equal to the share that was revealed in Stage 5. If both conditions hold, $\mathcal{O}$ recovers $\widehat{v}$ from $\widehat{\boldsymbol{w}}$ and returns $\widehat{v}$ to $\mathcal{A}$. Otherwise, $\mathcal{O}$ returns $\widehat{v} := \bot$ to $\mathcal{A}$. We note that $\mathcal{O}$ does not extract the committed values of the Stage 2 commitments in such right sessions.

The expected running time of $H_2^b(n,z)$ is at most $\mathsf{poly}(n^{\log n}) \cdot T_i(n)$ after the start of Stage 2 on the left.

Hybrid $H_3^b(n,z)$ is the same as $H_2^b(n,z)$ except that in Stage 2 on the left, the left committer commits to 0 instead of s_j for each $j \notin \Gamma$.

Since $\mathcal{A}$ receives no information about $\{s_j\}_{j \notin \Gamma}$ on the left in $H_3^0(n,z)$ and $H_3^1(n,z)$, and since $\boldsymbol{s}$ is $(n+1)$-out-of-$10n$ secret sharing, the output of $H_3^0(n,z)$ and that of $H_3^1(n,z)$ are identically distributed. Then, we consider the following claims. In what follows, we use $\mathsf{H}_i^b(n,z)$ to denote the output of experiment $H_i^b(n,z)$.

Claim 4. *For each $b \in \{0,1\}$, $\{\mathsf{H}_0^b(n,z)\}_{n\in\mathbb{N},z\in\{0,1\}^*}$ and $\{\mathsf{H}_1^b(n,z)\}_{n\in\mathbb{N},z\in\{0,1\}^*}$ are computationally indistinguishable.*

Claim 5. *For each $b \in \{0,1\}$, $\{\mathsf{H}_1^b(n,z)\}_{n\in\mathbb{N},z\in\{0,1\}^*}$ and $\{\mathsf{H}_2^b(n,z)\}_{n\in\mathbb{N},z\in\{0,1\}^*}$ are statistically indistinguishable.*

Claim 6. *For each $b \in \{0,1\}$, $\{\mathsf{H}_2^b(n,z)\}_{n\in\mathbb{N},z\in\{0,1\}^*}$ and $\{\mathsf{H}_3^b(n,z)\}_{n\in\mathbb{N},z\in\{0,1\}^*}$ are computationally indistinguishable.*

The proposition follows from these claims. □

Proof (of Claim 4). The view of $\mathcal{A}$ in $H_0^b(n,z)$ and that of $\mathcal{A}$ in $H_1^b(n,z)$ differ only in the committed values of CECom_{i+2} on the left. In addition, the running time of $H_0^b(n,z)$ and

[8] Since the running time of $\mathcal{A}$ and $\mathcal{O}$ is at most $\mathsf{poly}(n) \cdot T_{i+1.5}(n) \ll T_{i+2}(n)$, the strong computational binding property of $\mathsf{CCACom}_{i+3}^{1:1}$ guarantees that the Stage 1 commitment has at most one committed value except with negligible probability.

that of $H_1^b(n,z)$ are $\mathsf{poly}(n) \cdot T_{i+1.5}(n) \ll T_{i+2}(n)$ (except for the brute force extraction of the Stage 1 commitment on the left in $H_1^b(n,z)$). Thus, by considering Γ as non-uniform advice, we can prove indistinguishability from the hiding property of CECom_{i+2}.

Formally, assume for contradiction that there exists a polynomial $p(\cdot)$ such that for infinitely many n, there exists $z \in \{0,1\}^*$ and a PPT distinguisher $\mathcal{D}$ such that

$$\left|\Pr\left[\mathcal{D}(\mathsf{H}_0^b(n,z))=1\right] - \Pr\left[\mathcal{D}(\mathsf{H}_1^b(n,z))=1\right]\right| \geq \frac{1}{p(n)} \ .$$

In the following, we fix any such n and z. Let ρ be a prefix of the transcript of $\mathsf{H}_0^b(n,z)$ and $\mathsf{H}_1^b(n,z)$ such that immediately after ρ, Stage 3 starts on the left. (Recall that $\mathsf{H}_0^b(n,z)$ and $\mathsf{H}_1^b(n,z)$ proceed identically until the start of Stage 3 on the left.) Let prefix_ρ be the event that a prefix of the transcript is ρ. From an average argument, with probability $1/2p(n)$ over the choice of ρ, we have

$$\left|\Pr\left[\mathcal{D}(\mathsf{H}_0^b(n,z))=1 \mid \mathsf{prefix}_\rho\right] - \Pr\left[\mathcal{D}(\mathsf{H}_1^b(n,z))=1 \mid \mathsf{prefix}_\rho\right]\right| \geq \frac{1}{2p(n)} \ . \tag{1}$$

Then, from the strong computational binding property of $\mathsf{CCACom}_{i+3}^{1:1}$, with probability $1/2p(n) - \mathsf{negl}(n)$ over the choice of ρ, we have Equation (1) and ρ uniquely determines the committed value Γ of the Stage 1 commitment of the left session. We fix any such ρ.

Let us consider the following $T_{i+2}(n)$-time adversary $\mathcal{B}$ against the (parallel) hiding property of CECom_{i+2}. On non-uniform advice ρ and Γ, adversary $\mathcal{B}$ internally invokes $\mathcal{A}$ and honestly simulates $H_1^b(n,z)$ after ρ except for the following:

- In Stage 3 on the left, $\mathcal{B}$ sends $(s_j)_{j \notin \Gamma}$ and $(0,\ldots,0)$ to the external committer of CECom_{i+2}. Then, since the external committer commits to either $(s_j)_{j\notin\Gamma}$ or $(0,\ldots,0)$ in parallel, $\mathcal{B}$ forwards these commitments to $\mathcal{A}$. (At the same time, $\mathcal{B}$ computes commitments to $(s_j)_{j\in\Gamma}$ and sends them to $\mathcal{A}$.)

The output of $\mathcal{B}$ is that of the simulated $H_1^b(n,z)$.

We reach a contradiction by showing that $\mathcal{B}$ breaks the hiding property of CECom_{i+2}. The view of the internal $\mathcal{A}$ is identical with either that of $\mathcal{A}$ in $H_0^b(n,z)$ (when $\mathcal{B}$ receives commitments to $(s_j)_{j\notin\Gamma}$) or that of $\mathcal{A}$ in $H_1^b(n,z)$ (when $\mathcal{B}$ receives commitments to $(0,\ldots,0)$). Thus, from Equation (1), $\mathcal{B}$ distinguishes the commitments of $(s_j)_{j\notin\Gamma}$ from those of $(0,\ldots,0)$ with non-negligible probability. □

Next, we consider Claim 5. Note that $H_2^b(n,z)$ differs from $H_1^b(n,z)$ in that $\mathcal{O}$ computes the committed value of each right session from the extracted values of Stage 3 commitments instead of from those of Stage 2 commitments. We prove Claim 5 by showing that at the end of each right session, the value $\widehat{v}$ that $\mathcal{O}$ computes in $H_2^b(n,z)$ is the same as the value v that $\mathcal{O}$ computes in $H_1^b(n,z)$. Formally, for any right session, let $\boldsymbol{s}^{(2)} = (s_1^{(2)},\ldots,s_{10n}^{(2)})$ be the committed values of the Stage 2 commitments (if the committed value of the j-th commitment is not uniquely determined, $s_j^{(2)}$ is defined to be $\perp$) and let $\boldsymbol{s}^{(3)} = (s_1^{(3)},\ldots,s_{10n}^{(3)})$

be the committed values of the Stage 3 commitments. Then, for every $j \in [10n]$, we say that the j-th column of this session is *bad* if $s_j^{(2)} = \bot$, $s_j^{(3)} = \bot$, or $s_j^{(2)} \neq s_j^{(3)}$. In addition, we say that $\mathcal{A}$ *cheats* in this session if the session is accepted and the number of bad columns is at least n. Then, we prove the following subclaim.

