Journal of
**CRYPTOLOGY**

# On Black-Box Complexity of Universally Composable Security in the CRS Model*

Carmit Hazay

Faculty of Engineering, Bar-Ilan University, Ramat Gan, Israel
carmit.hazay@biu.ac.il

Muthuramakrishnan Venkitasubramaniam

University of Rochester, Rochester, NY14611, USA
muthuv@cs.rochester.edu

Communicated by Jonathan Katz.

**Abstract.** In this work, we study the intrinsic complexity of *black-box Universally Composable (UC) secure computation* based on *general assumptions*. We present a thorough study in various corruption modelings while focusing on achieving security in the common reference string (CRS) model. Our results involve the following:

- **Static UC secure computation.** Designing *the first* static UC oblivious transfer protocol based on public-key encryption and stand-alone semi-honest oblivious transfer. As a corollary, we obtain the first black-box constructions of UC secure computation assuming only two-round semi-honest oblivious transfer.
- **One-sided UC secure computation.** Designing adaptive UC two-party computation with single corruptions assuming public-key encryption with oblivious ciphertext generation.
- **Adaptive UC secure computation.** Designing adaptively secure UC commitment scheme assuming only public-key encryption with oblivious ciphertext generation. As a corollary, we obtain the first black-box constructions of adaptive UC secure computation assuming only (trapdoor) simulatable public-key encryption (as well as a variety of concrete assumptions). We remark that such a result was not known even under non-black-box constructions.

**Keywords.** UC secure computation, Black-box constructions, Oblivious transfer, UC commitments.

## 1. Introduction

Secure multi-party computation enables a set of parties to mutually run a protocol that computes some function $f$ on their private inputs, while preserving a number of security

---

properties. Two of the most important properties are privacy and correctness. The former implies data confidentiality; namely, nothing leaks by the protocol execution but the computed output. The later requirement implies that no corrupted party or parties can cause the output to deviate from the specified function. It is by now well known how to securely compute any efficient functionality [3,4,25,47,51] in various models and under the stringent simulation-based definitions (following the ideal/real paradigm). Security is typically proved with respect to two adversarial models, the semi-honest model (where the adversary follows the instructions of the protocol but tries to learn more than it should from the protocol transcript) and the malicious model (where the adversary follows an arbitrary polynomial-time strategy), and feasibility results are known in the presence of both types of attacks. The initial model considered for secure computation was of a static adversary where the adversary controls a fixed subset of the parties (who are called corrupted) before the protocol begins. In a stronger corruption model, the adversary is allowed to choose which parties to corrupt throughout the protocol execution and as a function of its view; such an adversary is called adaptive.

These feasibility results apply in most cases on stand-alone security, where a *single* set of parties run a *single* execution of the protocol. Moreover, the security of most cryptographic protocols proved in the stand-alone setting does not remain intact if many instances of the protocol are executed concurrently [39]. The strongest (but also the most realistic) setting for concurrent security is known by *Universally Composable* (UC) [4]. This setting considers the execution of an unbounded number of concurrent protocols in an arbitrary and adversarially controlled network environment. Unfortunately, stand-alone secure protocols typically fail to remain secure in the UC setting. In fact, without assuming some *trusted setup*, UC is impossible to achieve for most tasks [8,10,39]. Consequently, UC protocols have been constructed under various *trusted setup* assumptions in a long series of works; see [1,7,12,13,16,36,43] for few examples.

In this work, we are interested in understanding the intrinsic complexity of *UC secure computation*. Identifying the general assumptions required for a particular cryptographic task provides an abstraction of the functionality and the specific hardness that is exploited to obtain a secure realization of the task. The expressive nature of general assumptions allows the use of a large number of concrete assumptions of our choice, even one that may not have been considered at the time of designing the protocols. Constructions that are based on general assumptions are proved in two flavors:

| | |
|---|---|
| *Black-box usage* | A construction is black-box if it refers only to the input/output behavior of the underlying primitives. |
| *Non-black-box usage* | A construction is non-black-box if it uses the code computing the functionality of the underlying primitives. |

Typically, non-black-box constructions have been employed to demonstrate feasibility and derive the minimal assumptions required to achieve cryptographic tasks. Specifically, Lin et al. [43] provided a unified framework and minimal conditions under which UC security is feasible in a general setup. Moreover, the work of Damgard et al. [20] focused on identifying the necessary and sufficient assumptions for UC secure computation, in terms of both setup and computational assumptions. The former work identified the weakest assumptions in any setup known thus far, whereas the latter work identified tight upper and lower bounds on the hardness assumptions for the concrete common ref-

erence string and key registration models. Nevertheless, since both of these works rely on non-black-box techniques, an important theoretic question is whether or not non-black-box usage of the underlying primitives is necessary. Besides its theoretic importance, obtaining black-box constructions is related to the efficiency of the protocol as an undesirable effect of non-black-box constructions is that they are typically inefficient and unlikely to be implemented in practice.

Fortunately, in a line of works [24,27,32,49] the gap between what is achievable via non-black-box and black-box constructions under minimal assumptions has narrowed. More relevant to our context, the work of Ishai et al. [33] that provided the first black-box constructions of UC protocols in the static and adaptive settings assuming only one-way functions, in a model where all parties have access to an ideal oblivious transfer (OT) functionality. In the adaptive setting, the work of Choi et al. [6] provided a transformation from adaptively secure semi-honest oblivious transfer to one that is secure *in the stronger UC setting* against malicious adaptive adversaries while assuming that all parties have access to an ideal commitment functionality. These works make progress toward identifying the necessary minimal general computational assumptions in both the static and adaptive UC settings. In particular, it follows that, to answer the motivating question of identifying these minimal assumptions, it suffices to identify the minimal assumptions to realize the ideal oblivious transfer in the static setting as specified in [33] and the ideal commitment in the adaptive setting as specified in [6].

*Static setting* In the stand-alone (i.e., not UC) static setting, assuming only the existence of semi-honest oblivious transfer it has been shown in [27,28,32] how to construct secure multi-party computation protocols while relying on the underlying primitives in a black-box manner. In the UC setting, Canetti et al. [12] presented the first *non-black-box* constructions of static UC protocols assuming enhanced trapdoor permutations. In a later work, Choi et al. [6] (cf. Proposition 1) provided black-box constructions that are secure against static adversaries, where all parties have access to an ideal commitment functionality. This construction achieves a stronger security notion of straight-line simulation; however, it falls short of achieving static UC security (see more details in Sect. 3).

UC OT was studied in the influential paper by Peikert et al. [48], who presented a black-box framework in the local[1] common reference string (CRS) model for an oblivious transfer, based on dual-mode public-key encryption (PKE) schemes. Such PKE schemes can be concretely instantiated under the discrete Diffie–Hellman (DDH), quadratic residuosity (QR) and learning with errors (LWE) hardness assumptions. In a follow-up work, Choi et al. [11] present UC OT constructions in the global CRS model assuming DDH, $N$-residuosity and the Decision Linear Assumption (DLIN).

It is worth noting that while the works of Peikert et al. [48] and Choi et al. [11] provide abstractions of their assumptions, the assumptions themselves are not general enough to help understand the minimal assumptions required to achieve static UC security.

*Adaptive Setting* The only work that considered a single general assumption that implies adaptive UC security using non-black-box techniques is the result due to

---

[1]Namely, a distinct common reference string (CRS) per party.

Dachman-Soled et al. [16] that shows how to obtain adaptive UC commitments assuming simulatable PKE in the global CRS model.[2] Moreover, the best-known general assumptions required to achieve black-box UC security are adaptive semi-honest oblivious transfer and UC commitments [6,18]. Known minimal general assumptions that are required to construct these primitives are (trapdoor) simulatable PKE for adaptive semi-honest oblivious transfer [5] and mixed commitments for UC commitments [18] in the local CRS model. Finally, we remark that the commitment scheme of Damgard and Groth [15] based on Strong RSA is, in fact, an adaptive UC commitment in the global CRS model.

As such prior works leave the following important question open:

> *What are the minimal (general) assumptions required to construct UC protocols, given only* black-box *access to the underlying primitives?*

We note that this question is already well understood in the static setting when relaxing the black-box requirement. Namely, in [20] Damgård, Nielsen and Orlandi showed how to construct UC commitments assuming only semi-honest oblivious transfer in the global CRS model, while additionally assuming a preprocessing phase where the parties participate in a round-robin manner.[3] More recently, Lin et al. [44] have improved this result by removing any restricted preprocessing phase. In the same work, the authors showed how to achieve UC security in the global CRS model assuming only the existence of semi-honest oblivious transfer. In particular, this construction shows that static UC security can be achieved without assuming UC commitments when relying on non-black-box techniques.

## 1.1. *Our Results*

In this paper, we present a thorough study of black-box UC secure computation in the CRS model for different attack models; details follow. We note that our first and third results hold for the multi-party case, while the second result is for the two-party setting.

### 1.1.1. *Static UC Secure Computation*

Our first result is given in the static setting, where we demonstrate the feasibility of UC secure computation based on semi-honest oblivious transfer and extractable commitments. More concretely, we prove how to transform any statically semi-honest secure oblivious transfer into one that is secure in the presence of malicious adversaries, giving *only black-box access* to the underlying semi-honest oblivious transfer protocol. Our approach is inspired by the protocols from [28,42], where we observe that it is not required to use the full power of static UC commitments. Instead, we employ a weaker primitive that only requires straight-line input extractability. This weaker notion of security, denoted by extractable commitments [46], can be realized based on any CPA secure PKE. More precisely, we prove the following theorem.

---

[2]Here simulatable PKE is a public-key encryption scheme with additional properties that allow oblivious sampling of public keys and ciphertexts.

[3]In such a preprocessing phase, it is assumed that at most one party is allowed to transmit messages in any round.

**Theorem 1.1.** (Informal) *Assuming the existence of PKE and semi-honest oblivious transfer, then any functionality can be realized in the CRS model with static UC security, where the underlying primitives are accessed in a black-box manner.*

We remark here that this theorem makes a significant progress toward reducing the general assumptions required to construct UC protocols. Previously, the only general assumptions based on which we knew how to construct UC protocols were mixed commitments [17] and dual-mode PKE [48] both of which were tailor-made for the particular application. Toward understanding the required minimal assumptions, we recall the work Damgård and Groth in [15] who showed that the existence of UC commitments in the CRS model implies a stand-alone key agreement protocol. Moreover, under black-box constructions, the seminal work of Impagliazzo and Rudich [34] implies that key agreement cannot be based on one-way functions. Thus, there is reasonable evidence to believe that some public-key primitive is required for UC commitments. In that sense, our assumption regarding PKE is close to being optimal. Nevertheless, it is unknown whether plain model (i.e., without setup) semi-honest oblivious transfer assumption is required.

Our result is shown in two phases. At first, we compile the semi-honest oblivious transfer protocol into a new protocol with intermediate security properties in the presence of malicious adversaries. This transformation is an extension of the transformation from [28] that is only proved for bit oblivious transfer, whereas our proof works for string oblivious transfer. Next, we use the transformed oblivious transfer protocol in order to construct a fully secure (malicious) oblivious transfer. By combining our oblivious transfer protocol with the protocol from [33], we obtain a statically generic UC secure computation.

An important corollary is deduced from the work by Gertner et al. [23], who provided a black-box construction of PKE based on any two-round semi-honest oblivious transfer protocol. Specifically, the combination of their result with ours implies the following corollary, which demonstrates that two-round semi-honest oblivious transfer is sufficient in the CRS model to achieve black-box constructions of UC protocols. Namely,

**Corollary 1.2.** *(Informal) Assuming the existence of two-round semi-honest oblivious transfer, then any functionality can be UC realized in the CRS model, where the oblivious transfer is accessed in a black-box manner.*

The work of [6] shows how starting from a semi-honest oblivious transfer it is possible to obtain a black-box construction of an OT protocol that is secure against stand-alone static adversaries in the $\mathcal{F}_{\text{COM}}$-hybrid model. Moreover, $\mathcal{F}_{\text{COM}}$ can be directly realized in the $\mathcal{F}_{\text{EXTCOM}}$-hybrid using the notion of extractable trapdoor commitments [21,49].[4] We do not pursue this approach and instead directly realize OT in the $\mathcal{F}_{\text{EXTCOM}}$-hybrid because the main goal in this work is to identify the minimal assumptions required to construct UC OT. We remark that although the main result in [6] demonstrates UC

---

[4]Here the starting point of the latter work is a statistically hiding and statistically binding straight-line extractable commitment scheme which requires a physical assumption, and is therefore not applicable in the CRS model.

**Table 1.** Comparison with prior work on UC oblivious transfer in the CRS model against static corruptions.

| References | Assumption | Functionality | Setup | Black-box |
|---|---|---|---|---|
| [12] | Enhanced TDP | UC OT | Local CRS | No |
| [48] | Dual-mode PKE | UC OT | Local CRS | Yes |
| [11] | DDH/N-resid./DLIN | UC OT | Global CRS | Yes |
| [43,44] | SH-OT | UC OT | Global CRS | No |
| This work | PKE + SH-OT | UC OT | Local CRS | Yes |
| Follow-up [37] | PKE + SH-OT | UC OT | Global CRS | Yes |

security against adaptive corruptions, the same analysis fails to extend to the static setting. More concretely, while their protocol might be secure in the static setting (if we replace the underlying primitives with their analogues in the static setting), its security analysis is not sufficient. This is because Choi et al. modularly compose a weaker building block (adaptive semi-honest OT) to construct a UC OT. Furthermore, in the simulation of the final protocol, the simulator invokes the *adaptive* simulator of the weaker primitive on the fly. Such a simulation cannot be used in the static setting when the building blocks are instantiated with their analogues in the static setting.[5] We finally remark that the previous works of [6,28] require a three-step transformation, whereas our transformation is simpler with a single step transformation.

*Implications*     In what follows, we make a sequence of interesting observations that are implied by our result in the static UC setting which are summarized in Table 1.

- The important result by Canetti et al. [12], which assumes enhanced trapdoor permutations, can be extended assuming only PKE with oblivious ciphertext generation (which is PKE with the special property that a ciphertext can be obliviously sampled without the knowledge of the plaintext, and can be further realized using enhanced trapdoor permutation). In that sense, our result, assuming PKE with oblivious ciphertext generation, can be viewed as an improvement of Canetti et al. [12] when relying on this primitive in a *black-box* manner.
- The pair of works by Damgard et al. [20] and Lin et al. [44] demonstrate that *non-black-box* constructions of UC commitments, and more generally static UC secure computation, can be achieved in the CRS model assuming only semi-honest oblivious transfer. In comparison, our result shows that two-round semi-honest oblivious transfer protocols are sufficient for obtaining *black-box* UC secure computation in the CRS model. We note here that many semi-honest oblivious transfer protocols indeed involve only two-round of communication, e.g., [22,29].
- In [43,44], Lin, Pass and Venkitasubramaniam provided a unified framework for constructing UC protocols in any "trusted setup" model. Their result is achieved by capturing the minimal requirement that implies UC secure computation in the setup model. More precisely, they introduced the notion of a UC puzzle and showed that any setup model that admits a UC puzzle can be used to securely realize any functionality in the UC setting, while additionally assuming the existence of semi-honest

---

[5]This fact was confirmed with the authors of [6].

oblivious transfer. Moreover, they showed how to easily construct such puzzles in most models. We remark that our approach can be viewed as providing a framework to construct black-box UC protocols in other UC models. More precisely, we show that any setup model that admits the extractable commitment functionality can be used to securely realize any functionality assuming the existence of semi-honest oblivious transfer. In fact, our result easily extends to the chosen key registration authority (KRA) model [1], where it is assumed the existence of a trusted authority that samples public key–secret key pairs for each party and broadcasts the public key to all parties. We leave it for future work to instantiate our framework in other setup models.

- The fact that our construction only requires PKE and semi-honest oblivious transfer allows an easy translation of static UC security to various efficient implementations under a wide range of concrete assumptions. Specifically, both PKE and (two-round) semi-honest oblivious transfer can be realized under RSA, factoring Blum integers, LWE, DDH, $N$-residuosity, $p$-subgroup and coding assumptions. This is compared to prior results that could be based on the later five assumptions [11,14,19,48].

- Recently, Maji et al. [46] initiated the study of the cryptographic complexity of secure computation tasks, while characterizing the relative complexity of a task in the UC setting. Specifically, they established a zero–one law that states that any task is either trivial (i.e., it can be reduced to any other task), or complete (i.e., to which any task can be reduced to), where a functionality $\mathcal{F}$ is said to *reduce* to another functionality $\mathcal{G}$, if there is a UC protocol for $\mathcal{F}$ using ideal access to $\mathcal{G}$. More precisely, they showed that assuming the existence of semi-honest oblivious transfer, every finite two-party functionality is either trivial or complete. While their main theorem relies on the minimal assumption of semi-honest oblivious transfer, their use of the assumption is non-black-box and they leave it as an open problem to achieve the same while relying on oblivious transfer in a black-box manner. Our result makes progress toward establishing this. In more detail, their high-level approach is to identify complete functionalities using four categories, namely (1) $\mathcal{F}_{\text{XOR}}$ that abstracts a XOR-type functionality, (2) $\mathcal{F}_{\text{CC}}$ that abstracts a simple cut-and-choose functionality, (3) $\mathcal{F}_{\text{OT}}$ the oblivious transfer functionality and (4) $\mathcal{F}_{\text{COM}}$ the commitment functionality. They then show that each category can be used to securely realize any computational task.[6] Among these reductions, functionalities $\mathcal{F}_{\text{XOR}}$ and $\mathcal{F}_{\text{CC}}$ rely on oblivious transfer in a non-black-box way. In this work, we improve the reduction in functionality $\mathcal{F}_{\text{CC}}$. That is, we obtain this improvement by showing that the extractable commitment functionality $\mathcal{F}_{\text{EXTCOM}}$ and semi-honest oblivious transfer can be used in a black-box way to realize functionality $\mathcal{F}_{\text{OT}}$ and combine this with a reduction presented in [46] that reduces $\mathcal{F}_{\text{CC}}$ to the $\mathcal{F}_{\text{EXTCOM}}$ functionality in a black-box way.

### 1.1.2. *One-Sided UC Secure Computation*

In this stronger attack model, where at most one of the parties is adaptively corrupted [30,38], we prove that one-sided adaptive UC security is implied by PKE with

---

[6] Here it suffices to realize the $\mathcal{F}_{\text{OT}}$ functionality as it is known to be complete [35].

oblivious ciphertext generation, which implies semi-honest OT. Here we combine two observations: (1) In our malicious static oblivious transfer from the previous result, the actions of the parties depend on their real inputs only in the last phase of the protocol and (2) we do not need a full fledge NCE, rather only need one-sided non-committing encryption (NCE), which we know can be designed based on PKE with oblivious ciphertext generation [9,17]. In particular, NCE allows secure communication in the presence of adaptive attacks, which implies that the communication can be equivocated once the real message is handed to the simulator. Then, by encrypting part of our statically secure protocol using NCE, we obtain a generic protocol for any two-party functionality under the assumption specified above.[7] Namely,

**Theorem 1.3.** (Informal) *Assuming the existence of PKE with oblivious ciphertext generation, then any two-party functionality can be realized in the CRS model with one-sided adaptive UC security and black-box access to the PKE.*

### 1.1.3. *Adaptive UC Secure Computation*

Our last result is in the strongest corruption setting, where any number of parties can be adaptively corrupted. Here we design a new adaptively secure UC commitment scheme under the assumption of PKE with oblivious ciphertext generation, which is the first construction that achieves the stronger notion of adaptive security based on this hardness assumption. Our construction makes a novel usage of such a PKE together with Reed–Solomon codes, where the polynomial shares are encrypted using the PKE with oblivious ciphertext generation. Plugging in our UC commitment protocol into the transformation of [6], that generates adaptive malicious oblivious transfer given adaptive semi-honest oblivious transfer and UC commitments, implies adaptive UC oblivious transfer with malicious security based on semi-honest adaptive oblivious transfer and PKE with oblivious ciphertext generation, using only black-box access to these underlying primitives. That is,

**Theorem 1.4.** (Informal) *Assuming the existence of PKE with oblivious ciphertext generation and adaptive semi-honest oblivious transfer, then any functionality can be realized in the CRS model with adaptive UC security, where the underlying primitives are accessed in a black-box manner.*

We further recall the work of Choi et al. [5] that shows that the weakest general known assumption that is required to construct adaptively secure semi-honest oblivious transfer is trapdoor simulatable PKE.[8] Now, since such an encryption scheme admits PKE with oblivious ciphertext generation, we obtain the following corollary that unifies the two assumptions required to achieve adaptive UC security.

---

[7]We note that while in the plain model any statically secure protocol can be compiled into one-sided secure protocol by encrypting its entire communication using one-sided NCE, such a transformation cannot be applied generically in the UC setting as the trusted setup (e.g., CRS) might depend on the identity of the corrupted party.

[8]Trapdoor simulatable PKE is a simulatable PKE that requires a trapdoor to obliviously sample a public key or a ciphertext.

**Table 2.** Comparison with prior work on UC commitments and UC oblivious transfer in the CRS model against adaptive corruptions.

| References | Assumption | Functionality | Setup | Black-box |
|---|---|---|---|---|
| [17] | Mixed-Com | UC Com | Local CRS | Yes |
| [15] | Strong RSA | UC Com | Global CRS | Yes |
| [16] | Sim PKE | UC Com | Global CRS | No |
| [6] | Trapdoor Sim. PKE | UC OT | UC Com | Yes |
| This work | PKE w/OCG | UC Com | Local CRS | Yes |

**Corollary 1.5.** *Assuming the existence of (trapdoor) simulatable PKE, then any functionality can be realized in the CRS model with adaptive UC security and black-box access to the PKE.*

An additional interesting observation that is implied by our work is that our UC commitment scheme implies a construction that is secure in the adaptive setting when erasures are allowed and under the weaker assumption of PKE. Specifically, instead of obliviously sampling ciphertexts in the commitment phase, the committer encrypts arbitrary plaintexts and then erases the plaintexts and randomness used for these computations. Our proof follows easily for this case as well. Combining our UC commitment scheme together with the semi-honest with erasures OT from [40] and the transformation of Choi et al. [6], we obtain the following result.

**Theorem 1.6.** (Informal) *Assuming the existence of PKE and semi-honest oblivious transfer secure against an adaptive adversary assuming erasures, then any functionality can be realized in the CRS model with adaptive UC security assuming erasures, where the underlying primitives are accessed in a black-box manner.*

Noting that OT secure against adaptive adversaries assuming erasures can be realized under assumptions sufficient for achieving the same with respect to the weaker static adversaries, this theorem shows that achieving UC security against adaptive adversaries in the presence of erasures does not require any additional assumption beyond what is required to secure against static adversaries.

*Implications*    Next, we specify a sequence of interesting observations that are implied by our result in the adaptive UC setting which are summarized in Table 2.

- Previously, Dachman-Soled et al. [16] showed that adaptively secure UC protocols can be constructed in the CRS model assuming the existence of simulatable PKE. Our result improves this result in terms of complexity assumptions by showing that simulatable PKE is sufficient, and provides new constructions based on concrete assumptions that were not known before. Nevertheless, we should point out that while the work of Dachman-Soled et al. is constructed in the global CRS model using a non-black-box construction, our result provides a black-box construction in a CRS model where the length of the reference string is linear in the number of parties.

- Analogous to our result on static UC security, it is possible to extend this result to the chosen key registration authority (KRA) model, where we assume the existence of a trusted party that samples public keys and secret keys for each party, and broadcasts the public key to all parties.
- It is important to note that this result provides the first evidence that adaptively secure UC commitment is theoretically easier to construct than stand-alone adaptively secure semi-honest oblivious transfer. Namely, on the one hand, enhanced trapdoor permutations are sufficient to construct PKE with oblivious ciphertext generation which in turn are sufficient to realize adaptive UC commitment in the CRS model by Theorem 1.4. On the other hand, a result due to Lindell and Zarosim [45] (regarding static versus adaptive oblivious transfer) separates adaptively secure oblivious transfer from enhanced trapdoor permutation under black-box reductions.
- Regarding concrete assumptions, previously, adaptive UC commitments without erasures were constructed based on $N$-residuosity and $p$-subgroup hardness assumptions [18] and Strong RSA [15]. On the other hand, our result demonstrates the feasibility of this primitive under DDH, LWE, factoring Blum integers and RSA assumptions. When considering adaptive corruption with erasures, the work of Blazy et al. [2], extending the work of Lindell [41], shows how to construct highly efficient UC commitments based on the DDH assumption. On the other hand, assuming erasures, we are able to construct an adaptive UC commitment scheme based on any CPA secure PKE.

