A Black-Box Construction of Fully-Simulatable, Round-Optimal Oblivious Transfer from Strongly Uniform Key Agreement

Daniele Friolo[1], Daniel Masny*[2], and Daniele Venturi[1]

[1] *Sapienza University of Rome, Italy*
[2] *VISA Research, USA*

Abstract

We show how to construct maliciously secure oblivious transfer (M-OT) from a strengthening of key agreement (KA) which we call *strongly uniform* KA (SU-KA), where the latter roughly means that the messages sent by one party are computationally close to uniform, even if the other party is malicious. Our transformation is black-box, almost round preserving (adding only a constant overhead of up to two rounds), and achieves standard simulation-based security in the plain model.

As we show, 2-round SU-KA can be realized from cryptographic assumptions such as low-noise LPN, high-noise LWE, Subset Sum, DDH, CDH and RSA—all with polynomial hardness—thus yielding a black-box construction of fully-simulatable, round-optimal, M-OT from the same set of assumptions (some of which were not known before).

By invoking a recent result of Benhamouda and Lin (EUROCRYPT 2017), we also obtain (non-black-box) 5-round maliciously secure MPC in the plain model, from the same assumptions.

*Part of the work was done at UC Berkeley, funded by the Center for Long-Term Cybersecurity (CLTC, UC Berkeley).

Contents

1 Introduction

Oblivious transfer (OT) is a very simple functionality between two parties: a sender with input two strings (s_0, s_1), and a receiver with input a choice bit b; the output for the receiver equals s_b, while the sender learns nothing (i.e., the receiver's choice bit remains hidden) [Rab81, EGL82]. The standard security definition for OT compares an execution of the protocol in the real world—where either the sender or the receiver might act maliciously—with an execution in the ideal world where a trusted third party simply implements the above functionality. Following previous work, we call *"fully simulatable"* an OT protocol that meets this notion.

Surprisingly, OT turned out to be sufficient for constructing secure multi-party computation (MPC) for *arbitrary* functionalities [Yao82, Yao86, Kil88, IPS08, IKO+11, BL18, GS18]. For this reason, constructing OT has been an important objective and received much attention. Nevertheless, previous constructions of fully-simulatable OT suffer from diverse shortcomings (cf. also §1.4): (i) They require *trusted setup*, or are based on *random oracles* (as, e.g., in [JS07, PVW08]); (ii) They have *high round complexity* (as, e.g., in [Hai08]), while the optimal number of rounds would be 4 [IKO+11, GMPP16]; (iii) They are *non-black-box*, in that they are obtained by generically transforming semi-honestly secure OT (SH-OT)—which in turn can be constructed from special types of PKE [GKM+00]—to fully-simulatable OT via (possibly interactive) zero-knowledge proofs (*á la* GMW [GMW91]); (iv) They are tailored to *specific hardness assumptions* (as, e.g., in [Lin08, BD18]).

One exception is the work of Ostrovsky, Richelson and Scafuro [ORS15], that provide a black-box construction of 4-round, fully-simulatable OT in the plain model from *certified trapdoor permutations* (TDPs) [BY92, LMRS04, CL18], which in turn can be instantiated from the RSA assumption under some parameter regimes [KKM12, CL18]. This draws our focus to the question:

Can we obtain 4-round, fully-simulatable OT in a black-box way from minimal assumptions, without assuming trusted setup or relying on random oracles?

1.1 Our Contribution

We give a positive answer to the above question by leveraging a certain type of key agreement (KA) protocols, which intuitively allow two parties to establish a secure channel in the presence of an eavesdropper. The influential work by Impagliazzo and Rudich [IR88] showed a (black-box) separation between secret-key cryptography and public-key cryptography and KA. Ever since, it is common sense that public-key encryption (PKE) requires stronger assumptions than the existence of one-way functions, and thus secure KA is the weakest assumption from which public-key cryptography can be obtained. More recent research efforts have only provided further confidence in this conviction [GMMM18].

In more details, our main contribution is a construction of fully-simulatable OT (a.k.a. *maliciously secure* OT, or M-OT) from a strengthening of KA protocols, which we term *strongly uniform* (SU); our protocol is fully *black-box* and essentially *round-preserving*, adding only a constant overhead of at most two rounds. In particular, we show:

Theorem 1. *For any odd $t \in \mathbb{N}$, with $t > 1$, there is a black-box construction of a $(t + 1)$-round, fully-simulatable oblivious transfer protocol in the plain model, from any t-round strongly uniform key agreement protocol and a perfectly binding commitment scheme.*[1]

[1] Statistically binding commitment schemes are implied by perfectly-correct KA protocols [LS19]. Both LWE and low-noise LPN implie statistically binding commitment schemes as well [GHKW17].

Since, as we show, 2-round and 3-round SU-KA can be instantiated from several assumptions, including low-noise (ring) LPN, high-noise (ring) LWE, Subset Sum, CDH, DDH, and RSA—all with polynomial hardness—a consequence of our result is that we obtain round-optimal M-OT in the plain model under the same set of assumptions (in a black-box way). In particular, this yields the *first* such protocols from LPN, LWE (with modulus noise ratio $\sqrt{n}$), CDH, and Subset Sum.[2] Note that our LWE parameter setting relates to an approximation factor of $n^{1.5}$ for SIVP in lattices of dimension n [Reg05], which is the weakest LWE assumption known to imply PKE.

In our construction, we use a special kind of *"commit-and-open"* protocols which were implicitly used in previous works [Kil92, ORS15]. As a conceptual contribution, we formalize their security properties, which allows for a more modular presentation and security analysis.

1.2 Technical Overview

We proceed to a high level overview of the techniques behind our main result, starting with the notion of strong uniformity and the abstraction of commit-and-open protocols, and landing with the intuition behind our construction of M-OT (cf. Fig. 1).

Strong uniformity. As an important stepping stone to our main result, in §3, we introduce the notion of strong uniformity. Recall that a KA protocol allows Alice and Bob to share a key over a public channel, in such a way that the shared key is indistinguishable from uniform to the eyes of a passive eavesdropper. Strong uniformity here demands that, even if Bob is malicious, the messages sent by Alice are computationally close to uniform over an efficiently sampleable group.[3] This flavor of security straightforwardly translates to SH-OT and PKE, yielding so-called SUSH-OT and SU-PKE. In the case of SUSH-OT, it demands that all messages of the receiver have this property (even if the sender is malicious). For SU-PKE, we distinguish two types, which are a strengthening of the types defined by Gertner *et al.* [GKM$^+$00].[4]

- **Type-A PKE:** The distribution of the public key is computationally indistinguishable from uniform. This type of PKE is known to exist under DDH [Gam84] and CDH [GM84] over efficiently sampleable groups,[5] LWE [Reg05], low-noise LPN [Ale03], and Subset Sum [LPS10].
- **Type-B PKE:** The encryption of a uniformly random message w.r.t. a maliciously chosen public key is computationally close to the uniform distribution over the ciphertext space. This type of PKE is harder to obtain, and can be constructed from enhanced certified TDPs, and from CDH and DDH over efficiently sampleable groups. In case of a TDP f, a ciphertext has the form $(f(r), h(r) \oplus m)$, where h is a hardcore predicate for f, and r is a random element from the domain of f. Under CDH or DDH, a ciphertext is defined as

[2] We can also base our construction on Factoring when relying on the hardness of CDH over the group of signed quadratic residues [HK09], but this requires a trusted setup of this group which is based on a Blum integer.

[3] We call a group efficiently sampleable if we can efficiently sample uniform elements from the group and, given a group element, we can simulate this sampling procedure. A reverse sampleable group [GR13] would suffice. In the context of public-key encryption a similar property is called oblivious key generation [DN00]. In our construction, we require a stronger property where the public keys are additionally computationally indistinguishable from uniform.

[4] The difference is that the notions in [GKM$^+$00] only ask for oblivious sampleability, rather than our stronger requirement of computational uniformity over efficiently sampleable groups.

[5] These are groups for which one can directly sample a group element without knowing the discrete logarithm with respect to some generator. The latter requires non black-box access to the group, which is also needed when using ElGamal with messages that are encoded as group elements and not as exponents. Though we need the stronger property of sampleability of elements that are computationally close to uniform.

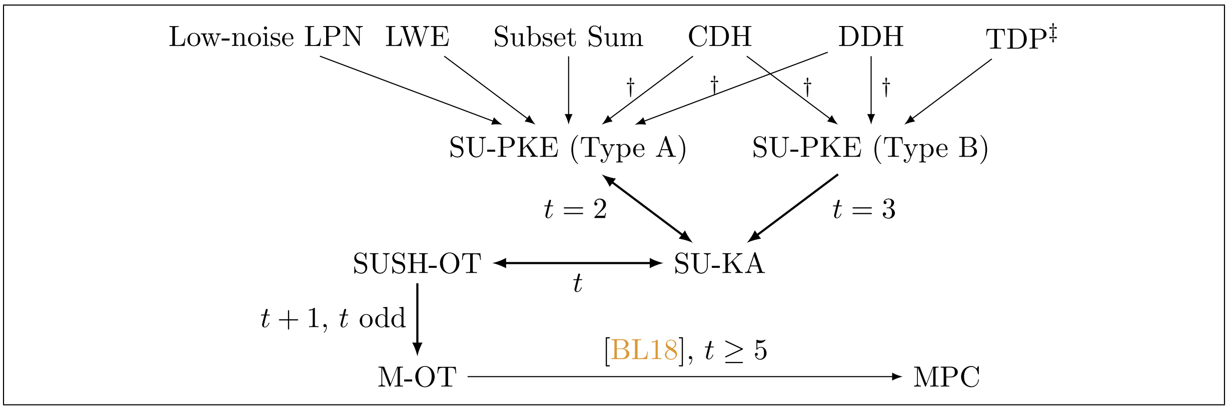

Figure 1: Overview over equivalence and implications of the notion of strong uniformity. The value $t \in \mathbb{N}$ denotes the round complexity. † This holds over efficiently sampleable groups. ‡ We need an enhanced certified TDP.

g^r and $h(g^{xr}) \cdot m$, $g^{xr} \cdot m$ respectively, where g^r is a uniform group element, and g^x is the public key. Clearly, for a uniform message m, these ciphertexts are uniform even under maliciously chosen public keys.

In §3, we show that SU Type-A and SU Type-B PKE imply, respectively, 2-round and 3-round SU-KA, whereas 2-round SU-KA implies SU Type-A PKE. Further, we prove that SU-KA is equivalent to SUSH-OT. The latter implies that strong uniformity is a sufficiently strong notion to bypass the black-box separation between OT and KA, in a similar way as Type-A and Type-B PKE bypass the impossibility of constructing OT from PKE [GKM+00].

Commit-and-open protocols. A 1-out-of-2 commit-and-open (C&O) protocol is a 3-round protocol with the following structure: (1) In the first round, the prover, with inputs two messages m_0, m_1 and a bit d, sends a string γ (called "commitment") generated with m_d but independent of m_{1-d} to the verifier; (2) In the second round, the verifier sends a value β to the prover (called "challenge"); (3) In the third round, the prover sends a tuple (δ, m_0, m_1) to the verifier (called "opening"). Security requires two properties. The first property, called *existence of a committing branch*, demands that a malicious prover must be committed to at least one message, i.e. m_d, already after having sent γ. The second property, called *committing branch indistinguishability*, asks that a malicious verifier cannot learn the committing branch, i.e. d, of an honest prover.

A construction of C&O protocols for single bits is implicit in Kilian [Kil92]. This has been extended to strings by Ostrovsky *et al.* [ORS15]. Both constructions make black-box use of a statistically binding commitment scheme, and allow a prover to equivocally open one of the messages. In §A of the appendix, we revisit the protocol and proof by Ostrovsky *et al.* to show that it indeed satisfies the two security notions sketched above.

M-OT from SUSH-OT: A warm up. In order to explain the main ideas behind our construction of M-OT, we describe below a simplified version of our protocol for the special case of $t = 2$, i.e. when starting with a 2-round SUSH-OT $(\mathsf{S}', \mathsf{R}')$; here, we denote with ρ the message sent by the receiver, and with σ the message sent by the sender, and further observe that for the case of 2 rounds the notion of strong uniformity collapses to standard semi-honest security with the additional property that the distribution of ρ is (computationally close to) uniform to the eyes of an eavesdropper. We then construct a 4-round OT protocol (S, R), as informally described below:

1. ($\mathsf{R} \to \mathsf{S}$): The receiver picks a uniformly random value $m_{1-b} \in \mathcal{M}$, where b is the choice bit, and runs the prover of the C&O protocol upon input m_{1-b}, obtaining a commitment γ that is forwarded to the sender.

2. ($\mathsf{S} \to \mathsf{R}$): The sender samples a challenge β for the C&O protocol, as well as uniformly random elements $r_0, r_1 \in \mathcal{M}$. Hence, it forwards (β, r_0, r_1) to the receiver.

3. ($\mathsf{R} \to \mathsf{S}$): The receiver runs the receiver R' of the underlying 2-round OT protocol with choice bit fixed to 0, obtaining a value ρ_b which is used to define the message $m_b = \rho_b - r_b$ required to complete the execution of the C&O protocol in the non-committing branch b. This results in a tuple (δ, m_0, m_1) that is forwarded to the sender.

4. ($\mathsf{S} \to \mathsf{R}$): The sender verifies that the transcript $T = (\gamma, \beta, (\delta, m_0, m_1))$ is accepting for the underlying C&O protocol. If so, it samples $u_0, u_1 \in \mathcal{M}$ uniformly at random, and runs the sender S' of the underlying 2-round OT protocol twice, with independent random tapes: The first run uses input strings (s_0, u_0) and message $m_0 + r_0$ from the receiver, resulting in a message σ_0, whereas the second run uses input strings (s_1, u_1) and message $m_1 + r_1$ from the receiver, resulting in a message σ_1. Hence, it sends (σ_0, σ_1) to the receiver.

5. Output: The receiver runs the receiver R' of the underlying 2-round OT protocol, upon input message σ_b from the sender, thus obtaining s_b.

Correctness is immediate. In order to prove simulation-based security we proceed in two steps. In the first step, we show the above protocol achieves a weaker security flavor called *receiver-sided simulatability* [NP05, ORS15] which consists of two properties: (1) The existence of a simulator which by interacting with the ideal OT functionality can fake the view of any efficient adversary corrupting the receiver in a real execution of the protocol (i.e., standard simulation-based security w.r.t. corrupted receivers); (2) Indistinguishability of the protocol transcripts with choice bit of the receiver equal to zero or one, for any efficient adversary corrupting the sender in a real execution of the protocol (i.e., game-based security w.r.t. corrupted senders). In the second step, we rely on a *round-preserving* black-box transformation given in [ORS15], which allows to boost receiver-sided simulatability to fully-fledged malicious security. To show (1), we consider a series of hybrid experiments:

- In the first hybrid, we run the first 3 rounds of the protocol, yielding a partial transcript $\gamma, (\beta, r_0, r_1), (\delta, m_0, m_1)$. Hence, after verifying that $T = (\gamma, \beta, (\delta, m_0, m_1))$ is a valid transcript of the C&O protocol, we rewind the adversary to the end of the first round and continue the execution of the protocol from there using a fresh challenge (β', r_0', r_1'), except that after the third round we artificially abort if there is no value $\hat{b} \in \{0,1\}$ such that $m_{\hat{b}} = m_{\hat{b}}'$, where (δ', m_0', m_1') is the third message sent by the adversary after the rewinding.

  Notice that an abort means that it is not possible to identify a committing branch for the C&O protocol, which however can only happen with negligible probability; thus this hybrid is computationally close to the original experiment.

- In the second hybrid, we modify the distribution of the value r_{1-b}' (right after the rewinding) to $r_{1-b}'' = \rho_{1-b} - m_{1-b}$, where we set $1 - b \stackrel{\text{def}}{=} \hat{b}$ from the previous hybrid, and where ρ_{1-b} is obtained by running the receiver R' of the underlying 2-round OT protocol with choice bit fixed to 1.

  To argue indistinguishability, we exploit the fact that the distribution of m_{1-b} is independent from that of r_{1-b}', and thus by strong uniformity we can switch $r_{1-b}' + m_{1-b}$ with ρ_{1-b} from the receiver R'.

- In the third hybrid, we use the simulator of the underlying 2-round SH-OT protocol to compute the messages σ_{1-b} sent by the sender. Note that in both the third and the second hybrid the messages $(\rho_{1-b}, \sigma_{1-b})$ are computed by the honest sender, and thus any efficient algorithm telling apart the third and the second hybrid violates semi-honest security of $(\mathsf{S}', \mathsf{R}')$.

In the last hybrid, a protocol transcript is independent of s_{1-b} but still yields a well distributed output for the malicious receiver, which immediately implies a simulator in the ideal world.

To show (2), we first use the strong uniformity property of $(\mathsf{S}', \mathsf{R}')$ to sample m_b uniformly at random at the beginning of the protocol. Notice the this implies that the receiver cannot recover the value s_b of the sender anymore. Finally, we use the committing branch indistinguishability of the C&O protocol to argue that the transcripts with $b = 0$ and $b = 1$ are computationally indistinguishable.

M-OT from SUSH-OT: The general case. There are several difficulties when trying to extend the above protocol to the general case where we start with a t-round SUSH-OT. In fact, if we would simply iterate sequentially the above construction, where one iteration counts for a message from R' to S' and back, the adversary could use different committing branches from one iteration to the other. This creates a problem in the proof, as the simulator would need to be consistent with both choices of possible committing branches from the adversary, which however requires knowing both inputs from the sender.

We resolve this issue by having the receiver sending all commitments γ_i for the C&O protocol in the first round, where each value γ_i is generated including a random message m^i_{1-b} concatenated with the full history $m^{i-1}_{1-b}, \ldots, m^1_{1-b}$. Hence, during each iteration, the receiver opens one commitment as before. As we show, this prevents the adversary from switching committing branch from one iteration to the next one. We refer the reader to §4.1 for a formal description of our protocol, and to §4.2 for a somewhat detailed proof intuition. The full proof appears in §4.3.

1.3 Application to Round-Efficient MPC

Since M-OT implies maliciously secure MPC [BL18, GS18] and very recently, the work of Choudhuri et al. [CCG+19], a direct consequence of Theorem 1 is the following:

Corollary 1. *For any odd $t \in \mathbb{N}$, there is a non-black-box construction of a $(t+1)$-round maliciously secure multi-party computation protocol in the plain model, from any t-round strongly uniform key agreement protocol.*

Corollary 1 yields 4-round maliciously secure MPC from any of low-noise LPN, high-noise LWE, Subset Sum, CDH, DDH, and RSA, all with polynomial hardness. Previously to our work, it was known how to get maliciously secure MPC in the plain model, for arbitrary functionalities:

- Using 5 rounds, via interactive ZK proofs and SH-OT [BL18], assuming polynomially-hard LWE with super-polynomial noise ratio and adaptive commitments [BHP17], polynomially-hard DDH [ACJ17], and enhanced certified trapdoor permutations (TDP) [ORS15, BL18];

- Using 4 rounds, assuming sub-exponentially-hard LWE with super-polynomial noise ratio and adaptive commitments [BHP17], polynomially-hard LWE with a SIVP approximation factor of $n^{3.5}$ [BD18], sub-exponentially-hard DDH and one-way permutations [ACJ17], polynomially-hard DDH/QR/DCR [BGJ+18], and either polynomially-hard QR or QR together with any of LWE/DDH/DCR (all with polynomial hardness) [HHPV18].

1.4 Related Work

Maliciously secure OT. Jarecki and Shamtikov [JS07], and Peikert, Vaikuntanathan, and Waters [PVW08], show how to construct 2-round M-OT in the common reference string model.

A result by Haitner *et al.* [Hai08, HIK+11] gives a black-box construction of M-OT from SH-OT. While being based on weaker assumptions (i.e., plain SH-OT instead of SUSH-OT), assuming the starting OT protocol has round complexity t, the final protocol requires 4 additional rounds for obtaining an intermediate security flavor known as "defensible privacy", plus 4 rounds for cut and choose, plus 2 times the number of rounds required for running coin tossing, plus a final round to conclude the protocol. Assuming coin tossing can be done in 5 rounds [KO04], the total accounts to $t + 19$ rounds, and thus yields 21 rounds by setting $t = 2$.

