Two-Round Adaptively Secure MPC from Isogenies, LPN, or CDH

Navid Alamati[1], Hart Montgomery[2], Sikhar Patranabis[3], Pratik Sarkar[4]

[1] Visa Research[***]
[2] Linux Foundation
[3] IBM Research India[†]
[4] Boston University

Abstract. We present a new framework for building round-optimal (two-round) *adaptively* secure MPC. We show that a relatively weak notion of OT that we call *indistinguishability OT with receiver oblivious sampleability* (r-iOT) is enough to build two-round, adaptively secure MPC against *malicious* adversaries in the CRS model. We then show how to construct r-iOT from CDH, LPN, or isogeny-based assumptions that can be viewed as group actions (such as CSIDH and CSI-FiSh). This yields the first constructions of two-round adaptively secure MPC against malicious adversaries from CDH, LPN, or isogeny-based assumptions. We further extend our non-isogeny results to the plain model, achieving (to our knowledge) the first construction of two-round adaptively secure MPC against semi-honest adversaries in the plain model from LPN.

Our results allow us to build a two-round adaptively secure MPC against malicious adversaries from essentially all of the well-studied assumptions in cryptography. In addition, our constructions from isogenies or LPN provide the first post-quantum alternatives to LWE-based constructions for round-optimal adaptively secure MPC. Along the way, we show that r-iOT also implies non-committing encryption (NCE), thereby yielding the first constructions of NCE from isogenies or LPN.

[***] Most of the work was done while the author was affiliated with UC Berkeley.
[†] Most of the work was done while the author was affiliated with ETH Zürich.

Table of Contents

1 Introduction

Secure multiparty computation (MPC) allows mutually distrusting parties to jointly evaluate functions of their secret inputs in a manner that doesn't reveal any information outside of the final output. More precisely, an MPC protocol involves n parties $P_1, \ldots, P_n$ with private inputs $x_1, \ldots, x_n$ such that, at the end of the protocol, each party P_i learns an output of the form $f_i(x_1, \ldots, x_n)$ but nothing else about the private inputs of any other party.

MPC has been extensively studied since the 1980s [Yao86, GMW87] and is currently used in practice for a wide variety of applications, such as privacy-preserving studies for social good [LJA+18], privacy-preserving online advertising [IKN+17], distributed key management [unb], and securely instantiating blockchain protocols [CCD+20].

MPC constructions are closely related to (and often based upon) another widely studied primitive called *oblivious transfer* (OT) [Rab05, EGL82]. Informally speaking, an OT protocol involves a *sender* holding two messages m_0 and m_1, and a receiver holding a bit b. At the end of the protocol, the receiver should only learn the message m_b and nothing about m_{1-b}, while the sender should learn nothing about the bit b. Due to its wide range of applications, OT has been studied extensively in a long line of works [NP01, PVW08, BD18, FMV19, DGH+20, LGdSG21, CSW20, ADMP20].

Models and Round Complexity. Given the ubiquity of MPC in cryptography, it is no surprise that MPC protocols have been studied in many different security models. Examples of such models include *semi-honest/malicious* as well as *static/adaptive* adversarial corruptions. MPC has also been studied in a variety of computational models such as the plain model and the common reference string (CRS) model. An important feature of any MPC protocol is its *round complexity* (i.e., the number of rounds of communication between the parties during protocol execution). Minimal round complexity is desirable when communication time dominates computational cost, which is the case in many practical protocols. So, designing *round-optimal* MPC protocols is widely regarded to be an important topic in MPC research.

The Static Corruption Model. In the *static* corruption model for MPC, the adversary is allowed to corrupt a pre-determined set of parties. A long line of works have shown how to design round-optimal MPC protocols in this model from a variety of assumptions in the CRS model [GGHR14, MW16, CPV17a]. Notably, [BL18, GS18] showed how to construct two-round MPC protocols from two-round OT protocols in different security models and computational settings.