Subclaim 3. *In any right session of $H_1^b(n, z)$, $\mathcal{A}$ cheats with at most negligible probability.*

Proof. At first sight, it seems that we can prove this subclaim by simply using the hiding property of $\mathsf{CCACom}_{i+3}^{1:1}$ and the property of cut-and-choose technique (i.e., it seems that, since the committed value Γ of the Stage 1 commitment on the right is hidden from $\mathcal{A}$, the probability that there are at least n bad columns but the session is accepted is negligible). However, $\mathcal{A}$ interacts with the left committer as well as with $\mathcal{O}$, and the left committer "cheats" in the left session (i.e., on the left, the committed values of the Stage 2 commitments and those of the Stage 3 commitments are not 0.9-close). Thus, $\mathcal{A}$ may be able to cheat in a right session by using the messages received on the left. A key to prove this subclaim is that the left session can be simulated by using the committed-value oracle of $\mathsf{CCACom}_{i+3}^{1:1}$ (i.e., if we know the committed value Γ of the Stage 1 commitment on the left, we can simulate the later stages in polynomial time). Thus, the one-one CCA security of $\mathsf{CCACom}_{i+3}^{1:1}$ guarantees that $\mathcal{A}$ cannot break the hiding property of $\mathsf{CCACom}_{i+3}^{1:1}$ even with the messages of the left session. We can therefore use the cut-and-choose technique to prove the subclaim.

Formally, assume for contradiction that in $H_1^b(n, z)$ there exists a right session in which $\mathcal{A}$ cheats with non-negligible probability. Then, since the number of right sessions is at most $\mathsf{poly}(n)$, $\mathcal{A}$ cheats with non-negligible probability in a randomly chosen session.

Let us consider the following $T_{i+3}(n)$-time adversary $\mathcal{B}$ against one-one CCA security of $\mathsf{CCACom}_{i+3}^{1:1}$. $\mathcal{B}$ internally invokes $\mathcal{A}$ and simulates $H_1^b(n, z)$ for $\mathcal{A}$ as follows. In Stage 1 on the left, $\mathcal{B}$ forwards the commitment from $\mathcal{A}$ to the committed-value oracle $\mathcal{O}$ (of $\mathsf{CCACom}_{i+3}^{1:1}$) and receives Γ from $\mathcal{O}$. (If $\mathcal{O}$ returns $\bot$, $\mathcal{B}$ let Γ be a random subset.) Then, $\mathcal{B}$ honestly simulates the later stages on the left by using Γ. On the right, $\mathcal{B}$ honestly simulates $H_1^b(n, z)$ in every right session except in a randomly chosen one. In Stage 1 of this randomly chosen session, $\mathcal{B}$ sends random subsets $\Gamma_0, \Gamma_1 \subseteq [10n]$ of size n to the external $\mathsf{CCACom}_{i+3}^{1:1}$ committer and forwards the $\mathsf{CCACom}_{i+3}^{1:1}$ commitment from the external committer to $\mathcal{A}$ (the committed value is either Γ_0 or Γ_1). Then, $\mathcal{B}$ honestly simulates $H_1^b(n, z)$ in Stage 2 and in Stage 3, and at the end of Stage 3, $\mathcal{B}$ extracts the committed values $\boldsymbol{s}^{(2)} = (s_1^{(2)}, \ldots, s_{10n}^{(2)})$ and $\boldsymbol{s}^{(3)} = (s_1^{(3)}, \ldots, s_{10n}^{(3)})$ by brute force. If $s_j^{(2)} = s_j^{(3)} \neq \bot$ for all $j \in \Gamma_1$ and the number of bad columns is at least n, $\mathcal{B}$ outputs 1. Otherwise, $\mathcal{B}$ outputs 0. We note that the running time of $\mathcal{B}$ is $\mathsf{poly}(n) \cdot T_{i+2.5}(n) \ll T_{i+3}(n)$.

We reach a contradiction by showing that $\mathcal{B}$ breaks the one-one CCA security of $\mathsf{CCACom}_{i+3}^{1:1}$. Since $\mathcal{B}$ perfectly simulates $H_1^b(n, z)$ for $\mathcal{A}$, the internal $\mathcal{A}$ cheats with non-negligible probability in the session that $\mathcal{B}$ chooses. Therefore, when $\mathcal{B}$ receives a commitment to Γ_1, $\mathcal{B}$ outputs 1 with non-negligible probability. On the other hand, when $\mathcal{B}$ receives a commitment to Γ_0, since the internal $\mathcal{A}$ receives no information about Γ_1, the probability that the number of bad

columns is at least n but we have $s_j^{(2)} = s_j^{(3)} \neq \perp$ for all $j \in \Gamma_1$ is exponentially small. Thus, when $\mathcal{B}$ receives a commitment to Γ_0, $\mathcal{B}$ outputs 1 with at most negligible probability. $\square$

Now, we are ready to prove Claim 5.

Proof (of Claim 5). As noted above, we prove Claim 5 by showing that at the end of each right session, the value computed by $\mathcal{O}$ in $H_2^b(n,z)$ is equal to the value computed by $\mathcal{O}$ in $H_1^b(n,z)$. Recall that if some of the Stage 3 commitments are invalid, there is over-extraction. To show that the equality holds even with over-extraction, we use the technique used in the analysis of the previous black-box constructions [CDSMW08, CDSMW09, Wee10, LP12]. Note that Subclaim 3 guarantees that there are at most n bad columns (and thus there are not many invalid commitments). Then, since we have carefully defined the behavior of $\mathcal{O}$ in $H_2^b(n,z)$ (in particular, since $\mathcal{O}$ checks whether the share is *0.8-close* to a valid codeword in $H_2^b(n,z)$), $\mathcal{O}$ can correctly decide the validity of each session in $H_2^b(n,z)$ even with over-extraction.

Formally, to show the indistinguishability between $\{\mathsf{H}_1^b(n,z)\}_{n\in\mathbb{N},z\in\{0,1\}^*}$ and $\{\mathsf{H}_2^b(n,z)\}_{n\in\mathbb{N},z\in\{0,1\}^*}$, let us consider the following hybrid experiment:

Hybrid $H_{1.5}^b(n,z)$ is the same as $H_2^b(n,z)$ except that in every right session on every thread,[9] if the committed values of CECom_{i+2} are extracted, then (1) $\mathcal{O}$ computes both v (as in $H_1^b(n,z)$) and $\widehat{v}$ (as in $H_2^b(n,z)$) and (2) $H_{1.5}^b(n,z)$ terminates and outputs fail when $v \neq \widehat{v}$.

In the following, we fix a thread in $H_{1.5}^b(n,z)$ and show that in every right session on this thread, $H_{1.5}^b(n,z)$ outputs fail with at most negligible probability. Since the only difference between the view of $\mathcal{A}$ in $H_1^b(n,z)$ and that of $\mathcal{A}$ in the main thread of $H_2^b(n,z)$ is the values that $\mathcal{O}$ returns to $\mathcal{A}$, Claim 5 follows.

