Endemic Oblivious Transfer via Random Oracles, Revisited

Zhelei Zhou[1,2], Bingsheng Zhang[1,2(✉)], Hong-Sheng Zhou[3(✉)], and Kui Ren[1,2]

[1] Zhejiang University, Hangzhou, China
{zl_zhou,bingsheng,kuiren}@zju.edu.cn
[2] ZJU-Hangzhou Global Scientific and Technological Innovation Center, Hangzhou, China
[3] Virginia Commonwealth University, Richmond, USA
hszhou@vcu.edu

Abstract. The notion of Endemic Oblivious Transfer (EOT) was introduced by Masny and Rindal (CCS'19). EOT offers a weaker security guarantee than the conventional random OT; namely, the malicious parties can fix their outputs arbitrarily. The authors presented a 1-round UC-secure EOT protocol under a tailor-made and non-standard assumption, Choose-and-Open DDH, in the RO model.

In this work, we systematically study EOT in the UC/GUC framework. We present a new 1-round UC-secure EOT construction in the RO model under the DDH assumption. Under the GUC framework, we propose the first 1-round EOT construction under the CDH assumption in the Global Restricted Observable RO (GroRO) model proposed by Canetti *et al.* (CCS'14). We also provide an impossibility result, showing there exist *no* 1-round GUC-secure EOT protocols in the Global Restricted Programmable RO (GrpRO) model proposed by Camenisch *et al.* (Eurocrypt'18). Subsequently, we provide the first round-optimal (2-round) EOT protocol with adaptive security under the DDH assumption in the GrpRO model. Finally, we investigate the relations between EOT and other cryptographic primitives.

As side products, we present the first 2-round GUC-secure commitment in the GroRO model as well as a separation between the GroRO and the GrpRO models, which may be of independent interest.

1 Introduction

The security of a cryptographic protocol is typically analyzed under the simulation paradigm [27], where the "formal specification" of the security requirements is modeled as an ideal process, and a real-world protocol is said to securely realize

Z. Zhou and B. Zhang—Work supported by the National Key R&D Program of China (No. 2021YFB3101601), the National Natural Science Foundation of China (Grant No. 62072401), "Open Project Program of Key Laboratory of Blockchain and Cyberspace Governance of Zhejiang Province", and Input Output (iohk.io).
H.-S. Zhou—Work supported in part by NSF grant CNS-1801470, and a Google Faculty Research Award.

C. Hazay and M. Stam (Eds.): EUROCRYPT 2023, LNCS 14004, pp. 303–329, 2023.
https://doi.org/10.1007/978-3-031-30545-0_11

the specification if it "emulates" the ideal process. In the past decades, many variants were proposed: Initially, protocol security was considered in the *standalone* setting, in the sense that the challenged protocol is executed in isolation. Later, *Universal Composibility* (UC) [6] was introduced to analyze protocol security in arbitrary execution environments; in particular, multiple protocol sessions may be executed concurrently in an adversarially coordinated way. Note that protocols in the UC framework must be *subroutine respecting*, in the sense that all the underlying subroutines are only created for the challenged protocol instance and cannot be directly accessed by any other protocols or even the other instances of the same protocol. To address this drawback, Canetti *et al.* [7] proposed the Generalized Universal Composibility (GUC) framework.

Endemic Oblivious Transfer. The notion of *Endemic Oblivious Transfer* (EOT) was introduced by Masny and Rindal [34] as a weaker version of Random OT (ROT). In an EOT protocol, the sender has no input, and the receiver inputs a choice bit $b \in \{0,1\}$; at the end of EOT, the sender outputs two random elements (m_0, m_1), and the receiver outputs m_b. Although EOT looks similar to the conventional ROT, EOT offers a weaker security guarantee—the malicious sender can fix its output (m_0, m_1) arbitrarily, and the malicious receiver can fix its output m_b arbitrarily. The first 1-round[1] (a.k.a. non-interactive) EOT/ROT protocol was proposed by Bellare and Micali [1]. It achieves standalone security against semi-honest adversaries under the DDH assumption in the Common Reference String (CRS) model. As shown in [25], this scheme can also be transformed to achieve malicious security using the Groth-Sahai proof [28]. Later, Garg *et al.* proposed several 1-round UC-secure EOT protocols under the well-understood assumptions, (e.g., Decisional Diffie-Hellman (DDH), Quadratic Residuosity (QR) and Learning With Errors (LWE)), in the CRS model [23]. Recently, Masny and Rindal [34] demonstrated a generic construction for 1-round EOT by using any non-interactive key exchange scheme in the Random Oracle (RO) model; however, their generic construction only achieves standalone security. Masny and Rindal [34,35] then provided a 1-round *UC-secure* EOT protocol but under a tailor-made computational assumption called "*Choose-and-Open DDH* (CODDH)", in the RO model. We remark that, different from the DDH, the CODDH is a new assumption, and its hardness is yet to be further studied.

(Global) Random Oracles. Random oracle (RO) model [2] is a popular idealized setup model that has been widely used to justify the security of efficient cryptographic protocols. In spite of its known inability to provide provable guarantees when RO is instantiated with a real-world hash function [8], RO is still a promising setup since it is generally accepted that security analysis in the RO model does provide strong evidences to the resilience of the protocol in

[1] In this work, we consider the simultaneous communication model with a rushing adversary, where both parties can send messages to each other within the same round. The rushing adversary can delay sending messages on behalf of corrupted parties in a given round until the messages sent by all the uncorrupted parties in that round have been received. Note that this is different from the simultaneous messaging requirement in [30], which deals with a non-rushing adversary.

question in the presence of practical attacks [9]. In fact, RO model draws increasing attention along with recent advancement of the blockchain technology.

Local RO Model vs. Global RO Models. The "local" RO model is often used in the UC framework where the simulator is allowed to simulate it in the ideal world, and it grants the simulator two advantages: (i) observability: the simulator can see what values the parties query the RO on; (ii) programability: the simulator can program RO query responses as long as they "look" indistinguishable from the real ones. In the GUC framework [7], a "global" RO is external to the simulator; to facilitate simulation, some "extra power" needs to be granted to the simulator. In the literature, two main strengthened variants of the global RO model were proposed: global RO with restricted observability (GroRO) model proposed by Canetti *et al.* [9] and global RO with restricted programmability (GrpRO) model proposed by Camenisch *et al.* [5]. Here, the restricted observability and programmability stand for the "extra power" that the simulator has but the adversary does not have.

1.1 Problem Statement

Constructing EOT in (Global) RO Models. As mentioned above, it is known that one can build a 1-round UC-secure EOT protocol under the well-known assumptions in the local CRS model [23]; however, in the local RO model, the recent construction by Masny and Rindal [34,35] was based on a *non-standard* assumption i.e., the CODDH assumption. A natural question to ask is: can we construct a 1-round UC-secure EOT protocol under well-understood assumptions (e.g., DDH assumption) in the local RO model?

Compared to local setups (e.g., local CRS and local RO), global setups are more practical in real life applications. However, very little research work has been done for constructing EOT protocols under a global setup. Our main goal here is to construct a 1-round EOT protocol using global setups. We emphasize that local setups are helpful for us to construct a provably secure 1-round EOT protocol. For example, in the local CRS model, both parties can utilize the shared string, i.e., the CRS, to generate the correlated information for the remaining protocol execution. In other words, *the CRS can be viewed as an extra round of communication messages* during the protocol execution. Intuitively, the security analysis can go through: the simulator is allowed to generate the CRS along with the trapdoor; then the trapdoor information will help the simulator to complete the simulation. In the local RO model, the situation is similar: in the protocol execution, the protocol players may query the RO at certain predefined points to obtain corresponding responses; in a very fuzzy way, it also can be viewed as an extra round of communication messages. In the security analysis, the simulator is allowed to program the RO on those predefined points; this gives the simulator advantages over the adversary which will help the simulator to complete the simulation.

The situation is very different when we use a **global** setup for constructing 1-round EOT protocols. First, we remark that, as already proven in [7], it is impossible to construct a non-trivial two-party computation protocols (including EOT) using a global CRS. To bypass this impossibility, Canetti *et al.* proposed the Augmented CRS (ACRS) model [7]; however, known technique of building non-trivial two-party computation protocols in the ACRS model requires coin-flipping [7,18], which increases round complexity. The good news is that it might be possible to construct a 1-round EOT protocol using a global RO model; note that, different global RO models (e.g., the GroRO [9] and the GrpRO [5]) have been introduced for constructing non-trivial two-party computation protocols. We must remark that, technical difficulty remains. Typically, a global RO is instantiated with a predefined hash function. It seems that the aforementioned design and analysis ideas using local ROs still work: both parties may still be able to utilize the shared hash function on some predefined points to generate the corresponding responses for the remaining protocol execution; unfortunately, it is not true. Below, we provide our elaboration: (1) in the GroRO model, the simulator is not allowed to program the global RO and thus cannot obtain the "trapdoor" of the corresponding responses; as a result, it is unclear how we will be able to complete the security analysis; (2) in the GrpRO model, the simulator is only allowed to program the unqueried points, and the simulator may not be able to program the global RO on those predefined points since the environment may have already queried them before the protocol execution. Given the technical difficulty, we ask the following major research question:

> *In the GUC framework, does there exist a 1-round EOT protocol under well-understood assumptions in the GroRO/GrpRO model?*

For completeness, we also construct new 1-round UC-secure EOT protocols in the local RO model.

Understanding the Complexity of EOT. In addition to the concrete protocol constructions, we are also interested in understanding the complexity, including the power and the limits, of the cryptographic task of EOT. More precisely, what are the relations between EOT and other well-known secure computation tasks? For example, is EOT fundamentally different from ROT or (1-out-of-2) OT? In [34], Masny and Rindal have already initialized the investigation of this interesting problem: They proposed a new OT notion called Uniform OT (UOT) which also looks similar to the conventional ROT, except that it offers a strong security guarantee that no adversary can bias the distribution of the ROT outputs. They showed that it is possible to build UOT based on an EOT and a coin-tossing protocol; however, it is unclear if the coin-tossing protocol can be built from an EOT protocol. We thus ask the following question:

> *What is the relation between the EOT and other cryptographic primitives (such as coin-tossing and UOT etc.)?*

Understanding the Complexity of Global RO Models. Finally, let us go back to the global setups we used in this work. Recall that, the GroRO and the GrpRO models provide different aspects of "extra power" to the simulator. Are these two different global RO models, essentially equivalent? Or one is strictly stronger than the other? It raises our last question:

What is the relation between the GroRO model and the GrpRO model?

Our goal is to provide a comprehensive and thorough investigation of constructing EOT via ROs. From a practical point of view, if the above questions could be answered, we would see highly efficient constructions for EOT. From a theoretical point of view, if (some of) the above questions could be answered, we would have a better understanding of the relation between EOT and many secure computation tasks; we could also have a better understanding of the power and limits of different global RO models.

1.2 Our Results

In this work, we investigate the above problems. Our results can be summarized as follows.

Constructing EOT via (Global) ROs. Table 1 depicts a selection of our new constructions.

Table 1. Comparison with state-of-the-art round-optimal EOT protocols under computational assumptions that related to the cyclic groups.

Protocol	#Round	Security	Computational Assumption	Setup Assumption
Garg *et al.* [23,24][a]	1	UC+Static	DDH	CRS
Masny and Rindal [34,35]	1	UC+Static	CODDH [b]	RO
Canetti *et al.* [11]	1	UC+Adaptive	DDH	GrpRO+CRS
$\Pi_{\mathsf{EOT\text{-}RO}}$(Sect. 3)	1	UC+Static	DDH	RO
$\Pi_{\mathsf{EOT\text{-}GroRO}}$(Sect. 5.1)[c]	1	GUC+Static	CDH	GroRO
$\Pi_{\mathsf{EOT\text{-}GrpRO}}$(Sect. 5.2)[d]	2	GUC+Adaptive	DDH	GrpRO

[a] Garg et al's constructions can be instantiated from different assumptions (e.g., DDH, LWE and QR); but in this table, we focus on constructions using (cyclic) group based assumptions.
[b] Here, CODDH refers to the "Choose-and-Open DDH" assumption which is not known to be reducible to the DDH assumption.
[c] Although protocol $\Pi_{\mathsf{EOT\text{-}GroRO}}$ uses a weaker computational assumption and a less idealized setup than protocol $\Pi_{\mathsf{EOT\text{-}RO}}$ does, the former is less efficient than the latter.
[d] This construction is round-optimal due to Theorem 5, below.

Next, we provide the technical overview for our EOT protocol constructions in the (global) RO models. We first show how to construct a 1-round UC-secure EOT protocol under DDH assumption in the RO model against static adversaries. After that, we turn to the global RO models and show how to construct a 1-round GUC-secure EOT protocol under CDH assumption in the GroRO model. Note that, the situation in the GrpRO model is complicated: We find that there exists no 1-round GUC-secure EOT protocols in the GrpRO model even with static security, and we give a round-optimal (2-round) EOT protocol under DDH assumption against adaptive adversaries.