Lindell [Lin08] gives constructions of M-OT with 7 rounds, under the DDH assumption, the Nth residuosity assumption, and the assumption that homomorphic PKE exists. Camenish, Neven, and shelat [CNS07], and Green and Hohenberger [GH07], construct M-OT protocols, some of which even achieve adaptive security, using computational assumptions over bilinear groups.

There are also several efficient protocols for OT that guarantee only privacy (but not simulatability) in the presence of malicious adversaries, see, e.g. [KO97, NP01, AIR01, Kal05, BD18].

Round-optimal MPC. Katz and Ostrovsky [KO04] proved that 5 rounds are necessary and sufficient for realizing general-purpose two-party protocols, without assuming a simultaneous broadcast channel (where the parties are allowed to send each other messages in the same round). Their result was later extended by Garg *et al.* [GMPP16] who showed that, assuming simultaneous broadcast, 4 rounds are optimal for general-purpose MPC. Together with a result by Ishai *et al.* [IKO+11]—yielding *non-interactive* maliciously secure two-party computation for arbitrary functionalities, in the OT-hybrid model—the latter implies that 4 rounds are optimal for constructing fully-simulatable M-OT in the plain model.

Ciampi *et al.* [COSV17b] construct a special type of 4-round M-OT assuming certified TDPs,[6] and show how to apply it in order to obtain (fully black-box) 4-round two-party computation with simultaneous broadcast. In a companion paper [COSV17a], the same authors further give a 4-round MPC protocol for the specific case of multi-party coin-tossing.

2 Preliminaries

2.1 Standard Notation

We use $\lambda \in \mathbb{N}$ to denote the security parameter, sans-serif letters (such as A, B) to denote algorithms, caligraphic letters (such as $\mathcal{X}$, $\mathcal{Y}$) to denote sets, and bold-face letters (such as $\mathbf{v}$, $\mathbf{A}$) to denote vectors and matrices; all vectors are by default row vectors, and $\mathbf{v}^\mathsf{T}$ denotes a column vector. An algorithm is *probabilistic polynomial-time* (PPT) if it is randomized, and its running time can be bounded by a polynomial in its input length. By $y \leftarrow_\$ \mathsf{A}(x)$, we mean that the value y is assigned to the output of algorithm A upon input x and fresh random coins. We implicitly assume that all algorithms are given the security parameter 1^λ as input.

A function $\nu : \mathbb{N} \to [0, 1]$ is negligible in the security parameter (or simply negligible) if it vanishes faster than the inverse of any polynomial in λ, i.e. $\nu(\lambda) \in O(1/p(\lambda))$ for all positive polynomials $p(\lambda)$. We often write $\nu(\lambda) \in \mathsf{negl}(\lambda)$ to denote that $\nu(\lambda)$ is negligible.

[6] They also claim [COSV17b, Footnote 3] that their OT protocol can be instantiated using PKE with special properties, however no proof of this fact is provided.

Ideal Functionality $\mathcal{F}_{\mathtt{OT}}$:

The functionality runs with Turing machines (S, R) and adversary Sim, and works as follows:

- Upon receiving message $(\mathtt{send}, s_0, s_1, \mathsf{S}, \mathsf{R})$ from S, where $s_0, s_1 \in \{0,1\}^\lambda$, store s_0 and s_1 and answer $\mathtt{send}$ to R and Sim.
- Upon receiving a message $(\mathtt{receive}, b)$ from R, where $b \in \{0,1\}$, send s_b to R and $\mathtt{receive}$ to S and Sim, and halt. If no message $(\mathtt{send}, \cdot)$ was previously sent, do nothing.

Figure 2: Oblivious transfer ideal functionality

For a random variable X, we write $\mathbb{P}[X = x]$ for the probability that X takes on a particular value $x \in \mathcal{X}$ (with $\mathcal{X}$ being the set where X is defined). The statistical distance between two random variables X and X' defined over the same set $\mathcal{X}$ is defined as $\Delta(X; X') = \frac{1}{2}\sum_{x\in\mathcal{X}} |\Pr[X = x] - \Pr[X' = x]|$. Given two ensembles $X = \{X_\lambda\}_{\lambda\in\mathbb{N}}$ and $Y = \{Y_\lambda\}_{\lambda\in\mathbb{N}}$, we write $X \equiv Y$ to denote that they are identically distributed, $X \approx_s Y$ to denote that they are statistically close (i.e., $\Delta(X_\lambda; Y_\lambda) \in \mathsf{negl}(\lambda)$), and $X \approx_c Y$ to denote that they are computationally indistinguishable—i.e., for all PPT distinguishers D there exists a negligible function $\nu : \mathbb{N} \to [0,1]$ such that $|\Pr[\mathsf{D}(X_\lambda) = 1] - \Pr[\mathsf{D}(Y_\lambda) = 1]| \leq \nu(\lambda)$.

We call a group efficiently sampleable if and only if there is a PPT sampling procedure Samp for the uniform distribution over the group, and moreover there exists a PPT simulator $\mathsf{SimSamp}$ that given an element of the group, outputs the randomness used by Samp. More precisely, $(r, \mathsf{Samp}(1^\lambda, r)) \approx_c (r', \mathsf{Samp}(1^\lambda, r'))$ where $r' \leftarrow\!\!\$\ \mathsf{SimSamp}(1^\lambda, \mathsf{Samp}(1^\lambda; r))$ and $r \leftarrow\!\!\$\ \{0,1\}^*$.[7] A group that is efficiently reverse sampleable (as in [GR13]) suffices.

2.2 Oblivious Transfer

An interactive protocol Π for the Oblivious Transfer (OT) functionality, features two interactive PPT Turing machines S, R called, respectively, the sender and the receiver. The sender S holds a pair of strings $s_0, s_1 \in \{0,1\}^\lambda$, whereas the receiver R is given a choice bit $b \in \{0,1\}$. At the end of the protocol, which might take several rounds, the receiver learns s_b (and nothing more), whereas the sender learns nothing.

Typically, security of OT is defined using the real/ideal paradigm. Specifically, we compare a real execution of the protocol, where an adversary might corrupt either the sender or the receiver, with an ideal execution where the parties can interact with an ideal functionality. The ideal functionality, which we denote by $\mathcal{F}_{\mathtt{OT}}$, features a trusted party that receives the inputs from both the sender and the receiver, and then sends to the receiver the sender's input corresponding to the receiver's choice bit. We refer the reader to Fig. 2 for a formal specification of the $\mathcal{F}_{\mathtt{OT}}$ functionality.

In what follows, we denote by $REAL_{\Pi,\mathsf{R}^*(z)}(\lambda, s_0, s_1, b)$ (resp., $REAL_{\Pi,\mathsf{S}^*(z)}(\lambda, s_0, s_1, b)$) the distribution of the output of the malicious receiver (resp., sender) during a real execution of the protocol Π (with s_0, s_1 as inputs of the sender, b as choice bit of the receiver, and z as auxiliary input for the adversary), and by $IDEAL_{\mathcal{F}_{\mathtt{OT}},\mathsf{Sim}^{\mathsf{R}^*(z)}}(\lambda, s_0, s_1, b)$ (resp., $IDEAL_{\mathcal{F}_{\mathtt{OT}},\mathsf{Sim}^{\mathsf{S}^*(z)}}(\lambda, s_0, s_1, b)$) the output of the malicious receiver (resp., sender) in an ideal execution where the parties (with analogous inputs) interact with $\mathcal{F}_{\mathtt{OT}}$, and where the simulator is given black-box access to the adversary.

[7]The existence of a simulator is crucial for constructing SUSH-OT from SU-KA; we solely use it for this purpose.

Definition 1 (OT with full simulation)**.** Let $\mathcal{F}_{\mathtt{OT}}$ be the functionality from Fig. 2. We say that a protocol $\Pi = (\mathsf{S}, \mathsf{R})$ securely computes $\mathcal{F}_{\mathtt{OT}}$ with *full simulation* if the following holds:

(a) For every non-uniform PPT malicious receiver R^*, there exists a non-uniform PPT simulator Sim such that

$$\left\{REAL_{\Pi,\mathsf{R}^*(z)}(\lambda, s_0, s_1, b)\right\}_{\lambda,s_0,s_1,b,z} \approx_c \left\{IDEAL_{\mathcal{F}_{\mathtt{OT}},\mathsf{Sim}^{\mathsf{R}^*(z)}}(\lambda, s_0, s_1, b)\right\}_{\lambda,s_0,s_1,b,z}$$

where $\lambda \in \mathbb{N}$, $s_0, s_1 \in \{0,1\}^\lambda$, $b \in \{0,1\}$, and $z \in \{0,1\}^*$.

(b) For every non-uniform PPT malicious sender S^*, there exists a non-uniform PPT simulator Sim such that

$$\left\{REAL_{\Pi,\mathsf{S}^*(z)}(\lambda, s_0, s_1, b)\right\}_{\lambda,s_0,s_1,b,z} \approx_c \left\{IDEAL_{\mathcal{F}_{\mathtt{OT}},\mathsf{Sim}^{\mathsf{S}^*(z)}}(\lambda, s_0, s_1, b)\right\}_{\lambda,s_0,s_1,b,z}$$

where $\lambda \in \mathbb{N}$, $s_0, s_1 \in \{0,1\}^\lambda$, $b \in \{0,1\}$, and $z \in \{0,1\}^*$.

Game-based security. One can also consider weaker security definitions for OT, where simulation-based security only holds when either the receiver or the sender is corrupted, whereas when the other party is malicious only game-based security is guaranteed. Below, we give the definition for the case of a corrupted sender, which yields a security notion known as *receiver-sided* simulatability. Intuitively, the latter means that the adversary cannot distinguish whether the honest receiver is playing with choice bit 0 or 1.

Definition 2 (OT with receiver-sided simulation)**.** Let $\mathcal{F}_{\mathtt{OT}}$ be the functionality from Fig. 2. We say that a protocol $\Pi = (\mathsf{S}, \mathsf{R})$ securely computes $\mathcal{F}_{\mathtt{OT}}$ with *receiver-sided simulation* if the following holds:

(a) Same as property (a) in Definition 1.

(b) For every non-uniform PPT malicious sender S^* it holds that

$$\left\{VIEW^{\mathsf{R}}_{\Pi,\mathsf{S}^*(z)}(\lambda, s_0, s_1, 0)\right\}_{\lambda,s_0,s_1,z} \approx_c \left\{VIEW^{\mathsf{R}}_{\Pi,\mathsf{S}^*(z)}(\lambda, s_0, s_1, 1)\right\}_{\lambda,s_0,s_1,z}$$

where $\lambda \in \mathbb{N}$, $s_0, s_1 \in \{0,1\}^\lambda$, and $z \in \{0,1\}^*$, and where $VIEW^{\mathsf{R}}_{\Pi,\mathsf{S}^*(z)}(\lambda, s_0, s_1, b)$ is the distribution of the view of S^* (with input s_0, s_1 and auxiliary input z) at the end of a real execution of protocol Π with the honest receiver R given b as input.

Receiver-sided simulatability is a useful stepping stone towards achieving full simulatability. In fact, Ostrovsky *et al.* [ORS15] show how to compile any 4-round OT protocol with receiver-sided simulatability to a 4-round OT protocol with full simulatability. This transformation can be easily extended to hold for any t-round protocol, with $t \geq 3$; the main reason is that the transform only relies on an extractable commitment scheme, which requires at least 3 rounds.

Theorem 2 (Adapted from [ORS15])**.** *Assuming $t \geq 3$, there is a black-box transformation from t-round OT with* receiver-sided *simulation to t-round OT with full simulation.*[8]

[8]They also need the existence of one-way functions. Since OT implies OT extension which implies one-way functions [LZ13, LZ18], OT implies one-way functions.

2.3 Commit-and-Open Protocols

We envision a 3-round protocol between a prover and a verifier where the prover takes as input two messages $m_0, m_1 \in \mathcal{M}$ and a bit $d \in \{0,1\}$. The prover speaks first, and the protocol is public coin, in the sense that the message of the verifier consists of uniformly random bits. Intuitively, we want that whenever the prover manages to convince the verifier, he must be committed to at least one of m_0, m_1 already after having sent the first message.

More formally, a 1-out-of-2 commit-and-open (C&O) protocol is a tuple of efficient interactive Turing machines $\Pi_{\mathsf{c\&o}} \stackrel{\text{def}}{=} (\mathsf{P} = (\mathsf{P}_0, \mathsf{P}_1), \mathsf{V} = (\mathsf{V}_0, \mathsf{V}_1))$ specified as follows. (i) The randomized algorithm P_0 takes m_d and returns a string $\gamma \in \{0,1\}^*$ and auxiliary state information $\alpha \in \{0,1\}^*$; (ii) The randomized algorithm V_0 returns a random string $\beta \leftarrow\!\$\, \mathcal{B}$; (iii) The randomized algorithm P_1 takes $(\alpha, \beta, \gamma, m_{1-d})$ and returns a string $\delta \in \{0,1\}^*$; (iv) The deterministic algorithm V_1 takes a transcript $(\gamma, \beta, (\delta, m_0, m_1))$ and outputs a bit.

We write $\langle \mathsf{P}(m_0, m_1, d), \mathsf{V}(1^\lambda) \rangle$ for a run of the protocol upon inputs (m_0, m_1, d) to the prover, and we denote by $T \stackrel{\text{def}}{=} (\gamma, \beta, (\delta, m_0, m_1))$ the random variable corresponding to a transcript of the interaction. Note that the prover does not necessarily need to know m_{1-d} before computing the first message. We say that $\Pi_{\mathsf{c\&o}}$ satisfies completeness if honestly generated transcripts are always accepted by the verifier, i.e. for all $m_0, m_1 \in \mathcal{M}$ and $d \in \{0,1\}$, we have $\Pr[\mathsf{V}_1(T) = 1 : \; T \leftarrow\!\$\, \langle \mathsf{P}(m_0, m_1, d), \mathsf{V}(1^\lambda) \rangle] = 1$, where the probability is over the randomness of $\mathsf{P}_0, \mathsf{V}_0$, and P_1.

Security properties. Roughly, a C&O protocol must satisfy two security requirements. The first requirement is that at the end of the first round, a malicious prover is committed to at least one message. This can be formalized by looking at a mental experiment where we first run the protocol with a malicious prover, yielding a first transcript $T = (\gamma, \beta, (\delta, m_0, m_1))$; hence, we rewind the prover to the point it already sent the first message, and give it a fresh challenge β' which yields a second transcript $T' = (\gamma, \beta', (\delta', m_0', m_1'))$. The security property now states that, as long as the two transcripts T and T' are valid, it shall exist at least one "committing branch" $\hat{d} \in \{0,1\}$ for which $m_{\hat{d}} = m'_{\hat{d}}$. The second requirement says that no malicious verifier can learn any information on the committing branch of the prover. The formal definitions appear below.

Definition 3 (Secure commit-and-open protocol)**.** Let $\Pi_{\mathsf{c\&o}} = (\mathsf{P}_0, \mathsf{P}_1, \mathsf{V}_0, \mathsf{V}_1)$ be a 1-out-of-2 commit-and-open protocol. We say that $\Pi_{\mathsf{c\&o}}$ is secure if, besides completeness, it satisfies the following security properties.

- **Existence of Committing Branch:** For every PPT malicious prover $\mathsf{P}^* = (\mathsf{P}_0^*, \mathsf{P}_1^*)$ there exists a negligible function $\nu : \mathbb{N} \to [0,1]$ such that

$$\Pr\left[\begin{array}{c} (\mathsf{V}_1(T) = 1) \wedge (\mathsf{V}_1(T') = 1) \\ \wedge (m_0 \neq m_0') \wedge (m_1 \neq m_1') \end{array} : \begin{array}{c} (\gamma, \alpha_0) \leftarrow\!\$\, \mathsf{P}_0^*(1^\lambda); \\ \beta, \beta' \leftarrow\!\$\, \mathsf{V}_0(1^\lambda); \\ (\delta, m_0, m_1) \leftarrow\!\$\, \mathsf{P}_1^*(\alpha_0, \beta); \\ (\delta', m_0', m_1') \leftarrow\!\$\, \mathsf{P}_1^*(\alpha_0, \beta') \end{array}\right] \leq \nu(\lambda),$$

  with $T = (\gamma, \beta, (\delta, m_0, m_1))$ and $T' = (\gamma, \beta, (\delta, m_0', m_1'))$, and where the probability is taken over the random coin tosses of P^* and V.

- **Committing Branch Indistinguishability:** For all PPT malicious verifiers V^*, and for all messages $m_0, m_1 \in \mathcal{M}$, we have that

$$\left\{\tilde{T} : \; \tilde{T} \leftarrow\!\$\, \langle \mathsf{P}(m_0, m_1, 0), \mathsf{V}^*(1^\lambda) \rangle\right\}_{\lambda \in \mathbb{N}} \approx_c \left\{\tilde{T} : \; \tilde{T} \leftarrow\!\$\, \langle \mathsf{P}(m_0, m_1, 1), \mathsf{V}^*(1^\lambda) \rangle\right\}_{\lambda \in \mathbb{N}}.$$

In §A.2 we show that a protocol by Ostrovsky *et al.* [ORS15] achieves this definition.

3 Strongly Uniform PKE, Key Agreement and OT

3.1 Strongly Uniform PKE

We start with defining strongly uniform public-key encryption (PKE). Here, we differ between two types of PKE. A Type-A PKE has a public key that is computationally close to uniform, while for a Type-B PKE this is the case for ciphertexts of uniform messages (under malicious public keys).

In general, a PKE scheme Π_{pke} consists of three efficient algorithms $(\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ specified as follows. (i) The probabilistic algorithm KGen takes as input the security parameter and outputs a pair of keys (pk, sk); (ii) The probabilistic algorithm Enc takes as input the public key pk and a message $\mu \in \mathcal{M}$, and returns a ciphertext $c \in \mathcal{C}$; (iii) The deterministic algorithm Dec takes as input the secret key sk and a ciphertext $c \in \mathcal{C}$, and returns a value $\mu \in \mathcal{M} \cup \{\perp\}$. We say that Π_{pke} meets correctness, if for all $\lambda \in \mathbb{N}$, all (pk, sk) output by $\mathsf{KGen}(1^\lambda)$, and all $\mu \in \mathcal{M}$ the following holds: $\mathbb{P}[\mathsf{Dec}(sk, \mathsf{Enc}(pk, \mu)) = \mu] = 1$.

Definition 4 (Strongly uniform Type-A PKE)**.** A PKE scheme $\Pi_{\mathsf{pke}} = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ is called a strongly uniform Type-A PKE if for any PPT distinguisher D the following holds:

$$|\Pr[\mathsf{D}(pk) = 1] - \Pr[\mathsf{D}(u) = 1]| \in \mathsf{negl}(\lambda),$$

where $(pk, sk) \leftarrow\$ \mathsf{KGen}(1^\lambda)$ and u is uniform over a suitable, efficiently sampleable group.

In case of strongly uniform Type-B PKE, we even ask that a ciphertext of a uniform message is indistinguishable from uniform to a distinguisher that chooses a public key for the encryption procedure in an arbitrary way.

Definition 5 (Strongly uniform Type-B PKE)**.** A PKE scheme $\Pi_{\mathsf{pke}} = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ is called a strongly uniform Type-B PKE if for any PPT distinguisher D the following holds:

$$|\Pr[\mathsf{D}(c) = 1] - \Pr[\mathsf{D}(u) = 1]| \in \mathsf{negl}(\lambda),$$

where $pk \in \{0,1\}^*$ is chosen by D, $\mu \leftarrow\$ \mathcal{M}$, $c \leftarrow\$ \mathsf{Enc}(pk, \mu)$, and u is uniform over a suitable, efficiently sampleable group.

When using PKE in the following, we also ask for standard security against chosen plaintext attacks, since this is not implied by the notion of strong uniformity in all generality.