First, we show that except with negligible probability, $H_{1.5}^b(n,z)$ does not outputs fail at the end of the first right session in which the committed values of CECom_{i+2} are extracted (the order of the right sessions is defined as the order of their completion; thus, the first right session of $H_{1.5}^b(n,z)$ is the first right session that $\mathcal{A}$ completes in $H_{1.5}^b(n,z)$). If the first right session is not accepted, we have $v = \widehat{v} = \perp$ and thus $H_{1.5}^b(n,z)$ does not output fail at the end of this session. Thus, we assume that the first right session is accepted. Since the view of $\mathcal{A}$ on any thread in $H_{1.5}^b(n,z)$ is identical with that of $\mathcal{A}$ in $H_1^b(n,z)$ until $\mathcal{A}$ completes the first right session in which the committed values of CECom_{i+2} are extracted, Subclaim 3 implies that the number of bad columns in the first right session on any thread in $H_{1.5}^b(n,z)$ is less than n except with negligible probability. Then, let us consider the following two cases:

- In Case 1, the committed shares $\boldsymbol{s}^{(2)} = (s_1^{(2)}, \ldots, s_{10n}^{(2)})$ of the Stage 2 commitment in the first right session is 0.9-close to a valid codeword $\boldsymbol{w} = (w_1, \ldots, w_{10n})$. In this case, except with negligible probability, the committed shares $\boldsymbol{s}^{(3)} = (s_1^{(3)}, \ldots, s_{10n}^{(3)})$ of the Stage 3 commitments is 0.8-close to $\boldsymbol{w}$. (Recall that, from Subclaim 3, $\boldsymbol{s}^{(3)}$ is 0.9-close to $\boldsymbol{s}^{(2)}$ except with negligible probability.) Then, except with negligible probability, the

[9] Recall that, as noted in Section 3.5, the extractor computes a sequence of "threads of execution" in the rewinding strategy of [PV08].

extracted shares $\widehat{\boldsymbol{s}}^{(3)} = (\widehat{s}_1^{(3)}, \ldots, \widehat{s}_{10n}^{(3)})$ is also 0.8-close to $\boldsymbol{w}$. (This is because (1) we have $\widehat{s}_j^{(3)} = s_j^{(3)}$ when $s_j^{(3)} \neq \bot$ and (2) we have $\left|\{j \mid s_j^{(3)} = w_j \neq \bot\}\right| \geq 8n$ except with negligible probability.) Thus, except with negligible probability, the codeword $\boldsymbol{w}$, which is computed from $\boldsymbol{s}^{(2)}$, is identical with the codeword $\widehat{\boldsymbol{w}}$, which is computed from $\widehat{\boldsymbol{s}}^{(3)}$. Thus, we have $v = \widehat{v}$ except with negligible probability.

- In Case 2, $\boldsymbol{s}^{(2)}$ is not 0.9-close to any valid codeword. In this case, we have $v = \bot$. If $\widehat{\boldsymbol{s}}^{(3)}$ is not 0.8-close to any valid codeword, we have $\widehat{v} = \bot$. Thus, we assume that $\widehat{\boldsymbol{s}}^{(3)}$ is 0.8-close to a valid codeword $\widehat{\boldsymbol{w}} = (\widehat{w}_1, \ldots, \widehat{w}_{10n})$. Note that $\boldsymbol{s}^{(2)}$ is not 0.9-close to $\widehat{\boldsymbol{w}}$. In addition, since Subclaim 3 guarantees that we have $\left|\{j \mid s_j^{(3)} \neq \bot\}\right| \geq 9n$ except with negligible probability, $\widehat{\boldsymbol{s}}^{(3)}$ and $\boldsymbol{s}^{(3)}$ are 0.9-close except with negligible probability; thus, $\widehat{\boldsymbol{w}}$ can be computed from $\boldsymbol{s}^{(3)}$ except with negligible probability. Then, as in the proof of Subclaim 3, we can show that except with negligible probability, there exists $j \in \Gamma$ such that $s_j^{(2)} \neq \widehat{w}_j$. Thus, we have $\widehat{v} = \bot$ except with negligible probability.

Therefore, at the end of the first right session in which the committed values of CECom_{i+2} are extracted, $H_{1.5}^b(n, z)$ outputs fail with at most negligible probability.

Next, we show that for any $k \in \mathbb{N}$, if $H_{1.5}^b(n, z)$ does not output fail at the end of the k'-th session for every $k' < k$ on the thread, $H_{1.5}^b(n, z)$ does not output fail at the end of the k-th session except with negligible probability. Since $H_{1.5}^b(n, z)$ does not output fail until $\mathcal{A}$ completes the k-th session, the view of $\mathcal{A}$ on the thread in $H_{1.5}^b(n, z)$ is identical with that of $\mathcal{A}$ in $H_1^b(n, z)$ until $\mathcal{A}$ completes the k-th session. We can therefore use the same argument as above to show that except with negligible probability, $H_{1.5}^b(n, z)$ does not output fail at the end of the k-th session.

We therefore conclude that $H_{1.5}^b(n, z)$ outputs fail with at most negligible probability. □

Finally, we prove Claim 6.

Proof (of Claim 6). $H_2^b(n, z)$ and $H_3^b(n, z)$ differ only in the committed values of Com_{i+1}. Since the running time of $H_2^b(n, z)$ and that of $H_3^b(n, z)$ are $\mathsf{poly}(n^{\log n}) \cdot T_i(n) \ll T_{i+1}(n)$ after the start of Stage 2 on the left, we can prove Claim 6 from the hiding property of Com_{i+1} (by considering Γ of the left session and the answers of $\mathcal{O}$ for some right sessions as non-uniform advice). Here, we use the fact that Com_{i+1} is a *2-round* commitment scheme. This fact enables us to rewind $\mathcal{A}$ in the right sessions of $H_2^b(n, z)$ without breaking the hiding property of Com_{i+1}.

Formally, assume for contradiction that there exists a polynomial $p(\cdot)$ such that for infinitely many n, there exists $z \in \{0,1\}^*$ such that $\mathsf{H}_3^b(n, z)$ can be distinguished from $\mathsf{H}_2^b(n, z)$ with probability at least $1/p(n)$. In the following, we fix any such n and z. Recall that both in $H_2^b(n, z)$ and $H_3^b(n, z)$, $\mathcal{A}$ is not rewound before Stage 2 starts in the left session. Let ρ be a prefix of the transcript of $H_2^b(n, z)$ and $H_3^b(n, z)$ such that immediately after ρ, Stage 2 starts on the left. Let prefix_ρ be the event that a prefix of the transcript is ρ. From an average

argument, with probability $1/2p(n)$ over the choice of ρ, we have

$$\left|\Pr\left[\mathcal{D}(\mathsf{H}_2^b(n,z))=1 \mid \mathsf{prefix}_\rho\right]-\Pr\left[\mathcal{D}(\mathsf{H}_3^b(n,z))=1 \mid \mathsf{prefix}_\rho\right]\right| \geq \frac{1}{2p(n)} \tag{2}$$

for a PPT distinguisher $\mathcal{D}$. Then, from strong computational binding property of $\mathsf{CCACom}_{i+3}^{1:1}$, with probability $1/2p(n)-\mathsf{negl}(n)$ over the choice of ρ, we have Equation (2) and ρ uniquely determines the committed value Γ of the Stage 1 commitment of the left session. We fix any such ρ. Let $\boldsymbol{s_1},\ldots,\boldsymbol{s_k}$ be the committed values of the Stage 2 commitments of the right sessions of which Stage 2 is contained in ρ.