New Technique: 1-round UC-secure EOT Protocol in the RO Model. We present a new technique that enables the first UC-secure 1-round EOT protocol in the RO model under the DDH assumption (cf. Sect. 3). The basic scheme achieves static security. Intuitively, our technique is as follows. We start with the two-round standalone ROT/EOT protocol in the RO model proposed in [13]. In the 1st round, the sender sends $h := g^s$ to the receiver; in the 2nd round, the receiver uses sender's message to compute $B := g^r h^b$ and sends B back, where $b \in \{0, 1\}$ is the choice bit; finally, the sender outputs $m_0 := \mathtt{Hash}(B^s)$ and $m_1 := \mathtt{Hash}((\frac{B}{h})^s)$, where $\mathtt{Hash}$ is a predefined hash function and it is modeled as a RO; the receiver outputs $m_b := \mathtt{Hash}(h^r)$. Although this protocol is simple and efficient, it cannot achieve UC security [26,33].

Our technique is presented as follows. The dependence of the sender's message in [13] can be eliminated such that the receiver's message can be produced simultaneously in the same round. The idea is to let the receiver produce the commitment key h instead of waiting it from the sender. How to generate a random group element and be oblivious to its discrete logarithm? This can be achieved by setting $h := \mathtt{Hash}(\mathsf{seed})$, where seed is some randomly sampled string. Similar technique can be found in [11]. Now the 1-round (non-interactive) version of [13] roughly works as follows. The sender sends $z := g^s$ to the receiver; meanwhile, the receiver picks $h := \mathtt{Hash}(\mathsf{seed})$ and computes $B := g^r h^b$, and then it sends (seed, B) to the sender; finally, the sender computes $h := \mathtt{Hash}(\mathsf{seed})$ and outputs $m_0 := \mathtt{Hash}(B^s)$ and $m_1 := \mathtt{Hash}((\frac{B}{h})^s)$; the receiver outputs $m_b := \mathtt{Hash}(z^r)$.

Further, to make the protocol UC-secure, certain *extractability* is needed: (i) when the sender is malicious, the simulator should be able to extract the sender's private randomness s, so the simulator can compute both m_0 and m_1; (ii) when the receiver is malicious, the simulator should be able to extract the receiver's choice bit. In order to extract the sender's s, we let the sender additionally generate a RO-based straight-line extractable NIZK argument [22,32,38]. In order to extract the receiver's choice bit b, we let the receiver computes the ElGamal encryption of b instead of the Pedersen commitment. We then let the receiver additionally generate a NIZK argument to ensure the correctness of the ElGamal encryption. Note that, we do not need the straight-line extractability here, since the simulator can program the RO to obtain $\log_g h$ and thus be able to decrypt the ElGamal ciphertext to extract b.

1-round GUC-secure EOT Protocol in the GroRO Model. Turning to the GUC setting, we propose the first 1-round EOT construction under the CDH assumption in the GroRO model (cf. Sect. 5.1). Compared to our UC-secure construction, this one requires *weaker* a computational assumption.

Recall that, in our UC-secure EOT protocol, we let the sender send $z := g^s$ together with a straight-line extractable NIZK argument. The straight-line extractable NIZK argument gives the simulator the ability to extract s. However, Pass showed that it is impossible to construct NIZK arguments in observable RO model [38], let alone NIZK arguments with straight-line extractability. The good news is that straight-line extractable NIWH argument is sufficient for our purpose, and it exists in the GroRO model [38]. Therefore, we let the sender generate a straight-line extractable NIWH argument of s such that $z = g^s$ instead. Next, to extract the receiver's choice bit, our UC-secure construction utilizes the programmability of RO; however, $\mathcal{G}_{\mathsf{roRO}}$ does not offer programability, so a different approach shall be taken. In particular, we let the receiver compute a Pedersen commitment to the choice bit $B := g^r h^b$, and generate a straight-line extractable NIWH argument of (r, b) such that $B = g^r h^b$. Analogously to the sender side, the straight-line extractable NIWH argument gives the simulator extractability.

Understanding the Power/Limits of Different Global ROs. Here we discuss the feasibility result and impossible result in the GrpRO model. In addition to that, we also reveal a separation between the GroRO and the GrpRO model.

A Separation Between the GroRO Model and the GrpRO Model. To show this separation, we first give a new impossibility result, showing that there exists *no* 1-round GUC-secure EOT protocol in the GrpRO model even with static security (cf. Sect. 5.2). By combining this negative result in the GrpRO model and the aforementioned positive result in the GroRO model, we demonstrate a separation between the GroRO model and the GrpRO model. More precisely, let $\mathcal{G}_{\mathsf{roRO}}, \mathcal{G}_{\mathsf{rpRO}}$ be the functionalities of the GroRO and the GrpRO model, we present the relation of these global RO models in Fig. 1.

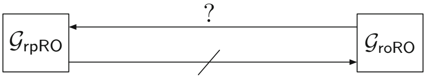


Fig. 1. The relation between the $\mathcal{G}_{\mathsf{roRO}}$ model and the $\mathcal{G}_{\mathsf{rpRO}}$ model. Here, "$\mathsf{A} \nrightarrow \mathsf{B}$" denotes that A does not imply B. In addition, "$\mathsf{A} \overset{?}{\rightarrow} \mathsf{B}$" denotes that whether A implies B remains unknown.

New Impossibility Results in the GrpRO Model. Here we will present more details about the aforementioned impossibility result in the GrpRO model. The impossibility is proven by contradiction (cf. Sect. 5.2). Suppose that there exists such a 1-round GUC-secure EOT protocol in the GrpRO model. Let us first consider the case where the receiver is corrupted, and the simulator needs to extract the choice bit of the receiver from its message. Recall that, the GrpRO only grants the simulator the restricted programmability: the simulator can program the unqueried points without being detected. More importantly, unlike local RO, the simulator cannot program a global RO on the fly, as it cannot see which point is queried at this moment. Thus, the simulator needs to find a way to enforce the corrupt receiver to query the simulator's programmed points. However, in a one simultaneous round protocol, the messages between parties have no dependency. Hence the simulator cannot enforce the corrupt receiver to produce its message on the programmed points, and has no advantages. If the simulator still succeeds to extract the corrupted receiver's choice bit, then we have the following attack. The adversary corrupts the sender, and instructs the sender to run the simulator algorithm above to extract the choice bit from the message sent by the receiver/simulator. However, the simulator has no idea about the real choice bit, thus with 1/2 probability the simulation would fail.

New Feasibility: Round-Optimal GUC-secure EOT Protocol in the GrpRO Model. To complete the picture, we also give a round-optimal (2-round) EOT protocol with adaptive security under the DDH assumption in the GrpRO model (cf. Sect. 5.2). Here, we do not consider simultaneous messaging in the same round. Our intuition comes from the UC-secure EOT protocol in the CRS+GrpRO model proposed by Canetti *et al.* [11]. In their protocol, the CRS consists of two group elements $g, h \in \mathbb{G}$, and the simulator knows $\log_g h$. The sender computes $z := g^r h^s$, while the receiver generates $(G, H) := \mathtt{Hash}(\mathsf{seed})$ and computes two Pedersen commitments to the choice bit using two sets of the parameter, i.e., (g, G) and (h, H), and the same randomness.

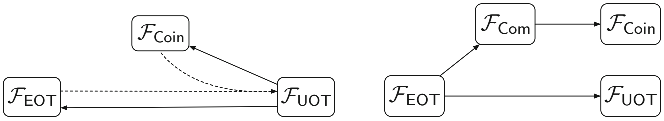
To eliminate the CRS, we let the sender generate the first set of the parameter $(g, h) := \mathtt{Hash}(\mathsf{seed}_1)$ where seed_1 is an uniformly sampled string. At the same time, the sender computes $z := g^r h^s$ using random $r, s \leftarrow \mathbb{Z}_q$ and sends seed, z to the receiver in the first round. In the second round, the receiver first checks if seed_1 is a programmed point. If not, the receiver generates the second set of the parameter $(G, H) := \mathtt{Hash}(\mathsf{seed}_2)$ where seed_2 is an uniformly sampled string. Then the receiver can compute two Pedersen commitments to the choice bit, i.e., $(B_1, B_2) := (g^x G^b, h^x H^b)$ using random $x \leftarrow \mathbb{Z}_q$. Finally, we let the receiver send $(\mathsf{seed}_2, B_1, B_2)$ to the sender. How to make the protocol simulatable in the GrpRO model? We show the simulation strategy as follows: when the receiver is malicious (and the sender is honest), the simulator can extract the receiver's choice bit b by programming the GrpRO (the simulator always succeeds to program the GrpRO since seed_1 is sampled by the honest sender itself) and knowing α such that $h = g^\alpha$; when the sender is malicious (and the receiver is honest), the simulator can compute both m_0 and m_1 by programming the GrpRO (the simulator always succeeds to program the GrpRO since seed_2 is sampled by the honest receiver itself) such that (g, h, G, H) is a DDH tuple.

Understanding the Relation Between EOT and Other Cryptographic Primitives. Here we discuss the complexity of EOT. Our results can be summarized as follows.

EOT Implies UOT and Commitment. In [34], the authors showed that UOT implies EOT. But the work on the opposite direction is incomplete. Let $\mathcal{F}_{\mathsf{EOT}}$, $\mathcal{F}_{\mathsf{UOT}}$ and $\mathcal{F}_{\mathsf{Coin}}$ be the ideal functionalities of EOT, UOT and coin-tossing protocol, respectively. They showed that a UOT protocol can be constructed in the $\{\mathcal{F}_{\mathsf{EOT}}, \mathcal{F}_{\mathsf{Coin}}\}$-hybrid world with unconditional security, and they constructed $\mathcal{F}_{\mathsf{Coin}}$ via only $\mathcal{F}_{\mathsf{UOT}}$. However, it remains unclear whether $\mathcal{F}_{\mathsf{Coin}}$ can be constructed via only $\mathcal{F}_{\mathsf{EOT}}$; therefore, it is still an open question on whether EOT implies UOT? We present the relations that they claimed in Fig. 2(a).

Recall that, Brzuska *et al.* proved that bit commitment can be constructed via 1-out-of-2 OT with unconditional security [3]. What about EOT? Nevertheless, surprisingly, we show that bit commitment can be constructed via a weaker primitive, i.e., EOT with unconditional security (cf. Sect. 4.1).

Our key observation is that the receiver's message can be viewed as the commitment to the choice bit b, and the locally computed message m_b together with b can be viewed as the opening. Typically, a commitment protocol requires both hiding and binding properties. The hiding property holds since the malicious receiver in the EOT cannot learn m_{1-b}, even if it can influence the distribution of m_b. The binding property holds since the malicious sender in the EOT cannot know which message is received by the receiver, even if it can influence the distributions of both m_0 and m_1.


$\mathcal{F}_{\mathsf{Coin}}$
$\mathcal{F}_{\mathsf{EOT}}$
$\mathcal{F}_{\mathsf{UOT}}$
$\mathcal{F}_{\mathsf{Com}}$
$\mathcal{F}_{\mathsf{Coin}}$
$\mathcal{F}_{\mathsf{EOT}}$
$\mathcal{F}_{\mathsf{UOT}}$

(a) The relations claimed in [34]

(b) The relations in this work.

Fig. 2. The relations between EOT and other primitives. "A → B" denotes that A implies B. "A ⇢ B" denotes that A can be transformed into B.

Since it is well-known how to construct $\mathcal{F}_{\mathsf{Coin}}$ via only $\mathcal{F}_{\mathsf{Com}}$, where $\mathcal{F}_{\mathsf{Com}}$ is the commitment functionality, we show that EOT implies UOT and completes the relation between EOT and UOT (cf. Sect. 4.2). We present the relations that explored in this work in Fig. 2(b).

Furthermore, as a side product, we present the first 2-round GUC-secure commitment in the GroRO model (cf. Sect. 5.1), which may be of independent interest. The previous state-of-the-art protocols need 3 rounds [36,42]. Note that this result does not contradict Zhou *et al.*'s impossibility result [42], as their work did not consider simultaneous communication model.

1.3 Related Work

In this work, we mainly focus on the EOT (and OT) protocols in the different variants of RO models, i.e. the local RO model, the GroRO model and the GrpRO model. The EOT (and OT) results in the CRS model can be found in the full version of this paper [41].

In terms of the local RO model, Chou and Orlandi proposed a 3-round OT protocol called "the simplest OT protocol" [13]. This protocol and the protocol in [29] have been found to suffer from a number of issues [4,26,33] and are not UC-secure. In the following, Masny and Rindal showed how to construct EOT protocols from the key exchange schemes in the local RO model [34]. In particular, they provided a 1-round UC-secure construction under a non-standard assumption, i.e., Choose-and-Open DDH (CODDH) assumption [34,35].