### 1.2. *Subsequent Work*

In subsequent work, Kiyoshima et al. [37] improve the results in the work for the static setting in the CRS model where they show that assuming PKE and semi-honest OT, UC is feasible in the global CRS model where there is a single CRS string chosen for all sessions.

## 2. Preliminaries

*Basic notations*  We denote the security parameter by $n$. We say that a function $\mu : \mathbb{N} \to \mathbb{N}$ is *negligible* if for every positive polynomial $p(\cdot)$ and all sufficiently large $n$ it holds that $\mu(n) < \frac{1}{p(n)}$. We use the abbreviation PPT to denote probabilistic polynomial time. We further denote by $a \leftarrow A$ the random sampling of $a$ from a distribution $A$ and by $[n]$ the set of elements $\{1, \ldots, n\}$. We specify next the definition of computationally indistinguishable.

**Definition 2.1.**  Let $X = \{X(a, n)\}_{a \in \{0,1\}^*, n \in \mathbb{N}}$ and $Y = \{Y(a, n)\}_{a \in \{0,1\}^*, n \in \mathbb{N}}$ be two distribution ensembles. We say that $X$ and $Y$ are *computationally indistinguishable*, denoted $X \overset{c}{\approx} Y$, if for every PPT machine $D$, every $a \in \{0, 1\}^*$, every positive polynomial $p(\cdot)$ and all sufficiently large $n$:

$$\left| \Pr\left[ D(X(a, n), 1^n) = 1 \right] - \Pr\left[ D(Y(a, n), 1^n) = 1 \right] \right| < \frac{1}{p(n)}.$$

## 2.1. *Public-Key Encryption Schemes*

We specify the definitions of public-key encryption, IND-CPA and public-key encryption with oblivious ciphertext generation.

**Definition 2.2.** *(PKE)* We say that $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ is a *public-key encryption scheme* if $\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}$ are polynomial-time algorithms specified as follows:

- $\mathsf{Gen}$, given a security parameter $n$ (in unary), outputs keys (PK, SK), where PK is a public key and SK is a secret key. We denote this by $(\text{PK}, \text{SK}) \leftarrow \mathsf{Gen}(1^n)$.
- $\mathsf{Enc}$, given the public key PK and a plaintext message $m$, outputs a ciphertext $c$ encrypting $m$. We denote this by $c \leftarrow \mathsf{Enc}_{\text{PK}}(m)$; when emphasizing the randomness $r$ used for encryption, we denote this by $c \leftarrow \mathsf{Enc}_{\text{PK}}(m; r)$.
- $\mathsf{Dec}$, given the public key PK, secret key SK and a ciphertext $c$, outputs a plaintext message $m$ s.t. there exists randomness $r$ for which $c = \mathsf{Enc}_{\text{PK}}(m; r)$ (or $\bot$ if no such message exists). We denote this by $m \leftarrow \mathsf{Dec}_{\text{PK},\text{SK}}(c)$.

For a public-key encryption scheme $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ and a non-uniform adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, we consider the following *indistinguishability game*:

$$(\text{PK}, \text{SK}) \leftarrow \mathsf{Gen}(1^n).$$
$$(m_0, m_1, history) \leftarrow \mathcal{A}_1(\text{PK}), \text{ s.t. } |m_0| = |m_1|.$$
$$c \leftarrow \mathsf{Enc}_{\text{PK}}(m_b), \text{ where } b \leftarrow_R \{0, 1\}.$$
$$b' \leftarrow \mathcal{A}_2(c, history).$$
$$\mathcal{A} \text{ wins if } b' = b.$$

Denote by $\text{ADV}_{\Pi,\mathcal{A}}(n)$ the probability that $\mathcal{A}$ wins the IND-CPA game.

**Definition 2.3.** *(IND-CPA)* A public-key encryption scheme $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ is *IND-CPA secure*, if for every non-uniform adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ there exists a negligible function $\mu(\cdot)$ such that for all sufficiently large $n$'s, $\text{ADV}_{\Pi,\mathcal{A}}(n) \leq \frac{1}{2} + \mu(n)$.

A public-key encryption with the property of oblivious ciphertext generation implies additional two algorithms: (1) oblivious ciphertext generator $\widetilde{\mathsf{Enc}}$ and (2) a corresponding ciphertext faking algorithm $\widetilde{\mathsf{Enc}}^{-1}$. Intuitively, the ciphertext faking algorithm is used to explain a legitimately generated ciphertext as an obliviously generated one. Formally,

**Definition 2.4.** *(PKE with oblivious ciphertext generation* [17]*)* A PKE $\Pi$ with oblivious sampling generation is defined by the tuple $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \widetilde{\mathsf{Enc}}, \widetilde{\mathsf{Enc}}^{-1})$ and has the following additional property,

- *Indistinguishability of oblivious and real ciphertexts* For any message $m$ in the appropriate domain, consider the experiment $(\text{PK}, \text{SK}) \leftarrow \mathsf{Gen}(1^n)$, $c_1 \leftarrow \widetilde{\mathsf{Enc}}_{\text{PK}}(r_1)$, $c_2 \leftarrow \mathsf{Enc}_{\text{PK}}(m; r_2)$, $r_2' \leftarrow \widetilde{\mathsf{Enc}}_{\text{PK}}^{-1}(c_2)$. Then, $(\text{PK}, r_1, c_1, m) \overset{c}{\approx} (\text{PK}, r_2', c_2, m)$.

To this end, we only employ encryption schemes with perfect decryption. This merely simplifies the analysis and can be relaxed by using PKE with a negligible decryption error instead.

## 2.2. *Secret Sharing*

A secret sharing scheme allows distribution of a secret among a group of $n$ players, each of whom in a *sharing phase* receive a share (or piece) of the secret. In its simplest form, the goal of secret sharing is to allow only subsets of players of size at least $t + 1$ to reconstruct the secret. More formally a $t+1$-out-of-$n$ secret sharing scheme comes with a sharing algorithm that on input a secret $s$ outputs $n$ shares $s_1, \ldots, s_n$ and a reconstruction algorithm that takes as input $(s_i)_{i \in S}$, $S$ where $|S| > t$ and outputs either a secret $s'$ or $\perp$. In this work, we will use the Shamir's secret sharing scheme [50] with secrets in $\mathbb{F} = GF(2^n)$. We present the sharing and reconstruction algorithms below:

| | |
|---|---|
| *Sharing algorithm* | For any input $s \in \mathbb{F}$, pick a random polynomial $f(\cdot)$ of degree $t$ in the polynomial field $\mathbb{F}[x]$ with the condition that $f(0) = s$ and output $f(1), \ldots, f(n)$. |
| *Reconstruction algorithm* | For any input $(s_i')_{i \in S}$ where none of the $s_i'$ are $\perp$ and $|S| > t$, compute a polynomial $g(x)$ such that $g(i) = s_i'$ for every $i \in S$. This is possible using Lagrange interpolation where $g$ is given by |

$$g(x) = \sum_{i \in S} s_i' \prod_{j \in S/\{i\}} \frac{x-j}{i-j}.$$

Finally the reconstruction algorithm outputs $g(0)$.

*Reed–Solomon code*    For integers $t$, $n$ and field $\mathbb{F}$, satisfying $0 < t \leq n < |\mathbb{F}|$, and a set of $n$ distinct elements $I = \{x_1, \ldots, x_n\} \subset \mathbb{F}$, the *Reed–Solomon code* $\mathcal{W}_{n,t}$ is defined by

$$\big\{ q(x_1), \ldots, q(x_n) \mid q(\cdot) \text{ is a degree } t \text{ polynomial in } \mathbb{F}[x] \big\}.$$

The Reed–Solomon code has minimum distance relative distance $1 - \frac{t}{n}$ where a corrupted code word with up to $\lceil \frac{n-t}{2} \rceil$ errors can be corrected using the Berlekamp–Welch algorithm. It follows easily that Shamir's secret sharing on $\mathbb{F}$ as described above results in a sequence of shares in the Reed–Solomon code $\mathcal{W}_{n,t}$.

## 2.3. *Oblivious Transfer*

1-out-of-2 oblivious transfer (OT) is an important functionality in the context of secure computation that is engaged between a sender Sen and a receiver Rec; see Fig. 1 for the description of functionality $\mathcal{F}_{\text{OT}}$. In this paper we are interested in reducing the hardness assumptions for general UC secure computation when using only black-box access to the underlying cryptographic primitives, such as the semi-honest OT. We use semi-honest OT as a building block for designing UC protocols in both static and adaptive

---

**Functionality $\mathcal{F}_{\text{OT}}$**

Functionality $\mathcal{F}_{\text{OT}}$ communicates with with sender Sen and receiver Rec, and adversary $\mathcal{S}$.

1. Upon receiving input (sender, $sid, v_0, v_1$) from Sen where $v_0, v_1 \in \{0, 1\}^t$, record $(sid, v_0, v_1)$.
2. Upon receiving (receiver, $sid, u$) from Rec, where a tuple $(sid, v_0, v_1)$ is recorded and $u \in \{0, 1\}$, send $(sid, v_u)$ to Rec and $sid$ to $\mathcal{S}$. Otherwise, abort.

---
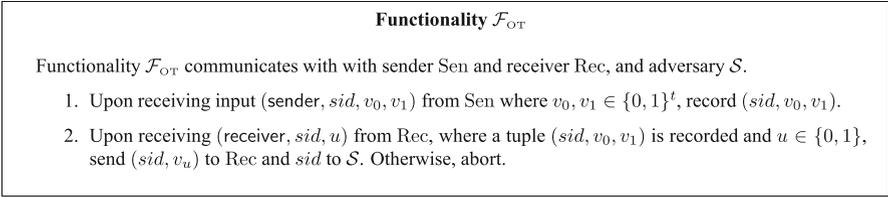
**Fig. 1.** Oblivious transfer functionality.

settings. In the static setting, we refer to the two-round protocol of [22] that is based on PKE with oblivious ciphertext generation (or enhanced trapdoor permutation). In the adaptive setting, we refer to the two-round protocol of [12] that is based on augmented non-committing encryption scheme.

We briefly recall that any two-round semi-honest OT implies PKE. This is demonstrated in two phases, starting with the claim that semi-honest OT implies a key agreement (KA) protocol. This statement has been proved in [23] in the static setting and holds for any number of rounds as well as in the presence of adaptive adversaries. Next, a well-established fact shows that in the static setting a two-round key agreement implies PKE. (In fact, these primitives are equivalent).

### 2.3.1. *Receiver Private Oblivious Transfer*

Receiver privacy is a weaker notion than malicious security and only requires that the receiver's input be hidden even against a malicious sender. It is weaker than malicious security in that it does not require a simulation of the malicious sender that extracts the sender's inputs. In particular, we will only require that a malicious sender cannot distinguish the cases where the receiver's input is 0 or 1. Formally stated,

**Definition 2.5.** *(Receiver private OT)* Let $\pi$ be a two-party protocol that is engaged between a sender Sen and a receiver Rec. We say that $\pi$ is a *receiver private* oblivious transfer protocol, if for every PPT adversary $\mathcal{A}$ that corrupts Sen, the following ensembles are computationally indistinguishable:

- $\{\mathbf{View}_{\mathcal{A},\pi}[\mathcal{A}(1^n), \text{Rec}(1^n, 0)]\}_{n \in \mathbb{N}}$
- $\{\mathbf{View}_{\mathcal{A},\pi}[\mathcal{A}(1^n), \text{Rec}(1^n, 1)]\}_{n \in \mathbb{N}}$,

where $\mathbf{View}_{\mathcal{A},\pi}[\mathcal{A}(1^n), \text{Rec}(1^n, b)]$ denotes $\mathcal{A}$'s view within $\pi$ whenever the receiver Rec inputs the bit $b$.

We point out that receiver privacy protects the receiver against a malicious sender and should be read as privacy against a malicious sender.

### 2.3.2. *Defensible Private Oblivious Transfer*

The notion of *defensible privacy* was introduced by Haitner in [27,28]. A defense in a two-party protocol $\pi = (P_1, P_2)$ execution is an input and random tape provided by the adversary after the execution concludes. A defense for a party controlled by the adversary is said to be *good* if whenever this party participated honestly in the

protocol using this input and random tape, then it would have resulted in the exact same messages that were sent by the adversary. In essence, this defense serves as a *proof* of honest behavior. Defensible privacy ensures that a protocol is private in the presence of defensible adversaries if the adversary learns nothing more than its prescribed output when it provides a good defense.

We begin with informally describing the notion of *good defense* for a protocol $\pi$; we refer to [28] for the formal definition. Let $\mathsf{trans} = (q_1, a_1, \ldots, q_\ell, a_\ell)$ be the transcript of an execution of a protocol $\pi$ that is engaged between $P_1$ and $P_2$ and let $\mathcal{A}$ denote an adversary that controls $P_1$, where $q_i$ is the $i$th message from $P_1$ and $a_i$ is the $i$th message from $P_2$ (that is, $a_i$ is the response for $q_i$). Then we say that $(x, r)$ constitutes a *good defense* of $\mathcal{A}$ relative to $\mathsf{trans}$ if the transcript generated by running the honest algorithm for $P_1$ with input $x$ and random tape $r$ against $P_2$'s messages $a_1, \ldots, a_\ell$ results exactly in $\mathsf{trans}$.

At a high level, an OT protocol is defensible private with respect to a corrupted sender if no adversary interacting with an honest receiver with input $b$ should be able to learn $b$, if at the end of the execution the adversary produces any good defense. Similarly, an OT protocol that is defensible private with respect to a corrupted receiver requires that any adversary interacting with an honest sender with input $(s_0, s_1)$ should not be able to learn $s_{1-b}$, if at the end of the execution the adversary produces a good defense with input $b$. Below we present a variant of the definition presented in [28]. We stress that while the [28] definition only considers bit OT (i.e., sender's inputs are bits) we consider *string OT*.

**Definition 2.6.** *(Defensible private string OT)* Let $\pi$ be a two-party protocol that is engaged between a sender Sen and a receiver Rec. We say that $\pi$ is a *defensible private* string oblivious transfer protocol, if for every PPT adversary $\mathcal{A}$ the following holds,

1. $\{\Gamma(\mathbf{View}_{\mathcal{A}}[\mathcal{A}(1^n), \mathrm{Rec}(1^n, U)], U)\} \overset{c}{\approx} \{\Gamma(\mathbf{View}_{\mathcal{A}}[\mathcal{A}(1^n), \mathrm{Rec}(1^n, U)], U')\}$ where $\Gamma(v, *)$ is set to $(v, *)$ if following the execution $\mathcal{A}$ outputs a good defense for $\pi$, and $\perp$ otherwise, and $U$ and $U'$ are independent random variables uniformly distributed over $\{0, 1\}$. This property is referred to as defensible private with respect to a corrupted sender.

2. $\{\Gamma(\mathbf{View}_{\mathcal{A}}[\mathrm{Sen}(1^n, (U_0^n, U_1^n)), \mathcal{A}(1^n)], U_{1-b}^n)\} \overset{c}{\approx} \{\Gamma(\mathbf{View}_{\mathcal{A}}[\mathrm{Sen}(1^n, (U_0^n, U_1^n)), \mathcal{A}(1^n)], \bar{U}^n)\}$ where $\Gamma(v, *)$ is set to $(v, *)$ if following the execution $\mathcal{A}$ outputs a good defense for $\pi$, and $\perp$ otherwise, $b$ is the Rec's input in this defense and $U_0^n, U_1^n, \bar{U}^n$ are independent random variables uniformly distributed over $\{0, 1\}^n$. This property is referred to as defensible private with respect to a corrupted receiver.

In our construction from Sect. 3, we will rely on an OT protocol that is receiver private and defensible private with respect to a corrupted receiver. In [28], Haitner et al. showed how to transform any semi-honest bit OT to one that is defensible private with respect to a corrupted receiver and malicious secure with respect to a corrupted sender. More formally, the following lemma is implicit in the work of [28].

**Lemma 2.1.** *(Implicit in Theorem 4.1 and Corollary 5.3 [28]) Assume the existence of a semi-honest oblivious transfer protocol $\pi$. Then there exists an oblivious transfer*

*protocol $\hat{\pi}$ that is defensible private with respect to the receiver and receiver private that relies on the underlying primitive in a black-box manner.*

Now, since receiver privacy is implied by malicious security with respect to a corrupted sender, this transformation yields a bit OT protocol with the required security guarantees. Nevertheless, our protocol crucially relies on the fact that the underlying OT is a string OT protocol. We therefore show in "Appendix A" how to transform any bit OT to a string OT protocol while preserving both defensible private with respect to a maliciously corrupted receiver and receiver privacy.

### 2.4. *Commitment Schemes*

Commitment schemes are used to enable a party, known as the *sender*, to commit itself to a value while keeping it secret from the *receiver* (this property is called hiding). Furthermore, in a later stage when the commitment is opened, it is guaranteed that the "opening" can yield only a single value determined in the committing phase (this property is called binding). In this work, we consider commitment schemes that are statistically binding; namely, while the hiding property only holds against computationally bounded (non-uniform) adversaries, the binding property is required to hold against unbounded adversaries. More precisely, a pair of PPT machines Com is said to be a commitment scheme if the following two properties hold.

*Computational hiding* For every (expected) PPT machine $R^*$, it holds that the following ensembles are computationally indistinguishable over $n \in N$.

$\{\text{view}_{\text{Com}}^{R^*}(v_1, z)\}_{n \in N, v_1, v_2 \in \{0,1\}^n, z \in \{0,1\}^*}$
$\{\text{view}_{\text{Com}}^{R^*}(v_2, z)\}_{n \in N, v_1, v_2 \in \{0,1\}^n, z \in \{0,1\}^*}$,

where $\text{view}_{\text{Com}}^{R^*}(v, z)$ denotes the random variable describing the output of $R^*$ after receiving a commitment to $v$ using Com.

*Statistical binding* Informally, the statistical binding property asserts that, with overwhelming probability over the coin tosses of the receiver $R$, the transcript of the interaction fully determines the value committed to by the sender.

We say that a commitment is *valid* if there exists a unique committed value that a (potentially malicious) committer can open to successfully. We refer the reader to [26] for more details.

### 2.5. *UC Commitment Schemes*

The notion of UC commitments was introduced by Canetti and Fischlin in [8]. The formal description of functionality $\mathcal{F}_{\text{COM}}$ is depicted in Fig. 2.

### 2.6. *Extractable Commitments*

Our result in the static setting requires the notion of (static) extractable UC commitments, which is a weaker security property than UC commitments in the sense that it does not require equivocality. Namely, the simulator is not required to commit to one

---

**Functionality $\mathcal{F}_{\text{COM}}$**

Functionality $\mathcal{F}_{\text{COM}}$ communicates with with sender Sen and receiver Rec, and adversary $\mathcal{S}$.

1. Upon receiving input (commit, $sid, m$) from Sen where $m \in \{0, 1\}^t$, internally record $(sid, m)$ and send message $(sid, \text{Sen}, \text{Rec})$ to the adversary. Upon receiving approve from the adversary send $sid$, to Rec. Ignore subsequent (commit, ., ., .) messages.

2. Upon receiving (reveal, $sid$) from Sen, where a tuple $(sid, m)$ is recorded, send message $m$ to adversary $\mathcal{S}$ and Rec. Otherwise, ignore.

---

**Fig. 2.** String commitment functionality.

message and then later convince the receiver that it committed to a different value. It is a real challenge to define this notion since it is hard to capture the notion of extractability in the ideal setting. In what follows, we recall the definition for the ideal functionality $\mathcal{F}_{\text{EXTCOM}}$ from [46]. To the best of our knowledge, this is the only definition that captures straight-line extractability, statistically binding and computationally (stand-alone) hiding. Toward introducing this definition, Maji et al. introduced some notions first. More concretely,

**Definition 2.7.** A protocol is a *syntactic commitment protocol* if:

- It is a two-phase protocol between a sender and a receiver (using only plain communication channels).
- At the end of the first phase (commitment phase), the sender and the receiver output a transcript trans. Furthermore, the sender receives an output (which will be used for opening the commitment).
- In the decommitment phase the sender sends a message $\gamma$ to the receiver, who extracts an output value opening(trans, $\gamma$) $\in \{0, 1\}^n \cup \{\bot\}$.

**Definition 2.8.** Two syntactic commitment protocols $(\omega_L, \omega_R)$ form a pair of complementary statistically binding commitment protocols if the following hold:

- $\omega_R$ is a statistically binding commitment scheme (with stand-alone security).
- In $\omega_L$, at the end of the commitment phase the receiver outputs a string $z \in \{0, 1\}^n$. If the receiver is honest, it is only with negligible probability that there exists $\gamma$ such that opening(trans, $\gamma$) $\neq \bot$ and opening(trans, $\gamma$) $\neq z$.

As noted in [46], $\omega_L$ by itself is not an interesting cryptographic goal, as the sender can simply send the committed string in the clear during the commitment phase. Nevertheless, in defining $\mathcal{F}_{\text{EXTCOM}}$ below, there exists a single protocol that satisfies both the security guarantees. We are now ready to introduce the notion of extractable commitments in Fig. 3 that is parameterized by $(\omega_L, \omega_R)$. We additionally include a function pp that will be used as an initialization phase to set up the public parameters for $\omega_L$ and $\omega_R$.

In "Appendix 2.6.1" we show how to realize $\mathcal{F}_{\text{EXTCOM}}$ based on IND-CPA secure PKE.

---

**Functionality $\mathcal{F}_{\text{EXTCOM}}$ parameterized by $(\text{pp}, \omega_L, \omega_R)$**

$\mathcal{F}_{\text{EXTCOM}}$ is running with parties $P_1, \ldots, P_n$ and an adversary $\mathcal{S}$: Upon receiving a message $(\text{init} - \text{commit}, sid, ssid, P_i, P_j)$ from $P_i$, it first checks if there is a tuple $(\text{public} - \text{params}, sid, P_i, (pp, sp))$. If yes, it sends $(\text{init} - \text{commit}, sid, ssid, P_i, P_j)$ to $P_j$. If not, it runs $(pp, sp) \leftarrow \text{pp}(1^n)$ and sends $(\text{init} - \text{commit}, sid, P_i, pp)$ to $P_i$, $P_j$ and $\mathcal{S}$. It stores $(\text{public} - \text{params}, sid, P_i, (pp, sp))$. We denote $P_i$ by the sender and $P_j$ by the receiver in this interaction. Next, the functionality behaves as follows, depending on which party is corrupted.

- $P_i$ IS HONEST AND $P_j$ IS HONEST.

  **Commit Phase:** Upon receiving $(\text{commit}, sid, ssid, P_i, P_j, m)$ from $P_i$, it internally simulates a session of $\omega_R$ (simulating both the sender and receiver in $\omega_R$), with the sender's input fixed to $m$. It gives $(\text{transcript}, sid, ssid, \text{trans}, \gamma)$ to $P_i$ and $(\text{receipt}, sid, ssid, P_i, P_j, \text{trans})$ to $P_j$ and $\mathcal{S}$.

  **Reveal Phase:** Upon receiving $(\text{decommit}, sid, ssid, \cdot)$ from the sender, it sends $(\text{decommit}, sid, ssid, P_i, P_j, z)$ to $P_j$ and $\mathcal{S}$.

- $P_i$ IS CORRUPTED AND $P_j$ IS HONEST.

  **Commit Phase:** It runs the commitment $\omega_L$ with the sender, playing the part of the receiver in $\omega_L$, to obtain $(sid, ssid, \text{trans}, z)$. It sends $(\text{receipt}, sid, ssid, P_i, P_j, \text{trans})$ to $P_j$ and $\mathcal{S}$.

  **Reveal Phase:** Upon receiving $(\text{decommit}, sid, ssid, \gamma)$ from the sender, if $\text{opening}(\text{trans}, \gamma) = z$, it sends $(\text{decommit}, sid, ssid, P_i, P_j, z)$ to $P_j$ and $\mathcal{S}$. Otherwise ignore.

- $P_i$ IS HONEST AND $P_j$ IS CORRUPT.

  **Commit Phase:** Upon receiving $(\text{commit}, sid, ssid, P_i, P_j, m)$ from $P_i$, it runs the commitment phase of $\omega_R$ with $P_j$, playing the sender's role in $\omega_R$ with $m$ as input. It obtains the output $(\text{trans}, \gamma)$ at the end of this phase, and sends $(\text{transcript}, sid, ssid, \text{trans}, \gamma)$ to $P_i$.

  **Reveal Phase:** Upon receiving $(\text{decommit}, sid, ssid)$ from the sender it sends $(\text{decommit}, sid, ssid, P_i, P_j, (\gamma, z))$ to $P_j$ and $\mathcal{S}$.

The functionality does not do anything when both the sender and the receiver are corrupted.