Let us consider the following $T_{i+1}(n)$-time adversary $\mathcal{B}$ against the hiding property of Com_{i+1}. On non-uniform advice ρ, Γ and $\boldsymbol{s_1},\ldots,\boldsymbol{s_k}$, adversary $\mathcal{B}$ internally invokes $\mathcal{A}$ and simulates $H_2^b(n,z)$ after ρ as follows.

- The right sessions are simulated honestly except that in the sessions of which Stage 2 completes before the start of Stage 2 of the left session, the answers of $\mathcal{O}$ are computed by using non-uniform advice $\boldsymbol{s_1},\ldots,\boldsymbol{s_k}$.
- The left session is simulated honestly except that in Stage 2, $\mathcal{B}$ sends $(s_j)_{j\notin\Gamma}$ and $(0,\ldots,0)$ to the external Com_{i+1} committer and forwards $9n$ commitments from the external committer to $\mathcal{A}$ (the committed values are either $(s_j)_{j\notin\Gamma}$ or $(0,\ldots,0)$). When $\mathcal{B}$ rewinds $\mathcal{A}$ during the simulation of $H_2^b(n,z)$ and $\mathcal{A}$ requires new Stage 2 commitments, $\mathcal{B}$ does the same thing again by receiving new Com_{i+1} commitments from the external committer.[10]

If the running time of the simulated $H_2^b(n,z)$ exceeds $4p(n)T(n) \ll T_{i+1}(n)$, where $T(n)=\mathsf{poly}(n^{\log n})\cdot T_i(n)$ is the expected running time of $H_2^b(n,z)$, $\mathcal{B}$ outputs fail_1. Otherwise, $\mathcal{B}$ outputs whatever the simulated $H_2^b(n,z)$ outputs.

We reach a contradiction by showing that $\mathcal{B}$ breaks the hiding property of Com_{i+1}. From its construction, $\mathcal{B}$ either perfectly simulates $H_2^b(n,z)$ (when $\mathcal{B}$ receives commitments to $(s_j)_{j\notin\Gamma}$) or perfectly simulates $H_3^b(n,z)$ (when $\mathcal{B}$ receives commitments to $(0,\ldots,0)$). In addition, from Marcov's inequality, $\mathcal{B}$ outputs fail_1 with probability at most $1/4p(n)$. Thus, from our assumption, the value that $\mathcal{B}$ outputs when $\mathcal{B}$ receives commitments to $(0,\ldots,0)$ can be distinguished with probability at least $1/4p(n)$ from the value that $\mathcal{B}$ outputs when $\mathcal{B}$ receives commitments to $(s_j)_{j\notin\Gamma}$. □

5.2 Proof of Proposition 2

Like the robustness of previous CCA-secure commitment schemes [CLP10,LP12], the robustness of CCACom_i can be shown by using the techniques in the proof of its CCA security.

Proof (of Proposition 2). We show that there exists a PQT simulator $\mathcal{S}$ such that for any $T_i(n)$-time adversary $\mathcal{A}$ and any κ-round PPT ITM B, the following are computationally indistinguishable.

[10] Since Com_{i+2} is a 2-round commitment, we can assume without loss of generality that $\mathcal{A}$ always receives the second-round message of Com_{i+1} immediately after $\mathcal{A}$ sends the first-round message of Com_{i+1}. Thus, $\mathcal{B}$ can always simulate Stage 2 by receiving new Com_{i+1} commitments.

– $\{\mathsf{output}_{B,\mathcal{A}^{\mathcal{O}}}[\langle B(y), \mathcal{A}^{\mathcal{O}}(z)\rangle(1^n, x)]\}_{n\in\mathbb{N},x,y,z\in\{0,1\}^n}$
– $\{\mathsf{output}_{B,\mathcal{S}^{\mathcal{A}}}[\langle B(y), \mathcal{S}^{\mathcal{A}}(z)\rangle(1^n, x)]\}_{n\in\mathbb{N},x,y,z\in\{0,1\}^n}$

Given oracle access to $\mathcal{A}$, simulator $\mathcal{S}$ simulates the interaction between B and $\mathcal{A}^{\mathcal{O}}$ as follows. On the left, $\mathcal{S}$ forwards messages from B to $\mathcal{A}$ and forwards those from $\mathcal{A}$ to B. On the right, $\mathcal{S}$ simulates each session between $\mathcal{A}$ and $\mathcal{O}$ honestly except for the following.

– In Stage 3, $\mathcal{S}$ extracts $\widehat{\boldsymbol{s}}^{(3)} = (\widehat{s}_1^{(3)}, \ldots, \widehat{s}_{10n}^{(3)})$ from the CECom_{i+1} commitments *without rewinding the left interaction* by using the technique of [LP09]. That is, $\mathcal{S}$ extracts $\widehat{\boldsymbol{s}}^{(3)}$ by using a slot that does not contain any message of the left interaction.[11] There must exist at least three such slots, since CECom_{i+1} in Stage 3 has $(\kappa + 3)$ slots.
– At the end of the session, $\mathcal{S}$ checks whether the following conditions hold: (1) $\widehat{\boldsymbol{s}}^{(3)}$ is 0.8-close to a valid codeword $\widehat{\boldsymbol{w}} = (\widehat{w}_1, \ldots, \widehat{w}_{10n})$ and (2) for every $j \in \Gamma$ (where Γ is the value that $\mathcal{S}$ sends to $\mathcal{A}$ in Stage 4), $\widehat{w}_j$ is equal to the share that was revealed in Stage 5. If both conditions hold, $\mathcal{S}$ recovers $\widehat{v}$ from $\widehat{\boldsymbol{w}}$ and returns $\widehat{v}$ to $\mathcal{A}$. Otherwise, $\mathcal{S}$ returns $\widehat{v} := \bot$.

We show that $\mathcal{S}$ correctly simulates the interaction between $\mathcal{B}$ and $\mathcal{A}^{\mathcal{O}}$. First, as in the proof of Subclaim 3, we can show that when $\mathcal{A}^{\mathcal{O}}$ interacts with B, in any right session between $\mathcal{A}$ and $\mathcal{O}$, $\mathcal{A}$ cheats with at most negligible probability (we use the hiding property of $\mathsf{CCACom}_{i+3}^{1:1}$ instead of one-one CCA-secure property). Then, as in the proof of Claim 5, we can show that the view of the internal $\mathcal{A}$ (in $\mathcal{S}$) is statistically close to the view of $\mathcal{A}^{\mathcal{O}}$ that interacts with B. □

6 One-One CCA Security for Long Tags from CCA Security for Short Tags

In this section, we consider a transformation from any CCA-secure commitment scheme for tags of length $t(n)$ to a one-one CCA-secure commitment scheme for tags of length $2^{t(n)-1}$. The transformation is essentially the same as those in [LPV08], which shows a transformation from any concurrent NM commitment scheme for short tags to a NM commitment scheme for long tags.

Lemma 3. *Let $\epsilon < 1$ be a positive constant, and assume the existence of one-way functions that are secure against 2^{n^ϵ}-time adversaries. Let $r(\cdot)$ and $t(\cdot)$ be arbitrary functions such that $t(n) \leq O(\log n)$, let $T_i(n) \stackrel{\text{def}}{=} 2^{(\log n)^{(2/\epsilon)^{10i+1}}}$ for $i \in \mathbb{N}$, and let CCACom_{i+1} be an $r(n)$-round commitment scheme that satisfies the following for any $T_{i+1}(n)$-time adversary.*

– *Statistical binding property, and*

[11] Recall that in the rewinding strategy of [PV08], the extractor rewinds a single slot that contains only "small" number of other slots. (Such a slot always exist when the number of slots is at least 3.) Then, $\mathcal{S}$ extracts $\widehat{\boldsymbol{s}}^{(3)}$ by rewinding a slot that (1) contains only "small" number of other slots and (2) does not contain any left message.