Regarding the GroRO model, Canetti *et al.* proposed a 2-round OT protocol under DDH assumption [9], but their protocol is only one-sided GUC-simulatable. Later, fully GUC-secure OT protocols in the GroRO model are proposed [16,19]. Their protocols only need CDH assumption but require no less than 5 rounds of communication. To achieve round-optimal, Canetti *et al.* proposed a 2-round GUC-secure OT protocol in the GroRO model [11], but their protocol requires a stronger assumption, i.e., DDH assumption.

As for the GrpRO model, Canetti *et al.* proposed an adaptive-secure 1-round EOT protocol in the GrpRO+CRS hybrid model [11], but their protocol is only UC-secure since their simulator must know the trapdoor of the CRS.

2 Preliminaries

2.1 Notations

We denote by $\lambda \in \mathbb{N}$ the security parameter. We say that a function $\mathsf{negl} : \mathbb{N} \to \mathbb{N}$ is negligible if for every positive polynomial $\mathrm{poly}(\cdot)$ and all sufficiently large λ, it holds that $\mathsf{negl}(\lambda) < \frac{1}{\mathrm{poly}(\lambda)}$. We use the abbreviation PPT to denote probabilistic polynomial-time. For an NP relation $\mathcal{R}$, we denote by $\mathcal{L}$ its associate language, i.e. $\mathcal{L} = \{x \mid \exists w \text{ s.t. } (x, w) \in \mathcal{R}\}$. We denote by $y := \mathsf{Alg}(x; r)$ the event where the algorithm Alg on input x and randomness r, outputs y. We denote by $y \leftarrow \mathsf{Alg}(x)$ the event where Alg selects a randomness r and sets $y := \mathsf{Alg}(x; r)$. We denote by $y \leftarrow S$ the process for sampling y uniformly at random from the set S. Let q be a λ-bit prime, and $p = 2q + 1$ also be a prime. Let $\mathbb{G}$ be a subgroup of order q of $\mathbb{Z}_p^*$ with the generator g.

2.2 Universal Composability

We formalize and analyze the security of our protocols in the Canetti's Universal Composability (UC) framework [6] and Canetti *et al*'s Generalized UC (GUC) framework [7]. The main difference between the UC and the GUC framework is that the environment $\mathcal{Z}$ cannot have direct access to the setups in the UC framework, whereas $\mathcal{Z}$ is "unconstrained" and can access the setups directly in the GUC framework. The local setups in the UC framework are often modeled as ideal functionalities, whereas the global setups in the GUC framework are often modeled as the *shared functionalities* which are completely analogous to ideal functionalities, except that they may interact with more than one protocol sessions. For that reason, the simulator in the UC framework can simulate the local setups and have the full control over it; whereas, the simulator in the GUC framework has no control over the global setups. We refer interesting readers to see more details in [6,7].

Adversarial Model. In this work, we consider both static corruption (where the adversary corrupts the parties at the beginning of the protocol) and adaptive corruption (where the adversary corrupts the parties at any time). We also consider *rushing* adversaries, who may delay sending messages on behalf of corrupted parties in a given round until the messages sent by all the uncorrupted parties in that round have been received [30].

Secure Communication Model. Many UC-secure protocols assume the parties are interconnected with secure or authenticated channels [7,10]. The secure channel and authenticated channel can be modeled as ideal functionalities $\mathcal{F}_{\mathsf{SC}}$ and $\mathcal{F}_{\mathsf{Auth}}$ respectively [6]. In this work, most of our protocols are designed in the simultaneous communication channel with rushing adversaries, which is different from that [30] deals with non-rushing adversaries. For this reason, we often assume the synchronous channel which can be modeled as $\mathcal{F}_{\mathsf{Syn}}$ [6]. Note that, intuitively, $\mathcal{F}_{\mathsf{Syn}}$ can be viewed an authenticated communication network with storage, which proceeds in a round-based fashion [6,31]. For readability, we will mention which secure communication channel is used in the context and omit it in the protocol description.

2.3 Ideal Functionalities

In this section, we provide ideal functionalities that will be used in UC/GUC security analysis.

OT, UOT and EOT. We start with the Oblivious Transfer (OT). In a OT protocol, there is a sender S holding two private input $m_0, m_1 \in \{0,1\}^\lambda$ and a receiver R holding a choice bit $b \in \{0,1\}$. At the end of the honest execution of the OT protocol, the receiver R will compute m_b. At the same time, the sender should learn nothing about b while the receiver should learn nothing about m_{1-b}. We present the OT functionality $\mathcal{F}_{\mathsf{OT}}$ in Fig. 3.

Functionality $\mathcal{F}_{\mathsf{OT}}$

It interacts with two parties S, R and an adversary $\mathcal{S}$.

Transfer. Upon receiving $(\textsc{Send}, \mathsf{sid}, S, R, m_0, m_1)$ from the sender S, do:

- Record $(\mathsf{sid}, S, R, m_0, m_1)$, and send $(\textsc{Send}, \mathsf{sid}, S, R)$ to R and the adversary $\mathcal{S}$.
- Ignore any subsequent $\textsc{Send}$ commands.

Choose. Upon receiving $(\textsc{Receive}, \mathsf{sid}, S, R, b)$ from R where $b \in \{0,1\}$, do:

- Record (sid, S, R, b), and send $(\textsc{Receive}, \mathsf{sid}, S, R)$ to the sender S and the adversary $\mathcal{S}$.
- Ignore any subsequent $\textsc{Receive}$ commands.

Process. When both $(\mathsf{sid}, S, R, m_0, m_1)$ and (sid, S, R, b) are recorded, do:

- Send $(\textsc{Proceed}?, \mathsf{sid}, S, R)$ to the adversary $\mathcal{S}$.
- Upon receiving $(\textsc{Proceed}, \mathsf{sid}, S)$ from the adversary $\mathcal{S}$, output $(\textsc{Received}, \mathsf{sid}, S, R)$ to the sender S; Upon receiving $(\textsc{No}, \mathsf{sid}, S)$ from the adversary $\mathcal{S}$, output $(\textsc{Abort}, \mathsf{sid}, S)$ to the sender S. Upon receiving $(\textsc{Proceed}, \mathsf{sid}, R)$ from the adversary $\mathcal{S}$, output $(\textsc{Received}, \mathsf{sid}, S, R, m_b)$ to R; Upon receiving $(\textsc{No}, \mathsf{sid}, R)$ from the adversary $\mathcal{S}$, output $(\textsc{Abort}, \mathsf{sid}, R)$ to R.

Fig. 3. The Ideal Functionality $\mathcal{F}_{\mathsf{OT}}$ for Oblivious Transfer

In [34], Masny and Rindal proposed two notions that called Uniform OT (UOT) and Endemic OT (EOT). Both of them are similar to OT, except that the senders have no inputs. The main difference between the UOT and the EOT is that they provide different levels of security guarantees. We describe the UOT first. The UOT functionality samples two uniformly random strings m_0, m_1, and outputs m_0, m_1 to the (potentially malicious) sender and m_b to the (potentially malicious) receiver. The UOT gives a strong security guarantee that any malicious party cannot influence the distribution of the OT messages. Formally, we put the UOT functionality $\mathcal{F}_{\mathsf{UOT}}$ in Fig. 4.

Functionality $\mathcal{F}_{\mathsf{UOT}}$

It interacts with two parties S, R and an adversary $\mathcal{S}$.

Transfer. Upon receiving $(\textsc{Send}, \mathsf{sid}, S, R)$ from S, do:

- Sample $m_0, m_1 \leftarrow \{0,1\}^\lambda$, record $(\mathsf{sid}, S, R, m_0, m_1)$, and send $(\textsc{Send}, \mathsf{sid}, S, R)$ to R and the adversary $\mathcal{S}$.
- Ignore any subsequent Send commands.

Choose. Upon receiving $(\textsc{Receive}, \mathsf{sid}, S, R, b)$ from R where $b \in \{0,1\}$, do:

- Record (sid, S, R, b), and send $(\textsc{Receive}, \mathsf{sid}, S, R)$ to the sender S and the adversary $\mathcal{S}$.
- Ignore any subsequent Receive commands.

Process. When both $(\mathsf{sid}, S, R, m_0, m_1)$ and (sid, S, R, b) are recorded, do:

- Send $(\textsc{Proceed?}, \mathsf{sid}, S, R)$ to the adversary $\mathcal{S}$.
- Upon receiving $(\textsc{Proceed}, \mathsf{sid}, S)$ from the adversary $\mathcal{S}$, output $(\textsc{Received}, \mathsf{sid}, S, R, m_0, m_1)$ to the sender S; Upon receiving $(\textsc{No}, \mathsf{sid}, S)$ from the adversary $\mathcal{S}$, output $(\textsc{Abort}, \mathsf{sid}, S)$ to the sender S. Upon receiving $(\textsc{Proceed}, \mathsf{sid}, R)$ from the adversary $\mathcal{S}$, output $(\textsc{Received}, \mathsf{sid}, S, R, m_b)$ to R; Upon receiving $(\textsc{No}, \mathsf{sid}, R)$ from the adversary $\mathcal{S}$, output $(\textsc{Abort}, \mathsf{sid}, R)$ to R.

Fig. 4. The Ideal Functionality $\mathcal{F}_{\mathsf{UOT}}$ for Uniform Oblivious Transfer

Now let us turn to EOT. Compared to UOT, the EOT functionality gives a weak security guarantee: no matter whether the sender or the receiver is malicious, the malicious party can always determine the distribution of the OT messages. Roughly speaking, if both sender and receiver are honest, the EOT functionality acts as the UOT functionality. If the sender is malicious and the receiver is honest, the EOT functionality lets the adversary determine the message strings m_0, m_1, and it returns the adversarial chosen m_b to the honest receiver after receiving b. If the receiver is malicious and the sender is honest, the EOT functionality lets the adversary determine the message string m_b, and it returns the adversarial chosen m_b and an uniformly sampled m_{1-b} to the honest sender. If both sender and receiver are malicious, the EOT functionality simply aborts. Formally, we put the EOT functionality $\mathcal{F}_{\mathsf{EOT}}$ in Fig. 5.

Random Oracles. Here we introduce two well-known global RO models: Global Restricted Programmable Random Oracle (GrpRO) model proposed by Camenisch *et al.* [5] and Global Restricted Observable Random Oracle (GroRO) model proposed by Canetti *et al.* [9]. We omit the formal description of the well-known local RO functionality $\mathcal{F}_{\mathsf{RO}}$.

Functionality $\mathcal{F}_{\mathsf{EOT}}$

It interacts with two parties S, R and an adversary $\mathcal{S}$.

Transfer/Choose. Upon receiving $(\textsc{Send}, \mathsf{sid}, S, R)$ from the sender S or $(\textsc{Receive}, \mathsf{sid}, S, R, b)$ from the receiver R, do the same as $\mathcal{F}_{\mathsf{UOT}}$ that depicted in Figure 4.

Process. When both $(\mathsf{sid}, S, R, m_0, m_1)$ and (sid, S, R, b) are recorded, do:

- If both the sender S and the receiver R are honest, output $(\textsc{Received}, \mathsf{sid}, S, R, m_0, m_1)$ to the sender S, $(\textsc{Received}, \mathsf{sid}, S, R, m_b)$ to R and $(\textsc{Received}, \mathsf{sid}, S, R)$ to the adversary $\mathcal{S}$.
- Else if the sender S is corrupted and the receiver R is honest, send $(\textsc{Proceed?}, \mathsf{sid}, R)$ to the adversary $\mathcal{S}$. Upon receiving $(\textsc{Proceed}, \mathsf{sid}, R, m_0^*, m_1^*)$ from the adversary $\mathcal{S}$, set $m_0 := m_0^*$, $m_1 := m_1^*$, and output $(\textsc{Received}, \mathsf{sid}, S, R, m_0, m_1)$ to the sender S, $(\textsc{Received}, \mathsf{sid}, S, R, m_b)$ to R; Upon receiving $(\textsc{No}, \mathsf{sid}, R)$ from the adversary $\mathcal{S}$, output $(\textsc{Abort}, \mathsf{sid}, R)$ to R.
- Else if the sender S is honest and the receiver R is corrupted, send $(\textsc{Proceed?}, \mathsf{sid}, S)$ to the adversary $\mathcal{S}$. Upon receiving $(\textsc{Proceed}, \mathsf{sid}, S, m_b^*)$ from the adversary $\mathcal{S}$, set $m_b := m_b^*$, and output $(\textsc{Received}, \mathsf{sid}, S, R, m_0, m_1)$ to the sender S, $(\textsc{Received}, \mathsf{sid}, S, R, m_b)$ to R; Upon receiving $(\textsc{No}, \mathsf{sid}, S)$ from the adversary $\mathcal{S}$, output $(\textsc{Abort}, \mathsf{sid}, S)$ to the sender S.
- Else if both the sender S and the receiver R are corrupted, halt.