3.2 Strongly Uniform Key Agreement

Let $\Pi_{\mathsf{ka}} = (\mathsf{P}_1, \mathsf{P}_2)$ be a key agreement (KA) protocol, where P_1 sends messages during t' rounds, which we denote by $\rho^1, \dots, \rho^{t'}$. The messages from P_2 to P_1 are denoted with $\sigma^1, \dots, \sigma^{t'+1}$, and are at most $t' + 1$. W.l.o.g. we will assume that P_2 sends the last message.

More precisely, algorithms P_1 and P_2 are stateful interactive Turing machines such that for each $i \in [t']$: (i) Algorithm P_1 takes the current state information $\alpha^{i-1}_{\mathsf{P}_1}$ (where $\alpha^0_{\mathsf{P}_1}$ is equal to P_1's input 1^λ) and a message σ^{i-1} from P_2 (with σ^0 empty), and returns ρ^i together with updated state information $\alpha^i_{\mathsf{P}_1}$; (ii) Algorithm P_2 takes the current state information $\alpha^{i-1}_{\mathsf{P}_2}$ (where $\alpha^0_{\mathsf{P}_2}$ is equal to P_2's input 1^λ) and message ρ^i from the receiver, and returns σ^i together with updated state information $\alpha^i_{\mathsf{P}_2}$.

For strong uniformity, we ask that P_1's messages are computationally close to uniform over an efficiently sampleable group $\mathcal{M}$. For simplicity, we assume that this is the same group for all messages. Our results still hold when the messages are uniform in different groups.

Additionally, we ask that given a transcript, one cannot distinguish the key P_1 and P_2 agreed upon from a uniformly random string.

Definition 6 (Strongly uniform secure key agreement)**.** A KA protocol $\Pi_{\mathtt{ka}} = (\mathsf{P}_1, \mathsf{P}_2)$ as defined above is a strongly uniform secure KA if there exists an efficiently samplable group $\mathcal{M}$ such that the messages $(\rho^1, \dots, \rho^{t'})$ sent by P_1 in a honest execution of the protocol are distributed over $\mathcal{M}$, and moreover the following conditions are met:

(a) **Key Indistinguishability:** For an honest execution of the protocol with agreed key K,

$$\left(\langle \mathsf{P}_1(1^\lambda), \mathsf{P}_2(1^\lambda)\rangle, K\right) \approx_c \left(\langle \mathsf{P}_1(1^\lambda), \mathsf{P}_2(1^\lambda)\rangle, U\right),$$

where U is uniform and independent of the view of P_1 and P_2.

(b) **Uniformity w.r.t. Malicious Interaction:** For all PPT distinguishers D the following quantity is negligible:

$$\Bigg| \Pr\left[\mathsf{D}(\alpha_{\mathsf{D}}^{t'}, (\rho^i, \sigma^i)_{i\in[t']}) = 1 : \begin{array}{c} \forall i \in [t'], (\alpha_{\mathsf{P}_1}^i, \rho^i) \leftarrow\$ \mathsf{P}_1(\alpha_{\mathsf{P}_1}^{i-1}, \sigma^i) \\ \wedge\ (\alpha_{\mathsf{D}}^i, \sigma^i) \leftarrow\$ \mathsf{D}(\alpha_{\mathsf{D}}^{i-1}, \rho^{i-1}) \end{array}\right] - \Pr\left[\mathsf{D}(\alpha_{\mathsf{D}}^{t'}, (\rho^i, \sigma^i)_{i\in[t']}) = 1 : \begin{array}{c} \forall i \in [t'], \rho^i \leftarrow\$ \mathcal{M} \\ \wedge\ (\alpha_{\mathsf{D}}^i, \sigma^i) \leftarrow\$ \mathsf{D}(\alpha_{\mathsf{D}}^{i-1}, \rho^{i-1}) \end{array}\right] \Bigg|,$$

where ρ^0 is the empty string, and $\alpha_{\mathsf{P}_1}^0 = \alpha_{\mathsf{P}_2}^0 = 1^\lambda$.

We show now that the property of strong uniformity is preserved within known construction of KA from Type-A or Type-B PKE, as well as Type-A PKE from KA. Both of the following lemmata are straightforward, and therefore we forego a more formal proof and just sketch them.

Lemma 1. *There exists a* 2*-round strongly uniform secure KA if and only if there exists a strongly uniform CPA-secure Type-A PKE (constructive).*

Proof. It is a well known fact that 2-round KA implies PKE and vice versa. What we will show is that this construction preserves strong uniformity. In the construction of KA from PKE the receiver sends a public key and receives back an encryption of a uniform key. If the public key is indistinguishable from uniform with all but negligible probability, then all the receivers messages are, and hence the KA is strongly uniform.

In the construction of PKE from KA, one uses the first message of the KA as public key. In a 2-round strongly uniform KA this message is indistinguishable from uniform with all but negligible probability by definition. Hence, the public key is computationally indistinguishable from uniform with all but negligible probability. □

Lemma 2. *If there exists a strongly uniform CPA-secure Type-B PKE, then there exists a* 3*-round strongly uniform secure KA (constructive).*

Gertner *et al.* [GKM+00] showed a similar lemma, namely that Type-B PKE implies 3-round semi-honestly secure OT. For simplicity, we prefer showing that there is a 3-round strongly uniform secure KA given a strongly uniform CPA-secure Type-B PKE.

Proof. The idea is simple and similar to the proof of Lemma 1. P_2 sends a public key, P_1 sends an encryption of a uniform key. Finally, P_2 decrypts the ciphertext and sends a dummy message. The last message is required by the definition of strongly uniform KA, which asks that P_1's messages are indistinguishable from uniform, where P_2 sends the last message.

In order to achieve strongly uniform KA, even for maliciously chosen public key, the ciphertext needs to be indistinguishable from uniform with all but negligible probability. Type-B PKE has this property, and hence the described protocol is strongly uniform. Security follows trivially, and for identical reasons, as in Lemma 1. □

3.3 Strongly Uniform OT

In an OT protocol $\Pi = (\mathsf{S}, \mathsf{R})$ we can w.l.o.g. assume that the sender S always speaks last. We use the same notation as described above for a key agreement protocol. In particular, $\rho^1, \ldots, \rho^{t'}$ are the messages from R to S, and $\sigma^1, \ldots, \sigma^{t'+1}$ the messages from S to R. The initial states are identical with the inputs, i.e. $\alpha_{\mathsf{R}}^0 = b \in \{0,1\}$ and $\alpha_{\mathsf{S}}^0 = (s_0, s_1) \in \{0,1\}^{2\lambda}$.

Correctness means that for all $b \in \{0,1\}$, and for all $s_0, s_1 \in \{0,1\}^\lambda$, the following probability is overwhelming:

$$\Pr\left[\rho^{t'+1} = s_b : \forall i \in [t'+1], (\alpha_{\mathsf{R}}^i, \rho^i) \leftarrow\!\!\$\ \mathsf{R}(\alpha_{\mathsf{R}}^{i-1}, \sigma^i) \wedge (\alpha_{\mathsf{S}}^i, \sigma^i) \leftarrow\!\!\$\ \mathsf{S}(\alpha_{\mathsf{S}}^{i-1}, \rho^{i-1})\right],$$

where ρ^0 is the empty string, and $\alpha_{\mathsf{S}}^0 = (s_0, s_1)$, $\alpha_{\mathsf{R}}^0 = b$.

As for security, we require two properties. The first property is equivalent to simulation-based security for honest-but-curious receivers. The second property says that a malicious sender cannot distinguish the case where it is interacting with the honest receiver, from the case where the messages from the receiver are replaced by uniform elements over an efficiently sampleable group $\mathcal{M}$.

Definition 7 (Strongly Uniform semi-honestly secure OT)**.** An OT protocol $\Pi = (\mathsf{S}, \mathsf{R})$ as defined above is a strongly uniform semi-honestly secure OT if there exists an efficiently samplable group $\mathcal{M}$ such that the messages $(\rho^1, \ldots, \rho^{t'})$ sent by R in a honest execution of the protocol are distributed over $\mathcal{M}$, and moreover the following conditions are met:

(a) **Security w.r.t. Semi-Honest Receivers:** There exists a PPT simulator $\mathsf{Sim}_{\mathsf{R}}$ such that for all $b \in \{0,1\}$ and for all $s_0, s_1 \in \{0,1\}^\lambda$ the following holds:

$$\left\{\mathsf{Sim}_{\mathsf{R}}(1^\lambda, b, s_b)\right\}_{\lambda, b, s_b} \approx_c \left\{\mathit{VIEW}_{\Pi}^{\mathsf{R}}(\lambda, s_0, s_1, b)\right\}_{\lambda, s_0, s_1, b},$$

where $\mathit{VIEW}_{\Pi}^{\mathsf{R}}(\lambda, s_0, s_1, b)$ denotes the distribution of the view of the honest receiver at the end of the protocol.

(b) **Uniformity w.r.t. Malicious Senders:** For all PPT distinguishers D, and for all $b \in \{0,1\}$, the following quantity is negligible:

$$\left| \Pr\left[\mathsf{D}(\alpha_{\mathsf{D}}^{t'}, (\rho^i, \sigma^i)_{i\in[t']}) = 1 : \begin{array}{c} \forall i \in [t'], (\alpha_{\mathsf{R}}^i, \rho^i) \leftarrow\!\!\$\ \mathsf{R}(\alpha_{\mathsf{R}}^{i-1}, \sigma^{i-1}) \\ \wedge\ (\alpha_{\mathsf{D}}^i, \sigma^i) \leftarrow\!\!\$\ \mathsf{D}(\alpha_{\mathsf{D}}^{i-1}, \rho^{i-1}) \end{array}\right] \right.$$
$$\left. - \Pr\left[\mathsf{D}(\alpha_{\mathsf{D}}^{t'}, (\rho^i, \sigma^i)_{i\in[t']}) = 1 : \begin{array}{c} \forall i \in [t'], \rho^i \leftarrow\!\!\$\ \mathcal{M} \\ \wedge\ (\alpha_{\mathsf{D}}^i, \sigma^i) \leftarrow\!\!\$\ \mathsf{D}(\alpha_{\mathsf{D}}^{i-1}, \rho^{i-1}) \end{array}\right] \right|,$$

where ρ^0 is the empty string, and $\alpha_{\mathsf{R}}^0 = b$.

Note that the second property implies game-based security w.r.t. malicious senders (i.e., property (b) of Definition 2). Furthermore, for the special case of $t' = 1$ the above definition collapses to standard semi-honest security, as the only message sent by the malicious sender plays no role in distinguishing the two distributions.

Next, we show a lemma that is not very surprising, namely that strongly uniform secure KA can be constructed from strongly uniform semi-honestly secure OT.

Lemma 3. *If there exists a t-round strongly uniform semi-honestly secure OT, then there exists a t-round strongly uniform secure KA (constructive).*

Proof. We construct a t-round KA $\Pi_{\mathtt{ka}}$ from a t-round OT Π as follows. P_1 and P_2 run Π, where P_1 takes the role of the receiver with choice bit 0. P_2 takes the role of the sender, with inputs equal to a uniform key k, i.e. $s_0 = k$, and a uniformly random string u, i.e. $s_1 = u$.

Given that Π is semi-honestly secure, Gertner *et al.* [GKM⁺00, Theorem 5] have shown that $\Pi_{\mathtt{ka}}$ is indeed a secure KA. The rough idea is to first switch the receiver's choice bit to 1 by using the game-based security of Π against honest-but-curious senders (which in our case is implied by strong uniformity). Afterwards, we can use the security against an honest-but-curious receiver to argue that an eavesdropper cannot distinguish $s_0 = k$ from random anymore, since even the receiver can only learn s_1 but has no information about s_0. Therefore $\Pi_{\mathtt{ka}}$ is a secure KA. It remains to prove strong uniformity.

Claim 1. *Assuming Π is strongly uniform, so is $\Pi_{\mathtt{ka}}$.*

Proof. Let PPT D' break the strong uniformity of $\Pi_{\mathtt{ka}}$, then we construct a PPT distinguisher D that breaks the strong uniformity of Π as follows. Distinguisher D chooses k and u uniformly, and interacts as a honest sender in Π, where the receiver's messages are either distributed according to the protocol description or uniform. Hence, P_1's messages are either conform with the protocol or uniform. Distinguisher D' receives the view of P_2 generated by D. Now, if D' distinguishes the messages of P_1 being conform with the protocol from uniform, D breaks the strong uniformity of Π. □

□

The next lemma is more surprising, as it implies that strongly uniform secure KA is equivalent to strongly uniform semi-honestly secure OT. Hence, the notion of strong uniformity is sufficiently strong to bypass the black-box separation of KA and OT by Gertner *et al.* [GKM⁺00, Corollary 7], which is a consequence of the separation between PKE and OT, and the fact that 2-round KA implies PKE. The above also implies that 2-round secure KA is separated as well from 2-round strongly uniform secure KA.

Lemma 4. *If there exists a t-round strongly uniform secure KA, then there exists a t-round strongly uniform semi-honestly secure OT (constructive).*

Proof. We construct an OT protocol Π using two parallel executions of a KA protocol $\Pi_{\mathtt{ka}}$, which we denote with $\Pi_{\mathtt{ka}}^0$ and $\Pi_{\mathtt{ka}}^1$. The receiver of the OT acts in both executions as P_1. For his choice bit b, he runs $\Pi_{\mathtt{ka}}^b$ according to the protocol description, and in $\Pi_{\mathtt{ka}}^{1-b}$ he samples and sends uniform messages.

In the last round the sender sends $k_0 + s_0$ and $k_1 + s_1$, where for $j \in \{0,1\}$ the key k_j is the exchanged key in $\Pi_{\mathtt{ka}}^j$, and s_0, s_1 are the OT inputs of the sender. Notice that this is a t-round protocol, since the sender can send his masked inputs together with his last messages of the KA protocols.

Claim 2. *Assuming $\Pi_{\mathtt{ka}}$ is strongly uniform, so is Π.*

Proof. Let there be a PPT distinguisher D' that distinguishes the receiver's messages in Π from uniform. We construct a PPT distinguisher D for P_1's messages using D'. Distinguisher D acts in $\Pi_{\mathtt{ka}}$ as P_2, where P_1's messages are either distributed according to the protocol description or uniform. Hence, D picks $b \leftarrow\!\!\$ \{0,1\}$ and uses the messages sent by D' in Π to interact with P_1 in $\Pi_{\mathtt{ka}}^b$. For $\Pi_{\mathtt{ka}}^{1-b}$, distinguisher D sends uniform messages as in the protocol description. Finally, D outputs the output of D'. Hence, if D' is successful, then so is D. □

Claim 3. *Assuming $\Pi_{\mathtt{ka}}$ is strongly uniform and secure, then Π is secure against honest-but-curious receivers.*

Proof. We use the following hybrids, where a simulator Sim generates a view of the receiver. In the last hybrid, Sim only uses s_b but not s_{1-b} and therefore implements a simulator $\mathsf{Sim}(1^\lambda, b, s_b)$ as required for security against honest-but-curious receivers.

$HYB_1(\lambda)$**:** Sim generates the receivers messages in $\Pi_{\mathtt{ka}}^{1-b}$ as in an actual key agreement $\Pi_{\mathtt{ka}}$, i.e. not uniform as in Π. The receiver's view only contains the messages, not the randomness used in $\Pi_{\mathtt{ka}}$ to generate these messages.

$HYB_2(\lambda)$**:** Sim sends a uniform value u instead of $k_{1-b} + s_{1-b}$.

To prove the claim, we need to show that the receiver's view in the real protocol is indistinguishable from $HYB_1(\lambda)$, and that $HYB_1(\lambda)$ is indistinguishable from $HYB_2(\lambda)$.

Let D' be a PPT distinguisher that distinguishes the recevier's view in the real protocol from $HYB_1(\lambda)$ with non-negligible probability. We show that there is a PPT distinguisher D that breaks the strong uniformity of the KA $\Pi_{\mathtt{ka}}$ with the same probability. Distinguisher D runs Π, but replaces the interaction in $\Pi_{\mathtt{ka}}^{1-b}$ on the receiver's side with a challenge instance of $\Pi_{\mathtt{ka}}$ against the strong uniformity. To simulate the view of the receiver correctly, we need to simulate the sampling procedure of the uniform messages in the protocol given only the challenge messages. We can do this by using the simulator $\mathsf{SimSamp}$ of efficiently sampleable groups. The challenge messages are either uniform, as in the receiver's actual view in Π, or honestly generated, as in $HYB_1(\lambda)$. Otherwise, D acts exactly according to the protocol description of Π. If D' distinguishes the two cases, D breaks the uniformity of $\Pi_{\mathtt{ka}}$.

Now let D' be a PPT distinguisher that distinguishes $HYB_1(\lambda)$ from $HYB_2(\lambda)$ with non-negligible probability. Then we can construct a PPT distinguisher that breaks security, i.e. key indistinguishability, of Π with the same probability. Distinguisher D receives a transcript of $\Pi_{\mathtt{ka}}$ and a challenge z which is either the key k or uniform. Hence, D uses the transcript of $\Pi_{\mathtt{ka}}$ as transcript of $\Pi_{\mathtt{ka}}^{1-b}$, and the challenge z to generate the message $k_{1-b} + s_{1-b}$ as $z + s_{1-b}$. Finally, D generates the remaining parts of the receiver's view honestly. If $z = k$, then D simulates $HYB_1(\lambda)$, and if $z = u$, and hence $z + s_{1-b}$ is uniform, D simulates $HYB_2(\lambda)$. Hence, whenever D' distinguishes $HYB_1(\lambda)$ from $HYB_2(\lambda)$, then D distinguishes the actual key from uniform. □

□

4 From Strongly Uniform Semi-Honestly Secure OT to Maliciously Secure OT

Our protocol is described in §4.1. In §4.2 we provide a somewhat detailed proof sketch, whereas in §4.3 we formally show the protocol satisfies receiver-sided simulatability; recall that by using Theorem 2 we immediately get a fully simulatable OT protocol.

4.1 Protocol Description

Let $\Pi_{\mathsf{c\&o}} = (\mathsf{P}_0, \mathsf{P}_1, \mathsf{V}_0, \mathsf{V}_1)$ be a 1-out-of-2 C&O protocol and $\Pi' = (\mathsf{S}', \mathsf{R}')$ be a $(2t'+1)$-round OT protocol, where the first message σ^1 might be the empty string. Our OT protocol $\Pi = (\mathsf{S}, \mathsf{R})$ is depicted in Fig. 3 on the following page. The protocol consists of $(2t'+2)$ rounds as informally described below.

1. The receiver samples $m_{1-b,i} \in \mathcal{M}$ for all $i \in [t']$, where b is the choice bit. Then he runs the prover of the C&O protocol upon input $(m_{1-b,j})_{j\in[i]}$ for all $i \in [t']$, obtaining $(\gamma_i)_{i\in[t']}$ which are forwarded to the sender.