Commit Phase

The committer C and the receiver R receive common inputs 1^n and $\mathsf{id} \in \{0,1\}^{2^{t(n)-1}}$. To commit to $v \in \{0,1\}^n$, the committer C chooses random $v_1, \ldots, v_{2^{t(n)-1}}$ such that $v = \bigoplus_j v_j$, and for each $j \in [2^{t(n)-1}]$ in parallel, $\mathcal{C}$ commits to v_j by using CCACom_{i+1} with tag (j, id_j), where id_j is the j-th bit of id.

Decommit Phase

C sends v to R and decommits all the CCACom_{i+1} commitments.

Fig. 7. One-one CCA-secure commitment $\mathsf{CCACom}_i^{1:1}$.

– *CCA security for tags of length* $t(n)$.

Then, there exists an $r(n)$*-round commitment scheme* $\mathsf{CCACom}_i^{1:1}$ *that satisfies the following for any* $T_i(n)$*-time adversary.*

– *Statistical binding property, and*
– *One-one CCA security for tags of length* $2^{t(n)-1}$.

If CCACom_{i+1} *uses the underlying one-way function only in a black-box way, then* $\mathsf{CCACom}_i^{1:1}$ *uses the underlying one-way function only in a black-box way.*

Proof. $\mathsf{CCACom}_i^{1:1}$ is shown in Figure 7. The statistical binding property follows from that of CCACom_{i+1}. Thus, it remains to show that $\mathsf{CCACom}_i^{1:1}$ is one-one CCA secure.

We show that for any $T_i(n)$-time adversary $\mathcal{A}$ that interacts with $\mathcal{O}$ only in a single session, the following are computationally indistinguishable:

– $\{\mathsf{IND}_0(\mathsf{CCACom}_i^{1:1}, \mathcal{A}, n, z)\}_{n\in\mathbb{N}, z\in\{0,1\}^*}$
– $\{\mathsf{IND}_1(\mathsf{CCACom}_i^{1:1}, \mathcal{A}, n, z)\}_{n\in\mathbb{N}, z\in\{0,1\}^*}$

Without loss of generality, we assume that the tag that $\mathcal{A}$ chooses in the right session is always different from the tag that $\mathcal{A}$ chooses in the left session.

Assume for contradiction that there exist a PPT distinguisher $\mathcal{D}$ and a polynomial $p(\cdot)$ such that for infinitely many n, there exists $z \in \{0,1\}^*$ such that $\mathcal{D}$ distinguishes $\mathsf{IND}_1(\mathsf{CCACom}_i^{1:1}, \mathcal{A}, n, z)$ from $\mathsf{IND}_0(\mathsf{CCACom}_i^{1:1}, \mathcal{A}, n, z)$ with probability at least $1/p(n)$. In the following, we fix any such n and z.

Let us consider the following $T_{i+1}(n)$-time adversary $\mathcal{B}$ against CCA security of CCACom_{i+1}. $\mathcal{B}$ internally invokes $\mathcal{A}$ and simulates $\mathrm{IND}_0(\mathsf{CCACom}_i^{1:1}, \mathcal{A}, n, z)$ for $\mathcal{A}$ as follows. First, $\mathcal{B}$ chooses random $j^* \in [2^{t(n)-1}]$, and for each $j \in [2^{t(n)-1}] \setminus \{j^*\}$, $\mathcal{B}$ chooses random $v_j \in \{0,1\}^n$. Then, in the left session, when $\mathcal{A}$ outputs challenge values $m_0, m_1 \in \{0,1\}^n$ and tag $\mathsf{id} = (\mathsf{id}_1, \ldots, \mathsf{id}_{2^{t(n)-1}})$, $\mathcal{B}$ sets $v_{j^*}^{(b)} := m_b \oplus \bigoplus_{j\neq j^*} v_j$ for each $b \in \{0,1\}$ and sends challenge

$v_{j^*}^{(0)}, v_{j^*}^{(1)}$ and tag $(j^*, \mathsf{id}_{j^*}) \in \{0,1\}^{t(n)}$ to the external left committer. When $\mathcal{B}$ receives a CCACom_{i+1} commitment from the left committer (the committed value is either $v_{j^*}^{(0)}$ or $v_{j^*}^{(1)}$), $\mathcal{B}$ forwards it to $\mathcal{A}$. At the same time, $\mathcal{B}$ generates CCACom_{i+1} commitments to $(v_j)_{j \neq j^*}$ and sends them to $\mathcal{A}$. In the right session, when $\mathcal{A}$ outputs tag $\widetilde{\mathsf{id}}$, $\mathcal{B}$ terminates and outputs fail if $\mathsf{id}_{j^*} = \widetilde{\mathsf{id}}_{j^*}$. Otherwise, $\mathcal{B}$ forwards a $\mathsf{CCACom}_i^{1:1}$ commitment from $\mathcal{A}$ to $\mathcal{O}$ as $2^{t(n)-1}$ parallel commitments of CCACom_{i+1} with tags $\{(j, \widetilde{\mathsf{id}}_j)\}_{j=1}^{2^{t(n)-1}}$. Then, $\mathcal{B}$ receives $(v_1, \ldots, v_{2^{t(n)-1}})$ from $\mathcal{O}$, and if $v_j \neq \bot$ for all $j \in [2^{t(n)-1}]$, $\mathcal{B}$ returns $v := \bigoplus_j v_j$ to $\mathcal{A}$. If $v_j = \bot$ for some j, $\mathcal{B}$ returns $\bot$ to $\mathcal{A}$. Finally, $\mathcal{B}$ outputs $\mathcal{D}(y)$, where y is the output of the simulated $\text{IND}_0(\mathsf{CCACom}_i^{1:1}, \mathcal{A}, n, z)$.

We reach a contradiction by showing that $\mathcal{B}$ breaks the CCA security of CCACom_{i+1} with non-negligible probability. For $b \in \{0,1\}$, let Abort_b be the event that $\mathcal{B}$ outputs fail in $\text{IND}_b(\mathsf{CCACom}_{i+1}, \mathcal{B}, n, z)$. Then, from the hiding property of CCACom_{i+1}, we have

$$|\Pr[\mathsf{Abort}_0] - \Pr[\mathsf{Abort}_1]| \leq \mathsf{negl}(n) \ .$$

In addition, since we always have $\mathsf{id} \neq \widetilde{\mathsf{id}}$, for each $b \in \{0,1\}$ we have

$$\Pr[\neg\mathsf{Abort}_b] \geq \frac{1}{2^{t(n)-1}} \geq \frac{1}{\mathsf{poly}(n)} \ .$$