---

**Fig. 3.** Extractable commitment functionality.

### 2.6.1. *Extractable Commitments from PKE in the CRS Model*

We briefly discuss how to realize the $\mathcal{F}_{\text{EXTCOM}}$ functionality in the CRS model. At a high level, we obtain an extractable commitment using a IND-CPA PKE. Loosely speaking, the common reference string contains a public key that is sampled using the key generation algorithm. Moreover, the trapdoor for the CRS is the corresponding secret key. In the real world, no adversary knows that secret key, and hence, it does not know the corresponding CRS trapdoor. In order to implement extractable commitments, our protocol requires from the commitment sender to simply encrypt its message *m* using the public key that is placed in the CRS. Decommitment is carried out by asking the sender to provide the randomness used to encrypt *m*.

## 3. Static UC Secure Computation

In this section, we prove the feasibility of UC secure computation based on semi-honest OT and extractable commitments, where the latter can be constructed based on two-

round semi-honest OT (see Sects. 2.3 and 2.6 for more details). More concretely, we prove how to transform any statically semi-honest secure OT into one that is secure in the presence of malicious adversaries, giving *only black-box access* to the underlying semi-honest OT protocol. Our protocol is a variant of the protocol by Lin and Pass from [42] (which in turn is a variant of the protocol of [28]). In particular, in [42], the authors rely on a strong variant of a commitment scheme known as a CCA secure commitment in order to achieve extraction. We observe that it is not required to use the full power of such commitments, or for that matter UC commitments. Specifically, using a weaker primitive that only implies straight-line input extractability enables to solely rely on semi-honest OT. An important weakening in our commitment scheme compared to CCA secure commitments from [42] is that we allow invalid commitments to be made by the adversary. Our construction obtains a statically UC protocol for any well-formed functionality (see definition in [12]). Namely,

**Theorem 3.1.** *Assume the existence of static semi-honest oblivious transfer. Then for any multi-party well-formed functionality $\mathcal{F}$, there exists a protocol that UC realizes $\mathcal{F}$ in the presence of static, malicious adversaries in the $\mathcal{F}_{\mathrm{EXTCOM}}$-hybrid model using black-box access to the oblivious transfer protocol.*

The proof of Theorem 3.1 follows from combining our UC OT protocol with the [33] protocol. It seems possible to generalize our theorem to multi-session functionalities. Analogous to [8], this will allow us to extend our corollaries to the global CRS model by additionally assuming CCA encryption scheme; we leave this as future work.

### 3.1. *Static UC Oblivious Transfer*

In the following, we discuss a secure implementation of the oblivious transfer functionality (see Fig. 1) with static, malicious security in the $\mathcal{F}_{\mathrm{EXTCOM}}$-hybrid model (where $\mathcal{F}_{\mathrm{EXTCOM}}$ is stated formally in Fig. 3). Our goal in this section is to show that the security of malicious UC OT can be based on UC semi-honest OT, denoted by $\pi_{\mathrm{OT}}^{\mathrm{SH}}$, and extractable commitments. Our result is shown in two phases. At first, we compile the semi-honest OT protocol $\pi_{\mathrm{OT}}^{\mathrm{SH}}$ into a new protocol with the security properties that are specified in Sect. 2.3.2, extending the [28] transformation into string OT; denote the compiled OT protocol by $\widehat{\pi}_{\mathrm{OT}}$. This transformation in specified "Appendix A." In what follows, we use $\widehat{\pi}_{\mathrm{OT}}$ in order to construct a new protocol $\pi_{\mathrm{OT}}^{\mathrm{ML}}$ that is secure in the presence of malicious adversaries.

Our protocol is a variant of first step of the compilation in [6] which in turn is based on the work of [28]. At a high level, the compilation in [6,28] shows how to amplify the security of an oblivious transfer protocol against receiver corruption from semi-honest to malicious. In comparison, our protocol amplifies the security of an OT protocol that is defensible private against the sender and receiver to full security.

Loosely speaking, the parties first run a coin-tossing protocol in order to generate the input and randomness for both the receiver and the sender. Using cut-and-choose, which requires to repeat this process multiple times, we are able to extract these values in the simulation. The parties then run a sequence of random oblivious transfers using the values generated in the coin-tossing phase. Finally, the sender applies a combiner on

the remaining random OT inputs (namely for the positions that were not opened during the cut-and-choose opening phase), in order to transfer its real inputs. Details follow,

**Protocol 1.** *(Protocol $\pi_{\text{OT}}^{\text{ML}}$ with static security)*

    **Input:** *The sender* Sen *has input* $(v_0, v_1)$ *where* $v_0, v_1 \in \{0, 1\}^n$ *and the receiver* Rec *has input* $u \in \{0, 1\}$.
    **The protocol:**

1. **Coin-tossing:**
   - Receiver's random tape generation: *The parties use a coin-tossing protocol in order to generate the inputs and random tapes for the receiver.*

     – *The receiver commits to* $20n$ *strings of appropriate length, denoted by* $a_{\text{Rec}}^1, \ldots, a_{\text{Rec}}^{20n}$, *by sending* $\mathcal{F}_{\text{EXTCOM}}$ *the message* (commit, $sid$, $\widetilde{ssid}_i$, $a_{\text{Rec}}^i$) *for all* $i \in [n]$.
     – *The sender responds with* $20n$ *random strings of appropriate length* $b_{\text{Rec}}^1, \ldots, b_{\text{Rec}}^{20n}$.
     – *The receiver computes* $r_{\text{Rec}}^i = a_{\text{Rec}}^i \oplus b_{\text{Rec}}^i$ *and then interprets* $r_{\text{Rec}}^i = c_i || \tau_{\text{Rec}}^i$ *where* $c_i$ *determines the receiver's input for the $i$th OT protocol, whereas* $\tau_{\text{Rec}}^i$ *determines the receiver's random tape used for this execution.*

   - Sender's random tape generation: *The parties use a coin-tossing protocol in order to generate the inputs and random tapes for the sender.*

     – *The sender commits to* $20n$ *strings of appropriate length, denoted by* $a_{\text{Sen}}^1, \ldots, a_{\text{Sen}}^{20n}$, *by sending* $\mathcal{F}_{\text{EXTCOM}}$ *the message* (commit, $sid$, $\widetilde{ssid}_i'$, $a_{\text{Sen}}^i$) *for all* $i \in [n]$.
     – *The receiver responds with* $20n$ *random strings of appropriate length* $b_{\text{Sen}}^1, \ldots, b_{\text{Sen}}^{20n}$.
     – *The sender computes* $r_{\text{Sen}}^i = a_{\text{Sen}}^i \oplus b_{\text{Sen}}^i$ *and then interprets* $r_{\text{Sen}}^i = s_i^0 || s_i^1 || \tau_{\text{Sen}}^i$ *where* $(s_i^0, s_i^1)$ *determine the sender's input for the $i$th OT protocol, whereas* $\tau_{\text{Sen}}^i$ *determines the sender's random tape used for this execution.*

2. **Oblivious transfer:**
   - *The parties participate in* $20n$ *executions of the OT protocol* $\widehat{\pi}_{\text{OT}}$ *with the corresponding inputs and random tapes obtained from Stage 2. Let the output of the receiver in the $i$th execution be* $\tilde{s}_i$.

3. **Cut-and-choose:**
   - Sen *chooses a random subset* $q_{\text{Sen}} = (q_{\text{Sen}}^1, \ldots, q_{\text{Sen}}^n) \in \{1, \ldots, 20\}^n$ *and sends it to* Rec. *The string* $q_{\text{Sen}}$ *is used to define a set of indices* $\Gamma_{\text{Sen}} \subset \{1, \ldots, 20n\}$ *of size $n$ by grouping the indices into blocks of 20 and choosing element* $q_{\text{Sen}}^i$ *index in the $i$th block. More formally,* $\Gamma_{\text{Sen}} = \{20i - q_{\text{Sen}}^i\}_{i \in [n]}$. *The receiver then opens the commitments from Stage 1 that correspond to the indices within* $\Gamma_{\text{Sen}}$; *namely, the receiver decommits* $a_{\text{Rec}}^i$ *for all* $i \in \Gamma_{\text{Sen}}$. Sen *checks that the*

    *decommitted values are consistent with the inputs and randomness used for the OTs in Stage 2 by the receiver, and aborts in case of a mismatch.*

- Rec *chooses a random subset* $q_{\mathrm{Rec}} = (q_{\mathrm{Rec}}^1, \ldots, q_{\mathrm{Rec}}^n) \in \{1, \ldots, 20\}^n$ *and sends it to* Sen. *The string* $q_{\mathrm{Rec}}$ *is used to define a set of indices* $\Gamma_{\mathrm{Rec}} \subset \{1, \ldots, 20n\}$ *of size n in the following way:* $\Gamma_{\mathrm{Rec}} = \{20i - q_{\mathrm{Rec}}^i\}_{i \in [n]}$. *The sender then opens the commitments from Stage 1 that correspond to the indices within* $\Gamma_{\mathrm{Rec}}$; *namely, the sender decommits* $a_{\mathrm{Sen}}^i$ *for all* $i \in \Gamma_{\mathrm{Rec}}$. Rec *checks that the decommitted values are consistent with the inputs and randomness used for the OTs in Stage 2 by the sender, and aborts in case of a mismatch.*

- Rec *commits to another subset* $\Gamma \subset [20n]$ *denoted by* $(\Gamma^1, \ldots, \Gamma^n)$, *by sending* $\mathcal{F}_{\mathrm{EXTCOM}}$ *the message* (commit, $sid, ssid_i', \Gamma^i$) *for all* $i \in [n]$. *(The sender will reveal its inputs and randomness that are used in Stage 2 that correspond to the indices in* $\Gamma$ *later in Stage 5.)*

4. **Combiner:**

- *Let* $\Delta = [20n] - \Gamma_{\mathrm{Rec}} - \Gamma_{\mathrm{Sen}}$. *Then for every* $i \in \Delta$, *the receiver computes* $\alpha_i = u \oplus c_i$ *and sends it to the sender.*

- *The sender computes a* $10n$-*out-of*-$18n$ *secret sharing of* $v_0$, *denote the shares by* $\{\rho_i^0\}_{i \in \Delta}$. *Analogously, it computes a* $10n$-*out-of*-$18n$ *secret sharing of* $v_1$, *denote the shares by* $\{\rho_i^1\}_{i \in \Delta}$. *The sender computes* $\beta_i^b = \rho_i^b \oplus s_i^{b \oplus \alpha_i}$ *for all* $b \in \{0, 1\}$ *and* $i \in \Delta$, *and sends the outcome to the receiver.*

- *The receiver computes* $\tilde{\rho}_i = \beta_i^u \oplus \tilde{s}_i$ *for all* $i \in \Delta$. *Denote by* $\rho$ *these concatenated bits.*

5. **Final cut-and-choose:**

- *The receiver decommits* $\Gamma$ *and the sender sends the inputs and randomness it used in Stage 2 for the coordinates that correspond to* $\Delta \cap \Gamma$. *(Note that the sender needs only to reveal the indices that were not decommitted in Stage 3.)* Rec *checks that the sender's values are consistent with the inputs and randomness used for the OTs in Stage 2 and the combiner computation in Stage 4 made by the sender, and aborts in case of a mismatch.*

- *The receiver checks whether* $(\tilde{\rho}_i)_{i \in \Delta}$ *agrees with some code word* $w \in \mathcal{W}_{18n, 10n}$ *on* $17n$ *locations (where the code* $\mathcal{W}_{18n, 10n}$ *is induced by the secret sharing construction that we use in Stage 4; see Definition* 2.2 *for more details). Recall that the minimum distance of the code* $\mathcal{W}_{18n, 10n}$ *is at least* $18n - 10n > 8n$, *which implies that there will be at most one such code word* $w$. *Furthermore, since we can correct up to* $\frac{18n - 10n}{2} = 4n$ *errors, any code that is* $17n$ *close to a code word can be efficiently recovered using the Berlekamp–Welch algorithm. The receiver outputs that* $w$ *as its output in the OT protocol. If no such* $w$ *exists, the receiver returns a default value.*

We next prove the following theorem.

**Theorem 3.2.** *Assume that that the compiled* $\hat{\pi}_{\mathrm{OT}}$ *is defensible private (cf. Definition* 2.6*). Then Protocol* 1 *UC realizes* $\mathcal{F}_{\mathrm{OT}}$ *in the presence of static malicious adversaries in the* $\mathcal{F}_{\mathrm{EXTCOM}}$-*hybrid model using black-box access to the oblivious transfer protocol.*

We recall Lemma 2.1 and "Appendix A" that demonstrate the transformation from semi-honest OT to defensible private string OT. Specifically, our protocol relies on the existence of semi-honest OT and extractable commitments, where the later can be constructed based on any two-round semi-honest OT, e.g., [22], which implies PKE (see Sects. 2.3 and 2.6 for more details). Therefore, an immediate corollary from Theorem 3.2 implies that

**Corollary 3.3.** *Assume the existence of two-round static semi-honest oblivious transfer. Then there exists a protocol that securely realizes $\mathcal{F}_{OT}$ in the presence of static malicious adversaries in the CRS model using black-box access to the oblivious transfer protocol.*

*A high-level proof* We first provide an overview of the security proof; the complete proof is found in Sect. 3.2. Loosely speaking, in case the receiver is corrupted the simulator plays the role of the honest sender in Stages 1–3 and extracts the receiver's input $u$. Specifically, the simulator extracts all the committed values of the receiver within Stage 1 (relying on the fact that the commitment scheme is extractable) and then uses these values in order to obtain the inputs for the OT executions in Stage 2. Upon completing Stage 2, the simulator records the coordinates for which the receiver deviates from the prescribed input and random tape chosen in the coin-tossing phase. Denoting these set of coordinates by $\Phi$, we recall that a malicious receiver may obtain both of the sender's inputs with respect to the OT executions that correspond to the coordinates within $\Phi$ and $\Gamma$. On the other hand, it obtains only one of the two inputs with respect to the rest of the OT executions that correspond to the coordinates within $\Delta - \Phi - \Gamma$. Consequently, the simulator checks how many shares of $v_0$ and $v_1$ are obtained by the receiver and completes Stage 4 accordingly. In more detail,

- If the receiver obtains more than $10n$ shares of both inputs, then the simulator halts and outputs fail (we prove in Sect. 3.2 that this event only occurs with negligible probability).
- If the receiver obtains less than $10n$ shares of both inputs, then the simulator picks two random values for $v_0$ and $v_1$ of the appropriate length and completes the interaction, playing the role of the honest sender on these values. Note that in this case the simulator does not need to call the ideal functionality.
- Finally, if the receiver obtains more than $10n$ shares for only one input $u \in \{0, 1\}$, then the simulator sends $u$ to the ideal functionality $\mathcal{F}_{OT}$ and obtains $v_u$. The simulator then sets $v_{1-u}$ as a random string of the appropriate length and completes the interaction by playing the role of the honest sender on these values.

Recall that the only difference between the simulation and the real execution is in the way the messages in Stage 4 are generated. Specifically, in the simulation a value $u$ is extracted from the malicious receiver and then fed to the $\mathcal{F}_{OT}$ functionality. The simulation is then completed based on the output returned from the functionality. Intuitively, the cut-and-choose mechanism ensures that the receiver cannot deviate from the honest strategy in Stage 2 in more than $n$ OT sessions without getting caught with overwhelming probability. Moreover, the defensible privacy of the OT protocol implies that the receiver

can learn at most one of the two inputs of the sender relative to the OT executions in Stage 2 for which the receiver proceeded honestly.

In case the sender is corrupted, the simulator's strategy is to play the role of the honest receiver with a fixed input 0 until Stage 5 where the simulator extracts the sender's inputs. More specifically, the simulator first extracts the sender's input for the OT executions in Stage 1 (relying on the fact that the commitment scheme is extractable). Next, the simulator extracts the shares $\{\rho_i^0\}_{i \in \Delta}$ and $\{\rho_i^1\}_{i \in \Delta}$ that correspond to inputs $v_0$ and $v_1$. To obtain the actual values the simulator checks whether these shares agree with some code word relative to $16n$ locations. That is,

- Let $w_0$ and $w_1$ denote the corresponding code words. (If there are no such code words that agree with $v_0$ and $v_1$ on $16n$ locations, then the simulator uses a default code word instead.) Next, the simulator checks $w_0$ and $w_1$ against the final cut-and-choose. If any of the shares from $w_b$ are inconsistent with the opened shares that are opened by the sender in the final cut-and-choose, then $v_b$ is set to a default value; otherwise, $v_b$ is the value corresponding to the shared secret.

Finally, the simulator sends $(v_0, v_1)$ to the ideal functionality for $\mathcal{F}_{\mathrm{OT}}$. Security in this case is reduced to the privacy of the OT receiver. In addition, the difference between the simulation's strategy and the honest receiver's strategy is that the simulator extracts the sender's both inputs in all $i \in \Delta - \Phi$ and then finds code words that are $16n$-close to the extracted values, whereas the honest receiver finds a code word that is $17n$-close based on the inputs it received in Stages 2 and 5, and returns it. We thus prove that the value $u$ extracted by the simulator is identical to the reconstructed output of the honest receiver relying on the properties of the secret sharing scheme.

### 3.2. *Proof of Theorem 3.2*

Let $\mathcal{A}$ be a malicious probabilistic polynomial-time real adversary running protocol 1 in the $\mathcal{F}_{\mathrm{EXTCOM}}$-hybrid model. We construct an ideal model adversary $\mathcal{S}$ with access to $\mathcal{F}_{\mathrm{OT}}$ which simulates a real execution of protocol $\pi_{\mathrm{OT}}^{\mathrm{ML}}$ with $\mathcal{A}$ such that no environment $\mathcal{Z}$ can distinguish the ideal process with $\mathcal{S}$ and $\mathcal{F}_{\mathrm{OT}}$ from a real execution of $\pi_{\mathrm{OT}}^{\mathrm{ML}}$ with $\mathcal{A}$. $\mathcal{S}$ starts by invoking a copy of $\mathcal{A}$ and running a simulated interaction of $\mathcal{A}$ with environment $\mathcal{Z}$, emulating the honest party. We separately describe the actions of $\mathcal{S}$ for every corruption case.

*Simulating the communication with* $\mathcal{Z}$ Every message that $\mathcal{S}$ receives from $\mathcal{Z}$ it internally feeds to $\mathcal{A}$ and every output written by $\mathcal{A}$ is relayed back to $\mathcal{Z}$.

*Simulating the corrupted receiver* In this case, $\mathcal{S}$ proceeds as follows:

1. $\mathcal{S}$ emulates functionality $\mathcal{F}_{\mathrm{EXTCOM}}$ in Stage 1 and invokes $20n$ times the commitment scheme $\omega_L$ with $\mathcal{A}$ (that plays the role of the committer), obtaining $((\widetilde{\mathsf{trans}}_1, a_{\mathrm{Rec}}^1), \ldots, (\widetilde{\mathsf{trans}}_{20n}, a_{\mathrm{Rec}}^{20n}))$. It internally records $a_{\mathrm{Rec}}^1, \ldots, a_{\mathrm{Rec}}^{20n}$ and further picks $20n$ random strings $b_{\mathrm{Rec}}^1, \ldots, b_{\mathrm{Rec}}^{20n}$, forwarding them to the adversary. The simulator also computes $r_{\mathrm{Rec}}^i = a_{\mathrm{Rec}}^i \oplus b_{\mathrm{Rec}}^i$ and then views $r_{\mathrm{Rec}}^i = c_i || \tau_{\mathrm{Rec}}^i$

where $c_i$ is the input an honest receiver must use in the $i$th OT protocol execution in Stage 2, together with randomness $\tau_{\text{Rec}}^i$.

Next, the simulator picks $20n$ random strings $a_{\text{Sen}}^1, \ldots, a_{\text{Sen}}^{20n}$ and emulates the ideal functionality $\mathcal{F}_{\text{EXTCOM}}$ by invoking $20n$ times the commitment phase of $\omega_R$ with inputs $a_{\text{Sen}}^1, \ldots, a_{\text{Sen}}^{20n}$, against $\mathcal{A}$ that plays the role of receiver for the commitment scheme. At the end of this phase, $\mathcal{S}$ obtains the output $((\widetilde{\text{trans}}_1', \gamma_1), \ldots, (\widetilde{\text{trans}}_{20n}', \gamma_{20n}))$ and receives from the adversary $20n$ random strings $b_{\text{Sen}}^1, \ldots, b_{\text{Sen}}^{20n}$.

2. In Stage 2, the simulator participates with the adversary in $20n$ executions of the OT protocol $\widehat{\pi}_{\text{OT}}$, while playing the role of the honest sender. Note that due to the fact that the simulator knows the values of the input and randomness that the honest receiver must use in each of the OT executions, the simulator can identify the coordinates of which the receiver deviates, in which case the receiver learns both the inputs of the sender. We denote this set of coordinates by the set $\Phi$.

3. In Stage 3, the simulator picks $n$ random numbers $(q_{\text{Sen}}^1, \ldots, q_{\text{Sen}}^n)$ from $\{1, \ldots, 20\}^n$ and sends them to the receiver. Upon receiving the decommitments from the receiver, the simulator verifies the decommitments as would the honest sender do with respect to $\Gamma_{\text{Sen}}$ and halts in case of a mismatch, outputting the simulated transcript thus far. Next, it receives $(q_{\text{Rec}}^1, \ldots, q_{\text{Rec}}^n)$ from the receiver and decommits the subset of values that corresponds to the coordinates in $\Gamma_{\text{Rec}}$ as determined by $(q_{\text{Rec}}^1, \ldots, q_{\text{Rec}}^n)$, playing the role of the sender. Finally, it emulates functionality $\mathcal{F}_{\text{EXTCOM}}$ and invokes the commitment scheme $\omega_L$ with $\mathcal{A}$ (that plays the role of the committer) $n$ times, obtaining $((\text{trans}_1', \Gamma_1), \ldots, (\text{trans}_{20n}', \Gamma_{20n}))$. Let $\Delta = [20n] - \Gamma_{\text{Rec}} - \Gamma_{\text{Sen}}$.

4. In Stage 4, the simulator proceeds as follows. Observe first that $\Phi$ and $\Gamma_{\text{Sen}}$ are disjoint, since otherwise the simulator would have halted in the previous stage. We consider three cases here:

    (a) $\Phi \geq n$: In this case, the simulator halts and outputs fail.
    (b) $\Phi < n$: This implies that $\Delta - \Phi - \Gamma > 16n$ where by definition, the malicious receiver proceeds according to the honest OT receiver's strategy with respect to every coordinate in $\Delta - \Phi - \Gamma$. Note that in this case the adversary learns at most $\Delta + 2\Phi + 2\Gamma < 20n$ distinct shares of both the sender's inputs and the simulator knows precisely which share is learned for every coordinate relative to the set $\Delta - \Phi - \Gamma$. We consider two subcases:

        i. There exists a bit $u \in \{0, 1\}$ for which the adversary learns $10n$ shares of $v_u$. We recall that the adversary might learn both shares for the OT executions that correspond to the coordinates within the set $\Phi \cup \Gamma$ and exactly one share for every OT execution that corresponds to the coordinates within $\Delta - \Phi - \Gamma$. Now, since the simulator knows which share is learned for every coordinate within $\Delta - \Phi - \Gamma$, it can also compute an upper bound on the number of $v_u$ shares that are obtained by the receiver. In this case, the simulator forwards $u$ to the ideal functionality $\mathcal{F}_{\text{OT}}$ and receives back $v_u$. The simulator then sets $v_{1-u}$ as a random string of the appropriate length and completes the interaction by playing the role of the honest sender on these inputs.

ii. There does not exist a bit $u \in \{0, 1\}$ for which the adversary learns $10n$ shares of $v_u$. In this case, the simulator picks two random values for $v_0$ and $v_1$ of the appropriate length and completes the interaction, playing the role of the honest sender on these values. Note that in this case the simulator does not call the ideal functionality $\mathcal{F}_{\mathrm{OT}}$.

5. The simulator completes the simulation in Stage 5 similarly to Stage 3.

Note that the only difference between the simulation and the real execution is in the way the messages in Stage 4 are generated. In the simulation, a value $u$ is extracted from the malicious receiver and then fed to the $\mathcal{F}_{\mathrm{OT}}$ functionality, where the simulation is completed based on the output returned from the functionality. Furthermore, we recall that in Stage 5 the receiver learns both of the sender's inputs in all sessions $i \in \Gamma$; then, it holds that the receiver learns one such input for every session it behaved honestly and two inputs for all sessions it deviates or included in $\Gamma$. Proving that the event, for which the adversary deviates in more than $n$ OT executions, only happens with negligible probability, implies that it learns less than $20n$ shares in total. Therefore, at least one of the shared secrets is completely hidden due to the $10n$-out-of-$18n$ secret sharing scheme. To complete the simulation, the simulator identifies which of the two values $v_0$ and $v_1$ is learnt by the receiver (by identifying how many shares are obtained by that party) and fixes that to be the receiver's input. Finally, indistinguishability follows from the defensible privacy with respect to the receiver of the OT protocol.