Sender $\mathsf{S}(s_0, s_1)$		**Receiver** $\mathsf{R}(b)$
$u_0, u_1 \leftarrow\$ \mathcal{M}$		$\alpha^0_{\mathsf{R},b} = 0$
$\alpha^0_{\mathsf{S},0} = (s_0, u_0)$		$\forall i \in [t']$:
$\alpha^0_{\mathsf{S},1} = (s_1, u_1)$		$m_{1-b,i} \leftarrow\$ \mathcal{M}$
$(\alpha^1_{\mathsf{S},0}, \sigma^1_0) \leftarrow\$ \mathsf{S}'(\alpha^0_{\mathsf{S},0})$		$(\gamma_i, \alpha_i) \leftarrow\$ \mathsf{P}_0((m_{1-b,j})_{j\in[i]})$
$(\alpha^1_{\mathsf{S},1}, \sigma^1_1) \leftarrow\$ \mathsf{S}'(\alpha^0_{\mathsf{S},1})$	$\xleftarrow{(\gamma_i)_{i\in[t']}}$	
$\beta_1 \leftarrow\$ \mathsf{V}_0(1^\lambda)$		
$r_{0,1}, r_{1,1} \leftarrow\$ \mathcal{M}$	$\xrightarrow{(\beta_1, (r_{k,1}, \sigma^1_k)_{k\in\{0,1\}})}$	
......Repeat for each $i \in [t']$......		
		$(\alpha^i_{\mathsf{R},b}, \rho^i_b) \leftarrow\$ \mathsf{R}'(\alpha^{i-1}_{\mathsf{R},b}, \sigma^i_b)$
		$m_{b,i} = \rho^i_b - r_{b,i}$
if $\mathsf{V}(\gamma_i, \beta_i, (\delta_i, (m_{0,j})_{j\in[i]}, (m_{1,j})_{j\in[i]})) = 0$	$\xleftarrow{(\delta_i, m_{0,i}, m_{1,i})}$	$\delta_i \leftarrow\$ \mathsf{P}_1(\alpha_i, \beta_i, \gamma_i, (m_{b,j})_{j\in[i]})$
return $\perp$		
$(\alpha^{i+1}_{\mathsf{S},0}, \sigma^{i+1}_0) \leftarrow\$ \mathsf{S}'(\alpha^i_{\mathsf{S},0}, m_{0,i} + r_{0,i})$		
$(\alpha^{i+1}_{\mathsf{S},1}, \sigma^{i+1}_1) \leftarrow\$ \mathsf{S}'(\alpha^i_{\mathsf{S},1}, m_{1,i} + r_{1,i})$		
$\beta_{i+1} \leftarrow\$ \mathsf{V}_0(1^\lambda)$		
$r_{0,i+1}, r_{1,i+1} \leftarrow\$ \mathcal{M}$	$\xrightarrow{(\beta_{i+1}, (r_{k,i+1}, \sigma^{i+1}_k)_{k\in\{0,1\}})}$	
....		
		$(\alpha^{t'+1}_{\mathsf{R},b}, \rho^{t'+1}_b) \leftarrow\$ \mathsf{R}'(\alpha^{t'}_{\mathsf{R},b}, \sigma^{t'+1}_b)$
		output $s_b = \rho^{t'+1}_b$

Figure 3: $(2t'+2)$-round OT protocol achieving receiver-sided simulatability from $(2t'+1)$-round strongly uniform semi-honestly secure OT. Note that the initial state information $\alpha^0_{\mathsf{S},0}, \alpha^0_{\mathsf{S},1}$ and $\alpha^0_{\mathsf{R},b}$ is set to be equal, respectively to the inputs used by the sender and the receiver during the runs of the underlying OT protocol $(\mathsf{S}', \mathsf{R}')$. The values $\beta_{t'+1}, r_{0,t'+1}, r_{1,t'+1}$ are not needed and can be removed, but we avoided to do that in order to keep the protocol description more compact.

2. The sender samples uniform values $u_0, u_1 \leftarrow\$ \mathcal{M}$. Then, he runs the underlying $(2t'+1)$-round OT twice with inputs (s_0, u_0) and (s_1, u_1) to generate the first messages σ^1_0 and σ^1_1. Further, the sender samples a challenge β_1 for the C&O protocol, as well as two uniformly random group elements $r_{0,1}$, $r_{1,1}$ from $\mathcal{M}$, and forwards $(\beta_1, r_{0,1}, r_{1,1})$ to the receiver together with the first messages of the OTs (i.e. σ^1_0 and σ^1_1).

3. Repeat the following steps for each $i \in [t']$:

   (a) $(\mathsf{R} \to \mathsf{S})$: The receiver runs the receiver R' of the underlying $(2t'+1)$-round OT protocol with choice bit fixed to 0, and upon input message σ^i_b from the sender, obtaining a message ρ^i_b which is used to define the message $m_{b,i} = \rho^i_b - r_{b,i}$ required to complete the execution of the C&O protocol in the non-committing branch b. This results in a tuple $(\delta_i, m_{0,i}, m_{1,i})$ that is forwarded to the sender.

(b) $(\mathsf{S} \rightarrow \mathsf{R})$: The sender verifies that the transcript $T_i = (\gamma_i, \beta_i, (\delta_i, (m_{0,j})_{j\in[i]}, (m_{1,j})_{j\in[i]}))$ is accepting for the underlying C&O protocol. If so, he continues the two runs of the sender S' for the underlying $(2t'+1)$-round OT protocol. The first run uses state $\alpha^i_{\mathsf{S},0}$ and message $m_{0,i} + r_{0,i}$ from the receiver resulting in a message σ_0^{i+1} and state $\alpha^{i+1}_{\mathsf{S},0}$, whereas the second run uses state $\alpha^i_{\mathsf{S},1}$ and message $m_{1,i} + r_{1,i}$ from the receiver resulting in a message σ_1^{i+1} and state $\alpha^{i+1}_{\mathsf{S},1}$. Finally, the sender samples a challenge β_{i+1} for the C&O protocol, as well as another two uniformly random group elements $r_{0,i+1}$, $r_{1,i+1}$ from $\mathcal{M}$, and forwards $(\sigma_0^{i+1}, \sigma_1^{i+1})$ and β_{i+1}, $r_{0,i+1}$, $r_{1,i+1}$ to the receiver.

4. Output: The receiver runs the receiver R' of the underlying $(2t'+1)$-round OT protocol, upon input the $(t'+1)$-th message $\sigma_b^{t'+1}$ from the sender, thus obtaining an output $\rho_b^{t'+1}$.

Correctness follows by the fact that, when both the sender and the receiver are honest, by correctness of the C&O protocol the transcripts T_i are always accepting, and moreover the messages produced by the sender σ_b^i are computed using message $m_{b,i} + r_{b,i} = \rho_b^i$ from the receiver, so that each pair (ρ_b^i, σ_b^i) corresponds to the i-th interaction of the underlying $(2t'+1)$-round OT protocol with input strings (s_b, u_b) for the sender and choice bit 0 for the receiver, and thus at the end the receiver outputs s_b. As for security, we have:

Theorem 3 (Receiver-sided simulatability of Π)**.** *Assuming that Π' is a $(2t'+1)$-round strongly uniform semi-honestly secure OT protocol, and that $\Pi_{\mathsf{c\&o}}$ is a secure 1-out-of-2 commit-and-open protocol, then the protocol Π from Fig. 3 securely realizes $\mathcal{F}_{\mathtt{OT}}$ with receiver-sided simulation.*

4.2 Proof Intuition

We give a detailed proof in §4.3, and here provide some intuition. In order to show receiver-sided simulatability we need to prove two things: (1) The existence of a simulator Sim which by interacting with the ideal functionality $\mathcal{F}_{\mathtt{OT}}$ can fake the view of any efficient adversary corrupting the receiver in a real execution of the protocol; (2) Indistinguishability of the protocol transcripts with choice bit of the receiver equal to zero or one, for any efficient adversary corrupting the sender in a real execution of the protocol.

To show (1), we consider a series of hybrid experiments that naturally lead to the definition of a simulator in the ideal world. In order to facilitate the description of the hybrids, it will be useful to think of the protocol as a sequence of t' iterations, where each iteration consists of 2 rounds, as depicted in Fig. 3 on the previous page.

- In the first hybrid, we run a malicious receiver twice after he has sent his commitments. The purpose of the first run is to learn a malicious receiver's input bit, i.e. on which branch he is not committed. If he is committed on both branches, simulation will be easy since he will not be able to receive any of the sender's inputs. We use the second run to learn the output of a malicious receiver. We describe the two runs now.

  1. The first round of each iteration yields an opening $(\delta_i, m_{0,i}, m_{1,i})$. Hence, after verifying that the opening is valid, we rewind the adversary to the end of the first round of the i-th iteration to receive another opening $(\delta'_i, m'_{0,i}, m'_{1,i})$.
     Now, let $b \in \{0,1\}$ such that $m_{b,i} \neq m'_{b,i}$. By the security of the C&O protocol, there can be at most one such b. If there is no b we continue the first run. Otherwise, if there is such a b, we have learned the equivocal branch and start the second run.

2. We execute the second run according to the protcol with the difference that we now know the equivocal branch, i.e. b, from the very beginning, which will help us later to simulate correctly right from the start. Notice that by the security of the C&O protocol, a malicious receiver cannot change the equivocal branch in the second run. Obviously, he cannot change it during the same iteration since then he would be equivocal on both branches and contradict the security of the C&O protocol. He can also not change the equivocal branch of one of the later rounds $j > i$, since in the j-th commitment δ_j he cannot be committed to both $m_{b,i}$ and $m'_{b,i}$, so he needs to equivocally open δ_j as well. Thus, he needs to be committed on the other branch, i.e. branch $1 - b$.

- The values $m'_{k,i}$ (right after the rewinding) of each iteration of the first run for $k \in \{0, 1\}$, and second run for $k = 1 - b$, are identical to $m_{k,i}$. Moreover, $m'_{k,i} \neq m_{k,i}$ holds only for the second run for branch $k = b$. Therefore, in the second hybrid, we can change the distribution of $r'_{k,i}$ to $r'_{k,i} = \rho^i_k - m_{k,i}$ for $k \in \{0, 1\}$, and both runs except branch $k = b$ during the second run. The value ρ^i_k is obtained by running the simulator for the receiver of the underlying strongly uniform semi-honest OT protocol with choice bit 1 and input u_k. We can use the messages generated by this simulator on the sender's side as well.

  We will use the strong uniformity of the OT to argue that a malicious receiver cannot distinguish $r'_{k,i} = \rho^i_k - m_{k,i}$ from uniform. By the semi-honest security, the messages generated by the simulator are indistinguishable from the actual semi-honest OT. At the same time this simulator is independent of the sender's inputs s_0 and s_1. Note that in this hybrid, we only need to known s_b for the second run after having learned b.

In the last hybrid, a protocol transcript is independent of s_{1-b} but still yields a well distributed output for the malicious receiver, which directly yields a simulator in the ideal world.

To show (2), we first use the strong uniformity of the underlying OT protocol to sample $m_{b,i}$ uniformly at random at the beginning of the protocol. Notice that this implies that the receiver cannot recover the value s_b of the sender anymore. Further, we need the strong uniformity property here, since the receiver is interacting with a malicious sender who could influence the distribution of $m_{b,i}$ sent by the receiver. Once both messages, $m_{0,i}$ and $m_{1,i}$ for all iterations are known before the start of the protocol, we can challenge the choice bit indistinguishability of the C&O protocol. As a consequence, we can argue that the transcripts with $b = 0$ and $b = 1$ are computationally indistinguishable, which implies game-based security against a malicious sender.

4.3 Security Analysis

4.3.1 Simulatability Against a Malicious Receiver

We need to prove that for all non-uniform PPT malicious receivers R^*, there exists a PPT simulator Sim such that

$$\left\{REAL_{\Pi,\mathsf{R}^*(z)}(\lambda, s_0, s_1, b)\right\}_{\lambda,s_0,s_1,b,z} \approx_c \left\{IDEAL_{\mathcal{F}_{\mathtt{OT}},\mathsf{Sim}^{\mathsf{R}^*(z)}}(\lambda, s_0, s_1, b)\right\}_{\lambda,s_0,s_1,b,z}$$

where $\lambda \in \mathbb{N}$, $s_0, s_1 \in \{0, 1\}^*$, $b \in \{0, 1\}$, and $z \in \{0, 1\}^*$.

To this end, we introduce several hybrid experiments naturally leading to the definition of an efficient simulator in the ideal world. Let $HYB_0(\lambda)$ be the real world experiment with a malicious receiver R^*. (All experiments are further parameterized by the inputs s_0, s_1 for the sender, but we omit to explicitly write this for simplicity.)

First hybrid. Hybrid $HYB_1(\lambda)$ proceeds as follows.

1. The sender picks $u_0, u_1 \leftarrow\$ \mathcal{M}$ and lets $\tilde{\alpha}^0_{\mathsf{S},0} = \alpha^0_{\mathsf{S},0} = (s_0, u_0)$, $\tilde{\alpha}^0_{\mathsf{S},1} = \alpha^0_{\mathsf{S},1} = (s_1, u_1)$, and $b, b', b'' = \bot$.
2. R^* forwards $(\gamma_i)_{i\in[t']}$, to which the sender replies with $(\beta_1, r_{0,1}, r_{1,1}, \tilde{\sigma}^1_0, \tilde{\sigma}^1_1)$, where $(\tilde{\alpha}^1_{\mathsf{S},0}, \tilde{\sigma}^1_0) \leftarrow\$ \mathsf{S}'(1^\lambda, \tilde{\alpha}^0_{\mathsf{S},0})$, $(\tilde{\alpha}^1_{\mathsf{S},1}, \tilde{\sigma}^1_1) \leftarrow\$ \mathsf{S}'(1^\lambda, \tilde{\alpha}^0_{\mathsf{S},1})$.
3. Repeat the steps below, for each $i \in [t']$:
   (a) R^* sends a tuple $(\delta_i, m_{0,i}, m_{1,i})$. Let $T_i = (\gamma_i, \beta_i, (\delta_i, (m_{0,j})_{j\in[i]}, (m_{1,j})_{j\in[i]}))$. Hence:
      i. If $\mathsf{V}_1(T_i) = 0$, restart the experiment with fresh randomness for R^*. Since the protocol is correct with non-negligible probability, it will only take polynomial time to find a run where R^* never gets restarted within this step in any iteration.
      ii. Rewind R^* at the beginning of the current iteration, and send a freshly sampled tuple $(\beta'_i, r'_{0,i}, r'_{1,i})$ with the same distribution as before.
   (b) R^* replies with $(\delta'_i, m'_{0,i}, m'_{1,i})$. Let $T'_i = (\gamma_i, \beta'_i, (\delta'_i, (m'_{0,j})_{j\in[i]}, (m'_{1,j})_{j\in[i]}))$. Hence:
      i. If $\mathsf{V}_1(T'_i) = 0$, we restart R^* as in step 3(a)i (again this can be done in polynomial time). If $\mathsf{V}_1(T'_i) = 1$ and on both branches $(m'_{0,j})_{j\in[i]} \neq (m_{0,j})_{j\in[i]}$ *and* $(m'_{1,j})_{j\in[i]} \neq (m_{1,j})_{j\in[i]}$, the sender aborts.
      ii. Attempt to define b' as the binary value for which $(m'_{b',j})_{j\in[i]} \neq (m_{b',j})_{j\in[i]}$, but $(m'_{1-b',j})_{j\in[i]} = (m_{1-b',j})_{j\in[i]}$. If such value is found, halt and go directly to step 4 after setting $b \overset{\text{def}}{=} b'$.
   (c) The sender computes $(\tilde{\alpha}^{i+1}_{\mathsf{S},0}, \tilde{\sigma}^{i+1}_0) \leftarrow\$ \mathsf{S}'(\tilde{\alpha}^i_{\mathsf{S},0}, m'_{0,i}+r'_{0,i})$, $(\tilde{\alpha}^{i+1}_{\mathsf{S},1}, \tilde{\sigma}^{i+1}_1) \leftarrow\$ \mathsf{S}'(\tilde{\alpha}^i_{\mathsf{S},1}, m'_{1,i}+ r'_{1,i})$, samples $(\beta_{i+1}, r_{0,i+1}, r_{1,i+1})$ as in the original protocol, and forwards $(\tilde{\sigma}^{i+1}_0, \tilde{\sigma}^{i+1}_1, \beta_{i+1}, r_{0,i+1}, r_{1,i+1})$ to R^*.
4. Rewind R^* to step 2, and re-start running the experiment from there with the following differences applied to each iteration $i \in [t']$:
   (a) Denote by $(\beta''_i, r''_{0,i}, r''_{1,i})$ the new challenges sent to R^* in step 3(a)ii, and with $(\delta''_i, m''_{0,i}, m''_{1,i})$ the corresponding answer computed by R^* in step 3b. Also let $T''_i = (\gamma_i, \beta''_i, (\delta''_i, (m''_{0,j})_{j\in[i]}, (m''_{1,j})_{j\in[i]}))$.
   (b) If either $\mathsf{V}_1(T''_i) = 0$, or $\mathsf{V}_1(T''_i) = 1$ and on both branches $(m''_{0,j})_{j\in[i]} \neq (m_{0,j})_{j\in[i]}$ *and* $(m''_{1,j})_{j\in[i]} \neq (m_{1,j})_{j\in[i]}$, the sender aborts.
   (c) Attempt to define b'' as the binary value for which $(m_{b'',j})_{j\in[i]} \neq (m''_{b'',j})_{j\in[i]}$, but $(m_{1-b'',j})_{j\in[i]} = (m''_{1-b'',j})_{j\in[i]}$. If such value is found, but $b'' \neq b$ the sender aborts.
   (d) The sender aborts if $b'' \neq \bot$, but $(m''_{b'',j})_{j\in[i]} = (m_{b'',j})_{j\in[i]}$.
   (e) The sender computes $(\alpha^{i+1}_{\mathsf{S},0}, \sigma^{i+1}_0) \leftarrow\$ \mathsf{S}'(\alpha^i_{\mathsf{S},0}, m'_{0,i}+r'_{0,i})$, $(\alpha^{i+1}_{\mathsf{S},1}, \sigma^{i+1}_1) \leftarrow\$ \mathsf{S}'(\alpha^i_{\mathsf{S},1}, m'_{1,i}+ r'_{1,i})$ samples $(\beta_{i+1}, r_{0,i+1}, r_{1,i+1})$ as in the original protocol, and forwards $(\sigma^{i+1}_0, \sigma^{i+1}_1, \beta_{i+1}, r_{0,i+1}, r_{1,i+1})$ to R^*.
5. Experiment output: The output of R^*.

Lemma 5. $\{HYB_0(\lambda)\}_{\lambda\in\mathbb{N}} \approx_c \{HYB_1(\lambda)\}_{\lambda\in\mathbb{N}}$.

Proof. First notice that all the restarts and rewindings of R^* do not change R^*'s output distribution, they only decrease the probability of a protocol abort at the cost of a polynomial increase in the running time.

For $i \in [t']$, consider the following events defined over the probability space of $HYB_1(\lambda)$.

Event $W^i_{1,1}$: The event becomes true if the sender aborts during step 3(b)i, i.e. the values $(\delta_i, m_{0,i}, m_{1,i})$ and $(\delta'_i, m'_{0,i}, m'_{1,i})$ output by R^* are such that there is no $\hat{b} \in \{0,1\}$ for which $(m_{\hat{b},j})_{j\in[i]} = (m'_{\hat{b},j})_{j\in[i]}$, and furthermore both transcripts T_i and T'_i are valid transcripts for the underlying commit-and-prove protocol.

Event $W^i_{1,2}$: The event becomes true if the sender aborts during step 4b, i.e. the values $(\delta_i, m_{0,i}, m_{1,i})$ and $(\delta''_i, m''_{0,i}, m''_{1,i})$ output by R^* are such that there is no $\hat{b} \in \{0,1\}$ for which $(m_{\hat{b},j})_{j\in[i]} = (m''_{\hat{b},j})_{j\in[i]}$, and furthermore both transcripts T_i and T''_i are valid transcripts for the underlying commit-and-prove protocol.

Event $W^i_{1,3}$: The event becomes true if the sender aborts during step 4c, i.e., the non-committing branches b' and b'' are different for the two runs of the adversary (after rewinding).

Event $W^i_{1,4}$: The event becomes true if the sender aborts during step 4d, i.e., the value b'' was set in some previous iteration $k < i$, meaning that $(m_{b'',j})_{j\in[k]} \neq (m''_{b'',j})_{j\in[k]}$, but during the i-th iteration the same branch becomes again committing, meaning that $(m_{b'',j})_{j\in[i]} = (m''_{b'',j})_{j\in[i]}$.

Define $W^i_1 \stackrel{\text{def}}{=} W^i_{1,1} \vee W^i_{1,2} \vee W^i_{1,3} \vee W^i_{1,4}$. For all PPT distinguishers D, by a union bound, we can write

$$\Delta_{\mathsf{D}}(HYB_0(\lambda); HYB_1(\lambda)) \leq \Pr[\exists i \in [t'] : \; W^i_1] \leq \sum_{i=1}^{t'} \sum_{j=1}^{4} \Pr[W^i_{1,j}],$$

and thus it suffices to prove that each of the events happens with negligible probability for all $i \in [t']$. We show this fact below, which concludes the proof of the lemma.