In terms of concrete computational assumptions, two-round maliciously secure OT protocols in the static corruption model have been constructed from DDH, QR/DCR, and LWE [NP01, PVW08, HK12, BD18]. More recently, such OT protocols have been designed from the CDH and LPN assumptions [DGH+20], as well as from isogenies of elliptic curves [ADMP20]. To summarize, we can currently build round-optimal maliciously secure MPC in the static corruption model from essentially all of the commonly used computational assumptions.

Limitations of the Static Corruption Model. Unfortunately, the static corruption model for MPC is not strong enough for certain real-world applications. In particular, the static corruption model does not provide security against "hacking attacks" where an adversary might adaptively corrupt parties at different stages of the protocol. For instance, what happens if the adversary seizes control of the parties' machines through backdoor access? Secure erasures of the party's state upon corruption is one possible solution to tackle such an attack. However, it is an impractical solution as argued by [CFGN96] since it requires the party to detect an attack and honestly execute its erasure of internal state. This motivates designing MPC protocols that are secure in the *adaptive* corruption model without relying on secure erasures. In this work we refer to adaptive security in the non-erasure model as adaptive security.

The Adaptive Corruption Model. In the *adaptive* corruption model for MPC, the adversary is allowed to dynamically corrupt any set of parties at any time during the protocol execution. Canetti *et al.* [CDD+04] presented the first formal investigation of the adaptive corruption model for MPC, and the relationships between adaptive security and static security in several models of computation. Garay *et al.* [GWZ09] showed how to construct adaptively secure two-party computation protocols in a generic manner from OT protocols

satisfying a weaker notion of *semi-adaptive security*; they also showed how to obtain semi-adaptively secure OT protocols from somewhat non-committing encryption (NCE), which is a weaker variant of standard NCE [CLOS02]. Subsequently, Hazay *et al.* [HV15] showed that adaptively secure MPC protocols can be obtained from minimal assumptions like trapdoor simulatable public key encryption (PKE).

However, the scenario is different once round optimality is taken into consideration. It is currently open to design round-optimal maliciously secure MPC protocols even from certain commonly used computational assumptions such as CDH, LPN and isogeny-based assumptions.[5] Initial works on two round, adaptively secure MPC relied on indistinguishability obfuscation (and other standard assumptions) [CGP15, GP15, CPV17a, CsW19] or assuming secure erasures[6] [CsW19] of the party's internal states. There are adaptively secure protocols [GOS12, CSW22] for NIZKs as well from specific assumptions like pairings, DDH+LPN and LWE. There are adaptively secure constant round MPC protocols [GS12, CGPS21] in the plain model based on non-blackbox simulation techniques since it is impossible to obtain constant round adaptively secure protocol without setup with blackbox simulation. [LLW20] constructs a two-round information theoretic MPC protocol in the honest-majority setting against semi-honest adversaries which can be adaptively corrupt.

The work of Benhamouda *et al.* [BLPV18] was the first to show how to construct round-optimal adaptively universal composability (UC) [Can01] secure MPC protocols from certain standard computational assumptions without obfuscation and erasures. More concretely, they established the following:

- Against *semi-honest* adversaries, adaptively UC-secure two-round MPC in the plain model is implied by non-committing encryption (NCE) [CLOS02], which in turn can be built from CDH/DDH, LWE, and RSA [CDMW09].
- Against *malicious* adversaries, adaptively UC-secure two-round MPC in the CRS model can be built from a certain kind of two-round statically secure OT protocol with additional "oblivious sampleability" properties, which in turn can be based on DDH, QR, and LWE.

The work of [CSW20] constructs a two round adaptively secure MPC protocol based on the DDH assumption. It is currently open to construct round-optimal (i.e., two-round) maliciously secure MPC protocols in the adaptive corruption model from commonly studied assumptions such as CDH, LPN and isogeny-based assumptions. In particular, the constructions of Benhamouda *et al.* [BLPV18] crucially rely on certain primitives such as "obliviously sampleable" smooth projective hash functions (SPHFs) and "augmented" non-committing encryption (NCE) that are not known from some or all of these assumptions. More generally, it is not known how to construct such MPC protocols from a *single* generic primitive that can be built from commonly used computational assumptions.