More formally, we begin with a proof that the probability that the simulator returns fail is negligible and then neglect this event. Namely, we prove that the simulated and real views are computationally indistinguishable conditioned on the even that the simulator did not output fail.

**Claim 3.1.** *The probability $\mathcal{S}$ return* fail *is negligible.*

*Proof.* Note first that the only place where the simulator fails is in Step 4a, when $\phi \geq n$. We now show that this event occurs with negligible probability. In other words, we need to show that the probability that the corrupted receiver deviates from the honest receiver's strategy in at least $n$-out-of-$18n$ OT executions while not getting caught by the sender is negligible. Formally, let Bad denote the event for which the corrupted receiver deviates in least $n$ coordinates. Note first that the simulator can easily identify when event Bad occurs since it knows the random tapes and the inputs the receiver must use in all executions, and can therefore identify the coordinates for which the receiver deviates. Next, we show that conditioned on event Bad occurring, the probability that $\Gamma_{\mathrm{Sen}}$ does not contain one of the $n$ deviated coordinates is negligible. This implies that the probability that $\mathcal{S}$ returns fail is negligible.

Denote by $\Phi$ the set of $n$ coordinates in which the receiver deviates and define the bins $A_j = \{20(j-1)+1, \ldots, 20j\}$ for all $j \in [n]$. By the pigeonhole principle it holds that at least $\lfloor n/20 \rfloor$ bins intersect with $\Phi$. In addition, we recall that $\Gamma_{\mathrm{Sen}}$ is chosen by the simulator by picking one element from each bin independently of $\Phi$ and uniformly at random. Then the probability that $\Gamma_{\mathrm{Sen}} \cap \Phi = \emptyset$ is at most $(19/20)^{\lfloor n/20 \rfloor}$ which is negligible in $n$. This concludes the proof of the claim.                                              □

Next, we prove that the receiver's view for both executions is computationally indistinguishable assuming that the simulator did not abort the execution. More formally, denoting the simulated execution by $\pi_{\text{IDEAL}}$, then we prove the following statement,

**Claim 3.2.** *The following two distribution ensembles are computationally indistinguishable,*

$$\left\{\mathbf{View}^{\mathcal{F}_{\text{EXTCOM}}}_{\pi^{\text{ML}}_{\text{OT}}, \mathcal{A}, \mathcal{Z}}(n)\right\}_{n \in \mathbb{N}} \stackrel{c}{\approx} \left\{\mathbf{View}^{\mathcal{F}_{\text{OT}}}_{\pi_{\text{IDEAL}}, \mathcal{S}, \mathcal{Z}}(n)\right\}_{n \in \mathbb{N}}.$$

*Proof.* The security argument proceeds in a sequence of hybrids games starting from the simulated execution toward the real execution. We denote by $\pi_{\text{HYBRID}_i}$ the receiver's view in the $i$th hybrid game.

*Hybrid 1* In this game, we define a simulator $\mathcal{S}_1$ that is identical to $\mathcal{S}$ except for the way the sender's message is generated in Stage 4. More precisely, the simulator modifies the way $\beta_i^{1 \oplus u}$ is computed for all $i \in \Delta - \Gamma - \Phi$. Recall first that $\mathcal{S}$ sets

$$\beta_i^{1 \oplus u} = \rho_i^{1 \oplus u} \oplus s_i^{(1 \oplus u) \oplus \alpha_i} = \rho_i^{1 \oplus u} \oplus s_i^{1 \oplus c_i}.$$

Instead, $\mathcal{S}_1$ will choose $\beta_i^{1 \oplus u}$ at random, which can be viewed as using a masking element that is independent of $s_i^{1 \oplus c_i}$. Intuitively, we claim that the simulated view and the view generated in hybrid 1 are computationally indistinguishable because for every $i \in \Delta - \Gamma - \Phi$ the receiver generates the OT messages in session $i$ of Stage 2 honestly (i.e., using the input $c_i$ and random tape $\tau^i_{\text{Rec}}$), where by the defensible privacy of the receiver it cannot distinguish the input $s_i^{1 \oplus c_i}$ from a random input.

*Proof Intuition* The goal in this hybrid is to remove the real input from the receiver. The idea is that in all parallel OT executions where the sender does not cheat, by the defensible privacy, the sender should not be able to identify the receiver's input. The specific executions can be identified by the set $\Delta - \Gamma - \Phi$. In order to carry of the security reduction, we need to reduce a cheating sender to a sender that violates the defensible privacy of the underlying OT protocol. In the main sequence of hybrids $H_1^1, H_1^2, \ldots$, we change the inputs of the receiver in the executions corresponding to $\Delta - \Gamma - \Phi$ one at a time. To argue indistinguishability between the $i$th and $i + 1$st hybrids, we need to do two things. First we need to decouple the receiver's actions in these execution (that are simulated) from the coin-tossing stage. We can rely on the hiding of the commitment for this; however, to carry out this reduction we need to guess which is the $i$th index in $\Delta - \Gamma - \Phi$. Toward this, we consider a variant of the hybrid $\bar{H}_e^i$ and $\bar{H}_e^{i+1}$ where we make a guess and isolate the indistinguishability on the guessed coordinate. Next, we consider a nested hybrid $\tilde{H}_1^i$ and $\tilde{H}_1^{i+1}$ where we rely on the hiding of the commitment scheme by setting the receivers commitments to 0 in the coin-tossing stage. Formally, we prove this in the following claim.

**Claim 3.3.** *The following two distribution ensembles are computationally indistinguishable,*

$$\left\{\mathbf{View}_{\pi_{\mathrm{IDEAL}},\mathcal{S},\mathcal{Z}}^{\mathcal{F}_{\mathrm{OT}}}(n)\right\}_{n\in\mathbb{N}} \equiv \left\{\mathbf{View}_{\pi_{\mathrm{HYBRID}_1},\mathcal{S}_1,\mathcal{Z}}^{\mathcal{F}_{\mathrm{OT}}}(n)\right\}_{n\in\mathbb{N}}.$$

*Proof.* Toward proving this claim, we introduce a sequence of intermediate hybrid experiments $H_1^e$ for $e = 0, \ldots, 18n$, where in hybrid $H_1^e$ we consider a simulator $\mathcal{S}_1^e$ that proceeds identically to $\mathcal{S}$ with the exception that it follows $\mathcal{S}_1$'s strategy for the first $e$ indices in $\Delta - \Gamma - \Phi$ regarding the generation of $\beta_i^{1\oplus u}$ (i.e., for the first $e$ sessions where the receiver proceeded honestly). By definition, we have that experiment $H_1^0$ proceeds identically to the ideal simulation and $H_1^{18n}$ proceeds identically to hybrid 1. Denote the view output in hybrid $H_1^e$ by $\mathsf{hyb}_e(n)$ and assume by contradiction that there exist an adversary $\mathcal{A}$ (controlled by $\mathcal{Z}$), a distinguisher $D$, a polynomial $p(\cdot)$ and infinitely many $n$'s such that

$$\left| \Pr[D(\mathbf{View}_{\pi_{\mathrm{IDEAL}},\mathcal{S},\mathcal{Z}}^{\mathcal{F}_{\mathrm{OT}}}(n)) = 1] - \Pr[D(\mathbf{View}_{\pi_{\mathrm{HYBRID}_1},\mathcal{S},\mathcal{Z}}^{\mathcal{F}_{\mathrm{OT}}}(n)) = 1] \right| \geq \frac{1}{p(n)}.$$

Using a standard hybrid argument it follows that there exists an $e \in [18n]$ such that

$$\left| \Pr[D(\mathsf{hyb}_e(n)) = 1] - \Pr[D(\mathsf{hyb}_{e-1}(n)) = 1] \right| \geq \frac{1}{18np(n)}.$$

Next, we plan to exploit the above observation in order to construct a defensible adversary $\mathcal{A}'$ that violates the receiver's defensible privacy relative to $\widehat{\pi}_{\mathrm{OT}}$ in the sense of Definition 2.6. At a high level, $\mathcal{A}'$ picks a random $j \in [20n]$ and externally forwards $\mathcal{A}$'s messages within the $j$th execution of the OT protocol, where $j$ serves as the guess for the $e$th execution in $\Delta - \Gamma - \Phi$. $\mathcal{A}'$ then emulates the rest of the OT executions, playing the role of the sender. In order to simplify the analysis and allow $\mathcal{A}'$ carry out the reduction properly (where the generated randomness within the coin-tossing phase is disassociated from the OT executions), we consider the following additional hybrid executions.

First, we consider a slight variation of $H_i^{e-1}$ (resp., $H_i^e$) denoted by $\overline{H}_{e-1}$ (resp., $\overline{H}_e$), and a random variable $J$ that denotes a randomly chosen index from $[20n]$ which is picked at the onset of the hybrid execution. Moreover, the experiment is aborted if chosen index does not correspond to the $e$th execution in $\Delta - \Gamma - \Phi$. We say that index $J$ is Bad if the experiment aborts. Note that experiments $\overline{H}_{e-1}$ and $\overline{H}_e$ proceed identically to $H_{e-1}$ and $H_e$, respectively, conditioned on $J$ not being Bad. This is due to the fact that $J$ is chosen independently of the experiments. Moreover, relying on the fact that the $e$th execution can take at most $20n$ values we have that

$$\Pr[J \text{ is not Bad}] = \frac{1}{20\mathrm{n}}.$$

Therefore, if $\overline{\mathsf{hyb}}_{e-1}(n)$ and $\overline{\mathsf{hyb}}_e(n)$, respectively, correspond to $\mathcal{A}$'s views in $\overline{H}_{e-1}$ and $\overline{H}_e$, then

$$\Pr[D(\overline{\mathsf{hyb}}_e(n)) = 1] = \Pr[D(\overline{\mathsf{hyb}}_e(n)) = 1 \wedge J \text{ not Bad}]$$
$$+ \Pr[D(\overline{\mathsf{hyb}}_e(n)) = 1 \wedge J \text{ is Bad}]$$
$$= \left(\frac{1}{20n}\right)\Pr[D(\overline{\mathsf{hyb}}_e(n)) = 1 \mid J \text{ not Bad}]$$
$$+ \left(1 - \frac{1}{20n}\right)\Pr[D(\overline{\mathsf{hyb}}_e(n)) = 1 \mid J \text{ is Bad}]$$
$$= \left(\frac{1}{20n}\right)\Pr[D(\mathsf{hyb}_e(n)) = 1] + \left(1 - \frac{1}{20n}\right)\Pr[D(\bot) = 1].$$

Similarly,

$$\Pr[D(\overline{\mathsf{hyb}}_{e-1}(n)) = 1] = \left(\frac{1}{20n}\right)\Pr[D(\mathsf{hyb}_{e-1}(n)) = 1] + \left(1 - \frac{1}{20n}\right)\Pr[D(\bot) = 1].$$

Therefore,

$$\left| \Pr[D(\overline{\mathsf{hyb}}_e(n)) = 1] - \Pr[D(\overline{\mathsf{hyb}}_{e-1}(n)) = 1] \right|$$
$$= \left(\frac{1}{20n}\right)\left| \Pr[D(\mathsf{hyb}_e(n)) = 1] - \Pr[D(\mathsf{hyb}_{e-1}(n)) = 1] \right|$$
$$\geq \left(\frac{1}{20n}\right)\frac{1}{18np(n)}. \tag{1}$$

Before we provide the description of $\mathcal{A}'$, we consider our second modification and define hybrids $\widetilde{H}_{e-1}$ and $\widetilde{H}_e$ as follows. Namely, in these new experiments we slightly modify the sender's messages in the coin-tossing phase and ask the sender to commit to the all zeros string of appropriate length instead of committing to a uniform string $a_{\mathsf{Sen}}^j$. Recalling that $a_{\mathsf{Sen}}^j$ and $b_{\mathsf{Sen}}^j$ determine the sender's input in the $j$th execution of the OT protocol, we instruct the sender to commit to 0 so that $\mathcal{A}'$ can forward the $j$th's execution messages to an external OT sender in the reduction described next. More precisely, $\widetilde{H}_{e-1}$ (resp., $\widetilde{H}_e$) follows exactly as $\overline{H}_{e-1}$ (resp., $\overline{H}_e$) with the exception that we modify the honest sender's message in the coin-tossing stage, where it commits to the all zeros string instead of $a_{\mathsf{Sen}}^j$. Observe that this change does not affect the cut-and-choose phase where the sender is required to reveal randomness for indices in $\Gamma_{\mathsf{Rec}}$ because if $J \in \Gamma_{\mathsf{Rec}} \wedge \Gamma$ then the experiment is aborted by definition. Denote by the random variables $\widetilde{\mathsf{hyb}}_e(n)$ and $\widetilde{\mathsf{hyb}}_{e-1}(n)$ the views of adversary $\mathcal{A}$ in $\widetilde{H}_{e-1}$ and $\widetilde{H}_e$, respectively. Then from the computational hiding property of $\omega_L$ the commitment scheme used in the coin-tossing stage it follows that there exists a negligible function $\nu(\cdot)$ such that for all sufficiently large $n$'s,[9]

$$\left| \Pr[D(\overline{\mathsf{hyb}}_e(n)) = 1] - \Pr[D(\widetilde{\mathsf{hyb}}_e(n)) = 1] \right| \leq \nu(n) \tag{2}$$

---

[9] We remark here that we rely on the security of the commitment scheme $\omega_L$ against receivers with auxiliary input as in Section 2.4.

$$\left| \Pr[D(\overline{\mathsf{hyb}}_{e-1}(n)) = 1] - \Pr[D(\widetilde{\mathsf{hyb}}_{e-1}(n)) = 1] \right| \leq \nu(n) . \tag{3}$$

Using Eq. 1 we obtain that for all sufficiently large $n$'s,

$$\left| \Pr[D(\widetilde{\mathsf{hyb}}_{e}(n)) = 1] - \Pr[D(\widetilde{\mathsf{hyb}}_{e-1}(n)) = 1] \right| \geq \frac{1}{(18n)20np(n)} - 2\nu(n) \geq \frac{1}{q(n)}, \tag{4}$$

where $q(\cdot)$ is the polynomial $2 \times 18(20n)np(n)$. Fix an $n$ such that this happens. We now show how to define $\mathcal{A}'$ and distinguisher $D'$ that violate the defensible private with respect to a corrupted receiver in $\pi_{\mathrm{OT}}^{\mathrm{ML}}$. More specifically, $\mathcal{A}'$ internally emulates experiment $\widetilde{H}_1^{e-1}$ by running the simulation strategy of $\mathcal{S}_1^{e-1}$ with the malicious receiver $\mathcal{A}$. Let $(c_J, \tau_{\mathrm{Rec}}^J)$ denote the input and randomness that the honest receiver is supposed to use in the internal $J$th execution. Recall that this is determined by $a_{\mathrm{Rec}}^J \oplus b_{\mathrm{Rec}}^J$ and is known to the simulator, as it extracts the adversary's commitments. Next $\mathcal{A}'$ plays the role of the sender in the executions of $\widehat{\pi}_{\mathrm{OT}}$ with the exception that it externally relays the messages of the adversary (acting as the receiver) in the $J$th execution of the oblivious transfer protocol from Stage 2. Following the oblivious transfer executions, $\mathcal{A}'$ continues the internal emulation until the end of Stage 3. If the experiment aborts in the internal emulation (where this happens if $J$ is Bad), then $\mathcal{A}'$ aborts. Otherwise, there is a good defense for the receiver in the $J$th execution, namely $(c_J, \tau_{\mathrm{Rec}}^J)$. Let $\mathsf{STATE}$ be the complete view of experiment $\widetilde{H}_{e-1}$ which includes the input and random tape of $\mathcal{A}$ and the simulator (playing the sender), as well as the partial transcript of the messages exchanged with $\mathcal{A}$ until Stage 3. $\mathcal{A}'$ outputs $(c_J, \tau_{\mathrm{Rec}}^J)$ as its defense and $\mathsf{STATE}$ as its output.

Upon receiving $(\mathsf{view}, s)$, where $\mathsf{view}$ is $\mathcal{A}'$'s view and $s$ is a string (as specified in Definition 2.6), distinguisher $D'$ proceeds as follows. It first extracts state $\mathsf{STATE}$ from the view and then completes the internal emulation of the experiment by playing the role of the sender in Stages 4 and 5. We note that $D'$ has all the information it needs as part of $\mathsf{STATE}$ to complete the execution, except for the sender's inputs $(s_J^0, s_J^1)$ that are required to compute $\beta_J^0$ and $\beta_J^1$ in Stage 4. Note also that the distinguisher can use $\mathcal{A}'$'s valid defense $(c_J, \tau_{\mathrm{Rec}}^J)$ to compute one of the two sender's inputs, namely $s_J^{c_J}$. For the other input, $D'$ uses $s$, i.e., it sets $s_J^{1 \oplus c_J} = s$ and completes the experiment using these inputs. Finally, $D'$ invokes $D$ on $\mathcal{A}$'s view and outputs whatever $D$ outputs. It follows from the construction that the view on which $D$ is invoked is distributed identically to $\mathcal{A}$'s view in $\widetilde{\mathsf{hyb}}_{e-1}(n)$ if $s$ is the sender's other input, namely $s_J^{1 \oplus c_J}$. On the other hand, if $s$ is a random string then the view is distributed identically to $\widetilde{\mathsf{hyb}}_e(n)$.

$$D'(\Gamma(\mathbf{View}_{\mathcal{A}}[\mathrm{Sen}(1^n, (U_0^n, U_1^n)), \mathcal{A}(1^n)], U_{1-b}^n)) = D(\widetilde{\mathsf{hyb}}_{e-1}(n))$$

and

$$D'(\Gamma(\mathbf{View}_{\mathcal{A}}[\mathrm{Sen}(1^n, (U_0^n, U_1^n)), \mathcal{A}(1^n)], \bar{U}^n)) = D(\widetilde{\mathsf{hyb}}_e(n)),$$

where $\Gamma(v, *) = (v, *)$ if $v$ contains a valid defense for $\mathcal{A}'$. From Equation 4, it follows that the difference is non-negligible and that $\mathcal{A}'$ and $D'$ contradict the defensible privacy

of protocol $\widehat{\pi}_{\mathrm{OT}}$ with respect to a corrupted receiver. This concludes the proof of the claim. □

*Hybrid 2* In this hybrid game, there is no trusted party that computes functionality $\mathcal{F}_{\mathrm{OT}}$. Instead, we define a simulator $\mathcal{S}_2$ that is given the sender's real inputs $v_0$ and $v_1$. Furthermore, $\mathcal{S}_2$ uses these inputs in Stage 5 of the execution. Then we claim that the receiver's view in Hybrids 1 and 2 is statistically close because the probability that the receiver learns more than $10n$ shares for both $u = 0$ and $u = 1$ is negligible. More formally,

**Claim 3.4.** *The following two distribution ensembles are statistically indistinguishable,*

$$\left\{\mathbf{View}_{\pi_{\mathrm{HYBRID}_1},\mathcal{S}_1,\mathcal{Z}}^{\mathcal{F}_{\mathrm{OT}}}(n)\right\}_{n\in\mathbb{N}} \stackrel{\mathrm{s}}{\approx} \left\{\mathbf{View}_{\pi_{\mathrm{HYBRID}_2},\mathcal{S}_2,\mathcal{Z}}^{\mathcal{F}_{\mathrm{EXTCOM}}}(n)\right\}_{n\in\mathbb{N}}.$$

*Proof.* This follows from the facts that $|\Delta| = 18n$, $|\Gamma| = n$ and $|\Phi| \leq n$ with overwhelming probability (relying on the proof of Claim 3.2), and that the masking values that are used in Stage 5 are independent of the input to the OT executions in Stage 4. Specifically, the overall number of shares that the receiver learns is bounded by $|\Delta - \Gamma - \Phi| + 2|\Gamma| + 2|\Phi| \leq 20n$, where the rest of the shares are perfectly hidden (as their masking strings are not used elsewhere in the protocol). □

*Hybrid 3* In this game, we define a simulator $\mathcal{S}_3$ that is identical to $\mathcal{S}_2$ except for the way the sender's message is generated in Stage 4. More precisely, for all $i \in \Delta - \Gamma - \Phi$ it modifies the way $\beta_i^{1\oplus u}$ is computed. Recalling that $\mathcal{S}_2$ sets it to be random, then $\mathcal{S}_3$ will instead set

$$\beta_i^{1\oplus u} = \rho_i^{1\oplus u} \oplus s_i^{1\oplus c_i}.$$

Indistinguishability of hybrids 2 and 3 follows using the same proof as in Claim 3.3. Therefore, we have that the following ensembles are computationally indistinguishable.

$$\left\{\mathbf{View}_{\pi_{\mathrm{HYBRID}_2},\mathcal{S}_2,\mathcal{Z}}^{\mathcal{F}_{\mathrm{OT}}}(n)\right\}_{n\in\mathbb{N}} \stackrel{\mathrm{c}}{\approx} \left\{\mathbf{View}_{\pi_{\mathrm{HYBRID}_3},\mathcal{S}_3,\mathcal{Z}}^{\mathcal{F}_{\mathrm{EXTCOM}}}(n)\right\}_{n\in\mathbb{N}}.$$

Observe that hybrid 3 is identical to the real execution. This concludes the proof of Claim 3.2. □

*Simulating the corrupted sender:* In this case, $\mathcal{S}$ proceeds as follows:

1. $\mathcal{S}$ picks $20n$ random strings $a_{\mathrm{Rec}}^1, \ldots, a_{\mathrm{Rec}}^{20n}$ and emulates the ideal functionality $\mathcal{F}_{\mathrm{EXTCOM}}$ by invoking $20n$ times the commitment phase of $\omega_R$ with inputs $a_{\mathrm{Rec}}^1, \ldots, a_{\mathrm{Rec}}^{20n}$, against $\mathcal{A}$ that plays the role of receiver for the commitment scheme. At the end of this phase, $\mathcal{S}$ obtains the output $((\widetilde{\mathrm{trans}}_1, \gamma_1), \ldots, (\widetilde{\mathrm{trans}}_{20n}, \gamma_{20n}))$ and receives from the adversary $20n$ random strings $b_{\mathrm{Rec}}^1, \ldots, b_{\mathrm{Rec}}^{20n}$.

Next, $\mathcal{S}$ emulates functionality $\mathcal{F}_{\text{EXTCOM}}$ in Stage 1 and invokes $20n$ times the commitment scheme $\omega_L$ with $\mathcal{A}$ (that plays the role of the committer), obtaining $((\widetilde{\text{trans}'_1}, a^1_{\text{Sen}}), \ldots, (\widetilde{\text{trans}'_{20n}}, a^{20n}_{\text{Sen}}))$. It internally records $a^1_{\text{Sen}}, \ldots, a^{20n}_{\text{Sen}}$ and further picks $20n$ random strings $b^1_{\text{Sen}}, \ldots, b^{20n}_{\text{Sen}}$, forwarding them to the adversary. The simulator also computes $r^i_{\text{Sen}} = a^i_{\text{Sen}} \oplus b^i_{\text{Sen}}$ and then views $r^i_{\text{Sen}} = s^0_i || s^1_i || \tau^i_{\text{Sen}}$; $(s^0_i, s^1_i)$ is the input an honest sender must use in the $i$th OT protocol execution in Stage 3, together with randomness $\tau^i_{\text{Sen}}$.

2. In Stage 2, the simulator participates with the adversary in $20n$ executions of the OT protocol $\widehat{\pi}_{\text{OT}}$, while playing the role of the honest receiver. Note that due to the fact that the simulator knows the values of the input and randomness that the honest sender must use in each of the OT executions, the simulator can identify the coordinates of which the sender deviates. We denote this set of coordinates by the set $\Phi$.

3. In Stage 3, $\mathcal{S}$ receives $(q^1_{\text{Sen}}, \ldots, q^n_{\text{Sen}})$ from the sender and decommits the subset of values that corresponds to the coordinates in $\Gamma_{\text{Sen}}$ as determined by $(q^1_{\text{Sen}}, \ldots, q^n_{\text{Sen}})$, playing the role of the receiver. Next, the simulator picks $n$ random numbers $(q^1_{\text{Rec}}, \ldots, q^n_{\text{Rec}})$ from $\{1, \ldots, 20\}^n$ and sends them to the sender. Upon receiving the decommitments from the sender, the simulator verifies the decommitments as would the honest receiver do with respect to $(q^1_{\text{Rec}}, \ldots, q^n_{\text{Rec}})$ and halts in case of a mismatch, outputting the simulated transcript thus far. Finally, it samples a subset $\Gamma$ from $[20n]$ of size $n$ and emulates functionality $\mathcal{F}_{\text{EXTCOM}}$ by invoking the commitment scheme $\omega_R$ with $\mathcal{A}$ (that plays the role of the receiver) $n$ times on input $\Gamma$, obtaining $((\text{trans}'_1, \gamma_1), \ldots, (\text{trans}'_{20n}, \gamma_{20n}))$. Let $\Delta = [20n] - \Gamma_{\text{Rec}} - \Gamma_{\text{Sen}} - \Phi$.