Claim 4. *For all PPT R^*, and for all $i \in [t']$, we have that* $\Pr[W^i_{1,1}] \in \mathsf{negl}(\lambda)$.

Proof. The proof is down to the property of existence of a committing branch for the commit-and-prove protocol. By contradiction, assume that there is a pair $s_0, s_1 \in \{0,1\}^\lambda$, some $i \in [t']$, a non-uniform PPT adversary R^*, and an auxiliary input $z \in \{0,1\}^*$, such that $\mathsf{R}^*(z)$ provokes event $W^i_{1,1}$ in an execution of $HYB_1(\lambda)$ with non-negligible probability. We build a non-uniform PPT adversary P^* that, given $i \in [t']$, attacks the security of $\Pi_{\mathsf{c\&o}}$ as follows:[9]

1. Run $\mathsf{R}^*(z)$, and after receiving $(\gamma_i)_{i\in[t']}$, forward γ_i to the challenger, thus obtaining a challenge β.
2. Emulate a run of experiment $HYB_1(\lambda)$ with R^*, except that the value β_i is defined by embedding the value β received from the challenger.
3. Upon receiving $(\delta_i, m_{0,i}, m_{1,i})$ from R^*, check that $T_i = (\gamma_i, \beta_i, (\delta_i, (m_{0,j})_{j\in[i]}, (m_{1,j})_{j\in[i]}))$ is a valid transcript; if so, forward $(\delta_i, (m_{0,j})_{j\in[i]}, (m_{1,j})_{j\in[i]})$ to the challenger.
4. Upon receiving a fresh challenge β' for the commit-and-prove protocol from the challenger, rewind R^* as described in $HYB_1(\lambda)$, except that the value β'_i is defined by embedding the value β' received from the challenger.
5. Upon receiving $(\gamma'_i, m'_{0,i}, m'_{1,i})$ from R^*, check that $T'_i = (\gamma_i, \beta'_i, (\delta'_i, (m'_{0,j})_{j\in[i]}, (m'_{1,j})_{j\in[i]}))$ is a valid transcript; if so, forward $(\gamma'_i, (m'_{0,j})_{j\in[i]}, (m'_{1,j})_{j\in[i]})$ to the challenger.
6. Complete the remaining steps of the protocol with R^*, as described in $HYB_1(\lambda)$.

[9] We can also make the reduction uniform, at the cost of losing a polynomial factor in the computational distance between the two hybrids (which is needed to guess the index i for which event $W^i_{1,1}$ is provoked).

Notice that the above simulation is perfect; this is because the values (β, β') that the reduction embeds during the i-th iteration have exactly the same distribution as in an execution of experiment $HYB_1(\lambda)$, whereas all other iterations are perfectly distributed as in $HYB_1(\lambda)$. It follows that adversary R^* will provoke event $W^i_{1,1}$ with non-negligible probability, which means that both the transcripts T_i and T'_i are accepting, and moreover $(m_{0,j})_{j\in[i]} \neq (m'_{0,j})_{j\in[i]}$ and $(m_{1,j})_{j\in[i]} \neq (m'_{1,j})_{j\in[i]}$. Thus, P^* wins with non-negligible probability, which concludes the proof of the claim. □

Claim 5. *For all PPT* R^**, and for all* $i \in [t']$*, we have that* $\Pr[W^i_{1,2}] \in \mathsf{negl}(\lambda)$.

Proof. The proof is similar to the one of the previous claim, and therefore omitted. The only difference is that the challenge β' is now embedded by the reduction in β''_i, and also the tuple $(\gamma''_i, (m''_{0,j})_{j\in[i]}, (m''_{1,j})_{j\in[i]})$ is sent to the challenger after the rewinding. □

Claim 6. *For all PPT* R^**, and for all* $i \in [t']$*, we have that* $\Pr[W^i_{1,3}] \in \mathsf{negl}(\lambda)$.

Proof. Without loss of generality, assume that $b' = 0$ and $b'' = 1$. Notice that event $W^i_{1,3}$ means that both transcripts T_i and T''_i are accepting for the commit-and-prove protocol, and additionally $(m_{0,j})_{j\in[i]} \neq (m'_{0,j})_{j\in[i]}$, whereas $(m_{1,j})_{j\in[i]} \neq (m''_{0,j})_{j\in[i]}$. The latter contradicts the property of existence of a committing branch for the commit-and-prove protocol. The formal reduction is similar to the one given above, and is therefore omitted. □

Claim 7. *For all PPT* R^**, and for all* $i \in [t']$*, we have that* $\Pr[W^i_{1,4}] \in \mathsf{negl}(\lambda)$.

Proof. Notice that event $W^i_{1,4}$ means that, for some iteration $k < i$, both transcripts $T_k = (\gamma_k, \beta_k, (\delta_k, (m_{0,j})_{j\in[k]}, (m_{1,j})_{j\in[k]}))$ and $T''_k = (\gamma_k, \beta''_k, (\delta''_k, (m''_{0,j})_{j\in[k]}, (m''_{1,j})_{j\in[k]}))$ are accepting for the commit-and-prove protocol, and additionally there exists a value $b \in \{0,1\}$ such that branch b is non-committing, which means $(m_{1-b,j})_{j\in[k]} = (m''_{1-b,j})_{j\in[k]}$. However, during the i-th iteration, both transcripts T_i and T''_i are accepting for the commit-and-prove protocol, but branch b becomes committing again. The latter implies that there exist accepting transcripts T_k and T''_k for which both $(m_{1-b,j})_{j\in[i]} = (m''_{1-b,j})_{j\in[k]}$ and $(m_{b,j})_{j\in[k]} = (m''_{b,j})_{j\in[k]}$, which contradicts the property of existence of a committing branch for the commit-and-prove protocol. The formal reduction is similar to the one given above, and is therefore omitted. □

□

Second hybrid. Hybrid $HYB_2(\lambda)$ proceeds identically to $HYB_1(\lambda)$, except for the following differences.

1. In step 1, the sender additionally sets $\tilde{\alpha}^0_{\mathsf{R}',0} = 1$.
2. The distribution of the values $r'_{0,i}$ computed during step 3(a)ii is changed by evaluating $(\tilde{\alpha}^i_{\mathsf{R}',0}, \tilde{\rho}^i_0) \leftarrow\!\!\$\ \mathsf{R}'(\tilde{\alpha}^{i-1}_{\mathsf{R}',0}, \tilde{\sigma}^i_0)$, and by letting $r'_{0,i} = \tilde{\rho}^i_0 - m_{0,i}$.

Notice that the latter change is applied only to the first run of R^* (i.e., up to the point where the value b' is set). This means that the distribution of the values $(r''_{0,i})_{i\in[t']}$ is not modified.

Lemma 6. $\{HYB_1(\lambda)\}_{\lambda\in\mathbb{N}} \approx_c \{HYB_2(\lambda)\}_{\lambda\in\mathbb{N}}$.

Proof. Let W_2^i be the same event as W_1^i, but over the probability space of $HYB_2(\lambda)$. For all PPT distinguishers D, we can write:

$$\Delta_{\mathsf{D}}(HYB_1(\lambda); HYB_2(\lambda)) \leq \Delta_{\mathsf{D}}(HYB_1(\lambda); HYB_2(\lambda) | \forall i \in [t'] : \neg W_2^i) + \Pr[\exists i \in [t'] : W_2].$$

An argument similar to that used in the proof of Lemma 5 shows that $\Pr[\exists i \in [t'] : W_2^i]$ is negligible, hence it suffices to prove that $\Delta_{\mathsf{D}}(HYB_1(\lambda); HYB_2(\lambda) | \forall i \in [t'] : \neg W_2^i)$ is also negligible. Note that the only difference between the two experiments comes from the distribution of the messages $(r'_{0,j})_{j \in [i^*]}$, with $i^* \leq t'$ being the index corresponding to the round (if any) where the bit b' is set during a run of the protocol: In experiment $HYB_1(\lambda)$ these values are uniformly random, whereas in experiment $HYB_2(\lambda)$ they are set to $\tilde{\rho}_0^j - m_{0,j}$, where $\tilde{\rho}_0^j$ is generated by a fresh run of the receiver for Π' with choice bit fixed to one.

By contradiction, assume that there exists a pair of values $s_0, s_1 \in \{0,1\}^\lambda$, and a non-uniform PPT distinguisher D, such that D can tell apart $HYB_1(\lambda)$ and $HYB_2(\lambda)$ with non-negligible probability. We use D to construct a PPT distinguisher $\hat{\mathsf{D}}$ attacking the uniformity property (cf. property (b) in Definition 7) of protocol Π'. Actually, for this particular step of the proof we only need a weaker property where the distinguisher $\hat{\mathsf{D}}$ is honest but curious. The reduction works as follows:

1. Forward $\hat{b} = 1$, $\hat{s}_0 = s_0$, and uniform $\hat{s}_1 = u_0$ to the challenger.
2. Receive a challenge $((\hat{\rho}^i, \hat{\sigma}^i)_{i \in [t']}, \hat{\sigma}^{t'+1})$ from the challenger.
3. Run experiment $HYB_2(\lambda)$ with D, except that the changes below are applied to each iteration of the first run of the distinguisher:
   (a) During step 3(a)ii, the value $r'_{0,i}$ is set to be $r'_{0,i} = \hat{\rho}^i - m'_{0,i}$, whereas $r'_{1,i}$ is chosen uniformly at random in $\mathcal{M}$.
   (b) During step 3c, the value $\tilde{\sigma}_0^{i+1}$ is defined by embedding the value $\hat{\sigma}^{i+1}$ from the challenge.
4. Output the same as D(output of R^*).

By inspection, depending on each pair $(\hat{\rho}^i, \hat{\sigma}^i)$ being distributed either as in a honest execution of protocol Π' between $\mathsf{S}'(s_0, u_0)$ and $\mathsf{R}'(1)$, or as in an interaction between $\mathsf{S}'(s_0, u_0)$ and using uniformly random group elements for the messages of the receiver, the distribution generated by the reduction is identical either to that of $HYB_1(\lambda)$ or to that of $HYB_2(\lambda)$. The latter in particular holds since we are conditioning on the event W_2^i not happening for all $i \in [t']$, which means that in $HYB_2(\lambda)$ the values $\tilde{\sigma}_0^{i+1}$ are computed by running the honest sender $\mathsf{S}'(s_0, u_0)$ upon input $r'_{0,i} + m_{0,i} = (\tilde{\rho}_0^i - m_{0,i}) + m_{0,i} = \tilde{\rho}_0^i$.

It follows that $\hat{\mathsf{D}}$ makes a perfect simulation, and thus it retains the same distinguishing advantage as that of D, which concludes the proof of the lemma. □

Third hybrid. Hybrid $HYB_3(\lambda)$ proceeds identically to $HYB_2(\lambda)$, except for the following differences.

1. In step 1, the sender additionally sets $\tilde{\alpha}_{\mathsf{Sim}',0}^0 = (1, u_0)$ and defines $\tilde{\sigma}_0^1$ as $(\tilde{\alpha}_{\mathsf{Sim}',0}^1, \tilde{\sigma}_0^1) \leftarrow\!\!\$\ \mathsf{Sim}'_{\mathsf{R}'}(1^\lambda, \tilde{\alpha}_{\mathsf{Sim}',0}^0)$.
2. The distribution of the values $\tilde{\rho}_0^i$ defined during step 3(a)ii, and of the values $\tilde{\sigma}_0^i$ defined during step 3c is changed by evaluating $(\tilde{\alpha}_{\mathsf{Sim}',0}^{i+1}, \tilde{\rho}_0^i, \tilde{\sigma}_0^{i+1}) \leftarrow\!\!\$\ \mathsf{Sim}'_{\mathsf{R}'}(\tilde{\alpha}_{\mathsf{Sim}',0}^1)$.

Notice that the latter change is applied only to the first run of R^* (i.e., up to the point where the value b' is set). This means that the distribution of the values $(\rho_0^i, \sigma_0^i)_{i\in[t']}$ is not modified.

Lemma 7. *$HYB_2(\lambda) \approx_c HYB_3(\lambda)$.*

Proof. Let W_3^i be the same event as W_2^i, but over the probability space of $HYB_3(\lambda)$. For all PPT distinguishers D, we can write:

$$\Delta_{\mathsf{D}}(HYB_2(\lambda); HYB_3(\lambda)) \leq \Delta_{\mathsf{D}}(HYB_2(\lambda); HYB_3(\lambda)|\forall i \in [t'] : \neg W_3^i) + \Pr[\exists i \in [t'] : W_3].$$

An argument similar to that used in the proof of Lemma 5 shows that $\Pr[\exists i \in [t'] : W_3^i]$ is negligible, hence it suffices to prove that $\Delta_{\mathsf{D}}(HYB_2(\lambda); HYB_3(\lambda)|\forall i \in [t'] : \neg W_3^i)$ is also negligible. Note that the only difference between the two experiments comes from the distribution of the values $\tilde{\sigma}_0^{t'+1}$,$(\tilde{\rho}_0^j, \tilde{\sigma}_0^j)_{j\in[i^*]}$, with $i^* \leq t'$ being the index corresponding to the round (if any) where the bit b' is set during a run of the protocol: In experiment $HYB_2(\lambda)$ these values are generated through a honest execution of protocol Π' between receiver R' with choice bit fixed to 1 and sender S' with inputs (s_0, u_0), whereas in experiment $HYB_3(\lambda)$ they are generated by running the simulator $\mathsf{Sim}'_{\mathsf{R}'}$.

The proof is down to the security of the underlying $(2t'+1)$-round OT protocol $\Pi' = (\mathsf{S}', \mathsf{R}')$ w.r.t. semi-honest receivers (cf. property (a) of Definition 7). By contradiction, assume that there exists a pair of inputs $s_0, s_1 \in \{0,1\}^\lambda$, and a non-uniform PPT distinguisher D, such that D can tell apart $HYB_2(\lambda)$ and $HYB_3(\lambda)$ with non-negligible probability. We construct a PPT distinguisher $\hat{\mathsf{D}}$ that given (s_0, s_1) attacks semi-honest security of Π' as follows:

1. Forward $\hat{b} = 1$, $\hat{s}_0 = s_0$, and $\hat{s}_1 = s_1$ to the challenger.
2. Receive a challenge $(\hat{\rho}^i, \hat{\sigma}^i)_{i\in[t']}, \hat{\sigma}^{t'+1}$ from the challenger.
3. Run experiment $HYB_3(\lambda)$ with D, except that the changes below are applied to each iteration of the first run of the distinguisher:
   (a) During step 3(a)ii, the value $r'_{0,i}$ is set to be $r'_{0,i} = \hat{\rho}^i - m'_{0,i}$, whereas $r'_{1,i}$ is chosen uniformly at random in $\mathcal{M}$.
   (b) During step 3c, the value $\tilde{\sigma}_0^{i+1}$ is defined by embedding the value $\hat{\sigma}^{i+1}$ from the challenge.
4. Output the same as D(output of R^*).

By inspection, depending on each pair $(\hat{\rho}^i, \hat{\sigma}^i)$ being distributed either as in a honest execution of protocol Π' between $\mathsf{S}'(s_0, u_0)$ and $\mathsf{R}'(1)$, or as computed by the simulator $\mathsf{Sim}'_{\mathsf{R}'}$ with inputs $(1^\lambda, 1, u_0)$, the distribution generated by the reduction is identical either to that of $HYB_2(\lambda)$ or to that of $HYB_3(\lambda)$. The latter in particular holds since we are conditioning on the event W_3^i not happening for all $i \in [t']$, which means that in $HYB_2(\lambda)$ the values $\tilde{\sigma}_0^i$ are computed by running the honest sender $\mathsf{S}'(s_0, u_0)$ upon input $r'_{0,i} + m_{0,i} = (\tilde{\rho}_0^i - m_{0,i}) + m_{0,i} = \tilde{\rho}_0^i$.

It follows that $\hat{\mathsf{D}}$ makes a perfect simulation, and thus it retains the same distinguishing advantage as that of D, which concludes the proof of the lemma. □

Fourth hybrid. Hybrid $HYB_4(\lambda)$ proceeds identically to $HYB_3(\lambda)$, except for the following differences.

1. In step 1, the sender additionally sets $\tilde{\alpha}^0_{\mathsf{R}',1} = 1$.

2. The distribution of the values $r'_{1,i}$ computed during step 3(a)ii is changed by evaluating $(\tilde{\alpha}^i_{\mathsf{R}',1}, \tilde{\rho}^i_1) \leftarrow\!\$\ \mathsf{R}'(\tilde{\alpha}^{i-1}_{\mathsf{R}',1}, \tilde{\sigma}^i_1)$, and by letting $r'_{1,i} = \tilde{\rho}^i_1 - m_{1,i}$.

Notice that the latter change is applied only to the first run of R^* (i.e., up to the point where the value b' is set). This means that the distribution of the values $(r''_{1,i})_{i\in[t']}$ is not modified. The proof of the lemma below is identical to the proof of Lemma 6, and is therefore omitted.

Lemma 8. $\{HYB_3(\lambda)\}_{\lambda\in\mathbb{N}} \approx_c \{HYB_4(\lambda)\}_{\lambda\in\mathbb{N}}$.

Fifth hybrid. Hybrid $HYB_5(\lambda)$ proceeds identically to $HYB_4(\lambda)$, except for the following differences.

1. In step 1, the sender additionally sets $\tilde{\alpha}^0_{\mathsf{Sim}',1} = (1, u_1)$ and defines $\tilde{\sigma}^1_1$ as $(\tilde{\alpha}^1_{\mathsf{Sim}',1}, \tilde{\sigma}^1_1) \leftarrow\!\$\ \mathsf{Sim}'_{\mathsf{R}'}(1^\lambda, \tilde{\alpha}^0_{\mathsf{Sim}',1})$

2. The distribution of the values $\tilde{\rho}^i_1$ defined during step 3(a)ii, and of the values $\tilde{\sigma}^i_1$ defined during step 3c is changed by evaluating $(\tilde{\alpha}^{i+1}_{\mathsf{Sim}',1}, \tilde{\rho}^i_1, \tilde{\sigma}^{i+1}_1) \leftarrow\!\$\ \mathsf{Sim}'_{\mathsf{R}'}(\tilde{\alpha}^i_{\mathsf{Sim}',1})$.

Notice that the latter change is applied only to the first run of R^* (i.e., up to the point where the value b' is set). This means that the distribution of the values $(\rho^i_1, \sigma^i_1)_{i\in[t']}$ is not modified. The proof of the lemma below is identical to that of Lemma 7, and is therefore omitted.

Lemma 9. $\{HYB_4(\lambda)\}_{\lambda\in\mathbb{N}} \approx_c \{HYB_5(\lambda)\}_{\lambda\in\mathbb{N}}$.

Sixth hybrid. Hybrid $HYB_6(\lambda)$ proceeds identically to $HYB_5(\lambda)$, except for the following differences.

1. In step 4, the sender additionally sets $\alpha^0_{\mathsf{R}',1-b} = 1$. If $b = \bot$, set both $\alpha^0_{\mathsf{R}',0} = \alpha^0_{\mathsf{R}',1} = 1$.

2. The distribution of the values $r''_{1-b,i}$ computed during step 4a is changed by evaluating $(\alpha^i_{\mathsf{R}',1-b}, \rho^i_{1-b}) \leftarrow\!\$\ \mathsf{R}'(\alpha^{i-1}_{\mathsf{R}',1-b}, \sigma^i_{1-b})$, and by letting $r'_{1-b,i} = \tilde{\rho}^i_{1-b} - m_{1-b,i}$. If $b = \bot$, such a change is applied on both branches.

The proof of the lemma below is identical to the proof of Lemma 6, and is therefore omitted.

Lemma 10. $\{HYB_5(\lambda)\}_{\lambda\in\mathbb{N}} \approx_c \{HYB_6(\lambda)\}_{\lambda\in\mathbb{N}}$.