Fig. 5. The Ideal Functionality $\mathcal{F}_{\mathsf{EOT}}$ for Endemic Oblivious Transfer

The GrpRO Model. Compared to $\mathcal{F}_{\mathsf{RO}}$, the GrpRO is modeled as a shared functionality $\mathcal{G}_{\mathsf{rpRO}}$ which may interact with more than one protocol sessions. The $\mathcal{G}_{\mathsf{rpRO}}$ answers to the queries in the same way as $\mathcal{F}_{\mathsf{RO}}$: Upon receiving $(\textsc{Query}, \mathsf{sid}, x)$ from any party, $\mathcal{G}_{\mathsf{rpRO}}$ first checks whether the query (sid, x) has been queried before. If not, $\mathcal{G}_{\mathsf{rpRO}}$ selects a random value of pre-specified length $v \leftarrow \{0,1\}^{\ell_{\mathsf{out}}(\lambda)}$, answers with the value v and records the tuple (sid, x, v); otherwise, the previously chosen value v is returned again, even if the earlier query was made by another party. The simulator is only granted the restricted programmability: both the adversary and the simulator are allowed to program the unqueried points of the random oracle, but only the simulator can program it without being detected. More precisely, as depicted in Fig. 6, upon receiving $(\textsc{Program}, \mathsf{sid}, x, v)$ from the simulator/adversary, $\mathcal{G}_{\mathsf{rpRO}}$ first checks whether (sid, x) has been queried before. If not, $\mathcal{G}_{\mathsf{rpRO}}$ stores (sid, x, v) in the query-answer lists. Any honest party can check whether a point has been programmed or not by sending the $(\textsc{IsProgramed}, \mathsf{sid}, x)$ to $\mathcal{G}_{\mathsf{rpRO}}$. Thus, in the real world, the programmed points can always be detected. However, in the ideal world, the simulator $\mathcal{S}$ can escape the detection since it can return $(\textsc{IsProgramed}, \mathsf{sid}, 0)$ when the adversary invokes $(\textsc{IsProgramed}, \mathsf{sid}, x)$ to verify whether a point x has been programmed or not.

The GroRO Model. The GroRO is also modeled as a share functionality $\mathcal{G}_{\mathsf{roRO}}$, and it answers to the queries in the same way as $\mathcal{F}_{\mathsf{RO}}$. The simulator is only granted the restricted observability: some of the queries can be marked as "illegitimate" and potentially disclosed to the simulator. As depicted in Fig. 7, the $\mathcal{G}_{\mathsf{roRO}}$ interacts with a list of ideal functionalities $\bar{\mathcal{F}} = \{\mathcal{F}_1, \ldots, \mathcal{F}_n\}$, where $\mathcal{F}_1, \ldots, \mathcal{F}_n$ are the ideal functionalities for protocols. For any query (sid', x) from any party $P = (\mathsf{pid}, \mathsf{sid})$ where sid' is the content of the SID field, if $\mathsf{sid}' \neq \mathsf{sid}$, then this

Share Functionality $\mathcal{G}_{\mathsf{rpRO}}$

It interacts with a set of parties $\mathcal{P} = \{P_1, \ldots, P_n\}$ and an adversary $\mathcal{S}$. It is parameterized by the output length $\ell_{\mathsf{out}}(\lambda)$. It maintains two initially empty lists $\mathsf{List}, \mathsf{Prog}$.

Query. Upon receiving $(\textsc{Query}, \mathsf{sid}', x)$ from a party $P_i \in \mathcal{P}$ where $P_i = (\mathsf{pid}, \mathsf{sid})$, or the adversary $\mathcal{S}$:

- Check if $\exists\ v \in \{0,1\}^{\ell_{\mathsf{out}}(\lambda)}$ such that $(\mathsf{sid}, x, v) \in \mathsf{List}$. If not, select $v \leftarrow \{0,1\}^{\ell_{\mathsf{out}}(\lambda)}$ and record the tuple (sid', x, v) in List.
- Return $(\textsc{QueryConfirm}, \mathsf{sid}', v)$ to the requestor.

Program. Upon receiving $(\textsc{Program}, \mathsf{sid}, x, v)$ with $v \in \{0,1\}^{\ell_{\mathsf{out}}(\lambda)}$ from the adversary $\mathcal{S}$:

- Check if $\exists\ v' \in \{0,1\}^{\ell_{\mathsf{out}}(\lambda)}$ s.t. $(\mathsf{sid}, x, v') \in \mathsf{List}$ and $v \neq v'$. If so, ignore this input.
- Set $\mathsf{List} := \mathsf{List} \cup \{(\mathsf{sid}, x, v)\}$ and $\mathsf{Prog} := \mathsf{Prog} \cup \{(\mathsf{sid}, x)\}$.
- Return $(\textsc{ProgramConfirm}, \mathsf{sid})$ to $\mathcal{S}$.

IsProgramed. Upon receiving $(\textsc{IsProgramed}, \mathsf{sid}', x)$ from a party $P_i \in \mathcal{P}$ where $P_i = (\mathsf{pid}, \mathsf{sid})$, or the adversary $\mathcal{S}$:

- If the input was given by $P_i = (\mathsf{pid}, \mathsf{sid})$ and $\mathsf{sid} \neq \mathsf{sid}'$, ignore this input.
- If $(\mathsf{sid}', x) \in \mathsf{Prog}$, set $b := 1$; otherwise, set $b := 0$.
- Return $(\textsc{IsProgramed}, \mathsf{sid}', b)$ to the requester.

Fig. 6. The Global Restricted Programmable Random Oracle Model $\mathcal{G}_{\mathsf{roRO}}$

query is considered "illegitimate". After that, $\mathcal{G}_{\mathsf{roRO}}$ adds the tuple (sid', x, v) to the list of illegitimate queries for SID sid', which we denote as $\mathcal{Q}_{\mathsf{sid}'}$. The illegitimate queries $\mathcal{Q}_{\mathsf{sid}'}$ may be disclosed to an instance of ideal functionality $\mathcal{F} \in \bar{\mathcal{F}}$ whose SID is the one of the illegitimate queries, and the ideal functionality instance $\mathcal{F}$ may leak the illegitimate queries $\mathcal{Q}_{\mathsf{sid}'}$ to the simulator.

Share Functionality $\mathcal{G}_{\mathsf{roRO}}$

It interacts with a set of parties $\mathcal{P} = \{P_1, \ldots, P_n\}$ and an adversary $\mathcal{S}$. It is parameterized by the output length $\ell_{\mathsf{out}}(\lambda)$ and a list of ideal functionalities $\bar{\mathcal{F}} := \{\mathcal{F}_1, \ldots, \mathcal{F}_n\}$. It maintains an initially empty list List.

Query. Upon receiving $(\textsc{Query}, \mathsf{sid}', x)$ from a party $P_i \in \mathcal{P}$ where $P_i = (\mathsf{pid}, \mathsf{sid})$, or the adversary $\mathcal{S}$, do the same as $\mathcal{G}_{\mathsf{roRO}}$ depicted in Figure 6, except when $\mathsf{sid} \neq \mathsf{sid}'$, add the tuple (sid', x, v) to the (initially empty) list of illegitimate queries for SID sid', which we denote by $\mathcal{Q}_{\mathsf{sid}'}$.

Observe. Upon receiving a request from an instance of an ideal functionality $\mathcal{F}_i \in \bar{\mathcal{F}}$ with SID sid', return the list of illegitimate queries $\mathcal{Q}_{\mathsf{sid}'}$ for SID sid' to this instance $\mathcal{F}_i$.

Fig. 7. The Global Restricted Observable Random Oracle Model $\mathcal{G}_{\mathsf{roRO}}$

2.4 Building Blocks

In this work, we use the followings as the main building blocks: the Pedersen commitment [39], the ElGamal encryption [20], the Sigma-protocols [15], and the (straight-line extractable) NIZK/NIWH arguments in the RO model [38].

We also use the well-known CDH and DDH assumption [17]. Due to the space limit, here we do not provide the formal descriptions of the building blocks above, and we refer interesting readers to see them in the full version of this paper [41].

3 UC-Secure Endemic OT via Random Oracles

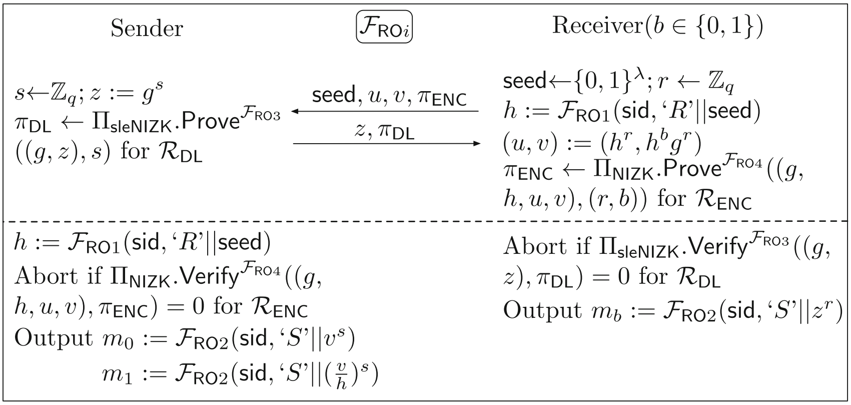
In this section, we provide a new 1-round UC-secure EOT protocol under standard assumptions in the RO model.

We start with the two-round standalone EOT protocol in [13]: in the first round, the sender sends $h := g^s$ using $s \leftarrow \mathbb{Z}_q$; in the second round, the receiver uses sender's message to compute $B := g^r h^b$ based on its choice bit b and its secret randomness $r \leftarrow \mathbb{Z}_q$; finally, the sender computes and outputs $m_0 := \mathcal{F}_{\mathsf{RO}}(B^s)$ and $m_1 := \mathcal{F}_{\mathsf{RO}}((\frac{B}{h})^s)$ while the receiver outputs $m_b := \mathcal{F}_{\mathsf{RO}}(h^r)$. Here we use to notation $y := \mathcal{F}_{\mathsf{RO}}(x)$ to describe the process for querying x to the random oracle $\mathcal{F}_{\mathsf{RO}}$ and obtaining the output y, which aligns with the notation in [11]. Our goals are: (i) reduce the round complexity of this protocol to one simultaneous round; (ii) add new mechanisms to make this protocol UC-secure.

In order to reduce the round complexity, we let the receiver generate h by invoking the RO on a randomly sampled string seed. In this way, the receiver can compute its message without the sender's message, thus only one simultaneous round is needed. This technique can be found in [11]. We then discuss how to provide UC security. The UC-secure EOT protocol requires *extractability*: (i) when the sender is malicious, the simulator should be able to extract the sender's secret randomness, so the simulator can compute both m_0 and m_1; (ii) when the receiver is malicious, the simulator should be able to extract the receiver's choice bit b. In order to extract the sender's secret randomness s, we let the sender additionally generate a straight-line extractable NIZK argument [22,32,38] of s such that $z = g^s$. The straight-line extractability relies on the observability of the RO model. In this way, the simulator can extract the malicious sender's secret randomness. In order to extract the receiver's choice bit b, we let the receiver generate an ElGamal encryption of bit b instead of a Pedersen commitment to bit b, i.e., the receiver computes $(u, v) := (h^r, h^b g^r)$ using $r \leftarrow \mathbb{Z}_q$. We also let the receiver generate a NIZK argument of (b, r) such that $(u, v) = (h^r, h^b g^r)$ to ensure that (u, v) is an ElGamal encryption of a bit b. In this way, the simulator knows $\log_g h$ by making use of the programmability of the RO model, and thus is able to extract b from (u, v).

Let g be the generator of $\mathbb{G}$. Let $\mathcal{F}_{\mathsf{RO1}} : \{0,1\}^* \rightarrow \mathbb{G}$ and $\mathcal{F}_{\mathsf{RO2}} : \{0,1\}^* \rightarrow \{0,1\}^\lambda$ be random oracles. Let $\mathcal{R}_{\mathsf{ENC}} := \{((g, h, u, v), (r, b)) \mid (b = 0 \wedge (u, v) = (h^r, g^r)) \vee (b = 1 \wedge (u, v) = (h^r, g^r h))\}$ and $\mathcal{R}_{\mathsf{DL}} := \{((g, z), s) \mid z = g^s\}$. We denote by Π_{sleNIZK} the straight-line extractable NIZK argument in the $\mathcal{F}_{\mathsf{RO3}}$-hybrid world. We denote by Π_{NIZK} the NIZK argument in the $\mathcal{F}_{\mathsf{RO4}}$-hybrid world. We note that, the domain and range of $\mathcal{F}_{\mathsf{RO3}}$ and $\mathcal{F}_{\mathsf{RO4}}$ depend on the concrete instantiations of the protocols, for that reason, we do not write them explicitly. Here we assume the synchronous channel $\mathcal{F}_{\mathsf{Syn}}$ is available to the protocol players.