Moreover, there are motivating concerns about efficient quantum computing and adaptive MPC. Currently, the only plausibly post-quantum secure constructions [BLPV18, CsW19] of two-round maliciously secure MPC protocols in the adaptive corruption model are based on LWE. This lack of diversity in post-quantum constructions is potentially concerning since a major advance in lattice cryptanalysis could substantially degrade (or in the worst case, invalidate) the security of LWE-based constructions for all practical parameter sets. Notably, the recent NIST competition to standardize post-quantum cryptosystems [CJL+16, AAAS+19, AASA+20] considers a wider class of post-quantum assumptions, including isogeny-based assumptions. In this paper, we ask the following question:

Can we construct two round adaptively UC-secure MPC protocols from a wider class of assumptions, such as CDH, LPN, and isogeny-based assumptions?

1.1 Our Contributions

We answer this above question in the affirmative. We establish a new route to achieving two round maliciously UC-secure MPC protocols in the adaptive corruption setting that relies on potentially weaker (or

[5] Note that constant round maliciously secure MPC against adaptive corruptions can only be achieved in the CRS model; see [GS12] for results establishing the impossibility of maliciously secure adaptive MPC in the plain model from black-box simulation.

[6] The secure erasures model allows erasing the internal state of an honest party when its gets adaptively corrupted by the adversary. It is a strictly weaker model than the one we consider, where erasing the party's state is not allowed.

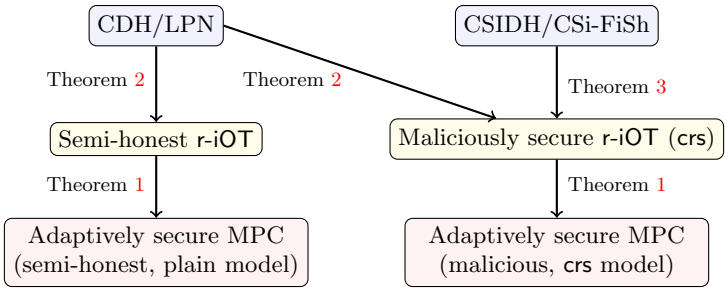

Fig. 1. A simplified overview of our results

"less structured") cryptographic primitives as compared to those used by Benhamouda *et al.* [BLPV18]. We also show how to instantiate these primitives from CDH, LPN, and certain families of isogeny-based assumptions (such as CSIDH [CLM+18] and CSI-FiSh [BKV19]). Our results thus establish the feasibility of realizing adaptively secure MPC from essentially *all* commonly used cryptographic assumptions.

We present our results in the "local" CRS model where every session of protocol execution has a local independently sampled CRS string. This is the same model in which Benhamouda *et al.* [BLPV18] described their constructions and proofs. The only other work [CSW20] in this setting is in the single common random string model, but it is solely based on DDH. We note here that Choi *et al.* [CKWZ13] achieved efficient, adaptively secure, composable OT protocols with a single, global CRS, albeit from a different set of concrete assumptions as compared to what we consider in this paper.

Our Ingredients. Our constructions of two-round, adaptively UC-secure MPC essentially rely on a *single* building block, which we refer to as *indistinguishability OT with receiver oblivious sampleability* (r-iOT). Informally, r-iOT is a two-message OT protocol that satisfies indistinguishability security [DGH+20] against the sender and the receiver in the *static* corruption model, while also satisfying an additional property called "receiver oblivious sampleability". At a high level, this property requires that it is possible to obliviously sample the OT receiver's message (without knowledge of any secret randomness and receiver's choice bit). This property also requires an algorithm for claiming that an honestly generated receiver's message was, in fact, obliviously sampled.