Seventh hybrid. Hybrid $HYB_7(\lambda)$ proceeds identically to $HYB_6(\lambda)$, except for the following differences.

1. In step 4, the sender additionally sets $\alpha^0_{\mathsf{Sim}',1-b} = (1, u_{1-b})$ and defines σ^1_{1-b} as $(\alpha^1_{\mathsf{Sim}',1-b}, \sigma^1_{1-b}) \leftarrow\!\$\ \mathsf{Sim}'_{\mathsf{R}'}(1^\lambda, \alpha^0_{\mathsf{Sim}',1-b})$. If $b = \bot$, set both $\alpha^0_{\mathsf{Sim}',0} = (1, u_0)$, $\alpha^0_{\mathsf{Sim}',1} = (1, u_1)$ and generate σ^1_0, σ^1_1 as $(\alpha^1_{\mathsf{Sim}',0}, \sigma^1_0) \leftarrow\!\$\ \mathsf{Sim}'_{\mathsf{R}'}(1^\lambda, \alpha^0_{\mathsf{Sim}',0})$, $(\alpha^1_{\mathsf{Sim}',1}, \sigma^1_1) \leftarrow\!\$\ \mathsf{Sim}'_{\mathsf{R}'}(1^\lambda, \alpha^0_{\mathsf{Sim}',1})$

2. The distribution of the values ρ^i_{1-b} defined during step 4a, and of the values σ^{i+1}_{1-b} defined during step 4e is changed by evaluating $(\alpha^{i+1}_{\mathsf{Sim}',1-b}, \rho^i_{1-b}, \sigma^{i+1}_{1-b}) \leftarrow\!\$\ \mathsf{Sim}'_{\mathsf{R}'}(\alpha^i_{\mathsf{Sim}',1-b})$. If $b = \bot$, such changes are applied on both branches.

The proof of the lemma below is identical to that of Lemma 7, and is therefore omitted.

Lemma 11. $\{HYB_6(\lambda)\}_{\lambda\in\mathbb{N}} \approx_c \{HYB_7(\lambda)\}_{\lambda\in\mathbb{N}}$.

Simulator. We are now ready to describe the simulator Sim, interacting with the ideal functionality $\mathcal{F}_{\mathtt{OT}}$. The simulator works as follows:

1. Pick $u_0, u_1 \leftarrow\!\$\, \mathcal{M}$, and let $\tilde{\alpha}^0_{\mathsf{Sim}',0} = (1, u_0)$, $\tilde{\alpha}^0_{\mathsf{Sim}',1} = (1, u_1)$, $(\tilde{\alpha}^1_{\mathsf{Sim}',0}, \tilde{\sigma}^1_0) \leftarrow\!\$\, \mathsf{Sim}'_{\mathsf{R}'}(1^\lambda, \tilde{\alpha}^0_{\mathsf{Sim}',0})$, $(\tilde{\alpha}^1_{\mathsf{Sim}',1}, \tilde{\sigma}^1_1) \leftarrow\!\$\, \mathsf{Sim}'_{\mathsf{R}'}(1^\lambda, \tilde{\alpha}^0_{\mathsf{Sim}',1})$, and $b, b', b'' = \perp$.

2. Upon receiving $(\gamma_i)_{i\in[t']}$ from R^*, sample $\beta_1 \leftarrow\!\$\, \mathsf{V}_0(1^\lambda)$, $r_{0,1}, r_{1,1} \leftarrow\!\$\, \mathcal{M}$, and send $(\beta_1, r_{0,1}, r_{1,1}, \tilde{\sigma}^1_0, \tilde{\sigma}^1_1)$ to R^*.

3. Repeat the steps below, for each $i \in [t']$:
   (a) Upon receiving a tuple $(\delta_i, m_{0,i}, m_{1,i})$ from R^*, let $T_i = (\gamma_i, \beta_i, (\delta_i, (m_{0,j})_{j\in[i]}, (m_{1,j})_{j\in[i]}))$. Hence:
      i. If $\mathsf{V}_1(T_i) = 0$, restart R^*.
      ii. Rewind R^* at the beginning of the current iteration, and send a tuple $(\beta'_i, r'_{0,i}, r'_{1,i})$ where $\beta'_i \leftarrow\!\$\, \mathsf{V}_0(1^\lambda)$, $r'_{0,i} = \tilde{\rho}^i_0 - m_{0,i}$ and $r'_{1,i} = \tilde{\rho}^i_1 - m_{1,i}$, for $(\tilde{\alpha}^{i+1}_{\mathsf{Sim}',0}, \tilde{\rho}^i_0, \tilde{\sigma}^{i+1}_0) \leftarrow\!\$\, \mathsf{Sim}'_{\mathsf{R}'}(\tilde{\alpha}^i_{\mathsf{Sim}',0})$ and for $(\tilde{\alpha}^{i+1}_{\mathsf{Sim}',1}, \tilde{\rho}^i_1, \tilde{\sigma}^{i+1}_1) \leftarrow\!\$\, \mathsf{Sim}'_{\mathsf{R}'}(\tilde{\alpha}^i_{\mathsf{Sim}',1})$.

   (b) Upon receiving a tuple $(\delta'_i, m'_{0,i}, m'_{1,i})$ from R^*, let $T'_i = (\gamma_i, \beta'_i, (\delta'_i, (m'_{0,j})_{j\in[i]}, (m'_{1,j})_{j\in[i]}))$. Hence:
      i. If $\mathsf{V}_1(T'_i) = 0$, restart R^*. If $\mathsf{V}_1(T'_i) = 1$ and on both branches $(m'_{0,j})_{j\in[i]} \neq (m_{0,j})_{j\in[i]}$ *and* $(m'_{1,j})_{j\in[i]} \neq (m_{1,j})_{j\in[i]}$, abort.
      ii. Attempt to define b' as the binary value for which $(m'_{b',j})_{j\in[i]} \neq (m_{b',j})_{j\in[i]}$, but $(m'_{1-b',j})_{j\in[i]} = (m_{1-b',j})_{j\in[i]}$. If such value is found, halt and go directly to step 4 after setting $b \stackrel{\text{def}}{=} b'$.

   (c) Forward $(\tilde{\sigma}^{i+1}_0, \tilde{\sigma}^{i+1}_1, \beta_{i+1}, r_{0,i+1}, r_{1,i+1})$ to R^*, where $\beta_{i+1} \leftarrow\!\$\, \mathsf{V}_0(1^\lambda)$, and $r_{0,i+1}, r_{1,i+1} \leftarrow\!\$\, \mathcal{M}$.

4. Query $\mathcal{F}_{\mathtt{OT}}$ upon input b, obtaining a value $s_b \in \{0,1\}^\lambda$.[10] Let $\alpha^0_{\mathsf{Sim}',1-b} = (1, u_{1-b})$, $\alpha^0_{\mathsf{S}',b} = (s_b, u_b)$ and define σ^1_0, σ^1_1 as $(\alpha^1_{\mathsf{Sim}',1-b}, \sigma^1_{1-b}) \leftarrow\!\$\, \mathsf{Sim}'_{\mathsf{R}'}(1^\lambda, \alpha^0_{\mathsf{Sim}',1-b})$,$(\alpha^1_{\mathsf{S}',b}, \sigma^1_b) \leftarrow\!\$\, \mathsf{S}'(1^\lambda, \alpha^0_{\mathsf{S}',b})$. Rewind R^* to step 2, sample $\beta_1 \leftarrow\!\$\, \mathsf{V}_0(1^\lambda)$, $r_{0,1}, r_{1,1} \leftarrow\!\$\, \mathcal{M}$, and send $(\beta_1, r_{0,1}, r_{1,1}, \sigma^1_0, \sigma^1_1)$ to R^*.

5. Repeat the steps below, for each $i \in [t']$:
   (a) Upon receiving a tuple $(\delta_i, m_{0,i}, m_{1,i})$ from R^*, let $T_i = (\gamma_i, \beta_i, (\delta_i, (m_{0,j})_{j\in[i]}, (m_{1,j})_{j\in[i]}))$. Hence:
      i. If $\mathsf{V}_1(T_i) = 0$, restart R^*.
      ii. Rewind R^* at the beginning of the current iteration, and send a tuple $(\beta''_i, r''_{0,i}, r''_{1,i})$ where $\beta''_i \leftarrow\!\$\, \mathsf{V}_0(1^\lambda)$, $r''_{1-b,i} = \rho^i_{1-b} - m_{1-b,i}$ and $r''_{b,i} \leftarrow\!\$\, \mathcal{M}$, for $(\alpha^{i+1}_{\mathsf{Sim}',1-b}, \rho^i_{1-b}, \sigma^{i+1}_{1-b}) \leftarrow\!\$\, \mathsf{Sim}'_{\mathsf{R}'}(\alpha^i_{\mathsf{Sim}',1-b})$.

   (b) Upon receiving a tuple $(\delta''_i, m''_{0,i}, m''_{1,i})$ from R^*, let $T''_i = (\gamma_i, \beta''_i, (\delta''_i, (m''_{0,j})_{j\in[i]}, (m''_{1,j})_{j\in[i]}))$. Hence:
      i. If either $\mathsf{V}_1(T''_i) = 0$, or $\mathsf{V}_1(T''_i) = 1$ and on both branches $(m''_{0,j})_{j\in[i]} \neq (m_{0,j})_{j\in[i]}$ *and* $(m''_{1,j})_{j\in[i]} \neq (m_{1,j})_{j\in[i]}$, abort.

[10] In case $b = \perp$, it is not necessary to query the ideal functionality. In fact, the latter means that in all iterations of the first run with the adversary, both branches for the C&O protocol are committing, and so they will be in the second run. Thus, the simulator can simply use the simulation strategy for the committing branch, which is independent of the sender's input, on both branches.

ii. Attempt to define b'' as the binary value for which $(m''_{b'',j})_{j\in[i]} \neq (m_{b'',j})_{j\in[i]}$, but $(m''_{1-b'',j})_{j\in[i]} = (m_{1-b'',j})_{j\in[i]}$. If such value is found, but $b'' \neq b$, abort.

iii. If $b'' \neq \perp$, but $(m''_{b'',j})_{j\in[i]} = (m_{b'',j})_{j\in[i]}$.

(c) Forward $(\sigma_0^{i+1}, \sigma_1^{i+1}, \beta_{i+1}, r_{0,i+1}, r_{1,i+1})$ to R^*, where $\beta_{i+1} \leftarrow\!\!\$\ \mathsf{V}_0(1^\lambda)$, and $r_{0,i+1}, r_{1,i+1} \leftarrow\!\!\$\ \mathcal{M}$, and further $(\alpha^{i+1}_{\mathsf{S}',b}, \sigma_b^{i+1}) \leftarrow\!\!\$\ \mathsf{S}'(\alpha^i_{\mathsf{S}',b}, m''_{b,i} + r''_{b,i})$, while σ^{i+1}_{1-b} was obtained in step 5(a)ii above.

6. Return the output of R^*.

The distribution of $HYB_7(\lambda)$ is identical to that of the ideal experiment $IDEAL_{\mathcal{F}_{\mathsf{OT}},\mathsf{Sim}^{\mathsf{R}^*(z)}}(\lambda, s_0, s_1, b)$ for the above defined simulator. This concludes the proof of property (a) in the definition of receiver-sided simulatability.

4.3.2 Indistinguishability Against a Malicious Sender

We need to show that given the view of a malicious sender it is hard to distinguish whether he has interacted with a receiver using choice bit $b = 0$ or $b = 1$. More precisely, for every non-uniform PPT malicious sender S^* it holds that

$$\left\{ VIEW^{\mathsf{R}}_{\Pi,\mathsf{S}^*(z)}(\lambda, s_0, s_1, 0)\right\}_{\lambda,s_0,s_1,z} \approx_c \left\{ VIEW^{\mathsf{R}}_{\Pi,\mathsf{S}^*(z)}(\lambda, s_0, s_1, 1)\right\}_{\lambda,s_0,s_1,z}$$

where $\lambda \in \mathbb{N}$, $s_0, s_1 \in \{0,1\}^\lambda$, and $z \in \{0,1\}^*$, and where $VIEW^{\mathsf{R}}_{\Pi,\mathsf{S}^*(z)}(\lambda, s_0, s_1, b)$ is the distribution of the view of S^* (with input s_0, s_1 and auxiliary input z) at the end of a real execution of Π with the honest receiver R (with input b).

Let $HYB_0(\lambda, b) \equiv VIEW^{\mathsf{R}}_{\Pi,\mathsf{S}^*(z)}(\lambda, s_0, s_1, b)$. To show the above, we define the following hybrid $HYB(\lambda, b)$.

1. The receiver picks for all $i \in [t']$ $m_{1-b,i} \leftarrow\!\!\$\ \mathcal{M}$. Then, he computes $(\gamma_i, \alpha_i) \leftarrow\!\!\$\ \mathsf{P}_0((m_{1-b,j})_{j\in[i]})$ and sends $(\gamma_i)_{i\in[t']}$.

2. Repeat for each $i \in [t']$: Upon receiving $(\sigma_0^i, \sigma_1^i, \beta_i, r_{0,i}, r_{1,i})$ the receiver picks $\rho_b^i \leftarrow\!\!\$\ \mathcal{M}$, sets $m_{b,i} = \rho_b^i - r_{b,i}$, computes $\delta_i \leftarrow\!\!\$\ \mathsf{P}_1(\alpha_i, \beta_i, \gamma_i, (m_{b,j})_{j\in[i]})$, and sends $(\delta_i, m_{0,i}, m_{1,i})$.

3. The experiment outputs the view of malicious sender S^*.

Notice that the output distribution of $HYB_1(\lambda, b)$ does not change when we sample $m_{b,i} \leftarrow\!\!\$\ \mathcal{M}$ during the first step and define $\rho_b^i = m_{b,i} + r_{b,i}$ in the second step.

Lemma 12. *For all $b \in \{0,1\}$, we have that $\{HYB_0(\lambda, b)\}_{\lambda\in\mathbb{N}} \approx_c \{HYB_1(\lambda, b)\}_{\lambda\in\mathbb{N}}$.*

Proof. By contradiction, assume that there exists a PPT distinguisher D, a bit $b \in \{0,1\}$, and a polynomial $p(\lambda) \in \mathsf{poly}(\lambda)$ such that for infinitely many values of $\lambda \in \mathbb{N}$:

$$|\Pr[\mathsf{D}(HYB_0(\lambda, b) = 1] - \Pr[\mathsf{D}(HYB_1(\lambda, b)) = 1]| \geq 1/p(\lambda).$$

We will construct a PPT distinguisher D' such that

$$\left| \mathbb{P}\left[\mathsf{D}'(\alpha_{\mathsf{D}}^{t'}, (\rho^i, \sigma^i)_{i\in[t']}) = 1 : \begin{array}{c} \forall i \in [t'], (\alpha^i_{\mathsf{R}}, \rho^i) \leftarrow\!\!\$\ \mathsf{R}(\alpha^{i-1}_{\mathsf{R}}, \sigma^i) \\ \wedge\ (\alpha^i_{\mathsf{D}}, \sigma^i) \leftarrow\!\!\$\ \mathsf{D}(\alpha^{i-1}_{\mathsf{D}}, \rho^i) \end{array}\right] - \mathbb{P}\left[\mathsf{D}'(\alpha_{\mathsf{D}}^{t'}, (\rho^i, \sigma^i)_{i\in[t']}) = 1 : \begin{array}{c} \forall i \in [t'], \rho^i \leftarrow\!\!\$\ \mathcal{M} \\ \wedge\ (\alpha^i_{\mathsf{D}}, \sigma^i) \leftarrow\!\!\$\ \mathsf{D}(\alpha^{i-1}_{\mathsf{D}}, \rho^i) \end{array}\right] \right| \geq 1/p(\lambda).$$

We define D' as follows. Distinguisher D' invokes D and acts as in the actual protocol, except for the way he samples the values ρ^i which are obtained from the challenger after forwarding each of the values σ^i sent by the malicious sender. Finally, D' outputs the same as D. It is easy to see that when ρ^i is generated by R, then D' simulates $HYB_0(\lambda, b)$, and when ρ^i is picked uniformly at random he generates $HYB_1(\lambda, b)$. Hence, D' has the same distinguishing advantage as that of D. This finishes the proof. □

In $HYB_1(\lambda, b)$ we can sample both messages $m_{1,i}$, $m_{0,i}$ for all $i \in [n]$ of the C&O protocol in the very beginning. Therefore we can use the choice bit indistinguishability to argue that the receivers choice bit is hidden. This fact is formalized in the lemma below.

Lemma 13. $\{HYB_1(\lambda, 0)\}_{\lambda\in\mathbb{N}} \approx_c \{HYB_1(\lambda, 1)\}_{\lambda\in\mathbb{N}}$.

Proof. The proof is by a standard hybrid argument. For each $j \in [0, t']$, let $HYB_{1,j}(\lambda, b)$ be the hybrid experiment that is identical to $HYB_1(\lambda, b)$ except that after sampling $(m_{0,i}, m_{1,i})_{i\in[t']}$ uniformly from $\mathcal{M}$, the receiver defines all commitments $(\gamma_i)_{i\le j}$ by running the prover P of the underlying C&O protocol upon input $(m_{1-b,i})_{i\le j}$, whereas the commitments $(\gamma_i)_{i>j}$ are defined by running the prover P upon input $(m_{b,i})_{i\le j}$. Observe that $HYB_{1,0}(\lambda, b) \equiv HYB_1(\lambda, 1-b)$ and $HYB_{1,t'}(\lambda, b) \equiv HYB_1(\lambda, b)$; hence, it suffices to show that $HYB_{1,j}(\lambda, b) \approx_c HYB_{1,j+1}(\lambda, b)$ holds for all $b \in \{0, 1\}$ and for all $j \in [0, t']$.

By contradiction, assume that there exists a PPT distinguisher D, a value $b \in \{0, 1\}$, an index $j \in [0, t']$, and a polynomial $p(\lambda) \in \mathsf{poly}(\lambda)$, such that for infinitely many values of $\lambda \in \mathbb{N}$:

$$|\mathbb{P}[\mathsf{D}(HYB_{1,j}(\lambda, b)) = 1] - \mathbb{P}[\mathsf{D}(HYB_{1,j+1}(\lambda, b)) = 1]|| \ge 1/p(\lambda).$$

We will construct a PPT distinguisher D' and a PPT malicious verifier V^* such that

$$\Big|\,\mathbb{P}\Big[\mathsf{D}'(\langle \mathsf{P}((m_{b,i})_{i\le j+1}, (m_{1-b,i})_{i\le j+1}, 0), \mathsf{V}^*(1^\lambda)\rangle) = 1\Big] - \mathbb{P}\Big[\mathsf{D}'(\langle \mathsf{P}((m_{b,i})_{i\le j+1}, (m_{1-b,i})_{i\le j+1}, 1), \mathsf{V}^*(1^\lambda)\rangle) = 1\Big]\,\Big| \ge 1/p(\lambda),$$

where for all $i \in [t']$, the values $m_{0,i}, m_{1,i}$ are uniformly sampled by D' from $\mathcal{M}$. Verifier V^* invokes D and emulates faithfully a run of $HYB_{1,j}(\lambda, b)$ except that it embeds the commitment received from the challenger in the value γ_{j+1} which is part of the first message sent to D, and similarly, after receiving $(\sigma_0^{j+1}, \sigma_1^{j+1}, \beta_{j+1}, r_{0,j+1}, r_{1,j+1})$ from D, it forwards β_{j+1} to the challenger, obtaining a value δ_{j+1} that is used together with $(m_{b,i})_{i\le j+1}$ and $(m_{1-b,i})_{i\le j+1}$ in order to terminate the execution of the experiment. In the end, D' outputs the output of D.

Clearly, when the challenger uses committing branch zero, the reduction perfectly simulates $HYB_{1,j}(\lambda, b)$, and when the challenger uses committing branch 1, the reduction perfectly simulates $HYB_{1,j+1}(\lambda, b)$. Since $t' \in \mathsf{poly}(\lambda)$, the statement follows. □

5 Conclusions

We have shown a construction of maliciously secure oblivious transfer (M-OT) protocol from a certain class of key agreement (KA) and semi-honestly secure OT (SH-OT) protocols that enjoy a property called *strong uniformity* (SU), which informally means that the distribution of the messages sent by one of the parties is computationally close to uniform, even in case the other party is malicious.