Protocol Description. We present our protocol $\Pi_{\mathsf{EOT\text{-}RO}}$ in Fig. 8; note that, in Fig. 8, we only cover the case where both sender and receiver are honest.



Sender | $\boxed{\mathcal{F}_{\mathsf{RO}i}}$ | Receiver$(b \in \{0,1\})$

$s \leftarrow \mathbb{Z}_q; z := g^s$
$\pi_{\mathsf{DL}} \leftarrow \Pi_{\mathsf{sleNIZK}}.\mathsf{Prove}^{\mathcal{F}_{\mathsf{RO3}}}$
$((g, z), s)$ for $\mathcal{R}_{\mathsf{DL}}$

$\xleftarrow{\mathsf{seed}, u, v, \pi_{\mathsf{ENC}}}$

$\xrightarrow{z, \pi_{\mathsf{DL}}}$

$\mathsf{seed} \leftarrow \{0,1\}^\lambda; r \leftarrow \mathbb{Z}_q$
$h := \mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}, \text{‘}R\text{’}||\mathsf{seed})$
$(u, v) := (h^r, h^b g^r)$
$\pi_{\mathsf{ENC}} \leftarrow \Pi_{\mathsf{NIZK}}.\mathsf{Prove}^{\mathcal{F}_{\mathsf{RO4}}}((g,$
$h, u, v), (r, b))$ for $\mathcal{R}_{\mathsf{ENC}}$

$h := \mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}, \text{‘}R\text{’}||\mathsf{seed})$
Abort if $\Pi_{\mathsf{NIZK}}.\mathsf{Verify}^{\mathcal{F}_{\mathsf{RO4}}}((g,$
$h, u, v), \pi_{\mathsf{ENC}}) = 0$ for $\mathcal{R}_{\mathsf{ENC}}$
Output $m_0 := \mathcal{F}_{\mathsf{RO2}}(\mathsf{sid}, \text{‘}S\text{’}||v^s)$
$m_1 := \mathcal{F}_{\mathsf{RO2}}(\mathsf{sid}, \text{‘}S\text{’}||(\frac{v}{h})^s)$

Abort if $\Pi_{\mathsf{sleNIZK}}.\mathsf{Verify}^{\mathcal{F}_{\mathsf{RO3}}}((g,$
$z), \pi_{\mathsf{DL}}) = 0$ for $\mathcal{R}_{\mathsf{DL}}$
Output $m_b := \mathcal{F}_{\mathsf{RO2}}(\mathsf{sid}, \text{‘}S\text{’}||z^r)$

Fig. 8. 1-round EOT protocol $\Pi_{\mathsf{EOT\text{-}RO}}$ in the $\{\mathcal{F}_{\mathsf{RO}}, \mathcal{F}_{\mathsf{Syn}}\}$-hybrid world, where $\mathcal{F}_{\mathsf{RO}} = \{\mathcal{F}_{\mathsf{RO}i}\}_{i\in[4]}$. Let g be the generator of $\mathbb{G}$. Let $\mathcal{F}_{\mathsf{RO1}} : \{0,1\}^* \to \mathbb{G}$ and $\mathcal{F}_{\mathsf{RO2}} : \{0,1\}^* \to \{0,1\}^\lambda$. Let $\mathcal{R}_{\mathsf{ENC}} := \{((g, h, u, v), (r, b)) \mid (b = 0 \wedge (u, v) = (h^r, g^r)) \vee (b = 1 \wedge (u, v) = (h^r, g^r h))\}$ and $\mathcal{R}_{\mathsf{DL}} := \{((g, z), s) \mid z = g^s\}$.

When sender (resp. receiver) is statically corrupted and receiver (resp. sender) is honest, after sending its message to $\mathcal{F}_{\mathsf{Syn}}$ and waiting for a long time, the honest receiver (resp. sender) will query $\mathcal{F}_{\mathsf{Syn}}$ to obtain the other party's message. If $\mathcal{F}_{\mathsf{Syn}}$ replies the desired message, the honest party will compute and output the local message according to Fig. 8; otherwise, the honest party simply aborts. The security of the protocol has been stated in Theorem 1.

Theorem 1. *Assume the DDH assumption holds in group* $\mathbb{G}$. *Let* $\mathcal{F}_{\mathsf{RO1}} : \{0,1\}^* \to \mathbb{G}$ *and* $\mathcal{F}_{\mathsf{RO2}} : \{0,1\}^* \to \{0,1\}^\lambda$ *be the random oracles. Let* Π_{NIZK} *be an NIZK argument in the* $\mathcal{F}_{\mathsf{RO3}}$*-hybrid world. Let* Π_{sleNIZK} *be a straight-line extractable NIZK argument in the* $\mathcal{F}_{\mathsf{RO4}}$*-hybrid world. The protocol* $\Pi_{\mathsf{EOT\text{-}RO}}$ *depicted in Fig. 8 UC-realizes the functionality* $\mathcal{F}_{\mathsf{EOT}}$ *depicted in Fig. 5 in the* $\{\mathcal{F}_{\mathsf{RO}}, \mathcal{F}_{\mathsf{Syn}}\}$*-hybrid world against static malicious corruption, where* $\mathcal{F}_{\mathsf{RO}} = \{\mathcal{F}_{\mathsf{RO}i}\}_{i\in[4]}$.

Proof. We leave the formal proof in the full version of this paper [41].

Instantiation. We instantiate Π_{sleNIZK} for relation $\mathcal{R}_{\mathsf{DL}}$ with the Schnorr's protocol [40] and the *randomized Fischlin transform* [32] which improves the efficiency and applicability of Fischlin transform [22]. We instantiate Π_{NIZK} for relation $\mathcal{R}_{\mathsf{ENC}}$ with the following techniques: we first employ the OR-composition [14] to the Chaum-Pedersen protocols [12] to prove either (g, h, v, u) is a DDH tuple (which means $b = 0$) or $(g, h, \frac{v}{h}, u)$ is a DDH tuple (which means $b = 1$), we then apply the the Fiat-Shamir transform [21] to remove the interaction.

Efficiency. Here we compare the efficiency in the amortized setting where the sender and the receiver can reuse some elements for multiple instances of the EOT protocol (in this protocol, the sender can reuse s, π_{DL} while the receiver can reuse the string seed). The amortized setting is also used in [11] for efficiency comparison. By taking the parameters (that achieves 128-bit security) from [32], our protocol requires 18 exponentiations w.r.t. computation and 10 group/field elements w.r.t. communication; while the state-of-the-art 1-round UC-secure RO-based protocol in [34] requires 4 exponentiations w.r.t. computation and 2 group elements w.r.t. communication. Note that, our protocol is based on a standard assumption; whereas the protocol in [34] is based on a non-standard assumption.

4 The Relations Between Endemic OT and Other Primitives

In this section, we first show how to construct a bit commitment protocol via EOT with unconditional security. Subsequently, we complete the picture of OT relations in [34], showing that UOT can be constructed via EOT with unconditional security.

4.1 From Endemic OT to Commitment

Recall that, Brzuska *et al.* proved that bit commitment can be constructed via 1-out-of-2 OT with unconditional security [3]. As remarked in [34], there is a separation between the EOT and OT in the standalone setting: there are no 1-round OT protocols while there are 1-round EOT protocols. Although there is such a separation, we show a surprising fact: bit commitment can also be constructed via a weaker primitive, i.e., EOT, with unconditional security.

We observe that the receiver's message can be viewed as the commitment to the receiver's choice bit b, and the locally computed message m_b together with b can be viewed as the opening. Typically, a commitment protocol requires two properties: hiding and binding. The hiding property comes from the fact: even if the malicious EOT receiver can influence the distribution of m_b, it cannot learn the other message m_{1-b}. The binding property comes from the fact: even if the malicious EOT sender can influence the distributions of both m_0 and m_1, it cannot tell which one is received by the receiver. Furthermore, if we use a UC-secure EOT protocol as the building block, the resulting commitment protocol is also UC-secure. Note that, we only assume authenticated channel $\mathcal{F}_{\mathsf{Auth}}$ is available to the protocol players, and we omit the formal description of the well-known commitment functionality $\mathcal{F}_{\mathsf{Com}}$.

Protocol Description. We present our protocol Π_{Com} in Fig. 9; note that, in Fig. 9 we only cover the case that both committer and receiver are honest. The remaining cases can be found in the full version of this paper [41]. The security of the protocol has been stated in Theorem 2.

Theorem 2. *The protocol* Π_{Com} *depicted in Fig. 9 UC-realizes the functionality* $\mathcal{F}_{\mathsf{Com}}$ *with unconditional security in the* $\{\mathcal{F}_{\mathsf{EOT}}, \mathcal{F}_{\mathsf{Auth}}\}$*-hybrid world against static malicious corruption.*

Proof. We leave the formal proof in the full version of this paper [41].

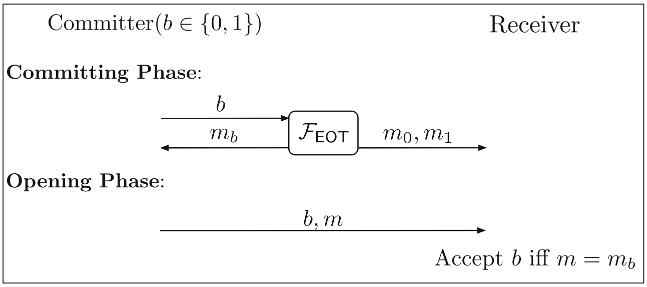


Fig. 9. Bit Commitment Protocol Π_{Com} in the $\{\mathcal{F}_{\mathsf{EOT}}, \mathcal{F}_{\mathsf{Auth}}\}$-Hybrid World

4.2 From Endemic OT to Uniform OT

In [34], the Masny and Rindal showed how to construct UOT with unconditional security in the $\{\mathcal{F}_{\mathsf{EOT}}, \mathcal{F}_{\mathsf{Coin}}, \mathcal{F}_{\mathsf{Auth}}\}$-hybrid world, where $\mathcal{F}_{\mathsf{Coin}}$ is the well-known coin-tossing functionality and we omit the formal description here. We recall the protocol construction in [34] in Fig. 10. However, they only showed how to construct the coin-tossing protocol via UOT. Therefore, whether EOT implies UOT remains an open question.

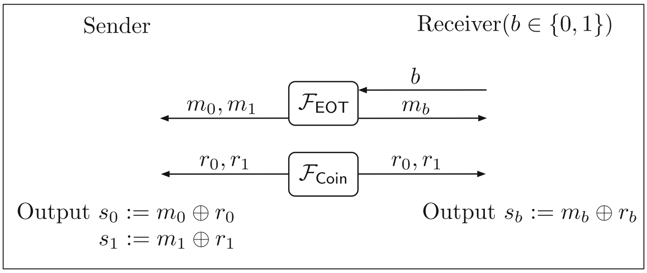

Sender
Receiver($b \in \{0,1\}$)
b
m_0, m_1
$\mathcal{F}_{\mathsf{EOT}}$
m_b
r_0, r_1
$\mathcal{F}_{\mathsf{Coin}}$
r_0, r_1
Output $s_0 := m_0 \oplus r_0$
$s_1 := m_1 \oplus r_1$
Output $s_b := m_b \oplus r_b$

Fig. 10. UOT Protocol Π_{UOT} in the $\{\mathcal{F}_{\mathsf{EOT}}, \mathcal{F}_{\mathsf{Coin}}, \mathcal{F}_{\mathsf{Auth}}\}$-Hybrid World from [34]

Lemma 1 ([34]). *The protocol Π_{UOT} depicted in Fig. 10 UC-realizes $\mathcal{F}_{\mathsf{UOT}}$ depicted in Fig. 4 with unconditional security in the $\{\mathcal{F}_{\mathsf{EOT}}, \mathcal{F}_{\mathsf{Coin}}, \mathcal{F}_{\mathsf{Auth}}\}$-hybrid world against static malicious corruption.*

In this section, we provide a positive answer to this unsolved question. Our solution is as follows: we have already showed that EOT implies commitment in Sect. 4.1, and the coin-tossing protocol can be easily constructed via only commitment; putting things together, we show that EOT implies UOT. Note that, we only assume $\mathcal{F}_{\mathsf{Auth}}$ is available to the protocol players, and we omit the formal description of the well-known coin-tossing functionality $\mathcal{F}_{\mathsf{Coin}}$.