If $\mathcal{B}$ does not output fail, $\mathcal{B}$ perfectly simulates either $\text{IND}_0(\mathsf{CCACom}_i^{1:1}, \mathcal{A}, n, z)$ or $\text{IND}_1(\mathsf{CCACom}_i^{1:1}, \mathcal{A}, n, z)$. In addition, if $\mathcal{B}$ does not output fail, each tag $(j, \widetilde{\mathsf{id}}_j)$, which was used on the right, is different from the tag (j^*, id_{j^*}), which was used on the left. Thus, we have

$$\begin{aligned}
&\left| \begin{array}{l} \Pr[\mathsf{IND}_0(\mathsf{CCACom}_{i+1}, \mathcal{B}, n, z) = 1] \\ -\Pr[\mathsf{IND}_1(\mathsf{CCACom}_{i+1}, \mathcal{B}, n, z) = 1] \end{array} \right| \\
&= \left| \begin{array}{l} \Pr[\mathsf{IND}_0(\mathsf{CCACom}_{i+1}, \mathcal{B}, n, z) = 1 \bigwedge \neg\mathsf{Abort}_0] \\ -\Pr[\mathsf{IND}_1(\mathsf{CCACom}_{i+1}, \mathcal{B}, n, z) = 1 \bigwedge \neg\mathsf{Abort}_1] \end{array} \right| \\
&\geq \left| \begin{array}{l} \Pr[\mathsf{IND}_0(\mathsf{CCACom}_{i+1}, \mathcal{B}, n, z) = 1 \mid \neg\mathsf{Abort}_0] \\ -\Pr[\mathsf{IND}_1(\mathsf{CCACom}_{i+1}, \mathcal{B}, n, z) = 1 \mid \neg\mathsf{Abort}_1] \end{array} \right| \times \frac{1}{\mathsf{poly}(n)} \\
&= \left| \begin{array}{l} \Pr\left[\mathcal{D}(\mathsf{IND}_0(\mathsf{CCACom}_i^{1:1}, \mathcal{A}, n, z)) = 1\right] \\ -\Pr\left[\mathcal{D}(\mathsf{IND}_1(\mathsf{CCACom}_i^{1:1}, \mathcal{A}, n, z)) = 1\right] \end{array} \right| \times \frac{1}{\mathsf{poly}(n)} \\
&\geq \frac{1}{p(n)\mathsf{poly}(n)} \ .
\end{aligned}$$

□

7 Constant-Round Black-Box Composable Protocol

In this section, we show a constant-round black-box construction of a general MPC protocol that satisfies angel-based UC security. Roughly speaking, the framework of angel-based UC

security (called $\mathcal{H}$-EUC framework) is the same as the UC framework except that both the adversary and the environment in the real and the ideal worlds have access to a super-polynomial-time angel $\mathcal{H}$.

To construct our protocol, we use the following theorem, which we obtain by combining Lemmas 1, 2, and 3.

Theorem 1. *Assume the existence of one-way functions that are secure against sub-exponential-time adversaries. Then, for any constant $\kappa \in \mathbb{N}$, there exists a constant-round commitment scheme that is CCA secure and κ-PQT-robust. This commitment scheme uses the underlying one-way functions only in a black-box way.*

We additionally use the following results of [CLP10] and [LP12].

Let $\langle C, R\rangle$ be any $r_{cca}(n)$-round commitment scheme that is CCA secure and κ-robust for any constant κ, $\langle S, R\rangle$ be any $r_{ot}(n)$-round semi-honest OT protocol, and $\mathcal{H}$ be an angel that breaks $\langle C, R\rangle$ essentially in the same way as the committed-value oracle of $\langle C, R\rangle$ does. Then, Lin and Pass [LP12] showed that there exists a black-box $O(\max(r_{ot}(n), r_{cca}(n)))$-round protocol that securely realizes the ideal OT functionality $\mathcal{F}_{OT}$ in the $\mathcal{H}$-EUC framework. By using essentially the same security proof as that of [LP12], we can show that even when $\langle C, R\rangle$ is CCA secure and only κ-PQT-robust for a sufficiently large κ, the protocol of [LP12] is still secure if $\langle S, R\rangle$ is secure against any PQT adversary.[12] Thus, we have the following theorem from [LP12].

Theorem 2. *Assume the existence of an $r_{cca}(n)$-round commitment scheme $\langle C, R\rangle$ that is CCA secure and κ-PQT-robust for a sufficiently large κ, and assume the existence of an $r_{ot}(n)$-round semi-honest oblivious transfer protocol $\langle S, R\rangle$ that is secure against any PQT adversary. Then, there exists an $O(\max(r_{cca}(n), r_{ot}(n)))$-round protocol that $\mathcal{H}$-EUC-realizes $\mathcal{F}_{OT}$. This protocol uses $\langle C, R\rangle$ and $\langle S, R\rangle$ only in a black-box way.*

In [CLP10], Canetti et al. showed the following.

Theorem 3 ([CLP10]). *For every well-formed functionality $\mathcal{F}$, there exists a constant-round $\mathcal{F}_{OT}$-hybrid protocol that $\mathcal{H}$-EUC-realizes $\mathcal{F}$.*

Then, by combining Theorems 1, 2, and 3, we obtain the following theorem.

Theorem 4. *Assume the existence of one-way functions that are secure against sub-exponential-time adversaries and constant-round semi-honest oblivious transfer protocols that are secure against quasi-polynomial-time adversaries. Then, there exists an angel $\mathcal{H}$ such that for every well-formed functionality $\mathcal{F}$, there exists a constant-round protocol that $\mathcal{H}$-EUC-realizes $\mathcal{F}$. This protocol uses the underlying one-way functions and oblivious transfer protocols only in a black-box way.*

[12] This is because κ-PQT-robustness guarantees that the committed-value oracle is useless in attacking any κ-round protocol *if the protocol is* PQT*-secure*.

8 Negative Result on CCA-Secure Commitments

In this section, we show that if we use a black-box reduction and assume only falsifiable polynomial-time hardness assumptions, we cannot prove the CCA security of any constant-round commitment scheme.

8.1 Preliminaries

First, we give a definition of the falsifiable polynomial-time hardness assumptions, which is essentially the same as the definition of the falsifiable assumptions of [GW11].

Definition 2. *A* falsifiable polynomial-time hardness assumption *is a pair* (Ch, c)*, where* Ch *is a* PPT *ITM called* challenger *and* c *is a constant such that* $0 \leq c < 1$*. For any (possibly super-polynomial-time) adversary* $\mathcal{A}$*, we say that* $\mathcal{A}$ breaks *an assumption* (Ch, c) *if there exists a polynomial* $p(\cdot)$ *such that for infinitely many* n*, we have*

$$\Pr\left[\mathsf{output}_{Ch}[\langle Ch, \mathcal{A}\rangle(1^n)] = 1\right] \geq c + 1/p(n) \ .$$

The assumption (Ch, c) *is true if and only if no* PPT *adversary can break* (Ch, c)*.* $\Diamond$

Next, we recall the definition of black-box reductions from [GW11]. For concreteness, we consider only black-box reductions showing the CCA security of a commitment scheme.

Definition 3. *A* black-box reduction *is a* PPT *oracle machine. We say that* a black-box reduction $\mathcal{R}$ shows the CCA security of a commitment scheme $\langle C, R\rangle$ based on an assumption (Ch, c) *if for any (possibly super-polynomial-time) adversary* $\mathcal{A}$ *that breaks the CCA security of* $\langle C, R\rangle$*,* $\mathcal{R}^{\mathcal{A}}$ *breaks the assumption* (Ch, c)*.* $\Diamond$

8.2 Our Negative Result

Theorem 5. *Let* $\langle C, R\rangle$ *be a* $o(\log n/\log\log n)$*-round commitment scheme. If there exists a black-box reduction* $\mathcal{R}$ *showing that* $\langle C, R\rangle$ *is CCA secure based on a falsifiable polynomial-time hardness assumption* (Ch, c)*, then the assumption* (Ch, c) *is false.*

In the proof of Theorem 5, we use a technique that Canetti et al. [CKPR02] used to show the impossibility of $o(\log n/\log\log n)$-round black-box concurrent zero-knowledge proofs for non-trivial languages.