When starting with 2-round or 3-round SUSH-OT or SU-KA, we obtain 4-round M-OT, and thus, invoking [CCG+19], 4-round maliciously secure MPC from standard assumptions including low-noise LPN, LWE, Subset Sum, CDH, DDH, and RSA (all with polynomial hardness).

Also, it is a natural question to see whether SU-KA with $t \ge 4$ rounds can be instantiated from concrete assumptions that do not imply PKE.

6 Acknowledgment

We would like to thank Silas Richelson for a discussion on their commit-and-open protocol. We also thank the anonymous reviewers who helped removing wrong claims and clarifying the presentation of our results.

References

[ACJ17] Prabhanjan Ananth, Arka Rai Choudhuri, and Abhishek Jain. A new approach to round-optimal secure multiparty computation. In *CRYPTO*, pages 468–499, 2017.

[AIR01] William Aiello, Yuval Ishai, and Omer Reingold. Priced oblivious transfer: How to sell digital goods. In *EUROCRYPT*, pages 119–135, 2001.

[Ale03] Michael Alekhnovich. More on average case vs approximation complexity. In *IEEE FOCS*, pages 298–307, 2003.

[BD18] Zvika Brakerski and Nico Döttling. Two-message statistically sender-private OT from LWE. In *TCC*, pages 370–390, 2018.

[BGJ+18] Saikrishna Badrinarayanan, Vipul Goyal, Abhishek Jain, Yael Tauman Kalai, Dakshita Khurana, and Amit Sahai. Promise zero knowledge and its applications to round optimal MPC. In *CRYPTO*, pages 459–487, 2018.

[BHP17] Zvika Brakerski, Shai Halevi, and Antigoni Polychroniadou. Four round secure computation without setup. In *TCC*, pages 645–677, 2017.

[BL18] Fabrice Benhamouda and Huijia Lin. k-round multiparty computation from k-round oblivious transfer via garbled interactive circuits. In *EUROCRYPT*, pages 500–532, 2018.

[BY92] Mihir Bellare and Moti Yung. Certifying cryptographic tools: The case of trapdoor permutations. In *CRYPTO*, pages 442–460, 1992.

[CCG+19] Arka Rai Choudhuri, Michele Ciampi, Vipul Goyal, Abhishek Jain, and Rafail Ostrovsky. Round optimal secure multiparty computation from minimal assumptions. Cryptology ePrint Archive, Report 2019/216, 2019. https://eprint.iacr.org/2019/216.

[CL18] Ran Canetti and Amit Lichtenberg. Certifying trapdoor permutations, revisited. In *TCC*, pages 476–506, 2018.

[CNS07] Jan Camenisch, Gregory Neven, and Abhi Shelat. Simulatable adaptive oblivious transfer. In *EUROCRYPT*, pages 573–590, 2007.

[COSV17a] Michele Ciampi, Rafail Ostrovsky, Luisa Siniscalchi, and Ivan Visconti. Delayed-input non-malleable zero knowledge and multi-party coin tossing in four rounds. In *TCC*, pages 711–742, 2017.

[COSV17b] Michele Ciampi, Rafail Ostrovsky, Luisa Siniscalchi, and Ivan Visconti. Round-optimal secure two-party computation from trapdoor permutations. In *TCC*, pages 678–710, 2017.

[DN00] Ivan Damgård and Jesper Buus Nielsen. Improved non-committing encryption schemes based on a general complexity assumption. In *CRYPTO*, pages 432–450, 2000.

[EGL82] Shimon Even, Oded Goldreich, and Abraham Lempel. A randomized protocol for signing contracts. In *CRYPTO*, pages 205–210, 1982.

[Gam84] Taher El Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *CRYPTO*, pages 10–18, 1984.

[GH07] Matthew Green and Susan Hohenberger. Blind identity-based encryption and simulatable oblivious transfer. In *ASIACRYPT*, pages 265–282, 2007.

[GHKW17] Rishab Goyal, Susan Hohenberger, Venkata Koppula, and Brent Waters. A generic approach to constructing and proving verifiable random functions. In *TCC*, pages 537–566, 2017.

[GKM+00] Yael Gertner, Sampath Kannan, Tal Malkin, Omer Reingold, and Mahesh Viswanathan. The relationship between public key encryption and oblivious transfer. In *IEEE FOCS*, pages 325–335, 2000.

[GM84] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *J. Comput. Syst. Sci.*, 28(2):270–299, 1984.

[GMMM18] Sanjam Garg, Mohammad Mahmoody, Daniel Masny, and Izaak Meckler. On the round complexity of OT extension. In *CRYPTO*, pages 545–574, 2018.

[GMPP16] Sanjam Garg, Pratyay Mukherjee, Omkant Pandey, and Antigoni Polychroniadou. The exact round complexity of secure computation. In *EUROCRYPT*, pages 448–476, 2016.

[GMW91] Oded Goldreich, Silvio Micali, and Avi Wigderson. Proofs that yield nothing but their validity for all languages in NP have zero-knowledge proof systems. *J. ACM*, 38(3):691–729, 1991.

[GR13] Oded Goldreich and Ron D. Rothblum. Enhancements of trapdoor permutations. *J. Cryptology*, 26(3):484–512, July 2013.

[GS18] Sanjam Garg and Akshayaram Srinivasan. Two-round multiparty secure computation from minimal assumptions. In *EUROCRYPT*, pages 468–499, 2018.

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

[HHPV18] Shai Halevi, Carmit Hazay, Antigoni Polychroniadou, and Muthuramakrishnan Venkitasubramaniam. Round-optimal secure multi-party computation. In *CRYPTO*, pages 488–520, 2018.

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

[HK09] Dennis Hofheinz and Eike Kiltz. The group of signed quadratic residues and applications. In *CRYPTO*, pages 637–653, 2009.

[IKO+11] Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, Manoj Prabhakaran, and Amit Sahai. Efficient non-interactive secure computation. In *EUROCRYPT*, pages 406–425, 2011.

[IPS08] Yuval Ishai, Manoj Prabhakaran, and Amit Sahai. Founding cryptography on oblivious transfer – Efficiently. In *CRYPTO*, pages 572–591, 2008.

[IR88] Russell Impagliazzo and Steven Rudich. Limits on the provable consequences of one-way permutations. In *CRYPTO*, pages 8–26, 1988.

[JS07] Stanislaw Jarecki and Vitaly Shmatikov. Efficient two-party secure computation on committed inputs. In *EUROCRYPT*, pages 97–114, 2007.

[Kal05] Yael Tauman Kalai. Smooth projective hashing and two-message oblivious transfer. In *EUROCRYPT*, pages 78–95, 2005.

[Kil88] Joe Kilian. Founding cryptography on oblivious transfer. In *ACM STOC*, pages 20–31, 1988.

[Kil92] Joe Kilian. A note on efficient zero-knowledge proofs and arguments (extended abstract). In *ACM STOC*, pages 723–732, 1992.

[KKM12] Saqib A. Kakvi, Eike Kiltz, and Alexander May. Certifying RSA. In *ASIACRYPT*, pages 404–414, 2012.

[KO97] Eyal Kushilevitz and Rafail Ostrovsky. Replication is NOT needed: SINGLE database, computationally-private information retrieval. In *IEEE FOCS*, pages 364–373, 1997.

[KO04] Jonathan Katz and Rafail Ostrovsky. Round-optimal secure two-party computation. In *CRYPTO*, pages 335–354, 2004.

[Lin08] Yehuda Lindell. Efficient fully-simulatable oblivious transfer. *Chicago J. Theor. Comput. Sci.*, 2008, 2008.

[LMRS04] Anna Lysyanskaya, Silvio Micali, Leonid Reyzin, and Hovav Shacham. Sequential aggregate signatures from trapdoor permutations. In *EUROCRYPT*, pages 74–90, 2004.

[LPS10] Vadim Lyubashevsky, Adriana Palacio, and Gil Segev. Public-key cryptographic primitives provably as secure as subset sum. In *TCC*, pages 382–400, 2010.

[LS19] Alex Lombardi and Luke Schaeffer. A note on key agreement and non-interactive commitments. Cryptology ePrint Archive, Report 2019/279, 2019. https://eprint.iacr.org/2019/279.

[LZ13] Yehuda Lindell and Hila Zarosim. On the feasibility of extending oblivious transfer. In *TCC*, pages 519–538, 2013.

[LZ18] Yehuda Lindell and Hila Zarosim. On the feasibility of extending oblivious transfer. *J. Cryptology*, 31(3):737–773, 2018.

[NP01] Moni Naor and Benny Pinkas. Efficient oblivious transfer protocols. In *SODA*, pages 448–457, 2001.

[NP05] Moni Naor and Benny Pinkas. Computationally secure oblivious transfer. *J. Cryptology*, 18(1):1–35, 2005.

[ORS15] Rafail Ostrovsky, Silas Richelson, and Alessandra Scafuro. Round-optimal black-box two-party computation. In *CRYPTO*, pages 339–358, 2015.

[PVW08] Chris Peikert, Vinod Vaikuntanathan, and Brent Waters. A framework for efficient and composable oblivious transfer. In *CRYPTO*, pages 554–571, 2008.

[Rab81] Michael O. Rabin. How to exchange secrets by oblivious transfer. Technical report, Harvard University, 1981.

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *ACM STOC*, pages 84–93, 2005.

[Yao82] Andrew Chi-Chih Yao. Protocols for secure computations (extended abstract). In *IEEE FOCS*, pages 160–164, 1982.

[Yao86] Andrew Chi-Chih Yao. How to generate and exchange secrets (extended abstract). In *IEEE FOCS*, pages 162–167, 1986.

A The ORS Commit-and-Open Protocol

A.1 Commitment Schemes

A non-interactive commitment scheme is an efficient randomized algorithm Commit taking as input a message $m \in \mathcal{M}$ together with random coins $r \in \{0,1\}^\lambda$, and returning a commitment com. The opening of a commitment com consists of strings (m, r) such that $com = \mathsf{Commit}(m; r)$; we sometimes write $\mathsf{Open}(m)$ to denote the randomness that is needed to open successfully a value com, i.e. $com = \mathsf{Commit}(m; \mathsf{Open}(m))$.

As for security, commitment schemes should satisfy two properties called hiding and binding. Intuitively, the first property says that a commitment does not leak any information on the committed message; the second property says that it should be hard to open a given commitment in two different ways. The formal definitions follow.

Definition 8 (Hiding of commitments)**.** A commitment scheme is perfectly (resp., computationally or statistically) hiding, if for all $m_0, m_1 \in \mathcal{M}$ it holds that the ensembles $\{\mathsf{Commit}(m_0; U_\lambda)\}_{\lambda\in\mathbb{N}}$ and $\{\mathsf{Commit}(m_1; U_\lambda)\}_{\lambda\in\mathbb{N}}$ are identically distributed (resp., computationally or statistically close), where U_λ denotes the uniform distribution over $\{0,1\}^\lambda$.

Definition 9 (Binding of commitments)**.** A commitment scheme is computationally binding, if for all PPT adversaries A there is a negligible function $\nu : \mathbb{N} \to [0,1]$ such that

$$\Pr\left[\mathsf{Commit}(m; r) = \mathsf{Commit}(m'; r') \wedge m \neq m' : \ ((m,r),(m',r')) \leftarrow\!\!\$\ \mathsf{A}(1^\lambda)\right] \leq \nu(\lambda).$$

In case the above probability equals zero for all even unbounded adversaries, we say that the commitment scheme is perfectly binding.

A.2 The ORS Construction

The ORS string C&O protocol $\Pi_{\mathsf{c\&o}} = (\mathsf{P}_0, \mathsf{P}_1, \mathsf{V}_0, \mathsf{V}_1)$ for string length n is depicted in Fig. 4. It relies on a statistically binding commitment scheme $(\mathsf{Commit}, \mathsf{Open})$ and a linear error detection

Verifier $\mathsf{V}(1^\lambda)$		**Prover** $\mathsf{P}(m_0, m_1, b)$
		$\hat{\mathbf{m}}_b := G(m_b), \mathbf{c}_{1-b} \leftarrow\$ \{0,1\}^n$ $\forall i \in [n]:$ $\mathbf{M}_{b,i} \leftarrow\$ \mathbb{Z}_q^{2\times q}$ s.t. $\forall j \in \mathbb{Z}_q:$ $\sum_{k\in\{0,1\}} \mathbf{M}_{b,i}[k,j] = \hat{\mathbf{m}}_b[i]$ $\sigma_i \leftarrow\$ \mathsf{Perm}(\mathbb{Z}_q)$ $\mathbf{M}_{1-b,i} \leftarrow\$ \mathbb{Z}_q^{2\times q}$ s.t. $\forall j \in \mathbb{Z}_q:$ $\sum_{k\in\{0,1\}} \mathbf{M}_{1-b,i}[k,j] = \sigma_i(j)$ $\mathbf{v}_{b,i} := \psi(\mathbf{M}_{b.i}[0,*])$ $\mathbf{v}_{1-b,i} := (-1)^{\mathbf{c}_{1-b}[i]}\psi(\mathbf{M}_{1-b.i}[\mathbf{c}_{1-b}[i],*])$ $\gamma := \{\mathbf{v}_{k,i}, \mathsf{Commit}(\mathbf{M}_{k,i})\}_{k\in\{0,1\},i\in[n]}$
	$\xleftarrow{\gamma}$	
$\beta \leftarrow\$ \{0,1\}^n$		
	$\xrightarrow{\beta}$	
		$\hat{\mathbf{m}}_{1-b} := G(m_{1-b}), \mathbf{c}_b := \beta - \mathbf{c}_{1-b}$ $\mathbf{d}_{1-b} \in \mathbb{Z}_q^n$ s.t. $\sum_{k\in\{0,1\}} \mathbf{M}_{1-b,i}[k, \mathbf{d}_{1-b}[i]] = \hat{\mathbf{m}}_{1-b}[i]$ $\mathbf{d}_b \leftarrow\$ \mathbb{Z}_q^n$ $\delta_{\mathbf{c}} := \mathsf{Open}(\mathbf{M}_{k,i}[\mathbf{c}_k[i],*])_{k\in\{0,1\},i\in[n]}$ $\delta_{\mathbf{d}} := \mathsf{Open}(\mathbf{M}_{k,i}[1-\mathbf{c}_k[i], \mathbf{d}_k[i]])_{k\in\{0,1\},i\in[n]}$
	$\xleftarrow{\delta, m_0, m_1}$	$\delta := (\mathbf{c}_0, \mathbf{c}_1, \mathbf{d}_0, \mathbf{d}_1, \delta_{\mathbf{c}}, \delta_{\mathbf{d}})$
$\forall \ell \in \{0,1\}, i \in [n]:$ $\hat{\mathbf{m}}_\ell[i] := \sum_{k\in\{0,1\}} \mathbf{M}_{\ell,i}[k, \mathbf{d}_\ell[i]]$ check if $\mathbf{c}_0 + \mathbf{c}_1 = \beta$ $\hat{\mathbf{m}}_0 = G(m_0), \hat{\mathbf{m}}_1 = G(m_1)$ $\forall k \in \{0,1\}, i \in [n]:$ $\psi(\mathbf{M}_{k,i}[\mathbf{c}_k[i],*]) = (-1)^{\mathbf{c}_k[i]}\mathbf{v}_{k,i}$ output 0 iff check fails		

Figure 4: The ORS string 1-out-of-2 commit-and-open protocol. $\mathsf{Perm}(\mathbb{Z}_q)$ is the set of permutations over $\mathbb{Z}_q$, and $\mathsf{Open}(m)$ denotes the randomness that is needed to open commitment $\mathsf{Commit}(m)$.

code G with minimal distance of at least $\frac{1}{2}(n+\kappa)$, which can be instantiated with, e.g., a Reed-Solomon code. In what follows, the code G is a public parameter of the protocol, and we write $G(m)$ to denote an encoding of message m under code G. For simplifying the presentation of the protocol, we use $\psi : \mathbb{Z}_q^q \to \mathbb{Z}_q^{q-1}$ to denote the linear map

$$(\mathbf{x}[0], \ldots, \mathbf{x}[q-1]) \mapsto (\mathbf{x}[1] - \mathbf{x}[0], \ldots, \mathbf{x}[q-1] - \mathbf{x}[0]),$$

where $\mathbf{x}[i]$ is the i-th entry of a vector $\mathbf{x} \in \mathbb{Z}_q^q$.

Remark 1. *In the ORS protocol, the prover does not need to know or fix $\hat{\mathbf{m}}_{1-b}$ till the second round. Nevertheless, during the committing branch indistinguishability experiment, both* *messages need to be fixed before the first round.*

Remark 2. *In order to simplify the notation, within this section we shall denote the input bit of the prover in the* 1*-out-of-*2 *C&O protocol with* b *(instead of* d*).*

Lemma 14 (Completeness of the ORS protocol)**.** *For any* $\epsilon \in [0,1)$*, assuming that the commitment scheme* Commit *is complete with probability at least* $1-\epsilon$*, then the ORS protocol from Fig. 4 is complete with probability at least* $(1-\epsilon)^{(q+1)n}$*.*

Proof. The verifier opens $(q+1)n$ commitments. By completeness, the openings will be correct with probability $(1-\epsilon)^{(q+1)n}$. In the following, we assume that this is the case. The protocol will succeed if and only if the checks do not fail, i.e. all of the below equations hold:

$$\mathbf{c}_0 + \mathbf{c}_1 = \beta \qquad \hat{\mathbf{m}}_0 = G(m_0) \qquad \hat{\mathbf{m}}_1 = G(m_1)$$
$$\forall k \in \{0,1\}, i \in [n]: \ \psi(\mathbf{M}_{k,i}[\mathbf{c}_k[i], *]) = (-1)^{\mathbf{c}_k[i]}\mathbf{v}_{k,i}.$$

By construction, it is easy to see that $\beta = \mathbf{c}_0 + \mathbf{c}_1$. Next we will show that in a honest execution of the protocol, both $\hat{\mathbf{m}}_0$ and $\hat{\mathbf{m}}_1$ will be codewords w.r.t. code G. The entries of $\hat{\mathbf{m}}_0$ and $\hat{\mathbf{m}}_1$ are computed by the verifier as

$$\hat{\mathbf{m}}_\ell[i] := \sum_{k\in\{0,1\}} \mathbf{M}_{\ell,i}[k, \mathbf{d}_\ell[i]]$$

for $\ell \in \{0,1\}$, $i \in [n]$. For branch $1-b$, the vector $\mathbf{d} \in \mathbb{Z}_q^n$ is chosen by the receiver such that

$$\sum_{k\in\{0,1\}} \mathbf{M}_{1-b,i}[k, \mathbf{d}_{1-b}[i]] = \hat{\mathbf{m}}'_{1-b}[i]$$

holds for all $i \in [n]$, where $\hat{\mathbf{m}}'_{1-b}[i]$ denotes $\hat{\mathbf{m}}_{1-b}[i]$ on the receiver's side. Further, such a vector $\mathbf{d}_{1-b}$ always exists due to the fact that there are q columns in $\mathbf{M}_{1-b,i}$ and each column sums to a different value in $\mathbb{Z}_q$. For branch b,

$$\sum_{k\in\{0,1\}} \mathbf{M}_{b,i}[k, \mathbf{d}_b[i]] = \hat{\mathbf{m}}'_b[i]$$

holds for any $\mathbf{d}_b \in \mathbb{Z}_q^n$. Therefore the vectors $\hat{\mathbf{m}}_0$ and $\hat{\mathbf{m}}_1$ computed by the verifier are identical to the vectors $\hat{\mathbf{m}}'_0$ and $\hat{\mathbf{m}}'_1$ computed by the prover, which are in particular chosen to be codewords w.r.t. code G for messages m_0 and m_1.