Protocol Description. We present our protocol Π_{Coin} in Fig. 11; note that, in Fig. 11 we only cover the case that both two players are honest. The remaining cases can be found in the full version of this paper [41]. The security of the protocol has been stated in Theorem 3.

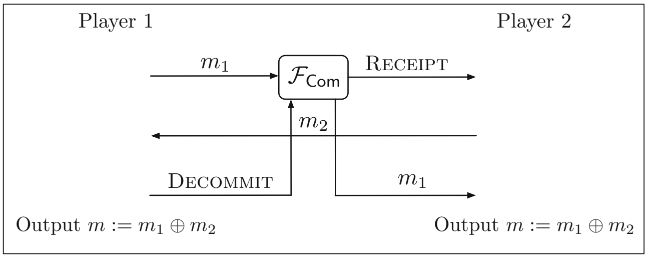

Player 1
Player 2
m_1
$\mathcal{F}_{\mathsf{Com}}$
RECEIPT
m_2
DECOMMIT
m_1
Output $m := m_1 \oplus m_2$
Output $m := m_1 \oplus m_2$

Fig. 11. Coin-Tossing Protocol Π_{Coin} in the $\{\mathcal{F}_{\mathsf{Com}}, \mathcal{F}_{\mathsf{Auth}}\}$-Hybrid World

Theorem 3. *The protocol Π_{Coin} depicted in Fig. 11 UC-realizes the functionality $\mathcal{F}_{\mathsf{Coin}}$ with unconditional security in the $\{\mathcal{F}_{\mathsf{Com}}, \mathcal{F}_{\mathsf{Auth}}\}$-hybrid world against static malicious corruption.*

Proof. We leave the formal proof in the full version of this paper [41].

Formally, we prove that EOT implies UOT through Corollary 1. The security proof of Corollary 1 directly comes from Lemma 1, Theorem 2 and Theorem 3, and thus we omit the trivial proof here.

Corollary 1. *The protocol Π_{UOT} depicted in Fig. 10 UC-realizes $\mathcal{F}_{\mathsf{UOT}}$ depicted in Fig. 4 with unconditional security in the $\{\mathcal{F}_{\mathsf{EOT}}, \mathcal{F}_{\mathsf{Auth}}\}$-hybrid world against static malicious corruption.*

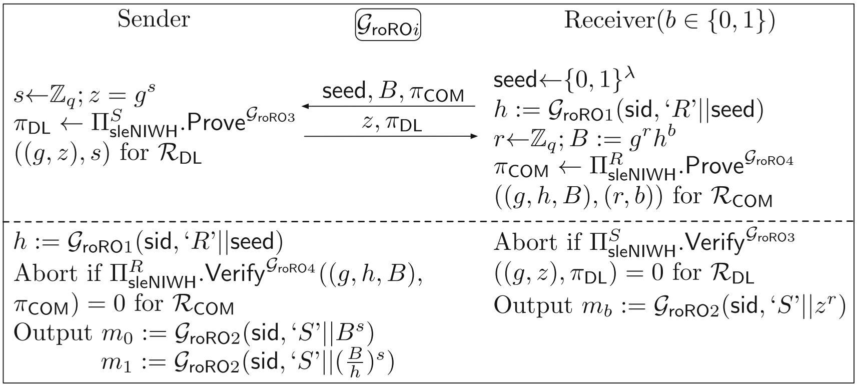
5 GUC-Secure Endemic OT via Global Random Oracles

In this section, we turn to global RO models to seek a stronger variant of UC security, i.e., GUC security. As for the GroRO model, we construct the *first* 1-round GUC-secure EOT protocol under CDH assumption against static adversaries. Basing on that, we propose the *first* 2-round GUC-secure commitment protocol in the GroRO model.

Regarding the GrpRO model, we prove that there exists *no* 1-round GUC-secure EOT protocol in the GrpRO model even with static security. By combining this negative result in the GrpRO model and the positive result in the GroRO model, we reveal a separation between these two models. Furthermore, we construct the *first* 2-round (round-optimal) GUC-secure EOT protocol under DDH assumption in the GrpRO model against adaptive adversaries.

5.1 Feasibility Results in the GroRO Model

Our EOT Protocol. We start with our UC-secure EOT protocol $\mathit{\Pi}_{\mathsf{EOT\text{-}RO}}$ depicted in Fig. 8. Recall that, we let the sender send $z := g^s$ using $s \leftarrow \mathbb{Z}_q$, together with a straight-line extractable NIZK argument of s such that $z = g^s$ in $\mathit{\Pi}_{\mathsf{EOT\text{-}RO}}$. The straight-line extractable NIZK argument gives the simulator chance of extracting the sender's secret randomness. However, Pass showed that it is impossible to construct NIZK arguments in observable RO model [38], let alone NIZK arguments with straight-line extractability. The good news is that we find that straight-line extractable NIWH argument is sufficient for our purpose, and it is possible in the GroRO model [38]. Therefore, we let the sender generate a straight-line extractable NIWH argument of s such that $z = g^s$. Now let us consider the receiver. In order to extract the receiver's choice bit, we make full use of the programmability of random oracles in $\mathit{\Pi}_{\mathsf{EOT\text{-}RO}}$. Since $\mathcal{G}_{\mathsf{roRO}}$ does not permit anyone to program the random oracle, we need to take a different strategy: we let the receiver generate h by invoking the $\mathcal{G}_{\mathsf{roRO}}$ on a randomly sampled string seed, compute a Pedersen commitment to the choice bit $B := g^r h^b$ using $r \leftarrow \mathbb{Z}_q$, and generate a straight-line extractable NIWH argument of (r, b) such that $B = g^r h^b$. Analogously to the sender side, the simulator can extract the malicious receiver's choice bit b.



Sender

$\boxed{\mathcal{G}_{\mathsf{roRO}i}}$

Receiver$(b \in \{0,1\})$

$s \leftarrow \mathbb{Z}_q; z = g^s$
$\pi_{\mathsf{DL}} \leftarrow \Pi^S_{\mathsf{sleNIWH}}.\mathsf{Prove}^{\mathcal{G}_{\mathsf{roRO3}}}$
$((g, z), s)$ for $\mathcal{R}_{\mathsf{DL}}$

$\xleftarrow{\mathsf{seed}, B, \pi_{\mathsf{COM}}}$

$\xrightarrow{z, \pi_{\mathsf{DL}}}$

$\mathsf{seed} \leftarrow \{0,1\}^\lambda$
$h := \mathcal{G}_{\mathsf{roRO1}}(\mathsf{sid}, \text{‘}R\text{’}||\mathsf{seed})$
$r \leftarrow \mathbb{Z}_q; B := g^r h^b$
$\pi_{\mathsf{COM}} \leftarrow \Pi^R_{\mathsf{sleNIWH}}.\mathsf{Prove}^{\mathcal{G}_{\mathsf{roRO4}}}$
$((g, h, B), (r, b))$ for $\mathcal{R}_{\mathsf{COM}}$

$h := \mathcal{G}_{\mathsf{roRO1}}(\mathsf{sid}, \text{‘}R\text{’}||\mathsf{seed})$
Abort if $\Pi^R_{\mathsf{sleNIWH}}.\mathsf{Verify}^{\mathcal{G}_{\mathsf{roRO4}}}((g, h, B),$
$\pi_{\mathsf{COM}}) = 0$ for $\mathcal{R}_{\mathsf{COM}}$
Output $m_0 := \mathcal{G}_{\mathsf{roRO2}}(\mathsf{sid}, \text{‘}S\text{’}||B^s)$
$m_1 := \mathcal{G}_{\mathsf{roRO2}}(\mathsf{sid}, \text{‘}S\text{’}||(\frac{B}{h})^s)$

Abort if $\Pi^S_{\mathsf{sleNIWH}}.\mathsf{Verify}^{\mathcal{G}_{\mathsf{roRO3}}}$
$((g, z), \pi_{\mathsf{DL}}) = 0$ for $\mathcal{R}_{\mathsf{DL}}$
Output $m_b := \mathcal{G}_{\mathsf{roRO2}}(\mathsf{sid}, \text{‘}S\text{’}||z^r)$

Fig. 12. 1-round EOT protocol $\mathit{\Pi}_{\mathsf{EOT\text{-}GroRO}}$ in the $\{\mathcal{G}_{\mathsf{roRO}}, \mathcal{F}_{\mathsf{Syn}}\}$-hybrid world, where $\mathcal{G}_{\mathsf{roRO}} = \{\mathcal{G}_{\mathsf{roRO}i}\}_{i \in [4]}$. Let g be the generator of $\mathbb{G}$. Let $\mathcal{G}_{\mathsf{roRO1}} : \{0,1\}^* \to \mathbb{G}$ and $\mathcal{G}_{\mathsf{roRO2}} : \{0,1\}^* \to \{0,1\}^\lambda$. Let $\mathcal{R}_{\mathsf{Com}} := \{((g, h, B), (r, b)) \mid B = g^r h^b\}$ and $\mathcal{R}_{\mathsf{DL}} := \{((g, z), s) \mid z = g^s\}$.

Let g be the generator of $\mathbb{G}$. Let $\mathcal{G}_{\mathsf{roRO1}} : \{0,1\}^* \to \mathbb{G}$ and $\mathcal{G}_{\mathsf{roRO2}} : \{0,1\}^* \to \{0,1\}^\lambda$. Let $\mathcal{R}_{\mathsf{Com}} := \{((g, h, B), (r, b)) \mid B = g^r h^b\}$ and $\mathcal{R}_{\mathsf{DL}} := \{((g, z), s) \mid z = g^s\}$. We denote by $\mathit{\Pi}^S_{\mathsf{sleNIWH}}$ the straight-line extractable NIWH argument in the $\mathcal{G}_{\mathsf{roRO3}}$-hybrid world which is used for generating the proof by sender. We denote by $\mathit{\Pi}^R_{\mathsf{sleNIWH}}$ the straight-line extractable NIWH argument in the $\mathcal{G}_{\mathsf{roRO4}}$-hybrid world which is used for generating the proof by receiver. We assume synchronous channel $\mathcal{F}_{\mathsf{Syn}}$ is available to the protocol players..

Protocol Description. We present our protocol $\mathit{\Pi}_{\mathsf{EOT\text{-}GroRO}}$ in Fig. 12; note that, in Fig. 12 we only cover the case that both sender and receiver are honest. The remaining cases can be found in the full version of this paper [41]. The security of the protocol has been stated in Theorem 4.

Before giving the theorem, we have to give the transferable EOT functionality $\mathcal{F}_{\mathsf{tEOT}}$ in Fig. 13. The main difference with the traditional EOT functionality is that in $\mathcal{F}_{\mathsf{tEOT}}$, the simulator can request the list of illegitimate queries, which fits the $\mathcal{G}_{\mathsf{roRO}}$ model.

Theorem 4. *Assume the CDH assumption holds in group $\mathbb{G}$. Let $\mathcal{G}_{\mathsf{roRO}1} : \{0,1\}^{\lambda} \rightarrow \mathbb{G}$ and $\mathcal{G}_{\mathsf{roRO}2} : \mathbb{G} \rightarrow \{0,1\}^{\lambda}$ be the random oracles. Let $\mathit{\Pi}^{S}_{\mathsf{sleNIWH}}$ be a straight-line extractable NIWH argument in the $\mathcal{G}_{\mathsf{roRO}3}$-hybrid world. Let $\mathit{\Pi}^{R}_{\mathsf{sleNIWH}}$ be a straight-line extractable NIWH argument in the $\mathcal{G}_{\mathsf{roRO}4}$-hybrid world. The protocol $\mathit{\Pi}_{\mathsf{EOT\text{-}GroRO}}$ depicted in Fig. 12 GUC-realizes the functionality $\mathcal{F}_{\mathsf{tEOT}}$ depicted in Fig. 13 in the $\{\mathcal{G}_{\mathsf{roRO}}, \mathcal{F}_{\mathsf{Syn}}\}$-hybrid world against static malicious corruption, where $\mathcal{G}_{\mathsf{roRO}} = \{\mathcal{G}_{\mathsf{roRO}i}\}_{i\in[4]}$.*

Proof. We leave the formal proof in the full version of this paper [41].

Functionality $\mathcal{F}_{\mathsf{tEOT}}$

The functionality interacts with two parties S, R and an adversary $\mathcal{S}$.
Transfer/Choose/Process. Same as $\mathcal{F}_{\mathsf{EOT}}$ depicted in Figure 5.

Observe. When asked by the adversary $\mathcal{S}$, obtain from $\mathcal{G}_{\mathsf{roRO}}$ the list of illegitimate queries $\mathcal{Q}_{\mathsf{sid}}$ that pertain to SID sid, and send $\mathcal{Q}_{\mathsf{sid}}$ to the adversary $\mathcal{S}$.