4. In Stage 4, the simulator proceeds as the honest receiver would do with input $u = 0$ and extracts the sender's inputs $v_0, v_1$. Specifically, the simulator knows all the inputs $\{(s^0_i, s^1_i)\}_{i \in \Delta}$ of the sender to the OT executions in Stage 2 and extracts the two sets of shares $\{\rho^0_i\}_{i \in \Delta}$ and $\{\rho^1_i\}_{i \in \Delta}$.

5. In Stage 5, the simulator plays the role of the honest receiver and checks whether the inputs and randomness revealed by the sender are consistent with the OT session that correspond to $\Delta \cap \Gamma$. In case of a mismatch the simulator halts, outputting the simulated transcript thus far. Next, the simulator checks that $\tilde{\rho}_0$ and $\tilde{\rho}_1$ agree with some respective code words $w_0$ and $w_1$ on $16n$ locations. In case of a non-agreement, the simulator records a default value; else, it records the code words $w_0$ and $w_1$. It then runs a second consistency check to verify whether these code words agree with $\beta^u_j \oplus s^{u \oplus \alpha_j}_j$ for all coordinates $j \in \Gamma$. If not, it records a default value. Finally, the simulator sends the recorded values to $\mathcal{F}_{\text{OT}}$.

We next prove the following,

**Claim 3.5.** *The following two distribution ensembles are computationally indistinguishable,*

$$\left\{ \mathbf{View}^{\mathcal{F}_{\text{EXTCOM}}}_{\pi^{\text{ML}}_{\text{OT}}, \mathcal{A}, \mathcal{Z}}(n) \right\}_{n \in \mathbb{N}} \stackrel{c}{\approx} \left\{ \mathbf{View}^{\mathcal{F}_{\text{OT}}}_{\pi_{\text{IDEAL}}, \mathcal{S}, \mathcal{Z}}(n) \right\}_{n \in \mathbb{N}}.$$

*Proof.* The security argument proceeds in a sequence of hybrid games starting from the simulated execution toward the real execution. We denote by $\pi_{\mathrm{HYBRID}_i}$ the receiver's view in the $i$th hybrid game.

**Hybrid 1:** In this hybrid game, there is no trusted party that computes functionality $\mathcal{F}_{\mathrm{OT}}$. Instead, we define a simulator $\mathcal{S}_1$ that is given the receiver's real input $u$ and proceeds identically as $\mathcal{S}$ except for the way it generates the receiver's message in Stage 4. More precisely, $\mathcal{S}_1$ uses the real input $u$ instead of 0 in order to compute $\alpha_i$ for all $i \in \Delta - \Phi$. Indistinguishability of the simulation from the view in hybrid 1 follows from the receiver privacy of the OT protocol.

We follow an approach similar to the proof of Claim 3.3 where we consider a sequence of hybrids $H_1^1, H_1^2, \ldots$ where we replace the sender's input in the executions in the parallel OT executions where the receiver proceeded honestly. Then we decouple actions of the sender from the coin-tossing stage by considering nested hybrids. Finally, we need an additional step where after replacing the sender's input from the executions in $\Delta - \Phi$ we rely on the secret sharing scheme to conclude that one of the two sender's inputs has been removed.

More formally, we prove the following claim.

**Claim 3.6.** *The following two distribution ensembles are computationally indistinguishable,*

$$\left\{\mathbf{View}^{\mathcal{F}_{\mathrm{OT}}}_{\pi_{\mathrm{IDEAL}},\mathcal{S},\mathcal{Z}}(n)\right\}_{n\in\mathbb{N}} \stackrel{c}{\approx} \left\{\mathbf{View}^{\mathcal{F}_{\mathrm{OT}}}_{\pi_{\mathrm{HYBRID}_1},\mathcal{S}_1,\mathcal{Z}}(n)\right\}_{n\in\mathbb{N}}.$$

*Proof.* Recall that the only difference between hybrid 1 and the simulated view is in the way that the messages in Stage 4 are generated. Specifically, in the simulated view $\mathcal{S}$ uses $u = 0$ in all sessions to compute $\alpha_i$. On the other hand, in game hybrid 1 the receiver uses the real $u$. Clearly, if the real input equals 0 then the views are identical and the proof of the claim follows immediately. Therefore, it suffices to consider the case $u = 1$. Toward proving this claim, we introduce a sequence of intermediate hybrid experiments $H_1^e$ for $e = 0, \ldots, 20n$. Namely, in hybrid $H_1^e$ we consider a simulator $\mathcal{S}^e$ that proceeds identically to $\mathcal{S}$ with the exception that it uses $u = 1$ in the first $e$ sessions in $\Delta$ in order to compute $\alpha_i$. By construction, we have that the experiment $H_1^0$ proceeds identically to the ideal simulation and $H_1^{20n}$ proceeds identically to hybrid 1. Denote the view output in hybrid $H_1^e$ by $\mathsf{hyb}_e(n)$ and assume by contradiction that there exist a distinguisher $D$, a polynomial $p(\cdot)$ and infinitely many $n$'s such that

$$|\Pr[D(\mathbf{View}^{\mathcal{F}_{\mathrm{OT}}}_{\pi_{\mathrm{IDEAL}},\mathcal{S},\mathcal{Z}}(n)) = 1] - \Pr[D(\mathbf{View}^{\mathcal{F}_{\mathrm{OT}}}_{\pi_{\mathrm{HYBRID}_1},\mathcal{S}_1,\mathcal{Z}}(n)) = 1]| \geq \frac{1}{p(n)} .$$

Using a standard hybrid argument, it follows that there exists an $e \in \{1, \ldots, 20n\}$ such that

$$|\Pr[D(\mathsf{hyb}_e(n)) = 1] - \Pr[D(\mathsf{hyb}_{e-1}(n)) = 1]| \geq \frac{1}{20np(n)} .$$

As in the proof of Claim 3.3, we consider experiments $\overline{H}_{e-1}$ and $\overline{H}_e$ where the simulator samples a random $J \in [3n+1]$ and aborts if $J$ is not the $e$th session in $\Delta$. Next, we consider modified hybrids $\widetilde{H}_{e-1}$ and $\widetilde{H}_e$, where we slightly modify the receiver's messages in the coin-tossing phase and ask the receiver to commit to the all zeros string of appropriate length instead of committing to a uniform string $a_{\mathrm{Rec}}^J$. Recalling that $a_{\mathrm{Rec}}^J$ and $b_{\mathrm{Rec}}^J$ determine the receiver's input in the $J$th execution of the OT protocol, we instruct the receiver to commit to 0 so that in the reduction we explain next we can forward the $J$th's execution messages to an external OT receiver in the reduction described next. More precisely, $\widetilde{H}_{e-1}$ (resp., $\widetilde{H}_e$) follows exactly as $\overline{H}_{e-1}$ (resp., $\overline{H}_e$) with the exception that we modify the honest receiver's message in the coin-tossing stage, where it commits to the all zeros string instead of $a_{\mathrm{Rec}}^j$. Denote by the random variables $\widetilde{\mathsf{hyb}}_e(n)$ and $\widetilde{\mathsf{hyb}}_{e-1}(n)$ the views of adversary $\mathcal{A}$ in $\widetilde{H}_{e-1}$ and $\widetilde{H}_e$, respectively. Then following the same proof as in Claim 3.3, we can conclude that there exists a polynomial $q(\cdot)$ such that

$$\left| \Pr[D(\widetilde{\mathsf{hyb}}_e(n)) = 1] - \Pr[D(\widetilde{\mathsf{hyb}}_{e-1}(n)) = 1] \right| \geq \frac{1}{q(n)}. \tag{5}$$

Without loss of generality, assume that

$$\Pr[D(\widetilde{\mathsf{hyb}}_e(n)) = 1] - \Pr[D(\widetilde{\mathsf{hyb}}_{e-1}(n)) = 1] \geq \frac{1}{q(n)} .$$

We use the above to construct a malicious sender $\mathcal{A}'$ that violates the privacy of the receiver relative to the oblivious transfer protocol $\widehat{\pi}_{\mathrm{OT}}$. Specifically, $\mathcal{A}'$ internally emulates the experiment $\widetilde{H}_1^{e-1}$ by running the simulation strategy of $\mathcal{S}_1^{e-1}$ with the malicious sender $\mathcal{A}$, except for the following difference. $\mathcal{A}'$ relays the messages of the sender in the $J$th execution of the oblivious transfer protocol from Stage 2 to an external receiver with input $c$. Following the oblivious transfer executions, it continues the internal emulation until Stage 4. If $J$ is not the $e$th session in $\Delta$, then $\mathcal{A}'$ follows hybrid $\widetilde{H}_1^{e-1}$ and sets the view as $\perp$. Otherwise, set $\alpha_J$ to be a random bit and continue the internal emulation to completion. It then invokes the distinguisher $D$ on $\mathcal{A}$'s internally generated view; denote by $b$ the bit output by $D$. Then, $\mathcal{A}'$ outputs $\alpha_j \oplus b$.

We proceed by analyzing the probability that $\mathcal{A}'$ correctly guesses $c$. Conditioned on not outputting $\mathsf{fail}$ and $\alpha_j \neq c$, the experiment emulated internally by $\mathcal{A}$ is identical to $\widetilde{H}_1^e$. Analogously, conditioned on not outputting $\mathsf{fail}$ and $\alpha_j = c$, the experiment emulated internally by $\mathcal{A}$ is identical to $\widetilde{H}_1^{e-1}$, where the probability that $\alpha_j = c$ (and $\alpha_j \neq c$) is $\frac{1}{2}$. Therefore,

$$\Pr[\, \mathcal{A}' \text{ guesses } c \text{ correctly} \,]$$
$$= \frac{1}{2} \Pr[D(\widetilde{\mathsf{hyb}}_e(n)) \oplus \alpha_j = c \mid \alpha_j \neq c\,]$$
$$+ \frac{1}{2} \Pr[D(\widetilde{\mathsf{hyb}}_{e-1}(n)) \oplus \alpha_j = c \mid \alpha_j = c\,]$$
$$= \frac{1}{2} \Pr[D(\widetilde{\mathsf{hyb}}_e(n)) = 1] + \frac{1}{2} \Pr[D(\widetilde{\mathsf{hyb}}_{e-1}(n)) = 0]$$

$$= \frac{1}{2} + \frac{1}{2} \left( \Pr[D(\widetilde{\mathsf{hyb}}_e(n)) = 1] - \Pr[D(\widetilde{\mathsf{hyb}}_{e-1}(n)) = 1] \right)$$

$$\geq \frac{1}{2} + \frac{1}{2} \left( \frac{1}{q(n)} \right).$$

Thus, we arrive at a contradiction. $\qquad\square$

Note that the only difference between the real execution and hybrid 1 is in the way that the receiver outputs $v_u$. Specifically, in hybrid 1 simulator $\mathcal{S}_1$ extracts $v_0, v_1$ and then outputs $v_u$, while in the real execution the receiver outputs the value that corresponds to its strategy in Stage 5. We now prove that the receiver's output in both experiments is statistically close. In more detail, the difference between the simulation's strategy and the honest receiver's strategy is that the simulator extracts the sender's both inputs in all $i \in \Delta - \Phi$ and then finds code words that are $16n$-close to the extracted values, whereas the honest receiver finds a code word that is $17n$-close based on the inputs it received in Stages 2 and 5, and returns it.

Observe that the sender's views in hybrid 1 and the real execution are identical. It is therefore suffices to show that the value $u$ extracted by the simulator and fed to $\mathcal{F}_{\mathrm{OT}}$ is identical the to the reconstructed output of the honest receiver. Let $v$ denote the value the honest receiver outputs and $v_u$ denote the value extracted by the simulator. These values are obtained in two steps:

- The honest receiver obtains shares of $v$ by computing $\tilde{\rho}_i = \beta_i^u \oplus \tilde{s}_i$ for $i \in \Delta$ where $\tilde{s}_i$ are its output from the OT sessions in Stage 2. On the other hand, the simulator computes $\overline{\rho}_i^u = \beta_j^u \oplus s_j^{u \oplus \alpha_j}$ where $s_j^b$'s are the inputs that the simulator extracted from Stage 1. (Note that these were the inputs that the sender was supposed to use in the OT sessions.)
- Next, the closest code word is computed from the shares. The honest receiver picks that code $\tilde{w}$ that is $17n$-close to $(\tilde{\rho}_i)_{i \in \Delta}$. The simulator, on the other hand, picks a code word $\overline{w}^u$ that is $16n$-close to $(\overline{\rho}_i^u)_{i \in \Delta}$.

We now show that $v \neq v_u$ only with negligible probability because of the final cut-and-choose stage. We consider two cases:

**Case 1:** *The honest receiver extracts a valid $v$ from $(\tilde{\rho}_i)_{i \in \Delta}$:* In this case, we know that there is a code word $w$ that is $17n$-close to $(\tilde{\rho}_i)_{i \in \Delta}$. Now, for every $i \in \Delta - \Phi$, we have that $\tilde{\rho}_i = \overline{\rho}_i$ since the sender proceeded honestly in those sessions. Following the same proof as in Claim 3.6, we can show that $|\Phi| \geq n$ only with negligible probability. Therefore, $|\Delta - \Phi| \geq 17n$ and $\tilde{\rho}_i$ and $\overline{\rho}_i^u$ agree on at least $17n$ locations in $\Delta - \Phi$. Now, since $w$ is $17n$-close to $(\tilde{\rho}_i)_{i \in \Delta}$, this means that $w$ is $16n$-close to $(\overline{\rho}_i^u)_{i \in [20n]}$ (because $|\Delta| = 18n$). Therefore, the simulator would have recovered the same code word and extracted the same value.

**Case 2:** *The honest receiver does not extract a valid $v$ from $(\tilde{\rho}_i)_{i \in \Delta}$:* This happens when $(\tilde{\rho}_i)_{i \in \Delta}$ is not $17n$-close to any code word. In this case, the receiver uses a default value for $v$. We need to show that in this case, even the simulation sets $\tilde{v}_u$ to a default value. Suppose that there exists $w$ that is $16n$-close to $(\overline{\rho}_i^u)_{i \in [20n]}$. We will argue that the simulator still sets $v_u$ to a default value.

Let $\psi$ be the locations where $w$ and $(\tilde{\rho}_i)_{i \in \Delta}$ differ. By our hypothesis that $(\tilde{\rho}_i)_{i \in \Delta}$ is not $17n$-close to any code word, we have that $|\psi| > n$. Nevertheless, since $\Gamma$ is a randomly chosen subset of size $n$ and based on the proof of Claim 3.6, we can show that $\psi \cap \Gamma \neq \emptyset$ except with negligible probability. In this case, there exists an index $j \in \psi \cap \Gamma$ such that the sender must reveal values $s_j^0, s_j^1$ that are consistent with the OT protocol in session $i$ in Stage 2. Therefore, for such a $j \in \psi \cap \Gamma$ it holds that

$$\tilde{\rho}_j = \beta_j^u \oplus \tilde{s}_i = \beta_j^u \oplus s_j^{u \oplus \alpha_j} \neq w_j.$$

This implies that the simulator would have noted that $\beta_j^u \oplus s_j^{u \oplus \alpha_j} \neq w_j$. In this case, the sender fails the second consistency check and the simulator should report that $\tilde{v}_u$ as the default value.

This concludes the proof of the claim.                                                                                        □

## 4. One-Sided Adaptive UC Secure Computation

In the two-party one-sided adaptive setting, at most one of the parties is adaptively corrupted [30,38]. In this section, we provide a simple transformation of our static UC protocol from Sect. 3 to a two-party UC protocol that is secure against one-sided adaptive corruption. Our first observation is that in Protocol 1 the parties use their real inputs to the OT protocol only in Phase 4. Therefore, simulation of the first three phases can be easily carried out by simply following the honest strategy. On the other hand, simulating the messages in Phase 4 requires some form of equivocation since if corruption takes place after this phase is concluded then the simulator needs to explain this message with respect to the real input of the corrupted party. It is important to note that while in the plain model any statically secure protocol can be compiled into one-sided secure protocol by encrypting its entire communication using non-committing encryption (NCE) [5,9,17], the same transformation does not hold in the UC setting due to the additional setup, e.g., a CRS, which may depend on the identity of the corrupted party. Nevertheless, in Phase 4 the parties only run a combiner for which the computation does not involve any usage of the CRS (which is induced by the extractable commitment). Therefore, the proof follows directly.

Our second observation is that in the context of one-sided adaptive security, it is sufficient to rely on a weaker variant of NCE, namely one that is secure against only a single adaptive corruption [30,38]. In particular, we take advantage of a construction presented in [9] and later refined in [17] that achieves equivocation with respect to only one party under the assumption of semi-honest OT with receiver equivocation (namely such OT implies that the receiver's messages can be explained with respect to both potential inputs $u = 0$ and $u = 1$ and some random string). We will briefly describe it now. Recall that in the fully adaptive case, the high-level idea is for the sender and receiver to mutually agree on a random bit. This process requires simulatable PKE schemes which implies the ability to both obliviously sample a public key without the knowledge of the secret key, as well as the ability to obliviously sample a ciphertext without the knowledge of the corresponding plaintext. In the simpler one-sided scenario,

Canetti et al. [9] observed that an oblivious transfer protocol can replace the oblivious generation of the public key. Specifically, the NCE receiver sends two public keys to the sender, and then, the parties invoke an OT protocol where the NCE receiver plays the role of the OT sender and enters the corresponding secret keys. To allow equivocation for the NCE sender, the OT must enable equivocation with respect to the OT receiver. The [22] OT protocol is an example for such a protocol. Here the OT receiver can pick the two ciphertexts so that it knows both plaintexts. Then equivocation is carried out by declaring that the corresponding ciphertext is obliviously sampled.

The advantage of this approach is that it removes the requirement of generating the public key obliviously, as now the randomness for its generation is split between the parties, where anyway only one of them is corrupted. This implies that the simulator can equivocate the outcome of the protocol execution without letting the adversary the ability to verify it. To conclude, it is possible to strengthen the security of Protocol 1 into the one-sided setting by simply encrypting the communication within the combiner phase using one-sided NCE which in turn can be constructed based on PKE with oblivious ciphertext generation. This implies the following theorem which further implies black-box one-sided UC secure computation from enhanced trapdoor permutation.

**Theorem 4.1.** *Assume the existence of PKE with oblivious ciphertext generation. Then for any two-party well-formed functionality $\mathcal{F}$, there exists a protocol that UC realizes $\mathcal{F}$ in the presence of one-sided adaptive, malicious adversaries in the CRS model using black-box access to the PKE.*

## 5. Adaptive UC Secure Computation

In this section, we demonstrate the feasibility of UC commitment schemes based on PKE with oblivious ciphertext generation (namely where it is possible to obliviously sample the ciphertext without knowing the plaintext). Our construction is secure even in the presence of adaptive corruptions and is the first to achieve the stronger notion of adaptive security based on this hardness assumption. As stated in the introduction, plugging in our UC commitment protocol into the transformation of [6], that generates adaptive malicious oblivious transfer given adaptive semi-honest oblivious transfer and UC commitments, implies malicious adaptive UC oblivious transfer based on semi-honest adaptive oblivious transfer and PKE with oblivious ciphertext generation, using only black-box access to these underlying primitives. Stating formally,

**Theorem 5.1.** *Assume the existence of adaptive semi-honest oblivious transfer and PKE with oblivious ciphertext generation. Then for any multi-party well-formed functionality $\mathcal{F}$, there exists a protocol that UC realizes $\mathcal{F}$ in the presence of adaptive, malicious adversaries in the CRS model using black-box access to the oblivious transfer protocol and the PKE.*

Noting that simulatable PKE implies both semi-honest adaptive OT [5,12] and PKE with oblivious ciphertext generation, we derive the following corollary (where simulatable PKE implies oblivious sampling of both public keys and ciphertexts),

**Corollary 5.2.** *Assume the existence of simulatable PKE. Then for any multi-party well-formed functionality $\mathcal{F}$, there exists a protocol that UC realizes $\mathcal{F}$ in the presence of adaptive, malicious adversaries in the CRS model using black-box access to the simulatable PKE.*

This in particular improves the result from [16] that relies on simulatable PKE in a non-black-box manner. Note also that our UC commitment can be constructed using a weaker notion than simulatable PKE where the inverting algorithms can require a trapdoor. This notion is denoted by trapdoor simulatable PKE [5] and can be additionally realized based on the hardness assumption of factoring Blum integers. This assumption, however, requires that we modify our commitment scheme so that the CRS includes $3n + 1$ public keys of the underlying PKE instead of just one, as otherwise the reduction to the security of the PKE does not follow for multiple ciphertexts. Specifically, at the cost of linear blowup (in the security parameter) of the CRS, we obtain adaptively secure UC commitments under a weaker assumption. Now, since trapdoor simulatable PKE implies adaptive semi-honest OT [5] it holds,

**Corollary 5.3.** *Assume the existence of trapdoor simulatable PKE. Then for any multi-party well-formed functionality $\mathcal{F}$, there exists a protocol that UC realizes $\mathcal{F}$ in the presence of adaptive, malicious adversaries in the CRS model using black-box access to the trapdoor simulatable PKE.*

Note that, since the best-known general assumptions for realizing adaptive semi-honest OT is trapdoor simulatable PKE, this corollary gives evidence that the assumptions for adaptive semi-honest OT are sufficient for adaptive UC security and makes a step toward identifying the minimal assumptions for achieving UC security in the adaptive setting. To conclude, we note that enhanced trapdoor permutations, which imply PKE with oblivious ciphertext generation, imply the following corollary,

**Theorem 5.4.** *Assume the existence of enhanced trapdoor permutation. Then $\mathcal{F}_{\mathrm{COM}}$ (cf. Fig. 2) can be UC realized in the CRS model in the presence of adaptive malicious adversaries.*

### 5.1. *UC Commitments from PKE with Oblivious Ciphertext Generation*

In this section, we demonstrate the feasibility of adaptively secure UC commitments for the message space $m \in \{0, 1\}$ from any public-key encryption scheme $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \widetilde{\mathsf{Enc}}, \widetilde{\mathsf{Enc}}^{-1})$ with oblivious ciphertext generation (cf. Definition 2.4) in the common reference string (CRS) model.

*Protocol Overview*   At a high level, the CRS contains two public keys of the encryption scheme, one used by the sender and the other by the receiver, and proceeds in two phases: an *input encoding phase*, where the sender encodes its input via a $n$-out-of-$(2n + 1)$ Shamir secret sharing scheme and commits to them in a specific way followed by a *cut-and-choose phase* where the receiver asks the sender to reveal $n$ of the shares. In slightly more detail, in the input encoding phase, the sender encodes its message $m$ via $n$-degree

polynomial $p(\cdot)$ such that $p(0) = m$ and commits to $p(1), \ldots, p(2n + 1)$ as follows: For each $i$, it sends two strings: One is a ciphertext containing an encryption of $p(i)$ under the public key in the CRS meant for the sender and the other is a random string. Furthermore, the sender randomly decides which one of the two strings is the encryption of $p(i)$. In the cut-and-choose phase, the parties engage in a coin toss where the receiver first encrypts its share for the coin-tossing using the receiver public key from the CRS, followed by the sender providing its share of the coin-tossing. Then the receiver opens its share and the result of the coin-tossing decided by the XOR of the shares determines a subset $\Gamma$ of $[2n + 1]$ of size $n$. The sender reveals the $p(i)$ for every $i \in \Gamma$ and the randomness used for generating the ciphertext. In the decommitment phase, the sender reveals the entire randomness used for the encoding phase.

Straight-line equivocation is achieved by considering encodings of both 0 and 1 and for each $i$, via polynomials $p(\cdot)$ and $q(\cdot)$, such that they agree on $n$ points randomly chosen from $\{1, \ldots, 2n + 1\}$, call this set $\Gamma^*$. Then the simulator encodes by having one ciphertext with the value $p(i)$ and the other $q(i)$. Finally, the simulator biases the coin-tossing outcome so that the outcome results is the set $\Gamma^*$. This can be achieved in a straight-line manner as the simulator will possess the secret key corresponding to the receiver public key in the CRS. Since the receiver encrypts its share first in the coin-tossing, the simulator can extract the value and send the sender's share accordingly to the biased outcome. Finally, since $p$ and $q$ agree on this set $\Gamma^*$, in the decommitment phase, the sender will be able to open either $p(\cdot)$ or $q(\cdot)$, depending on what the message is. Furthermore, we require one of the two strings in each coordinate to be random and this can be faked as the encryption scheme has pseudorandom ciphertexts.