First, we recall this technique (for details, see [CKPR02]). For any $k(n) = o(\log n/\log\log n)$-round zero-knowledge proof $\langle P, V\rangle$ and for any PPT black-box simulator $\mathcal{S}$, Canetti et al. constructed a family of cheating verifiers $\{V_{g,h}\}_{g\in G, h\in H}$, where G and H are families of hash functions. Each $V_{g,h}$ executes n^2 sessions of $\langle P, V\rangle$ in a specific schedule R_{n^2} (see Figure 8). The schedule R_{n^2} consists of n *recursive blocks*, and each recursive block consists of n sessions. In each session, $V_{g,h}$ interacts with the prover in the same way as the honest verifier does except that (1) randomness used in this session is determined by using h and a prefix of the transcript (called the *block prefix*) and (2) $V_{g,h}$ decides whether to abort this session by using

For any $m \leq n^2$, the schedule R_m is recursively defined as follows.

1. If $m < n$, sessions $1, \ldots, m$ are executed sequentially until they are all completed.
2. Otherwise, for $j = 1, \ldots, k(n)$:
   Message exchange: Each of the first n sessions exchanges two messages.
   Recursive call: If $j < k(n)$, the scheduling $\mathsf{R}_{\lceil (m-n)/(k(n)-1) \rceil}$ is applied recursively on $\lceil (m-n)/(k(n)-1) \rceil$ new sessions.

The set of n sessions that is explicitly executed during the message exchange phase is called a *recursive block*.

Fig. 8. Schedule R_m [CKPR02].

g and a prefix of the transcript (called the *iteration prefix*)[13]. $V_{g,h}$ accepts a recursive block if and only if $V_{g,h}$ accepted at least $n^{1/2}/4$ sessions in this recursive block. If $V_{g,h}$ rejects a recursive block, $V_{g,h}$ halts. If $V_{g,h}$ accepts all n recursive blocks, $V_{g,h}$ outputs accept. Then, Canetti et al. showed that with overwhelming probability over the choice of $g \in G$, $h \in H$, and randomness of $\mathcal{S}$, if $\mathcal{S}^{V_{g,h}}$ outputs an accepted transcript (i.e., a transcript in which $V_{g,h}$ outputs accept), there exists a session that was accepted but was not "rewound" in the execution of $\mathcal{S}^{V_{g,h}}$ (since otherwise the running time of $\mathcal{S}$ becomes super-polynomial).[14]

Next, we prove Theorem 5. In the proof, we use the idea behind $\{V_{g,h}\}_{g \in G, h \in H}$.

Proof (of Theorem 5). Let $\mathcal{R}$ be a black-box reduction showing CCA security of $\langle C, R \rangle$ based on a falsifiable polynomial-time hardness assumption (Ch, c). Then, for any (possibly super-polynomial-time) adversary $\mathcal{A}$ that breaks CCA security of $\langle C, R \rangle$, there exists a polynomial $p(\cdot)$ such that for infinitely many n, we have

$$\Pr\left[\mathsf{output}_{Ch}[\langle Ch, \mathcal{R}^{\mathcal{A}} \rangle(1^n)] = 1\right] \geq c + 1/p(n) \ .$$

First, let us consider the following family $\{\mathcal{A}^1_{g,h}\}_{g \in G, h \in H}$ of super-polynomial-time adversaries against CCA security of $\langle C, R \rangle$. In the left session, $\mathcal{A}^1_{g,h}$ honestly interacts with the left committer with randomly chosen challenge values. In the right sessions, $\mathcal{A}^1_{g,h}$ interacts with $\mathcal{O}$ in n^2 sessions in the schedule R_{n^2}. In the i-th right session, $\mathcal{A}^1_{g,h}$ chooses random $v_i \in \{0,1\}^n$ and commits to v_i. The randomness used in this session is generated as in [CKPR02] (i.e., using h and the block prefix) and $\mathcal{A}^1_{g,h}$ decides whether to abort this session as in [CKPR02] (i.e., using g and the iteration prefix). $\mathcal{A}^1_{g,h}$ accepts the i-th session if and only if $\mathcal{O}$ returns

[13] Roughly speaking, the block prefix is defined so that whenever $\mathcal{S}$ rewinds $V_{g,h}$ in a recursive block, the randomness used in higher-level recursive blocks is completely changed (and thus $\mathcal{S}$ needs to rewind $V_{g,h}$ in these recursive blocks as well), and the iteration prefix is defined so that whenever $\mathcal{S}$ rewinds $V_{g,h}$ in a session, $V_{g,h}$ aborts this session with a fixed probability (and thus $\mathcal{S}$ needs to rewind $V_{g,h}$ many times until $\mathcal{S}$ gets an accepted transcript of this session).

[14] Formally, Canetti et al. showed that in the execution of $\mathcal{S}^{V_{g,h}}$, there exists a prefix called a *useful block prefix*. For details, see [CKPR02].

v_i at the end of this session. $\mathcal{A}^1_{g,h}$ accepts a recursive block if and only if $\mathcal{A}^1_{g,h}$ accepted at least $n^{1/2}/4$ sessions in this recursive block. If $\mathcal{A}^1_{g,h}$ rejects a recursive block, $\mathcal{A}^1_{g,h}$ halts. If $\mathcal{A}^1_{g,h}$ accepts all n recursive blocks, $\mathcal{A}^1_{g,h}$ computes the committed value v in the left session by brute force and outputs v. (Thus, $\mathcal{A}^1_{g,h}$ breaks CCA security of $\langle C, R\rangle$.)

Next, let us consider the following family $\{\mathcal{A}^2_{g,h}\}_{g\in G, h\in H}$ of PPT adversaries. $\mathcal{A}^2_{g,h}$ is the same as $\mathcal{A}^1_{g,h}$ except that if $\mathcal{A}^2_{g,h}$ accepts all n blocks, $\mathcal{A}^2_{g,h}$ outputs a random string v.

Then, we show that with overwhelming probability over the choice of g and h, the following are computationally indistinguishable:

- $\{\mathsf{output}_{Ch}[\langle Ch, \mathcal{R}^{\mathcal{A}^1_{g,h}}\rangle(1^n)]\}_{n\in\mathbb{N}}$
- $\{\mathsf{output}_{Ch}[\langle Ch, \mathcal{R}^{\mathcal{A}^2_{g,h}}\rangle(1^n)]\}_{n\in\mathbb{N}}$

Since $\mathcal{A}^1_{g,h}$ and $\mathcal{A}^2_{g,h}$ differ only in the last message v, we can show the indistinguishability by showing that in the interaction with Ch, $\mathcal{R}^{\mathcal{A}^1_{g,h}}$ does not receive v from $\mathcal{A}^1_{g,h}$. Assume for contradiction that $\mathcal{R}^{\mathcal{A}^1_{g,h}}$ receives v from $\mathcal{A}^1_{g,h}$. Then, since $\mathcal{A}^1_{g,h}$ outputs v only if $\mathcal{A}^1_{g,h}$ accepts all n recursive blocks, and since the running time of Ch and that of $\mathcal{R}$ are polynomially bounded, from the analysis in [CKPR02], there exists a session that was accepted but was not rewound. Then, since $\mathcal{A}^1_{g,h}$ accepts a session only if $\mathcal{A}^1_{g,h}$ received the committed value of this session, $\mathcal{R}$ must have sent the committed value of this session to $\mathcal{A}^1_{g,h}$ without rewinding $\mathcal{A}^1_{g,h}$. Since the running time of Ch and that of $\mathcal{R}$ are polynomially bounded, this contradicts the hiding property of $\langle C, R\rangle$ (i.e., we can use Ch and $\mathcal{R}$ to break the hiding property of $\langle C, R\rangle$). We thus conclude that $\mathcal{R}$ does not receive v from $\mathcal{A}^1_{g,h}$, and therefore we conclude that the indistinguishability holds.