Fig. 13. The Transferable Ideal Functionality $\mathcal{F}_{\mathsf{tEOT}}$ for Endemic Oblivious Transfer

Instantiation. We instantiate $\mathit{\Pi}^{S}_{\mathsf{sleNIZK}}$ for relation $\mathcal{R}_{\mathsf{DL}}$ with the Schnorr's protocol and the randomized Fischlin transform as in Sect. 3. Note that, although we use the same instantiation as in Sect. 3, we only obtain a straight-line extractable NIWH argument, since here we use a observable RO model [38]. We instantiate $\mathit{\Pi}^{R}_{\mathsf{sleNIWH}}$ for relation $\mathcal{R}_{\mathsf{Com}}$ with the Okamoto's protocol [37] and the randomized Fischlin transform.

Efficiency. We consider the efficiency of our GUC-secure protocol $\mathit{\Pi}_{\mathsf{EOT\text{-}GroRO}}$ in the amortized setting here, just like we did in Sect. 3. By taking the parameter (that achieves 128-bit security) in [32], our GUC-secure protocol $\mathit{\Pi}_{\mathsf{EOT\text{-}GroRO}}$ requires 53 exponentiations w.r.t. computation and 41 group/field elements w.r.t. communication; while the state-of-the-art 2-round GroRO-based OT protocol in [11] requires 5 exponentiations w.r.t. computation and 2 group elements +

2λ bits string w.r.t. communication. Note that, our protocol only requires CDH assumption, whereas the protocol proposed in [11] requires the DDH assumption, which is stronger.

Our Commitment Protocol. Recall that, we construct a commitment protocol $\mathit{\Pi}_{\mathsf{Com}}$ depicted in Fig. 9 in the $\{\mathcal{F}_{\mathsf{EOT}}, \mathcal{F}_{\mathsf{Auth}}\}$-hybrid world with unconditional security (cf. Sect. 4.1). It is easy to see that if we replace $\mathcal{F}_{\mathsf{EOT}}$ with $\mathcal{F}_{\mathsf{tEOT}}$ and call the resulting protocol $\mathit{\Pi}_{\mathsf{tCom}}$, then the protocol $\mathit{\Pi}_{\mathsf{tCom}}$ will GUC-realize $\mathcal{F}_{\mathsf{tCom}}$ in the $\{\mathcal{F}_{\mathsf{tEOT}}, \mathcal{F}_{\mathsf{Auth}}\}$-hybrid world with unconditional security, where $\mathcal{F}_{\mathsf{tCom}}$ is the transferable commitment functionality introduced in [9] and here we omit the formal description of $\mathcal{F}_{\mathsf{tCom}}$. Formally, we have the following corollary, and its security proof is analogously to the proof of Theorem 2.

Corollary 2. *The protocol $\mathit{\Pi}_{\mathsf{tCom}}$ GUC-realizes the functionality $\mathcal{F}_{\mathsf{tCom}}$ with unconditional security in the $\{\mathcal{F}_{\mathsf{tEOT}}, \mathcal{F}_{\mathsf{Auth}}\}$-hybrid world against static malicious corruption.*

Instantiation. We instantiate $\mathcal{F}_{\mathsf{tEOT}}$ with our 1-round GUC-secure EOT protocol depicted in Fig. 12. Then we immediate obtain a 2-round GUC-secure commitment protocol $\mathit{\Pi}_{\mathsf{tCom}}$ in the GroRO model; note that, the first round messages are communicated over the synchronous channel $\mathcal{F}_{\mathsf{Syn}}$ and the second round message is communicated over the authenticated channel $\mathcal{F}_{\mathsf{Auth}}$. The security is guaranteed by Theorem 4 and Corollary 2.

Comparison. Our commitment protocol is the *first* 2-round GUC-secure commitment in the GroRO model, while the previous state-of-the-art protocols achieve 3 rounds [36,42]. Note that, Zhou *et al.* proved that it is impossible to construct 2-round GUC-secure commitment protocol in the GroRO model even with static security [42]; but they do not assume $\mathcal{F}_{\mathsf{Syn}}$ is available for protocol players. Our 2-round commitment protocol contains a simultaneous round, so we do not contradict their impossibility result. We also note that, our protocol and protocols in [36,42] are all 3-move static-secure protocols, but ours is the only one whose first two moves can be executed in one simultaneous round; hence, ours is the only one that can achieve 2-round.

5.2 Impossibility and Feasibility Results in the GrpRO Model

Our Impossibility Result. Here we show that there exists *no* 1-round GUC-secure EOT protocol against static adversaries in the GrpRO model.

We prove this impossibility by contradiction. Suppose that there exists such a 1-round GUC-secure EOT protocol. Let us first consider the case where the receiver is corrupted, and the simulator needs to extract the choice bit of the receiver from its message. Recall that, the $\mathcal{G}_{\mathsf{rpRO}}$ only grants the simulator the restricted programmability: although the simulator can program the unqueried points without being detected, the simulator is external to the $\mathcal{G}_{\mathsf{rpRO}}$ and it can not know in real time what queries other parties are sending to $\mathcal{G}_{\mathsf{rpRO}}$. Thus, the simulator needs to program the points in advance and find a way to enforce the corrupt receiver to generate its message on the simulator's programmed points. In that way, the simulator can have the chance of extracting the choice bit of the receiver. However, in a one simultaneous round protocol, the messages between parties have no dependency. Hence the simulator cannot enforce the corrupt receiver to produce its message on the programmed points, and has no advantages over the real world adversary. If the simulator still succeeds to extract the corrupted receiver's choice bit, then distinctions will be revealed when the adversary performs the following attacks. The adversary corrupts the sender, and instructs the sender to run the simulator algorithm above to extract the choice bit from the message sent by the receiver/simulator. However, the receiver/simulator has no idea about the real choice bit, thus with high probability the simulation would fail. Formally, we prove this impossibility through Theorem 5.

Theorem 5. *There exists no terminating 1-round protocol Π that GUC-realizes $\mathcal{F}_{\mathsf{EOT}}$ depicted in Fig. 5 with static security in the $\{\mathcal{G}_{\mathsf{rpRO}}, \mathcal{F}_{\mathsf{Syn}}\}$-hybrid world.*

Proof. We leave the formal proof in the full version of this paper [41].

By combining this negative result in the GrpRO model and the positive result in the GroRO model depicted in Sect. 5.1, we demonstrate a separation between the GroRO and the GrpRO model.

Our EOT Protocol. Theorem 5 rules out the possibility of 1-round GUC-secure EOT protocols in the $\{\mathcal{G}_{\mathsf{rpRO}}, \mathcal{F}_{\mathsf{Syn}}\}$-hybrid world. It makes us wonder if we do not let the sender and the receiver send their messages simultaneously but in a specific order, can we construct a 2-move (also 2-round) GUC-secure protocol?

We start with the UC-secure EOT protocol in the CRS+GrpRO hybrid model proposed by Canetti *et al.* [11]. Their CRS consists of two group elements $g, h \in \mathbb{G}$, and the simulator knows $\log_g h$. They let the receiver generate parameter G, H by invoking the RO on a randomly sampled string seed, and compute two instances of Pedersen commitment to the choice bit $(B_1, B_2) := (g^x G^b, h^x H^b)$ using two sets of different parameters $(g, G), (h, H)$ and the same randomness $x \leftarrow \mathbb{Z}_q$. As for the sender, they let the sender compute $z := g^r h^s$ using randomness $r, s \leftarrow \mathbb{Z}_q$. Finally, the sender outputs $m_0 := \mathcal{G}_{\mathsf{rpRO}}(B_1^r B_2^s)$ and $m_1 := \mathcal{G}_{\mathsf{rpRO}}((\frac{B_1}{G})^r (\frac{B_2}{H})^s)$ while the receiver outputs $m_b := \mathcal{G}_{\mathsf{rpRO}}(z^x)$.

Our goals are: (i) remove the CRS setup of this protocol; (ii) make this protocol GUC-secure in the GrpRO model. To achieve the former goal, we let the sender generate g, h by invoking random oracle on a randomly sampled string seed_1. Then the sender computes $z := g^r h^s$ using $r, s \leftarrow \mathbb{Z}_q$, and sends seed_1, z to the receiver. On the other hand, we let the receiver generate G, H by invoking $\mathcal{G}_{\mathsf{rpRO}}$ on another randomly sampled string seed_2, computes two instances of Pedersen commitment to the choice bit $(B_1, B_2) := (g^x G^b, h^x H^b)$ using the same randomness $x \leftarrow \mathbb{Z}_q$. The local computation is the same as Canetti *et al*'s protocol. In order to show that our modified protocol achieves the latter goal, we show the simulation strategy as follows: when the receiver is malicious, the simulator can extract the receiver's choice bit b by programming the $\mathcal{G}_{\mathsf{rpRO}}$ and knowing α such that $h = g^\alpha$. Then the simulator can extract b by the following strategy: if $B_2 = B_1^\alpha$, it sets $b := 0$; else if $\frac{B_2}{H} = (\frac{B_1}{G})^\alpha$, it sets $b := 1$; else, it sets $b := \perp$. Note that, when B_1, B_2 are not correctly constructed (i.e., the simulator sets $b := \perp$), the malicious receiver cannot compute either m_0 or m_1. When the sender is malicious, the simulator can compute both m_0 and m_1 by programming the $\mathcal{G}_{\mathsf{rpRO}}$ such that (g, h, G, H) is a DDH tuple, i.e., $G = g^t, H = h^t$. In this way, the simulator can compute $m_0 := \mathcal{G}_{\mathsf{rpRO}}(z^x)$ and $m_1 := \mathcal{G}_{\mathsf{rpRO}}(z^{x-t})$.

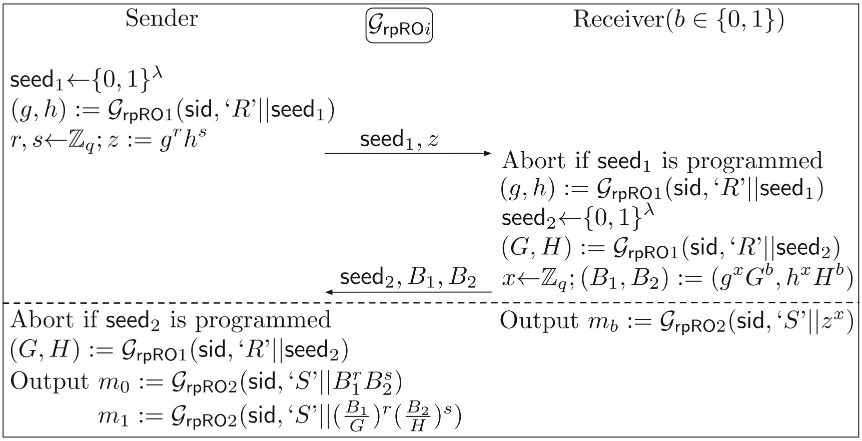

Sender
$\boxed{\mathcal{G}_{\mathsf{rpRO}i}}$
Receiver$(b \in \{0,1\})$
$\mathsf{seed}_1 \leftarrow \{0,1\}^\lambda$
$(g, h) := \mathcal{G}_{\mathsf{rpRO1}}(\mathsf{sid}, `R\text{'}||\mathsf{seed}_1)$
$r, s \leftarrow \mathbb{Z}_q; z := g^r h^s$
seed_1, z
Abort if seed_1 is programmed
$(g, h) := \mathcal{G}_{\mathsf{rpRO1}}(\mathsf{sid}, `R\text{'}||\mathsf{seed}_1)$
$\mathsf{seed}_2 \leftarrow \{0,1\}^\lambda$
$(G, H) := \mathcal{G}_{\mathsf{rpRO1}}(\mathsf{sid}, `R\text{'}||\mathsf{seed}_2)$
$\mathsf{seed}_2, B_1, B_2$
$x \leftarrow \mathbb{Z}_q; (B_1, B_2) := (g^x G^b, h^x H^b)$
Abort if seed_2 is programmed
$(G, H) := \mathcal{G}_{\mathsf{rpRO1}}(\mathsf{sid}, `R\text{'}||\mathsf{seed}_2)$
Output $m_0 := \mathcal{G}_{\mathsf{rpRO2}}(\mathsf{sid}, `S\text{'}||B_1^r B_2^s)$
$m_1 := \mathcal{G}_{\mathsf{rpRO2}}(\mathsf{sid}, `S\text{'}||(\frac{B_1}{G})^r(\frac{B_2}{H})^s)$
Output $m_b := \mathcal{G}_{\mathsf{rpRO2}}(\mathsf{sid}, `S\text{'}||z^x)$

Fig. 14. 2-round EOT protocol $\Pi_{\mathsf{EOT\text{-}GrpRO}}$ in the $\{\mathcal{G}_{\mathsf{rpRO}}, \mathcal{F}_{\mathsf{Auth}}\}$-hybrid world, where $\mathcal{G}_{\mathsf{rpRO}} = \{\mathcal{G}_{\mathsf{rpRO1}}, \mathcal{G}_{\mathsf{rpRO2}}\}$. Let $\mathcal{G}_{\mathsf{rpRO1}} : \{0,1\}^* \to \mathbb{G} \times \mathbb{G}$ and $\mathcal{G}_{\mathsf{rpRO2}} : \{0,1\}^* \to \{0,1\}^\lambda$.