Straight-line extraction, on the other hand, requires an information-theoretic lemma which states that there exists a unique set $\Gamma^*$ after the encoding phase that the sender needs as the outcome of the coin-tossing in the cut-and-choose phases for it to equivocate. First, using the semantic security of the receiver public key we show that the probability that the sender can bias the coin-tossing is negligible. Then we show that the simulator can extract the message of the sender by using the secret key corresponding to the sender public key used by the sender in the encoding phase. This will accomplish by using the $n$ values that have been revealed and finding a consistent polynomial for the remaining shares. (See [31] for more details.)

Finally, we remark that security against adaptive corruptions essentially follows from the pseudorandomness of the ciphertexts.

Our complete construction is shown in Fig. 4. Next, we prove that

**Theorem 5.5.** *Assume that* $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \widetilde{\mathsf{Enc}}, \widetilde{\mathsf{Enc}}^{-1})$ *is a PKE with oblivious ciphertext generation. Then protocol* $\pi_{\mathrm{COM}}$ *(cf. Fig. 4) UC realizes* $\mathcal{F}_{\mathrm{COM}}$ *in the CRS model in the presence of adaptive malicious adversaries.*

*Proof Overview* Intuitively, security requires proving both hiding and binding in the presence of static and adaptive corruptions. The hiding property follows from the IND-CPA security of the encryption scheme combined with the fact that the receiver only sees $n$ shares in a $(n + 1)$-out-of-$(3n + 1)$ secret sharing of the message in the commit phase. On the other hand, proving binding is much more challenging and reduces to the fact that

---

**Protocol** $\pi_{\text{COM}}$.

**CRS:** Two independent keys $\text{PK}, \widetilde{\text{PK}}$ that are in the range of $\text{Gen}(1^n)$.

**Sender's Input:** A message $m \in \{0, 1\}$ and a security parameter $1^n$.

**[Commitment phase:]**

    **Encoding phase:** The sender chooses a random $n$-degree polynomial $p(\cdot)$ over a field $\mathbb{F}[x]$ such that $p(0) = m$. Namely, it randomly chooses $a_i \leftarrow \mathbb{F}$ for all $i \in [n]$ and sets $a_0 = m$, and defines the polynomial $p(x) = a_0 + a_1 x + \cdots + a_n x^n$. The sender then creates a commitment to $m$ as follows. For every $i = [3n + 1]$, it first pick $b_i \leftarrow \{0, 1\}$ at random and then computes the following pairs:

$$\text{If } b_i = 0 \text{ then} \quad \begin{array}{rcl} c_i^0 &=& \text{Enc}_{\text{PK}}(p(i); t_i) \\ c_i^1 &=& r_i \end{array} \qquad \text{else,} \quad \text{if } b_i = 1 \text{ then} \quad \begin{array}{rcl} c_i^0 &=& r_i \\ c_i^1 &=& \text{Enc}_{\text{PK}}(p(i); t_i) \end{array}$$

where $t_i \leftarrow \{0, 1\}^n$ and $r_i \leftarrow \widetilde{\text{Enc}}(\cdot)$ is obliviously sampled. The sender sends $(c_0^0, c_0^1), \ldots, (c_{3n+1}^0, c_{3n+1}^1)$ to the receiver.

    **Coin-tossing phase:** The sender and receiver interact in a coin-tossing protocol that is carried out as follows.

        1. The receiver sends $c = \text{Enc}_{\widetilde{\text{PK}}}(\sigma_0; r_{\sigma_0})$ to the sender where $\sigma_0 \leftarrow \{0, 1\}^N$ is chosen uniformly at random.

        2. The sender picks $\sigma_1 \leftarrow \{0, 1\}^N$ at random and sends it in the clear to the receiver

        3. The receiver decrypts $c$ by revealing $\sigma_0$ and $r_{\sigma_0}$.

    Both the sender and the receiver compute $\sigma = \sigma_0 \oplus \sigma_1$ and use $\sigma$ as the random string to sample a random subset $S \subset [3n + 1]$ of size $n$. (Note that such sampling can be done in a simple way by partitioning the set of coordinates into $n$ sets of triples (where the last set includes 4 elements) and picking one element per set. Notably, this technique does not imply that any potential subset of size $n$ will be picked, rather it ensures that a subset is picked with a negligible probability in $n$, specifically $(1/3)^n$, which suffices for our proof.)

    **Cut-and-choose phase:** The sender decrypts the set $\{c_i^{b_i}\}_{i \in S}$ by sending the sequence $\{b_i, p(i), t_i\}_{i \in S}$. The receiver verifies that all the decryptions are correct and aborts otherwise.

**[Decommitment phase:]** Let $T = [3n + 1] - S$. The sender reveals its input $m$ and decrypts all the ciphertexts in $\{c_i^{b_i}\}_{i \in T}$. The receiver checks if all the decryptions are correct and aborts otherwise. Using the $n$ polynomial evaluations revealed relative to $i \in S$ and any additional polynomial evaluation that was revealed relative to $T$, the receiver reconstructs the polynomial $p(\cdot)$ (via polynomial interpolation of $n + 1$ points). Next, the receiver verifies whether $p(0) = m$, and that for every $i \in [3n + 1]$ the point $p(i)$ is the decrypted value within $c_i^{m_i}$.

---

**Fig. 4.** UC adaptively secure commitment scheme.

a corrupted sender cannot successfully predict exactly the $n$ indices from $\{1, \ldots, 3n+1\}$ that will be chosen in the coin-tossing protocol. In fact, if it can identify these $n$ indices, then it would be possible for the adversary to break binding. This is because it can create two different polynomials that intersect on these $n$ points, yet encode two different messages. An important information-theoretic argument that we prove here is that for a fixed encoding phase, no adversary can equivocate on two continuations from the encoding phase with different outcomes of the coin-tossing phase. Saying differently, for any given encoding phase there is exactly one outcome for the coin-tossing phase that will allow equivocation. Given this claim, binding now follows from the IND-CPA security of the encryption scheme used in the coin-tossing phase.

In addition, recall that in the UC setting the scheme must also support a simulation that allows straight-line extraction and equivocation. At a high level, the simulator sets the CRS to public keys for which it knows the corresponding secret keys. This will allow the simulator to extract all the values encrypted by the adversary. We observe that the simulator can fix the outcome of the coin-tossing phase to any $n$ indices of its choice by extracting the random string $\sigma_0$ encrypted by the receiver and choosing a random string

$\sigma_1$ so that $\sigma_0 \oplus \sigma_1$ is a particular string. Next, the simulator generates secret sharing for both 0 and 1 so that they overlap in the particular $n$ shares. To commit, the simulator encrypts the $n$ common shares within the $n$ indices to be revealed (which it knows in advance), and for the rest of the indices, it encrypts two shares: one that corresponds to the sharing of 0 and the other that corresponds to the sharing of 1. Finally, in the decommit phase, the simulator reveals that shares that correspond to the real message $m$, and exploits the invertible sampling algorithm to prove that the other ciphertexts were obliviously generated.

### 5.2. *Proof of Theorem 5.5*

Let $\mathcal{A}$ be a malicious probabilistic polynomial-time real adversary running the above protocol in the $\mathcal{F}_{\mathrm{CRS}}$-hybrid model. We construct an ideal model adversary $\mathcal{S}$ with access to $\mathcal{F}_{\mathrm{COM}}$ which simulates a real execution of protocol $\pi_{\mathrm{COM}}$ with $\mathcal{A}$ such that no environment $\mathcal{Z}$ can distinguish the ideal process with $\mathcal{S}$ and $\mathcal{F}_{\mathrm{COM}}$ from a real execution of $\pi_{\mathrm{COM}}$ with $\mathcal{A}$. $\mathcal{S}$ starts by invoking a copy of $\mathcal{A}$ and running a simulated interaction of $\mathcal{A}$ with environment $\mathcal{Z}$, emulating the honest party. We separately describe the actions of $\mathcal{S}$ for every corruption case.

**Initialization:** The common reference string (CRS) is chosen by $\mathcal{S}$ in the following way. It generates $(\mathrm{PK}, \mathrm{SK}) \leftarrow \mathsf{Gen}(1^n)$ and $(\widetilde{\mathrm{PK}}, \widetilde{\mathrm{SK}}) \leftarrow \mathsf{Gen}(1^n)$, and places $(\mathrm{PK}, \widetilde{\mathrm{PK}})$ in the CRS. The simulator further records $(\mathrm{SK}, \widetilde{\mathrm{SK}})$.

**Simulating the communication with $\mathcal{Z}$:** Every message that $\mathcal{S}$ receives from $\mathcal{Z}$ it internally feeds to $\mathcal{A}$ and every output written by $\mathcal{A}$ is relayed back to $\mathcal{Z}$.

**Simulating the commitment phase when the receiver is statically corrupted:** In this case $\mathcal{S}$ proceeds as follows:

1. *Encoding phase:* Upon receiving message $(sid, \mathrm{Sen}, \mathrm{Rec})$ from $\mathcal{F}_{\mathrm{COM}}$, the simulator picks a random subset $S^* \subset [3n+1]$ of size $n$ and two random $n$-degree polynomials $p_0(\cdot)$ and $p_1(\cdot)$ such that:

$$p_0(i) = p_1(i) \qquad \forall i \in S^*$$
$$p_0(0) = 0 \text{ and } p_1(0) = 1.$$

Note that the simulator can define these polynomials via interpolation, where a unique $n$-degree polynomial can be constructed given $n+1$ points. Let $T^* = [3n+1] - S^*$; then, the simulator defines the commitment as follows:

For every $i \in S^*$, the simulator proceeds as the honest sender would with polynomial $p_0(\cdot)$. Namely, it first picks $b_i \leftarrow \{0, 1\}$ at random and then sets the following pairs,

$$\text{If } b_i = 0 \text{ then } \begin{array}{l} c_i^0 = \mathsf{Enc}_{\mathrm{PK}}(p_0(i); t_i) \\ c_i^1 = r_i \end{array} \text{ else,}$$

$$\text{if } b_i = 1 \text{ then } \begin{array}{l} c_i^0 = r_i \\ c_i^1 = \mathsf{Enc}_{\mathrm{PK}}(p_0(i); t_i), \end{array}$$

where $b_i \leftarrow \{0, 1\}$, $t_i \leftarrow \{0, 1\}^n$ and $r_i \leftarrow \widetilde{\mathsf{Enc}}_{\mathrm{PK}}(\cdot)$ is obliviously sampled (we recall that $p_0(i) = p_1(i)$ for all $i \in S^*$).

For every $i \in T^*$, the simulator picks $b_i \leftarrow \{0, 1\}$ at random and then uses the points on both polynomials $p_0(\cdot)$ and $p_1(\cdot)$ to calculate the following pairs,

$$\text{If } b_i = 0 \text{ then } \begin{array}{l} c_i^0 = \mathsf{Enc}_{\mathrm{PK}}(p_0(i); t_i) \\ c_i^1 = \mathsf{Enc}_{\mathrm{PK}}(p_1(i); \tilde{t}_i) \end{array} \text{ else,}$$

$$\text{if } b_i = 1 \text{ then } \begin{array}{l} c_i^0 = \mathsf{Enc}_{\mathrm{PK}}(p_1(i); \tilde{t}_i) \\ c_i^1 = \mathsf{Enc}_{\mathrm{PK}}(p_0(i); t_i), \end{array}$$

where $b_i \leftarrow \{0, 1\}$ and $t_i, \tilde{t}_i \leftarrow \{0, 1\}^n$ are chosen uniformly at random.

Finally, the simulator sends the pairs $(c_0^0, c_0^1), \dots, (c_{3n+1}^0, c_{3n+1}^1)$ to the receiver.

2. *Coin-tossing phase:* The simulator biases the coin-tossing result so that the set $S$ that is chosen in this phase is identical to $S^*$. More precisely, the simulator extracts $\sigma_0$ from the receiver's ciphertext and then sets $\sigma_1$ so that $\sigma = \sigma_0 \oplus \sigma_1$ yields the set $S^*$.

3. *Cut-and-choose phase:* The simulator decrypts all the ciphertexts within $\{c_i^{b_i}\}_{i \in S^*}$.

**Simulating the decommitment phase where the receiver is statically corrupted:** Upon receiving a message $(\mathsf{reveal}, sid, m)$ from $\mathcal{F}_{\mathrm{COM}}$, $\mathcal{S}$ generates a simulated decommitment message as follows. Recall first that the simulator needs to reveal points on a polynomial $p(\cdot)$ and pairs $\{(b_i, t_i)\}_{i \in [3n+1]}$ such that $p(0) = m$ and $c_i^{b_i} = \mathsf{Enc}_{\mathrm{PK}}(p(i); t_i)$. Let $\hat{b}_i = b_i \oplus m$ for all $i \in T^*$, then $\mathcal{S}$ reveals $p_m(\cdot)$, $\{\hat{b}_i, t_i^{\hat{b}_i}, r_i = \widetilde{\mathsf{Enc}}_{\mathrm{PK}}^{-1}(c_i^{1-\hat{b}_i})\}_{i \in T^*}$.

Next, we prove that $\mathcal{Z}$ cannot distinguish an interaction of protocol $\pi_{\mathrm{COM}}$ with $\mathcal{A}$, corrupting the receiver, from an interaction of $\mathcal{S}$ with $\mathcal{F}_{\mathrm{COM}}$. Formally,

**Claim 5.1.** *The following two distribution ensembles are computationally indistinguishable,*

$$\left\{\mathbf{View}_{\pi_{\mathrm{COM}}, \mathcal{A}, \mathcal{Z}}^{\mathcal{F}_{\mathrm{CRS}}}(n)\right\}_{n \in \mathbb{N}} \stackrel{c}{\approx} \left\{\mathbf{View}_{\pi_{\mathrm{IDEAL}}, \mathcal{S}, \mathcal{Z}}^{\mathcal{F}_{\mathrm{COM}}}(n)\right\}_{n \in \mathbb{N}}.$$

*Proof.* We prove this claim using a sequence of hybrid games.

*Hybrid* 0: This is the real interaction of $\mathcal{Z}$ with $\mathcal{A}$ and Protocol $\pi_{\mathrm{COM}}$.

*Hybrid* 1: In this experiment, we define a simulator $\mathcal{S}_1$ that proceeds as follows. $\mathcal{S}_1$ uses $\mathcal{S}$'s strategy in the coin-tossing phase when simulating the corrupted receiver. Specifically, $\mathcal{S}_1$ emulates $\mathcal{F}_{\mathrm{CRS}}$ and generates (PK, SK) and $(\widetilde{\mathrm{PK}}, \widetilde{\mathrm{SK}})$ as in the simulation. Next, it picks at the beginning of the commit phase a random subset $S^*$ for which it wishes to bias the outcome of the coin-tossing phase. It then extracts the value $\sigma_0$ encrypted by the receiver in the coin-tossing phase using $\widetilde{\mathrm{SK}}$ and sets $\sigma_1$ so that $\sigma_0 \oplus \sigma_1$ results in

$S^*$. $S_1$ next follows the honest's sender role with input $m$ for the rest of the execution. We claim that the adversary's view in this hybrid game is identically distributed to its view in the prior hybrid. This is because $S^*$ is chosen independently of the hybrid game and uniformly at random. Therefore, given any particular $\sigma_0$ extracted from the adversary's commitment in the coin-tossing stage, $\sigma_1$ will be uniformly random (which is exactly how it is distributed in hybrid 0). Therefore, we have that the following distributions are identical,

$$\left\{\mathbf{View}^{\mathcal{F}_{\mathrm{CRS}}}_{\pi_{\mathrm{COM}}, \mathcal{A}, \mathcal{Z}}(n)\right\}_{n \in \mathbb{N}} \equiv \left\{\mathbf{View}^{\mathcal{F}_{\mathrm{COM}}}_{\pi_{\mathrm{IDEAL}}, S_1, \mathcal{Z}}(n)\right\}_{n \in \mathbb{N}}.$$

*Hybrid* 2: In this experiment, we define a simulator $S_2$ that is given the sender's message $m$, yet it carries out $S$'s strategy in the encoding phase instead of playing the role of the honest sender. More precisely, $S_2$ proceeds identically to $S_1$ with the exception that in the encoding phase, it defines polynomials $p_m(\cdot)$ and $p_{1-m}(\cdot)$ exactly as $S$ does in the simulation using the set $S^*$. Observe first that the outcome of the coin-tossing phase has already been fixed to $S^*$ in hybrid 1. Moreover, $S_2$ executes the decommitment phase exactly as the honest sender does by providing polynomial $p_m(\cdot)$. Then the differences between the receiver's view in hybrids 1 and 2 are with respect to the non-opened ciphertexts, namely the ciphertexts that are in positions $1 - b_i$'s, and denoted by $\{c_i^{1-b_i}\}_{i \in [3n+1]}$, which encode the polynomial $p_{1-m}(\cdot)$. These ciphertexts are obliviously picked in hybrid 1, yet computed using algorithm $\mathsf{Enc}$ in hybrid 1. We now prove that the receiver's views in these hybrids executions are computational indistinguishability due to the indistinguishability of ciphertexts using $\mathsf{Enc}$ and $\widetilde{\mathsf{Enc}}$. More precisely, we show the following claim:

**Claim 5.2.** *The following distributions are computationally indistinguishable.*

$$\left\{\mathbf{View}^{\mathcal{F}_{\mathrm{COM}}}_{\pi_{\mathrm{IDEAL}}, S_1, \mathcal{Z}}(n)\right\}_{n \in \mathbb{N}} \overset{\mathrm{c}}{\approx} \left\{\mathbf{View}^{\mathcal{F}_{\mathrm{COM}}}_{\pi_{\mathrm{IDEAL}}, S_2, \mathcal{Z}}(n)\right\}_{n \in \mathbb{N}}.$$

*Proof.* Assume by contradiction that there exist a PPT adversary $\mathcal{A}$, distinguisher $D$ and polynomial $p(\cdot)$ such that $D$ can distinguish the two distributions stated in the claim for infinitely many $n$'s with probability $\frac{1}{p(n)}$. Fix an $n$ for which this happens. Then we use $\mathcal{A}$ we construct an adversary $\mathcal{A}'$ that violates the indistinguishability of real and obliviously generated ciphertexts (cf. Definition 2.4). Toward this, we consider a sequence of intermediate hybrid games $H_2^0, \ldots, H_2^{3n+1}$ where in hybrid $H_2^j$ we define a simulator $S_2^j$ that proceeds identically to $S_2$ when generating $\{c_i^{1-b_i}\}_{i \in [j]}$; namely, it picks a polynomial $p_{1-m}(\cdot)$ and sets $c_i^{1-b_i}$ by $\mathsf{Enc}_{\mathrm{PK}}(p_{1-m}(i))$. Finally, $\{c_i^{1-b_i}\}_{i > j}$ are obliviously generated as in the real sender's strategy. Note that by our construction, $H_2^0$ and $H_2^{3n+1}$ proceed identically to Hybrids 1 and 2, respectively. Denoting the output of the execution in hybrid $H_2^j$ by $\mathsf{hyb}_j(n)$ and using a standard hybrid argument, it follows that there exists $j$ such that

$$\left| \Pr[D(\mathsf{hyb}_j(n)) = 1] - \Pr[D(\mathsf{hyb}_{j-1}(n)) = 1] \right| \geq \frac{1}{(3n+1)p(n)}. \qquad (6)$$

We now construct an adversary $\mathcal{A}'$ that violates the indistinguishability of obliviously generated ciphertexts and real ciphertexts. Specifically, recall that $\mathcal{A}'$ needs to distinguish $(\mathrm{PK}, r_1, c_1)$ from $(\mathrm{PK}, r_1', c_2)$ where $c_1 \leftarrow \widetilde{\mathsf{Enc}}_{\mathrm{PK}}(r_1)$ and $c_2 \leftarrow \mathsf{Enc}_{\mathrm{PK}}(m; r_2)$, $r_1' \leftarrow \widetilde{\mathsf{Enc}}^{-1}(c_2)$. Upon receiving $(\mathrm{PK}, r, c)$ $\mathcal{A}'$ proceeds as follows. It first emulates the execution as in hybrid 1 by setting the CRS to be $(\mathrm{PK}, \widetilde{\mathrm{PK}})$ for $(\widetilde{\mathrm{PK}}, \widetilde{\mathrm{SK}}) \leftarrow \mathsf{Gen}(1^n)$. It then emulates the internal execution by following the strategy of $\mathcal{S}_1^{j-1}$ with the exception that $c_j^{1-b_j}$ is set to $c$. Later, when $\mathcal{A}'$ needs to reveal $c_j^{1-b_j}$ it returns $r$ as the randomness used to obliviously generate $c$. Finally, $\mathcal{A}'$ invokes $D$ on $\mathcal{A}$'s view and outputs whatever $D$ outputs. We recall that the ciphertexts that correspond to the set $\{1 - b_i\}_{i \in [3n+1]}$ are always revealed as obliviously generated ciphertexts regardless of the way they were generated. It must also be noted that $\mathcal{A}'$ does not need to know SK in order to complete the simulation of the sender's messages since it never extracts here. Nevertheless, $\mathcal{A}'$ does need access to $\widetilde{\mathsf{Enc}}^{-1}$ in order to generate the randomness of the first $j-1$ ciphertexts, which by the definition of the encryption scheme requires only PK. To conclude, the internal emulation of $\mathcal{A}'$ upon receiving $(\mathrm{PK}, r_1, c_1)$ so that $c_1 \leftarrow \widetilde{\mathsf{Enc}}_{\mathrm{PK}}(r_1)$ is identically distributed to $H_j$, whereas when $(\mathrm{PK}, r_1', c_2)$ are generated so that $c_2 \leftarrow \mathsf{Enc}_{\mathrm{PK}}(m; r_2)$, $r_1' \leftarrow \widetilde{\mathsf{Enc}}_{\mathrm{PK}}^{-1}(c_2)$ with $m = p_{1-m}(j)$, $\mathcal{A}$'s view is distributed identically to $H_{j-1}$. Therefore, it follows from Equation 6 that

$$\left| \Pr[\mathcal{A}'(\mathrm{PK}, r_1, c_1) = 1] - \Pr[\mathcal{A}'(\mathrm{PK}, r_1', c_2) = 1] \right| \geq \frac{1}{(3n+1)p(n)}.$$

This implies a contradiction relative to the indistinguishability property of real and obliviously generated ciphertexts. □

*Hybrid* 3: This hybrid is the actual simulation with $\mathcal{S}$. Namely, here $\mathcal{S}_3$ does not have the honest sender's actual input $m$ and it computes two polynomials $p_0(\cdot)$ and $p_1(\cdot)$ as defined above. Furthermore, $\mathcal{S}_3$ reveals one of the polynomials $p_0(\cdot)$ or $p_1(\cdot)$ in the decommitment phase, depending on the value of $m$. Observe that the distribution of the messages sent by $\mathcal{S}_2$ and $\mathcal{S}_3$ is identical in both hybrids. We use the facts that at most $n$ shares are revealed in the commitment phase and that $p(\cdot)$ is an $n$-degree polynomial. Therefore, revealing these $n$ shares keeps $p_{1-m}(0)$ completely hidden and we have that

$$\left\{ \mathbf{View}_{\pi_{\mathrm{IDEAL}}, \mathcal{S}_2, \mathcal{Z}}^{\mathcal{F}_{\mathrm{COM}}}(n) \right\}_{n \in \mathbb{N}} \equiv \left\{ \mathbf{View}_{\pi_{\mathrm{IDEAL}}, \mathcal{S}_3, \mathcal{Z}}^{\mathcal{F}_{\mathrm{COM}}}(n) \right\}_{n \in \mathbb{N}}.$$

□

**Simulating the commit phase when the sender is statically corrupted:** Simulating the sender involves extracting the committed value as follows:

1. *Encoding phase:* The simulator proceeds honestly following the honest receiver's strategy, receiving pairs $(c_i^0, c_i^1)$ for all $i \in [3n+1]$. The simulator exploits the fact that it knows the secret key SK and decrypts all ciphertexts. Let $\beta_i^b = \mathsf{Dec}_{SK}(c_i^b)$.

2. *Coin-tossing phase:* The simulator proceeds honestly following the honest receiver's strategy. Let $S'$ be the outcome of the coin-tossing phase.

3. *Cut-and-choose phase:* The simulator proceeds as the honest receiver and verifies whether the openings are consistent with the ciphertexts sent in the encoding phase. Note that none of the revealed values should differ from what the simulator decrypted using SK due to the fact that $\Pr[\mathsf{Dec}_{SK}(\mathsf{Enc}_{PK}(m)) = m] = 1$.