For every $g \in G$ and $h \in H$, since $\mathcal{A}^1_{g,h}$ breaks CCA security of $\langle C, R\rangle$, there exists a polynomial $p(\cdot)$ such that for infinitely many n, we have

$$\Pr\left[\mathsf{output}_{Ch}[\langle Ch, \mathcal{R}^{\mathcal{A}^1_{g,h}}\rangle(1^n)] = 1\right] \geq c + 1/p(n) \ .$$

Then, from the above indistinguishability, for random $g \in G$ and $h \in H$, we have

$$\begin{aligned}
&\Pr\left[\mathsf{output}_{Ch}[\langle Ch, \mathcal{R}^{\mathcal{A}^2_{g,h}}\rangle(1^n)] = 1\right] \\
&\geq c + 1/p(n) - \mathsf{negl}(n) \\
&\geq c + 1/\mathsf{poly}(n)
\end{aligned}$$

with overwhelming probability over the choice of g and h. Since the running time of $\mathcal{R}$ and that of $\mathcal{A}^2_{g,h}$ are polynomially bounded, this fact implies that the assumption (Ch, c) is false. □

References

[BS05] Boaz Barak and Amit Sahai. How to play almost any mental game over the net—concurrent composition via super-polynomial simulation. In *FOCS*, pages 543–552, 2005.

[Can01] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *FOCS*, pages 136–145, 2001.

[CDSMW08] Seung Geol Choi, Dana Dachman-Soled, Tal Malkin, and Hoeteck Wee. Black-box construction of a non-malleable encryption scheme from any semantically secure one. In *TCC*, pages 427–444, 2008.

[CDSMW09] Seung Geol Choi, Dana Dachman-Soled, Tal Malkin, and Hoeteck Wee. Simple, black-box constructions of adaptively secure protocols. In *TCC*, pages 387–402, 2009.

[CF01] Ran Canetti and Marc Fischlin. Universally composable commitments. In *CRYPTO*, pages 19–40, 2001.

[CGGM00] Ran Canetti, Oded Goldreich, Shafi Goldwasser, and Silvio Micali. Resettable zero-knowledge. In *STOC*, pages 235–244, 2000.

[CKL03] Ran Canetti, Eyal Kushilevitz, and Yehuda Lindell. On the limitations of universally composable two-party computation without set-up assumptions. In *EUROCRYPT*, pages 68–86, 2003.

[CKPR02] Ran Canetti, Joe Kilian, Erez Petrank, and Alon Rosen. Black-box concurrent zero-knowledge requires (almost) logarithmically many rounds. *SIAM J. Comput.*, 32(1):1–47, 2002.

[CLOS02] Ran Canetti, Yehuda Lindell, Rafail Ostrovsky, and Amit Sahai. Universally composable two-party and multi-party secure computation. In *STOC*, pages 494–503, 2002.

[CLP10] Ran Canetti, Huijia Lin, and Rafael Pass. Adaptive hardness and composable security in the plain model from standard assumptions. In *FOCS*, pages 541–550, 2010.

[DDN00] Danny Dolev, Cynthia Dwork, and Moni Naor. Nonmalleable cryptography. *SIAM J. Comput.*, 30(2):391–437, 2000.

[GGJS12] Sanjam Garg, Vipul Goyal, Abhishek Jain, and Amit Sahai. Concurrently secure computation in constant rounds. In *EUROCRYPT*, pages 99–116, 2012.

[GLP+12] Vipul Goyal, Huijia Lin, Omkant Pandey, Rafael Pass, and Amit Sahai. Round-efficient concurrently composable secure computation via a robust extraction lemma. Cryptology ePrint Archive, Report 2012/652, 2012. `http://eprint.iacr.org/`.

[GMW87] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or a completeness theorem for protocols with honest majority. In *STOC*, pages 218–229, 1987.

[Goy11] Vipul Goyal. Constant round non-malleable protocols using one way functions. In *STOC*, pages 695–704, 2011.

[Goy13] Vipul Goyal. Non-black-box simulation in the fully concurrent setting. In *STOC*, pages 221–230, 2013.

[GW11] Craig Gentry and Daniel Wichs. Separating succinct non-interactive arguments from all falsifiable assumptions. In *STOC*, pages 99–108, 2011.

[Hai08] Iftach Haitner. Semi-honest to malicious oblivious transfer - the black-box way. In *TCC*, pages 412–426, 2008.

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

[IKLP06] Yuval Ishai, Eyal Kushilevitz, Yehuda Lindell, and Erez Petrank. Black-box constructions for secure computation. In *STOC*, pages 99–108, 2006.

[KMO14] Susumu Kiyoshima, Yoshifumi Manabe, and Tatsuaki Okamato. Constant-round black-box construction of composable multi-party computation protocol. In *TCC*, 2014. To appear.

[Lin11] Huijia Lin. *Concurrent Security*. PhD thesis, Cornell University, 2011.

[LP09] Huijia Lin and Rafael Pass. Non-malleability amplification. In *STOC*, pages 189–198, 2009.

[LP12] Huijia Lin and Rafael Pass. Black-box constructions of composable protocols without set-up. In *CRYPTO*, pages 461–478, 2012.

[LPV08] Huijia Lin, Rafael Pass, and Muthuramakrishnan Venkitasubramaniam. Concurrent non-malleable commitments from any one-way function. In *TCC*, pages 571–588, 2008.

[MMY06] Tal Malkin, Ryan Moriarty, and Nikolai Yakovenko. Generalized environmental security from number theoretic assumptions. In *TCC*, pages 343–359, 2006.

[MOSV06] Daniele Micciancio, Shien Jin Ong, Amit Sahai, and Salil P. Vadhan. Concurrent zero knowledge without complexity assumptions. In *TCC*, pages 1–20, 2006.

[Nao91] Moni Naor. Bit commitment using pseudorandomness. *J. Cryptology*, 4(2):151–158, 1991.

[Nao03] Moni Naor. On cryptographic assumptions and challenges. In *CRYPTO*, pages 96–109, 2003.

[Pas03] Rafael Pass. Simulation in quasi-polynomial time, and its application to protocol composition. In *EUROCRYPT*, pages 160–176, 2003.

[Pas04] Rafael Pass. Bounded-concurrent secure multi-party computation with a dishonest majority. In *STOC*, pages 232–241, 2004.

[PLV12] Rafael Pass, Huijia Lin, and Muthuramakrishnan Venkitasubramaniam. A unified framework for UC from only OT. In *ASIACRYPT*, pages 699–717, 2012.

[PRS02] Manoj Prabhakaran, Alon Rosen, and Amit Sahai. Concurrent zero knowledge with logarithmic round-complexity. In *FOCS*, pages 366–375, 2002.

[PS04] Manoj Prabhakaran and Amit Sahai. New notions of security: Achieving aniversal composability without trusted setup. In *STOC*, pages 242–251, 2004.

[PV08] Rafael Pass and Muthuramakrishnan Venkitasubramaniam. On constant-round concurrent zero-knowledge. In *TCC*, pages 553–570, 2008.

[PW09] Rafael Pass and Hoeteck Wee. Black-box constructions of two-party protocols from one-way functions. In *TCC*, pages 403–418, 2009.

[PW10] Rafael Pass and Hoeteck Wee. Constant-round non-malleable commitments from sub-exponential one-way functions. In *EUROCRYPT*, pages 638–655, 2010.

[Wee10] Hoeteck Wee. Black-box, round-efficient secure computation via non-malleability amplification. In *FOCS*, pages 531–540, 2010.