Let $\mathcal{G}_{\mathsf{rpRO1}} : \{0,1\}^* \to \mathbb{G} \times \mathbb{G}$ and $\mathcal{G}_{\mathsf{rpRO2}} : \{0,1\}^* \to \{0,1\}^\lambda$. Here we assume authenticated channels $\mathcal{F}_{\mathsf{Auth}}$ are available.

Protocol Description. We present our protocol $\Pi_{\mathsf{EOT\text{-}GrpRO}}$ in Fig. 14; note that, in Fig. 14 we only cover the case that both sender and receiver are honest. The remaining cases can be found in the full version of this paper [41]. The security of the protocol has been stated in Theorem 6.

Theorem 6. *Assume the DDH assumption holds in group $\mathbb{G}$. Let $\mathcal{G}_{\mathsf{rpRO1}} : \{0,1\}^\lambda \to \mathbb{G} \times \mathbb{G}$ and $\mathcal{G}_{\mathsf{rpRO2}} : \mathbb{G} \to \{0,1\}^\lambda$ be the random oracles. The protocol $\Pi_{\mathsf{EOT\text{-}GrpRO}}$ depicted in Fig. 14 GUC-realizes the functionality $\mathcal{F}_{\mathsf{EOT}}$ depicted in Fig. 5 in the $\{\mathcal{G}_{\mathsf{rpRO}}, \mathcal{F}_{\mathsf{Auth}}\}$-hybrid world against adaptive malicious corruption, where $\mathcal{G}_{\mathsf{rpRO}} = \{\mathcal{G}_{\mathsf{rpRO1}}, \mathcal{G}_{\mathsf{rpRO2}}\}$.*

Proof. We leave the formal proof in the full version of this paper [41].

Efficiency. We consider the efficiency of our protocol $\Pi_{\mathsf{EOT\text{-}GrpRO}}$ in the amortized setting here, just like we did in Sect. 3. Our protocol requires 5 exponentiations w.r.t. computation and 2 group elements w.r.t. communication; while the state-of-the-art 2-round GrpRO-based OT protocol in [11] requires the same computation and extra 2λ bits string w.r.t. communication compared to our protocol. We emphasize that our protocol achieves GUC security; whereas the protocol proposed in [11] achieves only UC security.

Acknowledgment. We thank anonymous reviewers of Eurocrypt 2023 for their helpful and constructive comments.

References

1. Bellare, M., Micali, S.: Non-interactive oblivious transfer and applications. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 547–557. Springer, New York (1990). https://doi.org/10.1007/0-387-34805-0_48
2. Bellare, M., Rogaway, P.: Random oracles are practical: a paradigm for designing efficient protocols. In: Denning, D.E., Pyle, R., Ganesan, R., Sandhu, R.S., Ashby, V. (eds.) ACM CCS 93, pp. 62–73. ACM Press, November 1993. https://doi.org/10.1145/168588.168596
3. Brzuska, C., Fischlin, M., Schröder, H., Katzenbeisser, S.: Physically uncloneable functions in the universal composition framework. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 51–70. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22792-9_4
4. Byali, M., Patra, A., Ravi, D., Sarkar, P.: Fast and universally-composable oblivious transfer and commitment scheme with adaptive security. Cryptology ePrint Archive, Report 2017/1165 (2017). https://eprint.iacr.org/2017/1165
5. Camenisch, J., Drijvers, M., Gagliardoni, T., Lehmann, A., Neven, G.: The wonderful world of global random oracles. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018. LNCS, vol. 10820, pp. 280–312. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-78381-9_11
6. Canetti, R.: Universally composable security: a new paradigm for cryptographic protocols. In: 42nd FOCS, pp. 136–145. IEEE Computer Society Press, October 2001. https://doi.org/10.1109/SFCS.2001.959888
7. Canetti, R., Dodis, Y., Pass, R., Walfish, S.: Universally composable security with global setup. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 61–85. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-70936-7_4
8. Canetti, R., Goldreich, O., Halevi, S.: The random oracle methodology, revisited (preliminary version). In: 30th ACM STOC, pp. 209–218. ACM Press, May 1998. https://doi.org/10.1145/276698.276741
9. Canetti, R., Jain, A., Scafuro, A.: Practical UC security with a global random oracle. In: Ahn, G.J., Yung, M., Li, N. (eds.) ACM CCS 2014, pp. 597–608. ACM Press, November 2014. https://doi.org/10.1145/2660267.2660374
10. Canetti, R., Lindell, Y., Ostrovsky, R., Sahai, A.: Universally composable two-party and multi-party secure computation. In: 34th ACM STOC, pp. 494–503. ACM Press, May 2002. https://doi.org/10.1145/509907.509980
11. Canetti, R., Sarkar, P., Wang, X.: Efficient and round-optimal oblivious transfer and commitment with adaptive security. In: Moriai, S., Wang, H. (eds.) ASIACRYPT 2020. LNCS, vol. 12493, pp. 277–308. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-64840-4_10
12. Chaum, D., Pedersen, T.P.: Wallet databases with observers. In: Brickell, E.F. (ed.) CRYPTO 1992. LNCS, vol. 740, pp. 89–105. Springer, Heidelberg (1993). https://doi.org/10.1007/3-540-48071-4_7

13. Chou, T., Orlandi, C.: The simplest protocol for oblivious transfer. In: Lauter, K., Rodríguez-Henríquez, F. (eds.) LATINCRYPT 2015. LNCS, vol. 9230, pp. 40–58. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-22174-8_3
14. Cramer, R., Damgård, I., Schoenmakers, B.: Proofs of partial knowledge and simplified design of witness hiding protocols. In: Desmedt, Y.G. (ed.) CRYPTO 1994. LNCS, vol. 839, pp. 174–187. Springer, Heidelberg (1994). https://doi.org/10.1007/3-540-48658-5_19
15. Damgård, I.: On Σ-protocols. Lecture Notes, University of Aarhus, Department for Computer Science, p. 84 (2002). https://www.cs.au.dk/ivan/Sigma.pdf
16. David, B., Dowsley, R.: Efficient composable oblivious transfer from CDH in the global random oracle model. In: Krenn, S., Shulman, H., Vaudenay, S. (eds.) CANS 2020. LNCS, vol. 12579, pp. 462–481. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-65411-5_23
17. Diffie, W., Hellman, M.E.: New directions in cryptography. IEEE Trans. Inf. Theory **22**(6), 644–654 (1976)
18. Dodis, Y., Shoup, V., Walfish, S.: Efficient constructions of composable commitments and zero-knowledge proofs. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 515–535. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-85174-5_29
19. Doerner, J., Kondi, Y., Lee, E., Shelat, A.: Secure two-party threshold ECDSA from ECDSA assumptions. In: 2018 IEEE Symposium on Security and Privacy, pp. 980–997. IEEE Computer Society Press, May 2018. https://doi.org/10.1109/SP.2018.00036
20. ElGamal, T.: A public key cryptosystem and a signature scheme based on discrete logarithms. In: Blakley, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 10–18. Springer, Heidelberg (1985). https://doi.org/10.1007/3-540-39568-7_2
21. Fiat, A., Shamir, A.: How to prove yourself: practical solutions to identification and signature problems. In: Odlyzko, A.M. (ed.) CRYPTO 1986. LNCS, vol. 263, pp. 186–194. Springer, Heidelberg (1987). https://doi.org/10.1007/3-540-47721-7_12
22. Fischlin, M.: Communication-efficient non-interactive proofs of knowledge with online extractors. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 152–168. Springer, Heidelberg (2005). https://doi.org/10.1007/11535218_10
23. Garg, S., Ishai, Y., Srinivasan, A.: Two-round MPC: information-theoretic and black-box. In: Beimel, A., Dziembowski, S. (eds.) TCC 2018. LNCS, vol. 11239, pp. 123–151. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-03807-6_5
24. Garg, S., Ishai, Y., Srinivasan, A.: Two-round MPC: information-theoretic and black-box. Cryptology ePrint Archive, Report 2018/909 (2018). https://eprint.iacr.org/2018/909
25. Garg, S., Srinivasan, A.: Garbled protocols and two-round MPC from bilinear maps. In: Umans, C. (ed.) 58th FOCS, pp. 588–599. IEEE Computer Society Press, October 2017. https://doi.org/10.1109/FOCS.2017.60
26. Genç, Z.A., Iovino, V., Rial, A.: "The simplest protocol for oblivious transfer" revisited. Inf. Process. Lett. **161**, 105975 (2020). https://doi.org/10.1016/j.ipl.2020.105975
27. Goldreich, O., Micali, S., Wigderson, A.: How to play any mental game or a completeness theorem for protocols with honest majority. In: Aho, A. (ed.) 19th ACM STOC, pp. 218–229. ACM Press, May 1987. https://doi.org/10.1145/28395.28420
28. Groth, J., Sahai, A.: Efficient non-interactive proof systems for bilinear groups. In: Smart, N. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 415–432. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-78967-3_24

29. Hauck, E., Loss, J.: Efficient and universally composable protocols for oblivious transfer from the CDH assumption. Cryptology ePrint Archive, Report 2017/1011 (2017). https://eprint.iacr.org/2017/1011
30. Katz, J.: On achieving the "best of both worlds" in secure multiparty computation. In: Johnson, D.S., Feige, U. (eds.) 39th ACM STOC, pp. 11–20. ACM Press, Jun 2007. https://doi.org/10.1145/1250790.1250793
31. Katz, J., Maurer, U., Tackmann, B., Zikas, V.: Universally composable synchronous computation. In: Sahai, A. (ed.) TCC 2013. LNCS, vol. 7785, pp. 477–498. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-36594-2_27
32. Kondi, Y., Shelat, A.: Improved straight-line extraction in the random oracle model with applications to signature aggregation. In: Agrawal, S., Lin, D. (eds.) ASIACRYPT 2022, pp. 279–309. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-22966-4_10
33. Li, B., Micciancio, D.: Equational security proofs of oblivious transfer protocols. In: Abdalla, M., Dahab, R. (eds.) PKC 2018. LNCS, vol. 10769, pp. 527–553. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-76578-5_18
34. Masny, D., Rindal, P.: Endemic oblivious transfer. In: Cavallaro, L., Kinder, J., Wang, X., Katz, J. (eds.) ACM CCS 2019, pp. 309–326. ACM Press, November 2019. https://doi.org/10.1145/3319535.3354210
35. Masny, D., Rindal, P.: Endemic oblivious transfer. Cryptology ePrint Archive, Report 2019/706 (2019). https://eprint.iacr.org/2019/706
36. Mohassel, P., Rosulek, M., Scafuro, A.: Sublinear zero-knowledge arguments for ram programs. In: Coron, J.-S., Nielsen, J.B. (eds.) EUROCRYPT 2017. LNCS, vol. 10210, pp. 501–531. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-56620-7_18
37. Okamoto, T.: Provably secure and practical identification schemes and corresponding signature schemes. In: Brickell, E.F. (ed.) CRYPTO 1992. LNCS, vol. 740, pp. 31–53. Springer, Heidelberg (1993). https://doi.org/10.1007/3-540-48071-4_3
38. Pass, R.: On deniability in the common reference string and random oracle model. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 316–337. Springer, Heidelberg (2003). https://doi.org/10.1007/978-3-540-45146-4_19
39. Pedersen, T.P.: Non-interactive and information-theoretic secure verifiable secret sharing. In: Feigenbaum, J. (ed.) CRYPTO 1991. LNCS, vol. 576, pp. 129–140. Springer, Heidelberg (1992). https://doi.org/10.1007/3-540-46766-1_9
40. Schnorr, C.P.: Efficient identification and signatures for smart cards. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 239–252. Springer, New York (1990). https://doi.org/10.1007/0-387-34805-0_22
41. Zhou, Z., Zhang, B., Zhou, H.S., Ren, K.: Endemic oblivious transfer via random oracles, revisited. Cryptology ePrint Archive, Paper 2022/1525 (2022). https://eprint.iacr.org/2022/1525
42. Zhou, Z., Zhang, B., Zhou, H.S., Ren, K.: Guc-secure commitments via random oracles: new impossibility and feasibility. In: Agrawal, S., Lin, D. (eds.) ASIACRYPT 2022, pp. 129–158. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-22972-5_5