4. *Input extraction:* Finally, the simulator extracts the sender's input as follows. $\mathcal{S}$ chooses an arbitrary index $j \in [3n+1] - S'$ and reconstructs two polynomials $q(\cdot)$ and $\widetilde{q}(\cdot)$ such that

$$q(i) = \widetilde{q}(i) = \beta_i^{b_i} \qquad \forall i \in S'$$
$$q(j) = \beta_j^0 \quad \text{and} \quad \widetilde{q}(j) = \beta_j^1 \quad \text{and} \quad q(0), \widetilde{q}(0) \in \{0, 1\}.$$

It then verifies whether for all $i \in [3n+1]$, $q(i) \in \{\beta_i^0, \beta_i^1\}$ and $\widetilde{q}(i) \in \{\beta_i^0, \beta_i^1\}$. The following cases arise:

**Case 1:** *Both $q(\cdot)$ and $\widetilde{q}(\cdot)$ satisfy the condition and $\widetilde{q}(0) \neq q(0)$.* Then $\mathcal{S}$ halts returning fail. Below we prove that the simulator outputs fail with negligible probability.

**Case 2:** *At most one of $q(\cdot)$ and $\widetilde{q}(\cdot)$ satisfies the condition or $\widetilde{q}(0) = q(0)$.* $\mathcal{S}$ sends (commit, $sid$, $q(0)$) to the $\mathcal{F}_{COM}$ functionality and stores the committed bit $q(0)$. Otherwise, $\mathcal{S}$ sends a default value.

**Case 3:** *Neither $q(\cdot)$ or $\widetilde{q}(\cdot)$ satisfy the condition.* $\mathcal{S}$ sends a default value to the ideal functionality and need not store the committed bit since it will never be decommitted correctly.

**Claim 5.3.** *Conditioned on case 1 not occurring, the sender can decommit to $b$ if and only if $\mathcal{S}$ sends $b$ to $\mathcal{F}_{COM}$.*

*Proof.* By the assumption in the claim, either case 2 or 3 occurs. We now show that if $\mathcal{A}$ decommits successfully, then it must be either with polynomial $q(\cdot)$ or $\widetilde{q}(\cdot)$ if both satisfy the conditions, or with the single satisfying polynomial. That would imply that the adversary can only decommit to whatever sent by the simulator to $\mathcal{F}_{COM}$. We will demonstrate our argument for the case that both polynomials satisfy the condition. The case of a single polynomial follows similarly. More formally, suppose that $q(\cdot)$ and $\widetilde{q}(\cdot)$ are as required by the above condition. Then polynomial $q^*(\cdot)$ that is revealed by $\mathcal{A}$ in the decommitment phase must take the same value as $q(\cdot)$ and $\widetilde{q}(\cdot)$ for all $i \in S'$. Focusing on the $j$th index that is specified above, it holds that either $q^*(j) = q(j)$ or $q^*(j) = \widetilde{q}(j)$ (because $\mathcal{A}$ can decrypt ciphertexts $c_j^0$ and $c_j^1$ only to the plaintexts $q(j)$ and $\widetilde{q}(j)$, respectively). This implies that either $q^*(\cdot)$ and $q(\cdot)$ share $n+1$ points or $q^*(\cdot)$

and $\widetilde{q}(\cdot)$ share $n + 1$ points. Consequently, $q^*(\cdot)$ becomes identically equal to either $q(\cdot)$ or $\widetilde{q}(\cdot)$ since it is an $n$-degree polynomial, and $\mathcal{A}$ can only decommit to $q(0)$ or $\widetilde{q}(0)$. □

**Claim 5.4.** *The probability that $\mathcal{S}$ outputs* fail *in case* 1 *is negligible.*

*Proof.* Assume for contradiction that there exists $\mathcal{A}$ that for infinitely many $n$'s generates ciphertexts for the encoding phase such that $\mathcal{S}$ obtains valid $q_0(\cdot)$ and $q_1(\cdot)$ such that both satisfy the conditions at the end of the commit phase with probability at least $\frac{1}{p(n)}$. Observe that in such a case, the transcript can be equivocated to both 0 and 1 using $q_0(\cdot)$ and $q_1(\cdot)$, respectively. We show how to construct $\mathcal{A}'$ that violates the privacy of the underlying encryption scheme. At a high level, we prove that $\mathcal{A}$ can successfully equivocate only if it biases the coin-tossing outcome, and this can be achieved only by breaking the privacy of the encryption scheme.

We consider first an alternative simulator $\widetilde{\mathcal{S}}$ that proceeds exactly as the real simulator $\mathcal{S}$ does, with the exception that it receives as input PK* that it internally sets as $\widetilde{\text{PK}}$ in the CRS. Observe that $\mathcal{S}$ does not use the $\widetilde{\text{SK}}$ in simulating the corrupted sender. Hence, the view generated by $\widetilde{\mathcal{S}}$ is identical to $\mathcal{S}$. This implies that the transcript that is obtained in the simulated commit phase of $\widetilde{\mathcal{S}}$ can be equivocated with probability $\frac{1}{p(n)}$, i.e., there are valid decommitments to both 0 and 1 relative to polynomials $q_0(\cdot)$ and $q_1(\cdot)$. Then, by applying a standard averaging argument it holds that the transcript from the commit phase can be equivocated with probability $\frac{1}{2p(n)}$ over a random continuation of some fixed transcript $\tau$, where $\tau$ is a partial transcript of protocol $\pi_{\text{COM}}$ that reaches right after the encoding phase and the probability is taken over the adversary's and honest receiver's randomness. (Specifically, the probability that $\frac{1}{2p(n)}$ portion of the partial transcripts lead to a successful equivocation is $\frac{1}{2p(n)}$.) Using this observation, we will construct an adversary $\mathcal{B}$ that wins the IND-CPA game for the scheme $\Pi$.

Our proof relies heavily of the following claim,

**Claim 5.5.** *Let $\tau$ be a fixed partial transcript as above. Then there exist no transcripts* trans$_1$, trans$_2$ *that satisfy the following conditions:*

1. trans$_1$ *and* trans$_2$ *are complete and accepting transcripts of $\pi_{\text{COM}}$ with $\tau$ being their prefix.*
2. *There exists two distinct sets $S_1$, $S_2$ such that $S_1$ and $S_2$ are the respective outcomes of the coin-tossing phase within* trans$_1$ *and* trans$_2$.
3. *There are valid decommitments to values 0 and 1 in* trans$_1$ *and* trans$_2$.

An important observation that follows from Claim 5.5 is that the sets chosen in the coin-tossing phase must be identical for any two complete trans$_1$, trans$_2$ that are defined relative to a fixed partial transcript $\tau$, on which a transcript can be equivocated. Clearly, given that the receiver's random string $\sigma_0$ in the coin-tossing phase is hidden from the sender, it must hold that the probability that the same set is chosen in two coin-tossing executions is exponentially small. On the other hand, with non-negligible probability over partial transcripts $\tau$, there are decommitments to both 0 and 1, and from Claim 5.5, we know that a successful equivocation implies a fixed joint set $S^*$ of size $n$. This

intuitively means that the adversary violates the IND-CPA security of the encryption scheme $\Pi$. Formally, we construct an adversary $\mathcal{B}$ that internally incorporates $\mathcal{A}$ and proceeds as in the IND-CPA game:

1. $\mathcal{B}$ externally receives a public key PK*. It follows $\widetilde{\mathcal{S}}$'s strategy and sets $\widetilde{\text{PK}}$ in the CRS as this input.
2. $\mathcal{B}$ emulates an execution with $\mathcal{A}$ following $\widetilde{\mathcal{S}}$'s strategy until the completion of the encoding phase. Denote the partial transcript obtained so far by $\tau$.
3. $\mathcal{B}$ samples $M_1 = np(n)$ transcripts with prefix $\tau$ as follows. It invokes $\mathcal{A}$, $M_1$ times, each time with independent randomness for $\widetilde{\mathcal{S}}$ (which specifically implies independent randomness in the coin-tossing phase). For each such execution, $\mathcal{B}$ checks whether there are two valid decommitments for 0 and 1. If there exists one such transcript, $\mathcal{B}$ stores the outcome $S^*$ of the coin-tossing phase on that transcript. If no such transcript is encountered, $\mathcal{B}$ outputs a random bit and halts.
4. $\mathcal{B}$ samples two random strings $\sigma_0^0$ and $\sigma_0^1$ independently at random from $\{0, 1\}^N$ and outputs these strings. Upon receiving a ciphertext $c$ from its oracle, $\mathcal{B}$ feeds $c$ internally as the receiver's message in the coin-tossing phase within the partial transcript $\tau$. It then invokes $\mathcal{A}$ on $(\tau, c)$ and completes the execution as follows. If $\mathcal{A}$ aborts, then $\mathcal{B}$ outputs a random bit and halts. Otherwise, let $\sigma$ be the string revealed by $\mathcal{A}$ in the coin-tossing phase. If $\sigma \oplus \sigma_0^{b'}$ does not result in $S^*$ as the outcome of the coin-tossing for some $b' \in \{0, 1\}$, then $\mathcal{B}$ outputs a random bit. Otherwise $\mathcal{B}$ outputs $b'$ and halts. (Note that in any case $\mathcal{B}$ aborts the execution right before it needs to decrypt $c$, since it cannot do that.)

We will now prove that $\mathcal{B}$ successfully identifies whether $c$ is an encryption of $\sigma_0^0$ or $\sigma_0^1$ with probability non-negligibly greater than $\frac{1}{2}$. Toward proving that we consider the following events, conditioned on $c$ being an encryption of $\sigma_0^b$:

$E_1$: There are decommitments to 0 and 1 conditioned on transcript $\tau$ that is generated in Step 2. We already argued above that the probability that $E_1$ occurs is at least $\frac{1}{2p(n)}$.

$E_2$: Here we consider the event that $\mathcal{B}$ successfully computes $S^*$ in Step 3. Note first that the probability a transcript, generated in Step 3, fails to reveal $S^*$ is at most $1 - \frac{1}{2p(n)}$. This is due to the fact that set $S^*$ can be efficiently extracted whenever there are two decommitments. Therefore,

$$\Pr[E_1 \mid E_2] = 1 - \left(1 - \frac{1}{2p(n)}\right)^{np(n)} = 1 - e^{-n/2}.$$

$E_3$: Finally, we consider the event that the coin-tossing phase results in $S^*$. Note that by Claim 5.5 whenever the transcript can be equivocated then it must be that the coin-tossing result is $S^*$, this implies that

$$\Pr[E_3 \mid E_2 \wedge E_1] \geq \Pr[\text{ transcript with prefix } \tau \text{ can be equivocated} \mid E_2 \wedge E_1]$$

$$\geq \frac{1}{2p(n)}.$$

From the definition of the events it follows that if $E_1 \wedge E_2 \wedge E_3$ occurs, then $\mathcal{B}$ wins the IND-CPA game with probability 1. Denote the joint events $E_1 \wedge E_2 \wedge E_3$ by Comp and note that if Comp does not occur then $\mathcal{B}$'s guess is correct with probability $\frac{1}{2}$. Then from the calculation above it holds that

$$
\begin{aligned}
\Pr[\text{Comp}] = \Pr[E_3 \wedge E_2 \wedge E_1] &= \Pr[E_3 \mid E_2 \wedge E_1] \cdot \Pr[E_2 \mid E_1] \cdot \Pr[E_1] \\
&\geq \frac{1}{2p(n)} \times (1 - e^{-n/2}) \times \frac{1}{2p(n)} \\
&\geq \frac{1}{8(p(n))^2}.
\end{aligned}
$$

Next, we compute the probability that $\mathcal{B}$ succeeds in its guess.

$$
\begin{aligned}
\text{ADV}_{\Pi,\mathcal{B}}(n) &= \Pr[\mathcal{B} \text{ succeeds} \mid \text{Comp}] \cdot \Pr[\text{Comp}] \\
&\quad + \Pr[\mathcal{B} \text{ succeeds} \mid \neg\text{Comp}] \cdot \Pr[\neg\text{Comp}] \\
&= 1 \cdot \Pr[\text{Comp}] + \frac{1}{2} \Pr[\neg\text{Comp}] \\
&= 1 \cdot \Pr[\text{Comp}] + \frac{1}{2}(1 - \Pr[\text{Comp}]) \\
&= \frac{1}{2} + \frac{1}{2} \Pr[\text{Comp}] \\
&\geq \frac{1}{2} + \frac{1}{16(p(n))^2}.
\end{aligned}
$$

This concludes the proof of Claim 5.6. It remains to prove Claim 5.5.

*Proof of Claim 5.5.* Assume for contradiction there exists a partial transcript $\tau$ of the encoding phase, complete transcripts $\text{trans}_1$ and $\text{trans}_2$ and sets $S_1 \neq S_2$ as in Claim 5.5. Let PK be the public key in the CRS and SK be the corresponding secret key. We define some notations first.

$$
\tau = \begin{pmatrix} c_1^0 & c_2^0 & \cdots & c_{3n+1}^0 \\ c_1^1 & c_2^1 & \cdots & c_{3n+1}^1 \end{pmatrix}
$$



| Decommitment to 0 | Decommitment to 0 |
| Decommitment to 1 | Decommitment to 1 |
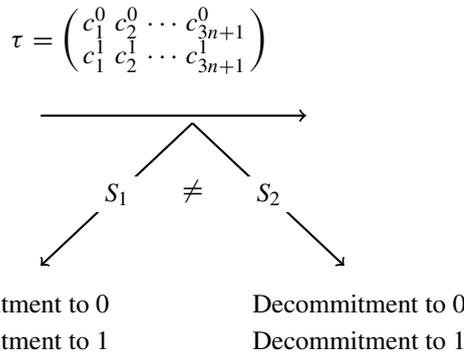
- We denote by $T_1 = [3n + 1] - S_1$ and $T_2 = [3n + 1] - S_2$.
- Recall that transcripts $\text{trans}_1$ and $\text{trans}_2$ include valid decommitments to both 0 and 1. Moreover, since we assume that the prefix $\tau$ is in common to both transcripts

and the decryption is prefect, then a ciphertext $c_i^b$ that is decrypted in either $\mathsf{trans}_1$ or $\mathsf{trans}_2$ must be correctly revealed into exactly one plaintext, which is determined by $\beta_i^b = \mathsf{Dec}_{SK}(c_i^b)$.

- By the assumption above, transcript $\mathsf{trans}_1$ induces two valid decommitments to both 0 and 1. We denote the $b_i$ values within the decommitment to 0 by $(b_1^0, \ldots, b_{3n+1}^0)$ and the values within the decommitment to 1 by $(b_1^1, \ldots, b_{3n+1}^1)$.
- Similarly, transcript $\mathsf{trans}_2$ induces two valid such decommitments. Then, let the $b_i$ values within the decommitment to 0 be denoted by $(\widehat{b}_1^0, \ldots, \widehat{b}_{3n+1}^0)$ and the values within the decommitment to 1 denoted by $(\widehat{b}_1^1, \ldots, \widehat{b}_{3n+1}^1)$.

Consider transcript $\mathsf{trans}_1$. Then the shares that correspond to the indices in $S_1$ and are revealed during the commitment phase imply that

$$\beta_i^0 = \beta_i^1 \quad \forall\, i \in S_1.$$

This further imply that

$$b_i^0 = b_i^1 \quad \forall\, i \in S_1. \tag{7}$$

The rest of the shares are revealed in the decommitment phase. Now, since we use an $(n + 1)$-out-of-$(3n + 1)$ secret sharing scheme, these $n$ shares that correspond to the indices in $S_1$, together with any additional revealed share $i \in T_1$, constitute $n + 1$ shares from which a unique polynomial can be reconstructed. Specifically, the reconstructed polynomials for decommitting to 0 and 1 have to be different (since the secrets are different), so it must be that the revealed plaintexts are also different for every $i \in T_1$, i.e.,

$$\beta_i^0 \neq \beta_i^1 \quad \forall\, i \in T_1.$$

This means that, for $i \in T_1$, $\{c_i^0, c_i^1\}$ must contain the plaintexts $\beta_i^0$ and $\beta_i^1$. Hence, $(c_i^0, c_i^1)$ must either be the encryption of $(\beta_i^0, \beta_i^1)$ or $(\beta_i^1, \beta_i^0)$. In either case, we have that

$$b_i^0 = 1 - b_i^1 \quad \forall\, i \in T_1. \tag{8}$$

Next, we consider transcript $\mathsf{trans}_2$ and recall that it shares the same encoding phase with $\mathsf{trans}_1$. It thus must be the case that for every $i \in [3n + 1]$ the revealed shares for both transcripts must correspond to either $\beta_i^0$ or $\beta_i^1$. From Equation 8 we know that for every $i \in T_1$, $b_i^0 \neq b_i^1$. Hence, for every $i \in T_1$, either $\widehat{b}_i^0 = b_i^0$ or $\widehat{b}_i^0 = b_i^1$. Relying on the fact that $|T_1| = 2n + 1$ and the pigeonhole principle, it must hold that there are $n + 1$ indices in $T_1$ for which either $\widehat{b}_i^0 = b_i^0$ for all these indices, or $\widehat{b}_i^0 = b_i^1$ for all these indices. This implies two cases:

$\widehat{b}_i^0 = b_i^0$ **on** $n+1$ **locations:** In this case, the revealed shares for these $n+1$ locations must be the same for both $\mathsf{trans}_1$ and $\mathsf{trans}_2$ when decommitting to 0. Note that if $n + 1$ shares are the identical then the polynomials revealed for both transcripts must be identical as well, and therefore, the revealed shares for every other index

must be identical. We additionally know that the plaintext shares $\beta_i^0 \neq \beta_i^1$ for any index $i \in T_1$. Combining this with the fact that $\widehat{b}_i^0$ must correspond to the same share as the one corresponding to $b_i^0$, it follows that $\widehat{b}_i^0 = b_i^0$ for all $i \in T_1$.

$\widehat{b}_i^0 = b_i^1$ **on** $n + 1$ **locations:** In this case, we conclude, analogously to the previous case, that the polynomial revealed when decommitting to 0 in $\text{trans}_2$ must be the same as when decommitting to 1 in $\text{trans}_1$. However, such a decommitment on $\text{trans}_2$ is invalid because the secret, which is the value of this polynomial evaluated at 0 is 1.

Therefore, we can conclude that it must be first case, where the revealed polynomials are identical so that $\widehat{b}_i^0 = b_i^0$ for all $i \in T_1$ (and not just for the $n + 1$ locations). Applying the same argument, we can prove that $\widehat{b}_i^1 = b_i^1$ for all $i \in T_1$. Now, since for every $i \in T_1$ we have that $b_i^0 = 1 - b_i^1$, it follows that:

$$\widehat{b}_i^0 = 1 - \widehat{b}_i^1 \quad \forall i \in T_1.$$

Next, we observe that $T_1 \cap S_2$ is non-empty since $S_1 \neq S_2$. We conclude with a contradiction by considering $i^* \in T_1 \cap S_2$. Specifically, for any such $i^*$ we have from the preceding argument that the fact that $i^* \in T_1$ implies that

$$\widehat{b}_{i^*}^0 = 1 - \widehat{b}_{i^*}^1.$$

On the other hand, $i^* \in S_2$ and the values for all the indices in $S_2$ are already revealed in the commitment phase of $\text{trans}_2$; thus, we have that

$$\widehat{b}_{i^*}^0 = \widehat{b}_{i^*}^1$$

which is a contradiction.                                                                           □

**Simulating the decommit phase when the sender is statically corrupted:** $\mathcal{S}$ first checks whether the decommitted value is the value stored during the commit phase and whether the decommitment is valid. If these two conditions are met, then $\mathcal{S}$ sends $(\text{reveal}, sid)$ to $\mathcal{F}_{\text{COM}}$. Otherwise, $\mathcal{S}$ ignores the message. Next, we prove that $\mathcal{Z}$ cannot distinguish an interaction of protocol $\pi_{\text{COM}}$ with $\mathcal{A}$, corrupting the sender, from an interaction of $\mathcal{S}$ with $\mathcal{F}_{\text{COM}}$. Formally,

**Claim 5.6.** *The following two distribution ensembles are computationally indistinguishable,*

$$\left\{\mathbf{View}_{\pi_{\text{COM}}, \mathcal{A}, \mathcal{Z}}^{\mathcal{F}_{\text{CRS}}}(n)\right\}_{n \in \mathbb{N}} \overset{c}{\approx} \left\{\mathbf{View}_{\pi_{\text{IDEAL}}, \mathcal{S}, \mathcal{Z}}^{\mathcal{F}_{\text{COM}}}(n)\right\}_{n \in \mathbb{N}}.$$

*Proof.* Here we need to argue that the simulator outputs fail with negligible probability and that the receiver outputs the same message in both executions. Recall first the event for which the simulator fails occurs when it computes the polynomials $q_0$ and $q_1$ and then finds out that both of them satisfy the condition in case 1. In this case, $\mathcal{A}$ can equivocate the committed message in the decommit phase. We thus prove that the probability that

$\mathcal{A}$ can generate such polynomials is negligible, which implies that the probability that $\mathcal{S}$ fails is negligible. More precisely, we prove in the following lemma that the probability that $\mathcal{A}$ can break the binding property is negligible probability. A key point in our proof relies on the fact that a malicious sender cannot bias the coin-tossing outcome (as opposed to the simulator). Formally, $\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\square$

**Simulating the commit phase when the sender is corrupted after the encoding phase:** Upon corruption, $\mathcal{S}$ receives the sender's input $m$. It then reveals the sender's randomness just as it would do when simulating a decommitment for an uncorrupted sender. Computational indistinguishability follows similarly to the case that the receiver is statically corrupted.

**Simulating the commit phase when the receiver is corrupted anywhere during the commit phase:** Recall that $\mathcal{S}$ honestly simulates the receiver messages, and thus, it can simply reveal the randomness of the receiver.

## A. From Bit OT to String OT

In this section, we demonstrate how to transform a bit OT protocol to a string OT protocol in a way that preserves both defensible private with respect to a corrupted receiver and receiver private. At a high level, in order to obtain an $n$-bit OT we repeat the bit OT protocol in parallel $n$ times. Given two $n$-bit strings $(s_0, s_1)$, the sender enters the $i$th bit of $s_0$ and $s_1$ as the bit inputs for the $i$th OT execution. In addition, the receiver must use the same index for all executions. We next prove that the receiver privacy of the transformed protocol follows easily using a simple hybrid argument. Defensible privacy, on the other hand, holds since the receiver is required to produce a good defense for the $n$ parallel executions simultaneously and a good defense should show that the receiver supplied the same index in all the executions. We note that our proof works for random OT, which is sufficient for our purposes, yet can be extended for the more general case. The definition of defensible privacy for $n$-bit OT is provided in Section 2.3. Below we present an equivalent game-based security formulation for bit OT defensible privacy, inspired by [28]. More formally, we consider the following experiment for a protocol $\pi$ and a PPT adversary $\mathcal{A}$:

**Experiment $\mathsf{Expt}_\pi(\mathcal{A})$ :**

1. Choose $s_0, s_1 \in \{0, 1\}$ uniformly at random.
2. Let $\rho_{\mathsf{Sen}}$ be a uniformly chosen random tape for the sender Sen, and let $\mathsf{trans}$ be a transcript of an interaction between the adversary $\mathcal{A}$ and $\mathsf{Sen}((s_0, s_1), \rho_{\mathsf{Sen}})$.
3. Let $((r, \rho_{\mathsf{Rec}}), s^*)$ be the output of $\mathcal{A}(1^n)$ on transcript $\mathsf{trans}$ where $(r, \rho_{\mathsf{Rec}})$ is a defense and $s^*$ is a guess for $s_{1-r}$.
4. Output 1 only if $(r, \rho_{\mathsf{Rec}})$ is a good defense for $\mathcal{A}$ in $\mathsf{trans}$ and $s^* = s_{1-r}$.

**Definition A.1.** An OT protocol is defensible private with respect to a corrupted receiver if for any PPT adversary $\mathcal{A}$ there exists a negligible function $\mu(\cdot)$ such that for all sufficiently large $n$'s,

$$\Pr[\mathsf{Expt}_\pi(\mathcal{A}(1^n)) = 1] \leq \frac{1}{2} + \mu(n).$$

We now provide our transformation and prove correctness. Let $\pi_{\mathrm{OT}}$ be a bit OT protocol. We construct a string OT protocol $\pi_{\mathrm{OT}}^n$ using $\pi_{\mathrm{OT}}$ as follows.

---

**Protocol 2.**   *(Protocol $\pi_{\mathrm{OT}}^n$)*

    **Input:** *The sender* Sen *has input* $(v_0, v_1)$ *where* $v_0, v_1 \in \{0,1\}^n$ *and the receiver* Rec *has input* $u \in \{0,1\}$.
    **The protocol:**

*The parties participate in n executions of the OT protocol $\pi_{\mathrm{OT}}$ where the receiver uses u as its input in all executions and the sender uses the ith bits of $v_0$ and $v_1$ as its input in execution i.*

---

**Lemma A.1.**   *Assume that $\pi_{\mathrm{OT}}$ is a bit OT protocol that is defensible private with respect to a corrupted receiver and receiver private. Then $\pi_{\mathrm{OT}}^n$ is a string OT protocol that is defensible private with respect to a corrupted receiver and receiver private.*

*Proof.*   We first prove receiver privacy of the string OT protocol.