The last part of the checking procedure checks whether the image of ψ for the two rows of $\mathbf{M}$ is indeed consistent with the transmitted value $\mathbf{v}_{k,i}$. More specifically, $\forall k \in \{0,1\}$, $i \in [n]$,

$$\psi(\mathbf{M}_{k,i}[\mathbf{c}_k[i], *]) = (-1)^{\mathbf{c}_k[i]}\mathbf{v}_{k,i}$$

must hold. Again, by construction this is true for all $i \in [n]$ and $k = 1-b$, simply because $\mathbf{v}_{1-b,i}$ is chosen such that it holds. In case $k = b$ it holds as well, since for each $i \in [n]$ all the columns of $\mathbf{M}_{b,i}$ sum to $\hat{\mathbf{m}}_{b,i}$ or equivalently for all $j \in \mathbb{Z}_q$, $\mathbf{M}_{b,i}[1,j] = \hat{\mathbf{m}}_{b,i} - \mathbf{M}_{b,i}[0,j]$. Due to this fact, for all $c \in \{0,1\}$, $i \in [n]$

$$\begin{aligned}\psi(\mathbf{M}_{b,i}[c,*]) &= (\mathbf{M}_{b,i}[c,1] - \mathbf{M}_{b,i}[c,0], \dots, \mathbf{M}_{b,i}[c,q-1] - \mathbf{M}_{b,i}[c,0]) \\ &= (-\mathbf{M}_{b,i}[1-c,1] + \mathbf{M}_{b,i}[1-c,0], \dots, -\mathbf{M}_{b,i}[1-c,q-1] + \mathbf{M}_{b,i}[1-c,0]) \\ &= (-1)\psi(\mathbf{M}_{b,i}[1-c,*])\end{aligned}$$

holds. Further, $\mathbf{v}_{b,i} := \psi(\mathbf{M}_{b.i}[0,*])$ and therefore

$$\psi(\mathbf{M}_{b,i}[\mathbf{c}_b[i], *]) = (-1)^{\mathbf{c}_b[i]}\mathbf{v}_{b,i}$$

holds for any choice of $\mathbf{c}_b \in \{0,1\}^n$. This concludes proving completeness. □

Lemma 15 (Existence of a committing branch for the ORS protocol)**.** *Let $\kappa \in \mathbb{N}$ be a statistical security parameter. Assuming that the commitment scheme* Commit *is statistically binding except with probability at most ϵ, and that code G has minimal distance $\frac{1}{2}(n+\kappa)$, then the ORS protocol from Fig. 4 satisfies the property of existence of a committing branch except with probability at most $2\epsilon + 2^{-\kappa}$.*

Proof. We define several hybrids to prove the lemma. In the first hybrid, a malicious prover P^* loses if, for any $i \in [n]$ and any $k \in \{0,1\}$, a partial message $\hat{\mathbf{m}}_k[i]$ differs from $\hat{\mathbf{m}}'_b[i]$ and the opened row of $\mathbf{M}_{k,i}$ differs as well, i.e. $\mathbf{c}_k[i] \neq \mathbf{c}'_k[i]$.

In the second hybrid, the adversary will lose as well if there are more than κ positions $i \in [n]$ for which both messages $\hat{\mathbf{m}}_0[i]$ and $\hat{\mathbf{m}}_1[i]$ differ from the messages $\hat{\mathbf{m}}'_0[i]$ and $\hat{\mathbf{m}}'_1[i]$ of the second run.

Hybrid $HYB_0(\lambda)$**:** This is the original security game, i.e.

$$\begin{aligned}&(\gamma, \alpha_0) \leftarrow\!\!\$\ \mathsf{P}_0^*(1^\lambda);\\&\beta, \beta' \leftarrow\!\!\$\ \mathsf{V}_0(1^\lambda);\\&(\delta, m_0, m_1) \leftarrow\!\!\$\ \mathsf{P}_1^*(\alpha_0, \beta);\\&(\delta', m'_0, m'_1) \leftarrow\!\!\$\ \mathsf{P}_1^*(\alpha_0, \beta')\end{aligned}$$

and the prover wins iff

$$\begin{aligned}&(\mathsf{V}_1(T) = 1) \wedge (\mathsf{V}_1(T') = 1)\\ \wedge&(m_0 \neq m'_0) \wedge (m_1 \neq m'_1).\end{aligned}$$

Hybrid $HYB_1(\lambda)$**:** Identical to $HYB_0(\lambda)$ except that the prover wins iff

$$\begin{aligned}&(\mathsf{V}_1(T) = 1) \wedge (\mathsf{V}_1(T') = 1)\\ \wedge&(m_0 \neq m'_0) \wedge (m_1 \neq m'_1)\\ \wedge&\underline{\forall i \in [n], k \in \{0,1\} : (\hat{\mathbf{m}}_k[i] = \hat{\mathbf{m}}'_k[i]) \vee (\mathbf{c}_k[i] = \mathbf{c}'_k[i])}.\end{aligned}$$

Hybrid $HYB_2(\lambda)$**:** Identical to $HYB_1(\lambda)$ except the prover wins iff

$$\begin{aligned}&(\mathsf{V}_1(T) = 1) \wedge (\mathsf{V}_1(T') = 1)\\ \wedge&(m_0 \neq m'_0) \wedge (m_1 \neq m'_1)\\ \wedge&\forall i \in [n], k \in \{0,1\} : (\hat{\mathbf{m}}_k[i] = \hat{\mathbf{m}}'_k[i]) \vee (\mathbf{c}_k[i] = \mathbf{c}'_k[i])\\ \wedge&\underline{|\{i \in [n] \mid \hat{\mathbf{m}}_0[i] \neq \hat{\mathbf{m}}'_0[i] \wedge \hat{\mathbf{m}}_1[i] \neq \hat{\mathbf{m}}'_1[i]\}| < \kappa}.\end{aligned}$$

Claim 8. $\Delta\left(HYB_0(\lambda); HYB_1(\lambda)\right) \leq 2\epsilon$.

Proof. There is a difference between the two hybrids if and only if there is an $i \in [n]$ and $k \in \{0,1\}$ such that

$$(\hat{\mathbf{m}}_k[i] \neq \hat{\mathbf{m}}'_k[i]) \wedge (\mathbf{c}_k[i] \neq \mathbf{c}'_k[i]).$$

By the checking procedure of the verifier, we have

$$\psi(\mathbf{M}_{k,i}[\mathbf{c}_k[i], *]) = (-1)^{\mathbf{c}_k[i]}\mathbf{v}_{k,i},$$

which implies the two equalities

$$\begin{aligned}\mathbf{v}[\mathbf{d}_k[i]] &= \mathbf{M}_{k,i}[0, \mathbf{d}_k[i]] - \mathbf{M}_{k,i}[0,0] = -\mathbf{M}_{k,i}[1, \mathbf{d}_k[i]] + \mathbf{M}_{k,i}[1,0],\\ \mathbf{v}[\mathbf{d}_k[i]] &= \mathbf{M}'_{k,i}[0, \mathbf{d}'_k[i]] - \mathbf{M}'_{k,i}[0,0] = -\mathbf{M}'_{k,i}[1, \mathbf{d}'_k[i]] + \mathbf{M}_{k,i}[1,0].\end{aligned}$$

Further,

$$\hat{\mathbf{m}}_k[i] = \mathbf{M}_{k,i}[0, \mathbf{d}_k[i]] + \mathbf{M}_{k,i}[1, \mathbf{d}_k[i]] = \mathbf{M}_{k,i}[0,0] + \mathbf{M}_{k,i}[1,0]$$

as well as $\hat{\mathbf{m}}'_k[i] = \mathbf{M}'_{k,i}[0,0] + \mathbf{M}'_{k,i}[1,0]$. Since $\hat{\mathbf{m}}_k[i] \neq \hat{\mathbf{m}}'_k[i]$, either $\mathbf{M}_{k,i}[0,0] \neq \mathbf{M}'_{k,i}[0,0]$ or $\mathbf{M}_{k,i}[1,0] \neq \mathbf{M}'_{k,i}[1,0]$ which breaks statistical binding. □

Claim 9. $\Delta\left(HYB_1(\lambda); HYB_2(\lambda)\right) \leq 2^{-\kappa}$.

Proof. A malicious prover P^* is successful in $HYB_1(\lambda)$ but not in $HYB_2(\lambda)$ if for set

$$\mathcal{S} := \{i \in [n] : \ \hat{\mathbf{m}}_0[i] \neq \hat{\mathbf{m}}'_0[i] \wedge \hat{\mathbf{m}}_1[i] \neq \hat{\mathbf{m}}'_1[i]\}$$

the inequality $|\mathcal{S}| \geq \kappa$ holds. To prove the claim, we show this bound on set $\mathcal{S}$.

For any $i \in [n]$ and $k \in \{0,1\}$, either $\hat{\mathbf{m}}_k[i] \neq \hat{\mathbf{m}}'_k[i]$ or $\mathbf{c}_k[i] \neq \mathbf{c}'_k[i]$ holds. Hence, for all elements i in $\mathcal{S}$, we necessarily have $\mathbf{c}_0[i] = \mathbf{c}'_0[i]$ and $\mathbf{c}_1[i] = \mathbf{c}'_1[i]$. This implies that challenge $\beta = \mathbf{c}_0 + \mathbf{c}_1$ is identical with β' on position i. Since β' is uniformly random, this is only the case with probability 1/2. If it is not the case, the verifier rejects. Since the size of $\mathcal{S}$ has to be at least κ, the probability of this to happen is at most $2^{-|\mathcal{S}|} \leq 2^{-\kappa}$. □

In $HYB_2(\lambda)$, the adversary's choice of $\hat{\mathbf{m}}_0$ and $\hat{\mathbf{m}}_1$ will both differ from $\hat{\mathbf{m}}'_0$ and $\hat{\mathbf{m}}'_1$ on at most κ positions. On all other positions, $\hat{\mathbf{m}}_0$ and $\hat{\mathbf{m}}_1$ will be identical to $\hat{\mathbf{m}}'_0$ and $\hat{\mathbf{m}}'_1$. Since there are $n - \kappa$ positions left, at least one of the pairs will be identical on at least $\frac{1}{2}(n-\kappa)$ positions. Let this be $\hat{\mathbf{m}}_b$.

Due to the minimal distance $\frac{1}{2}(n+\kappa)$ of code G, there is a unique codeword that matches these $\frac{1}{2}(n-\kappa)$ positions. Hence, in both runs, a malicious receiver is committed to $\hat{\mathbf{m}}_b = \hat{\mathbf{m}}'_b$, because if $\hat{\mathbf{m}}_b$ or $\hat{\mathbf{m}}'_b$ is not a codeword, the verifier rejects. Thus, $\hat{\mathbf{m}}_b = \hat{\mathbf{m}}'_b$ decodes to a unique message m_b and therefore for all unbounded provers P^* experiment $HYB_2(\lambda)$ returns 1 with zero probability, which concludes this proof. □

Lemma 16 (Committing branch indistinguishability of the ORS protocol)**.** *Assuming that the commitment scheme* Commit *satisfies computational hiding, the ORS protocol from Fig. 4 satisfies committing branch indistinguishability.*

Proof. To show indistinguishablity, we define a hybrid in which a prover commits to both messages and both branches will follow the same distribution. Let $HYB_0(\lambda, b)$ be the experiment defining committing branch indistinguishability, where the adversary V^* acts as a malicious verifier; our goal is to show that for all PPT V^*, we have $HYB_0(\lambda, 0) \approx_c HYB(\lambda, 1)$. Consider the hybrid experiment $HYB(\lambda, b)$ where in the first round the prover takes the following actions:

$$\begin{aligned}
&\hat{\mathbf{m}}_b := G(m_b), \mathbf{c}_{1-b} \leftarrow\!\!\$\ \{0,1\}^n, \underline{\hat{\mathbf{m}}_{1-b} := G(m_{1-b})} \\
&\mathbf{d}_b, \underline{\mathbf{d}_{1-b}} \leftarrow\!\!\$\ \mathbb{Z}_q^n \\
&\forall i \in [n], \underline{\ell \in \{0,1\}} : \\
&\quad \mathbf{M}_{\ell,i} \leftarrow\!\!\$\ \mathbb{Z}_q^{2\times q} \text{ s.t. } \forall j \in \mathbb{Z}_q : \\
&\qquad \sum_{k\in\{0,1\}} \mathbf{M}_{\ell,i}[k,j] = \hat{\mathbf{m}}_b[i] \\
&\quad \mathbf{v}_{\ell,i} := \psi(\mathbf{M}_{\ell.i}[0,*]) \\
&\gamma := \{\mathbf{v}_{k,i}, \mathsf{Commit}(\mathbf{M}_{k,i})\}_{k\in\{0,1\}, i\in[n]},
\end{aligned}$$

and moreover during the third round, the prover acts as follows:

$$\mathbf{c}_b = \beta - \mathbf{c}_{1-b}$$
$$\delta_{\mathbf{c}} := \mathsf{Open}(\mathbf{M}_{k,i}[\mathbf{c}_k[i], *])_{k\in\{0,1\}, i\in[n]}$$
$$\delta_{\mathbf{d}} := \mathsf{Open}(\mathbf{M}_{k,i}[1-\mathbf{c}_k[i], \mathbf{d}_k[i]])_{k\in\{0,1\}, i\in[n]}$$

Notice that sampling first $\mathbf{c}_{1-b}$ and setting $\mathbf{c}_b = \beta - \mathbf{c}_{1-b}$ has the same distribution as $\mathbf{c}_0, \mathbf{c}_1 \leftarrow\$ \{0,1\}^n$ conditioned on $\beta = \mathbf{c}_0 + \mathbf{c}_1$. Therefore both branches have the same distribution.

Claim 10. *For all PPT V^*, and for all $b \in \{0,1\}$, we have that $HYB_0(\lambda, b) \approx_c HYB_1(\lambda, b)$.*

Proof. We will define $n(q-1)$ sub-hybrids. For each $i \in [n]$, there are $q-1$ commitments in branch $b-1$ that are not opened in the third round. We will switch their committed value $\mathbf{M}_{1-\mathbf{c}_i,i}$ step by step from the distribution in HYB_0 to the distribution in HYB_1, i.e. from being uniform conditioned on summing to $\sigma_i(j)$ to summing to $\hat{\mathbf{m}}[i]$.

We denote the sub hybrids with $HYB_{0,0,0}(\lambda, b)$ to $HYB_{0,n,q}(\lambda, b)$, where $HYB_{0,0,0}(\lambda, b) \equiv HYB_0(\lambda, b)$ and $HYB_{0,n,q}(\lambda, b) \equiv HYB_1(\lambda, b)$. We switch from $HYB_{0,i,j}(\lambda, b)$ to $HYB_{0,i,j+1}(\lambda, b)$, and from $HYB_{0,i,q}(\lambda, b)$ to $HYB_{0,i+1,0}(\lambda, b)$. In the following, we will just show how to transition from $HYB_{0,i,j}(\lambda, b)$ to $HYB_{0,i,j+1}(\lambda, b)$. The other step is done analogously. Further notice that the the hybrids

$$HYB_{0,i,\mathbf{d}_{1-b}[i]-1}(\lambda, b) \quad \text{and} \quad HYB_{0,i,\mathbf{d}_{1-b}[i]}(\lambda, b)$$

are already distributed identically. Next, we show that for any $i^* \in [n]$, $j^* \in \mathbb{Z}_q$, and for all PPT V^* and $b \in \{0,1\}$, hybrids $HYB_{0,i^*,j^*}(\lambda, b)$ and $HYB_{0,i^*,j^*+1}(\lambda, b)$ are computationally close, which finishes the proof of the claim.

Recall that an adversary A against the hiding of the commitment scheme chooses two messages $\tilde{m}_0$ and $\tilde{m}_1$, and receives a commitment $\tilde{com}$ of one of the two messages. We denote this by $\tilde{com} \leftarrow\$ \mathcal{O}_{\mathsf{Commit}}(\tilde{m}_0, \tilde{m}_1)$. Attacker A wins if he successfully determines which message has been committed to. In what follows, we mostly ignore branch b since it has the same distribution in both hybrids. In the first round, A simulates the prover as follows.

$$\mathbf{c}_{1-b} \leftarrow\$ \{0,1\}^n, \hat{\mathbf{m}}_{1-b} := G(m_{1-b}), \mathbf{d}_{1-b} \leftarrow\$ \mathbb{Z}_q^n$$
$$\forall (i < i^* \vee (i = i^* \wedge j \leq j^*)), \sigma_i(j) := \hat{\mathbf{m}}_{1-b}[i]$$
$$\sigma' \leftarrow\$ \mathsf{Perm}(\mathbb{Z}_q) \text{ s.t. } \sigma'(\mathbf{d}_{1-b}[i^*]) = \hat{\mathbf{m}}_{1-b}[i^*]$$
$$\forall j > j^*, \sigma_{i^*}(j) := \sigma'(j)$$
$$\forall i > i^*, \sigma_i \leftarrow\$ \mathsf{Perm}(\mathbb{Z}_q) \text{ s.t. } \sigma_i(\mathbf{d}_{1-b}[i]) = \hat{\mathbf{m}}_{1-b}[i]$$
$$\forall i \in [n], \mathbf{M}_{1-b,i} \leftarrow\$ \mathbb{Z}_q^{2\times q} \text{ s.t.}$$
$$\sum_{k\in\{0,1\}} \mathbf{M}_{1-b,i}[k,j] = \sigma_i(j)$$
$$\forall i < i^*, \mathbf{v}_{1-b,i} := \psi(\mathbf{M}_{1-b,i}[0,*])$$
$$\forall i \geq i^*, \mathbf{v}_{1-b,i} := (-1)^{\mathbf{c}_{1-b}[i]} \psi(\mathbf{M}_{1-b.i}[\mathbf{c}_{1-b}[i], *])$$
$$\forall (i \neq i^* \vee j \neq j^* \vee k \neq \mathbf{c}_{1-b}[i]), com_{i,k,j} \leftarrow\$ \mathsf{Commit}(\mathbf{M}_{1-b,i}[k,j])$$
$$com_{i^*,\mathbf{c}_{1-b}[i^*],j^*} \leftarrow\$ \mathcal{O}_{\mathsf{Commit}}(\sigma'[j^*] - \mathbf{M}_{1-b,i^*}[\mathbf{c}_b[i^*], j^*], \mathbf{M}_{1-b,i^*}[\mathbf{c}_{1-b}[i^*], j^*])$$
$$\gamma := \{\mathbf{v}_{0,i}, \mathbf{v}_{1,i}, \mathsf{Commit}(\mathbf{M}_{b,i}), (com_{i,k,j})_{k\in\{0,1\}, j\in[q]}\}_{i\in[n]}.$$

Since A does not open $com_{i^*,\mathbf{c}_{1-b}[i^*],j^*}$, he can easily simulate the third round:

$$\mathbf{c}_b = \beta - \mathbf{c}_{1-b}$$
$$\delta_{\mathbf{c}} := \mathsf{Open}(\mathbf{M}_{k,i}[\mathbf{c}_k[i], *])_{k \in \{0,1\}, i \in [n]}$$
$$\delta_{\mathbf{d}} := \mathsf{Open}(\mathbf{M}_{k,i}[1 - \mathbf{c}_k[i], \mathbf{d}_k[i]])_{k \in \{0,1\}, i \in [n]}.$$

If the challenger of the commitment security game commits to message $\sigma'(j^*) - \mathbf{M}_{1-b,i^*}[\mathbf{c}_b[i^*], j^*]$, attacker A simulates hybrid $HYB_{0,i^*,j^*}(\lambda, b)$, and otherwise if the challenger commits to $\mathbf{M}_{1-b,i^*}[\mathbf{c}_{1-b}[i^*], j^*]$ the attacker simulates hybrid $HYB_{0,i^*,j^*+1}(\lambda, b)$. This concludes the proof of this claim. □

Clearly, the distribution of hybrid $HYB_1(\lambda, b)$ is independent of bit b. Therefore, $HYB_1(\lambda, 0) \equiv HYB_1(\lambda, 1)$. This and the previous claim result in the statement of the lemma. □