*Receiver privacy:*   Loosely speaking, receiver privacy requires that no malicious sender can distinguish the case when the receiver's input is 0 and 1, with non-negligible probability. Suppose for contradiction that there exist a PPT adversary $\mathcal{A}$, distinguisher $D$ and polynomial $p(\cdot)$ such that $D$ distinguishes the following distributions with probability at least $\frac{1}{p(n)}$ for infinitely many $n$'s,

- $\{\textbf{View}_{\mathcal{A}, \pi_{\mathrm{OT}}^n}[\mathcal{A}(1^n), \mathrm{Rec}(1^n, 0)]\}_{n \in \mathbb{N}}$,
- $\{\textbf{View}_{\mathcal{A}, \pi_{\mathrm{OT}}^n}[\mathcal{A}(1^n), \mathrm{Rec}(1^n, 1)]\}_{n \in \mathbb{N}}$.

Fix an $n$ for which this happens. We construct $\mathcal{A}'$ and distinguisher $D'$ using $\mathcal{A}$ and $D$ that violates the receiver privacy of $\pi_{\mathrm{OT}}$. We introduce a sequence of intermediate hybrid experiments $H_0, \ldots, H_n$, where in hybrid $H_i$ we consider a receiver $\mathrm{Rec}_i$ that follows the honest receiver's code in each of the $n$ parallel executions of $\pi_{\mathrm{OT}}$ with the exception that it uses the input 1 in the first $i$ executions and 0 in the remaining executions. Let $\mathsf{hyb}_i(n)$ denote the view of the adversary in hybrid $H_i$. Then by construction we have that

$$\mathsf{hyb}_0(n) = \textbf{View}_{\mathcal{A}, \pi_{\mathrm{OT}}^n}[\mathcal{A}(1^n), \mathrm{Rec}(1^n, 0)] \text{and}$$
$$\mathsf{hyb}_n(n) = \textbf{View}_{\mathcal{A}, \pi_{\mathrm{OT}}^n}[\mathcal{A}(1^n), \mathrm{Rec}(1^n, 1)].$$

Moreover, using a standard hybrid argument there exists $i$ such that $D$ distinguishes $\mathsf{hyb}_{i-1}(n)$ and $\mathsf{hyb}_i(n)$ with probability at least $\frac{1}{np(n)}$.

Then adversary $\mathcal{A}'$ is defined as follows. It internally emulates the hybrid experiment $\mathsf{hyb}_{i-1}(n)$ by playing the role of the honest receiver against $\mathcal{A}$, with the exception that it forwards $\mathcal{A}$'s messages in the $i$th execution to an external receiver. Consider the function reconstruct that on input the view of $\mathcal{A}'$ in an interaction using $\pi_{\mathrm{OT}}$ reconstructs the view of $\mathcal{A}$ in the internal emulation of $\mathcal{A}'$. It follows from our construction that if the receiver uses input 0 in the interaction with $\mathcal{A}'$, this view of $\mathcal{A}$ is identically distributed to $\mathsf{hyb}_{i-1}(n)$. If the receiver uses input 1, the view is identically distributed to $\mathsf{hyb}_i(n)$. More precisely,

$$\text{reconstruct}(\mathbf{View}_{\mathcal{A}',\pi_{\mathrm{OT}}}[\mathcal{A}(1^n), \mathrm{Rec}(1^n, 0)]) = \mathsf{hyb}_{i-1}(n)$$
$$\text{reconstruct}(\mathbf{View}_{\mathcal{A}',\pi_{\mathrm{OT}}}[\mathcal{A}(1^n), \mathrm{Rec}(1^n, 1)]) = \mathsf{hyb}_i(n).$$

Next, we construct a distinguisher $D'$. On input a view of $\mathcal{A}'$, runs the function reconstruct on the view and runs $D$ on the output of the function. Finally, $D'$ outputs what $D$ outputs. It now follows that

$$\Pr[D'(\mathbf{View}_{\mathcal{A}',\pi_{\mathrm{OT}}}[\mathcal{A}(1^n), \mathrm{Rec}(1^n, 0)]) = 1] = \Pr[D(\mathsf{hyb}_{i-1}(n)) = 1]$$
$$\Pr[D'(\mathbf{View}_{\mathcal{A}',\pi_{\mathrm{OT}}}[\mathcal{A}(1^n), \mathrm{Rec}(1^n, 1)]) = 1] = \Pr[D(\mathsf{hyb}_i(n)) = 1]$$

which implies that $D'$ distinguishes $\mathcal{A}'$'s view when the receiver's input are 0 and 1 because $D$ distinguishes $\mathsf{hyb}_{i-1}(n)$ and $\mathsf{hyb}_i(n)$ with probability at least $\frac{1}{np(n)}$, which contradicts the receiver privacy of $\pi_{\mathrm{OT}}$.

*Defensible privacy with respect to a corrupted receiver:* As mentioned before, we reduce the security of defensible privacy for string OT according to Definition 2.6 to the game-based formulation of defensible privacy for bit OT (cf. Definition A.1). Assume by contradiction that there exists an adversary $\mathcal{A}$ that violates the defensible privacy of $\pi_{\mathrm{OT}}^n$ with respect to a corrupted receiver. More precisely, suppose there exist a PPT adversary $\mathcal{A}$, distinguisher $D$ and polynomial $p(\cdot)$ such that for infinitely many $n$'s, $D$ distinguishes the following distributions with probability at least $\frac{1}{p(n)}$,

- $\{\Gamma(\mathbf{View}_{\mathcal{A}}[\mathrm{Sen}(1^n, (U_0^n, U_1^n)), \mathcal{A}(1^n)], U_{1-b}^n)\}$ and
- $\{\Gamma(\mathbf{View}_{\mathcal{A}}[\mathrm{Sen}(1^n, (U_0^n, U_1^n)), \mathcal{A}(1^n)], \bar{U}^n)\}$,

where $\Gamma(v, *)$ equals $(v, *)$ if when following the execution $\mathcal{A}$ outputs a good defense for $\pi$, and $\perp$ otherwise, $b$ is Rec's input in this defense and $U_0^n, U_1^n, \bar{U}^n$ are independent random variables that are uniformly distributed over $\{0, 1\}^n$. Fix $n$ for which this happens. Then we rewrite these distributions more explicitly:

$$\{\Gamma(\mathbf{View}_{\mathcal{A}}[\mathrm{Sen}(1^n, (U_0^n, U_1^n)), \mathcal{A}(1^n)], U_{1-b}^n)\}$$
$$= \{s_0 \leftarrow U_0^n; s_1 \leftarrow U_1^n, z \leftarrow \bar{U}^n; v$$
$$\leftarrow \mathbf{View}_{\mathcal{A}}[\mathrm{Sen}(1^n, (s_0, s_1)), \mathcal{A}(1^n)] : \Gamma(v, s_{1-b})\}$$
$$\{\Gamma(\mathbf{View}_{\mathcal{A}}[\mathrm{Sen}(1^n, (U_0^n, U_1^n)), \mathcal{A}(1^n)], \bar{U}^n)\}$$
$$= \{s_0 \leftarrow U_0^n; s_1 \leftarrow U_1^n, z \leftarrow \bar{U}^n; v$$
$$\leftarrow \mathbf{View}_{\mathcal{A}}[\mathrm{Sen}(1^n, (s_0, s_1)), \mathcal{A}(1^n)] : \Gamma(v, z)\}.$$

Next, we use $\mathcal{A}$ to construct $\mathcal{A}'$ that breaks the defensible privacy of $\pi_{\mathrm{OT}}$ with respect to a corrupted receiver. We use the experiment formulation of defensible privacy for the bit OT protocol. Toward this, we consider the following sequence of distributions:

$$\overline{\mathsf{hyb}}_i(n) = \{s_0 \leftarrow U_0^n; s_1 \leftarrow U_1^n, z \leftarrow \bar{U}^n; v \leftarrow \mathbf{View}_{\mathcal{A}}[\mathrm{Sen}(1^n, (s_0, s_1)), \mathcal{A}(1^n)] :$$
$$\Gamma(v, (z^1, \ldots, z^i, s_{1-b}^{i+1}, \ldots, s_{1-b}^n))\},$$

where $z = (z^1, \ldots, z^n)$ and $s_0 = (s_0^1, \ldots, s_0^n)$ and $s_1 = (s_1^1, \ldots, s_1^n)$.
Observe that

$$\overline{\mathsf{hyb}}_0(n) = \{\Gamma(\mathbf{View}_{\mathcal{A}}[\mathsf{Sen}(1^n, (U_0^n, U_1^n)), \mathcal{A}(1^n)], U_{1-b}^n)\}$$
$$\overline{\mathsf{hyb}}_n(n) = \{\Gamma(\mathbf{View}_{\mathcal{A}}[\mathsf{Sen}(1^n, (U_0^n, U_1^n)), \mathcal{A}(1^n)], \bar{U}^n)\}.$$

Then using a standard hybrid argument we can conclude that there exists an index $i$ such that $D$ distinguishes $\mathsf{hyb}_{i-1}(n)$ and $\mathsf{hyb}_i(n)$ with probability at least $\frac{1}{np(n)}$. More precisely,

$$\left| \Pr[D(\overline{\mathsf{hyb}}_i(n)) = 1] - \Pr[D(\overline{\mathsf{hyb}}_{i-1}(n)) = 1] \right| > \frac{1}{p(n)}.$$

Without loss of generality, we assume that

$$\Pr[D(\overline{\mathsf{hyb}}_i(n)) = 1] - \Pr[D(\overline{\mathsf{hyb}}_{i-1}(n)) = 1] > \frac{1}{p(n)}. \tag{9}$$

Now, consider a machine $\mathcal{A}'$ that is interacting externally with a sender on input $(s_0, s_1)$ in the protocol $\pi_{\mathrm{OT}}$. $\mathcal{A}'$ internally incorporates $\mathcal{A}$ and proceeds as follows. It starts by emulating an execution with $\mathcal{A}$ by supplying the sender's messages in $\pi_{\mathrm{OT}}^n$ which implies $n$ parallel OT executions. Specifically, $\mathcal{A}'$ supplies random inputs for the sender in all but the $i$th execution, for which it forwards externally to the sender that participates in $\pi_{\mathrm{OT}}$. Denote by $(s_0^j, s_1^j)$ the sender's selected inputs for every $j \neq i$. Upon completion, $\mathcal{A}'$ receives a defense from $\mathcal{A}$. If the defense is not good $\mathcal{A}'$ aborts. Else, $\mathcal{A}'$ computes $w = (w_1, \ldots, w_n)$ as follows:

- $w_j = z^j$ where $z^j$ is sampled at random from $\{0, 1\}$ for $j <= i$.
- $w_j = s_{u_j}^j$ for $j > i$, where $u_j$ is the receiver's input in the $j$th execution which can be obtained from the defense output by $\mathcal{A}$.

Next, $\mathcal{A}'$ invokes $D$ on input $(v, w)$ where $v$ is $\mathcal{A}$'s internally generated view. Let $b$ be the output of $D$ on these inputs and $(u_i, \rho_R)$ be the defense of $\mathcal{A}$ in the $i$th interaction. Then $\mathcal{A}'$ outputs a defense $(u_i, \rho_R)$ and defined its guess for the external sender's other input by $b \oplus w_i$. By construction, we have that $(v, w)$ are sampled in the internal emulation according to $\overline{\mathsf{hyb}}_i(n)$. This means that $\mathcal{A}'$ succeeds in the experiment $\mathsf{Expt}_{\pi_{\mathrm{OT}}}$ when it is given

$$D(\overline{\mathsf{hyb}}_i(n)) \oplus w_i = s_{1-u_i}.$$

Observe that if $w_i = s_{1-u_i}$ then $(v, w)$ in the internal emulation of $\mathcal{A}'$ is distributed according to $\overline{\mathsf{hyb}}_{i-1}(n)$. This means that:

$$\Pr[\mathcal{A}' \text{ wins } \mathsf{Expt}_{\pi_{\mathrm{OT}}} | w_i = s_{1-u_i}] = \Pr[D(\overline{\mathsf{hyb}}_{i-1}(n)) = 0]. \tag{10}$$

Next, we introduce a new distribution $\widehat{U}_j^n$ that is identical to $\widetilde{U}_j^n$ with the exception that the $(j + 1)^{st}$ bit in $\bar{U}_j^n$ is flipped. More precisely, for $j \in [n]$,

$$\widehat{\mathsf{hyb}}_j(n) = \{s_0 \leftarrow U_0^n; s_1 \leftarrow U_1^n, z \leftarrow \bar{U}^n; v \leftarrow \mathbf{View}_{\mathcal{A}}[\mathsf{Sen}(1^n, (s_0, s_1)), \mathcal{A}(1^n)] :$$
$$\Gamma(v, (z^1, \ldots, z^j, 1 \oplus s_{1-b}^{j+1}, \ldots, s_{1-b}^n))\},$$

where $z = (z^1, \ldots, z^n)$, $s_0 = (s_0^1, \ldots, s_0^n)$ and $s_1 = (s_1^1, \ldots, s_1^n)$. Now, since the bits in the $i$th position are complement of each other in $\widehat{U}_{i-1}^n$ and $\bar{U}_{i-1}^n$, and the $i$th bit is randomly distributed in $\widetilde{U}_i^n$, we have that

$$\Pr[D(\overline{\mathsf{hyb}}_i(n)) = 1] = \frac{1}{2}\Pr[D(\overline{\mathsf{hyb}}_{i-1}(n)) = 1] + \frac{1}{2}\Pr[D(\widehat{\mathsf{hyb}}_{i-1}(n)) = 1]. \quad (11)$$

Moreover, $\mathcal{A}'$ succeeds if $D(\widehat{\mathsf{hyb}}_{i-1}(n)) = 1$ when $w_i \neq s_{1-u_i}$. More precisely

$$\Pr[\mathcal{A}' \text{ wins } \mathsf{Expt}_{\pi_{\mathrm{OT}}} | w_i \neq s_{1-u_i}] = \Pr[D(\widehat{\mathsf{hyb}}_i(n)) = 1]. \quad (12)$$

Now, since both $s_{1-r}$ and $w_i$ are chosen at random we have that:

$$\Pr[\mathcal{A}' \text{ wins } \mathsf{Expt}_{\pi_{\mathrm{OT}}}]$$
$$= \Pr[\mathcal{A}' \text{ wins } \mathsf{Expt}_{\pi_{\mathrm{OT}}} | w_i \neq s_{1-u_i}] \Pr[w_i \neq s_{1-u_i}]$$
$$\quad + \Pr[\mathcal{A}' \text{ wins } \mathsf{Expt}_{\pi_{\mathrm{OT}}} | w_i = s_{1-r}] \Pr[w_i = s_{1-u_i}]$$
$$= \frac{1}{2}\Pr[D(\widehat{\mathsf{hyb}}_{i-1}(n)) = 1]$$
$$\quad + \frac{1}{2}\Pr[D(\overline{\mathsf{hyb}}_{i-1}(n)) = 0] \qquad \text{(Using Equations 12 and 10)}$$
$$= \left(\Pr[D(\mathsf{hyb}_i(n)) = 1] - \frac{1}{2}\Pr[D(\overline{\mathsf{hyb}}_{i-1}(n)) = 1]\right)$$
$$\quad + \frac{1}{2}\Pr[D(\overline{\mathsf{hyb}}_{i-1}(n)) = 0] \qquad \text{(Using Equation 11)}$$
$$= \left(\Pr[D(\mathsf{hyb}_i(n)) = 1] - \frac{1}{2}\Pr[D(\overline{\mathsf{hyb}}_{i-1}(n)) = 1]\right)$$
$$\quad + \left(\frac{1}{2} - \frac{1}{2}\Pr[D(\overline{\mathsf{hyb}}_{i-1}(n)) = 1]\right)$$
$$= \frac{1}{2} + \left(\Pr[D(\overline{\mathsf{hyb}}_i(n)) = 1] - \Pr[D(\overline{\mathsf{hyb}}_{i-1}(n)) = 1]\right)$$
$$\geq \frac{1}{2} + \frac{1}{p(n)} \qquad \text{(Using Equation 9)}.$$

This contradicts the defensible privacy against a corrupted receiver in protocol $\pi_{\mathrm{OT}}$. $\square$

## References

[1] B. Barak, R. Canetti, J.B. Nielsen, R. Pass, Universally composable protocols with relaxed set-up assumptions, in *FOCS*, (2004), pp. 186–195

[2]  O. Blazy, C. Chevalier, D. Pointcheval, D. Vergnaud, Analysis and improvement of lindell's uc-secure commitment schemes, in *ACNS*, (2013), pp. 534–551

[3]  D. Beaver, Foundations of secure interactive computing, in *CRYPTO*, (1991), pp. 377–391

[4]  R. Canetti, Universally composable security: a new paradigm for cryptographic protocols, in *FOCS*, (2001), pp. 136–145

[5]  S.G. Choi, D. Dachman-Soled, T. Malkin, H. Wee, Improved non-committing encryption with applications to adaptively secure protocols, in *ASIACRYPT*, (2009), pp. 287–302

[6]  S.G. Choi, D. Dachman-Soled, T. Malkin, H. Wee, Simple, black-box constructions of adaptively secure protocols, in *TCC*, (2009), pp. 387–402

[7]  R. Canetti, Y. Dodis, R. Pass, S. Walfish, Universally composable security with global setup, in *TCC*, (2007), pp. 61–85

[8]  R. Canetti, M. Fischlin, Universally composable commitments, in *CRYPTO*, (2001), pp. 19–40

[9]  R. Canetti, U. Feige, O. Goldreich, M. Naor, Adaptively secure multi-party computation, in *STOC*, (1996), pp. 639–648

[10]  R. Canetti, E. Kushilevitz, Y. Lindell, On the limitations of universally composable two-party computation without set-up assumptions. *J. Cryptol.* **19**(2), 135–167 (2006)

[11]  S.G. Choi, J. Katz, H. Wee, H.-S. Zhou, Efficient, adaptively secure, and composable oblivious transfer with a single, global CRS, in *PKC*, (2013), pp. 73–88

[12]  R. Canetti, Y. Lindell, R. Ostrovsky, A. Sahai, Universally composable two-party and multi-party secure computation, in *STOC*, (2002), pp. 494–503

[13]  R. Canetti, R. Pass, A. Shelat, Cryptography from sunspots: how to use an imperfect reference string, in *FOCS*, (2007), pp. 249–259

[14]  B. David, R. Dowsley, A.C.A. Nascimento, Universally composable oblivious transfer based on a variant of LPN, in *CANS*, (2014), pp. 143–158

[15]  I. Damgård, J. Groth, Non-interactive and reusable non-malleable commitment schemes, in *STOC*, (2003), pp. 426–437

[16]  D. Dachman-Soled, T. Malkin, M. Raykova, M. Venkitasubramaniam, Adaptive and concurrent secure computation from new adaptive, non-malleable commitments, in *ASIACRYPT*, (2013), pp. 316–336

[17]  I. Damgård, J.B. Nielsen, Improved non-committing encryption schemes based on a general complexity assumption, in *CRYPTO*, (2000), pp. 432–450

[18]  I. Damgård, J.B. Nielsen, Perfect hiding and perfect binding universally composable commitment schemes with constant expansion factor, in *CRYPTO*, (2002), pp. 581–596

[19]  B.M. David, A.C.A. Nascimento, J. Müller-Quade, Universally composable oblivious transfer from lossy encryption and the mceliece assumptions, in *ICITS*, (2012), pp. 80–99

[20]  I. Damgård, J.B. Nielsen, C. Orlandi, On the necessary and sufficient assumptions for UC computation, in *TCC*, (2010), pp. 109–127

[21]  I. Damgård, A. Scafuro, Unconditionally secure and universally composable commitments from physical assumptions, in *ASIACRYPT*, (2013), pp. 100–119

[22]  S. Even, O. Goldreich, A. Lempel, A randomized protocol for signing contracts. *Commun. ACM*, **28**(6), 637–647 (1985)

[23]  Y. Gertner, S. Kannan, T. Malkin, O. Reingold, M. Viswanathan, The relationship between public key encryption and oblivious transfer, in *FOCS*, (2000), pp. 325–335

[24]  V. Goyal, C.-K. Lee, R. Ostrovsky, I. Visconti, Constructing non-malleable commitments: a black-box approach, in *FOCS*, (2012), pp. 51–60

[25]  O. Goldreich, S. Micali, A. Wigderson, How to play any mental game or A completeness theorem for protocols with honest majority, in *STOC*, (1987), pp. 218–229

[26]  O. Goldreich, *Foundations of Cryptography: Basic Tools*. (Cambridge University Press, Cambridge, 2001)

[27]  I. Haitner, Semi-honest to malicious oblivious transfer—the black-box way, in *TCC*, (2008), pp. 412–426

[28]  I. Haitner, Y. Ishai, E. Kushilevitz, Y. Lindell, E. Petrank, Black-box constructions of protocols for secure computation. *SIAM J. Comput.* **40**(2), 225–266 (2011)

[29]  S. Halevi, Y.T. Kalai, Smooth projective hashing and two-message oblivious transfer. *J. Cryptol.*, 25(1):158–193 (2012)

[30]  C. Hazay, A. Patra, One-sided adaptively secure two-party computation, in *TCC*, (2014), pp. 368–393

[31] C. Hazay, M. Venkitasubramaniam, On black-box complexity of universally composable security in the CRS model. *IACR Cryptol. ePrint Arch.*, 2015, 488 (2015)

[32] Y. Ishai, E. Kushilevitz, Y. Lindell, E. Petrank, Black-box constructions for secure computation, in *STOC*, (2006), pp. 99–108

[33] Y. Ishai, M. Prabhakaran, A. Sahai, Founding cryptography on oblivious transfer—efficiently, in *CRYPTO*, (2008), pp. 572–591

[34] R. Impagliazzo, S. Rudich, Limits on the provable consequences of one-way permutations, in *CRYPTO*, (1988), pp. 8–26

[35] J. Kilian, Founding cryptography on oblivious transfer, in *STOC*, (1988), pp. 20–31

[36] Y.T. Kalai, Y. Lindell, M. Prabhakaran, Concurrent composition of secure protocols in the timing model. *J. Cryptol.*, **20**(4), 431–492 (2007)

[37] S. Kiyoshima, H. Lin, M. Venkitasubramaniam, A unified approach to constructing black-box UC protocols in trusted setup models, in *TCC*, (2017), pp. 776–809

[38] J. Katz, R. Ostrovsky, Round-optimal secure two-party computation, in *CRYPTO*, (2004), pp. 335–354

[39] Y. Lindell, General composition and universal composability in secure multi-party computation, in *FOCS*, (2003), pp. 394–403

[40] Y. Lindell, Adaptively secure two-party computation with erasures, in *CT-RSA*, (2009), pp. 117–132

[41] Y. Lindell, Highly-efficient universally-composable commitments based on the DDH assumption, in *EUROCRYPT*, (2011), pp. 446–466

[42] H. Lin, R. Pass, Black-box constructions of composable protocols without set-up, in *CRYPTO*, (2012), pp. 461–478

[43] H. Lin, R. Pass, M. Venkitasubramaniam, A unified framework for concurrent security: universal composability from stand-alone non-malleability, in *STOC*, (2009), pp. 179–188

[44] H. Lin, R. Pass, M. Venkitasubramaniam, A unified framework for UC from only OT, in *ASIACRYPT*, (2012), pp. 699–717

[45] Y. Lindell, H. Zarosim, Adaptive zero-knowledge proofs and adaptively secure oblivious transfer, in *TCC*, (2009), pp. 183–201

[46] H.K. Maji, M. Prabhakaran, M. Rosulek, A zero-one law for cryptographic complexity with respect to computational UC security, in *CRYPTO*, (2010), pp. 595–612

[47] S. Micali, P. Rogaway, Secure computation (abstract), in *CRYPTO*, (1991), pp. 392–404

[48] C. Peikert, V. Vaikuntanathan, B. Waters, A framework for efficient and composable oblivious transfer, in *CRYPTO*, (2008), pp. 554–571

[49] R. Pass, H. Wee, Black-box constructions of two-party protocols from one-way functions, in *TCC*, (2009), pp. 403–418

[50] A. Shamir, How to share a secret. *Commun. ACM*, **22**(11), 612–613 (1979)

[51] A.C.-C. Yao, How to generate and exchange secrets (extended abstract), in *FCOS*, (1986), pp. 162–167