We note that the concept of receiver oblivious sampleable OT was introduced and used in their constructions by Benhamouda *et al.* [BLPV18]. However, our constructions rely on a *strictly weaker* set of properties for our starting r-iOT protocol. First of all, the constructions in [BLPV18] assume that the starting OT protocol satisfies (simulation-based) UC-security against a semi-honest sender and a malicious receiver in the static corruption model, i.e. a malicious receiver's input can be extracted. On the other hand, our starting r-iOT protocol is only required to achieve a strictly weaker notion of indistinguishability security, which we subsequently bootstrap all the way to full-fledged UC security via a sequence of transformations. Additionally, the constructions in [BLPV18] assume that the starting OT protocol satisfies *both* receiver and sender oblivious sampleability, while we show a generic construction to build such an OT protocol (with both receiver and sender oblivious sampleability) starting our r-iOT protocol which satisfies only receiver oblivious sampleability.

Main Results. Figure 1 summarizes the main results of this paper. We compare the state-of-the-art results and our result in Fig. 2. Our first main result is a generic construction of UC-secure two-round adaptive MPC from any r-iOT protocol. In somewhat more detail, our first result can be summarized as follows:

Theorem 1 (Informal). *Assuming* r-iOT*, i.e, a two-message OT protocol that satisfies indistinguishability security and receiver oblivious sampleability against static corruption of the sender/receiver by malicious adversaries in the CRS model (resp. semi-honest adversaries in the plain model), there exists a two-round MPC protocol for any functionality f that satisfies UC security against adaptive corruption of any subset of the parties by malicious adversaries in the CRS model (resp. semi-honest adversaries in the plain model).*

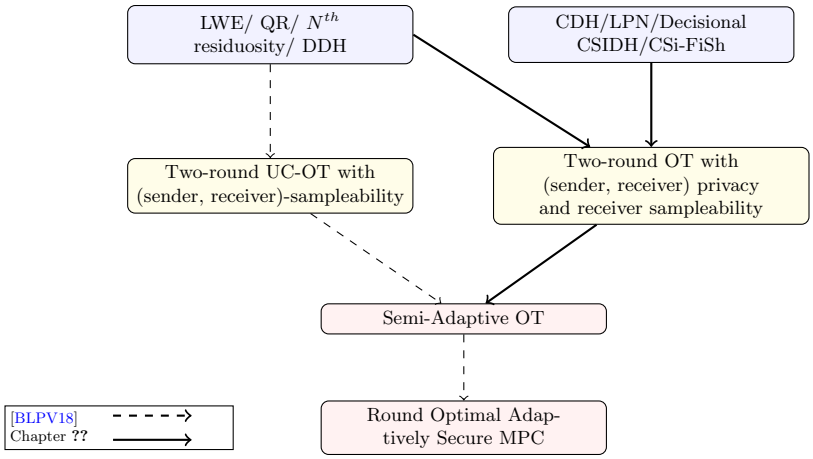

Fig. 2. Comparison of Adaptively Secure MPC Protocols

We achieve this result via a sequence of transformations that build progressively stronger OT protocols from weaker ones. These transformations use a number of additional cryptographic primitives, all of which we show can be built in a generic way from any r-iOT protocol in the appropriate model.

Next, we show how to instantiate an r-iOT protocol in various models from a variety of concrete assumptions, including CDH, LPN, and isogeny-based assumptions. In somewhat more details, our second main result can be summarized as follows:

Theorem 2 (Informal). *Assuming CDH or LPN, there exists a construction of* r-iOT *that is secure against malicious adversaries in the CRS model (resp. semi-honest adversaries in the plain model).*

Theorem 3 (Informal). *Under certain isogeny-based assumptions (notably, CSIDH [CLM+18] or CSI-FiSh [BKV19]), there exists a construction of* r-iOT *that is secure against malicious adversaries in the CRS model.*

Our constructions of r-iOT from CDH and LPN build upon previous work due to Döttling *et al.* [DGH+20] that realized UC-secure OT/MPC against static corruptions from the same set of assumptions. Our construction of r-iOT from isogeny-based assumptions is based on a novel usage of the *(restricted) effective group action* framework due to Alamati *et al.* [ADMP20]. In particular, we show how to use a trusted setup to bypass issues around sampling obliviously from the "set" of an effective group action, which is a well-known open problem in the isogeny literature [Pet17, DMPS19, CPV20].[7]

Combined with the previous theorem, we obtain as a corollary the *first* constructions of two-round adaptively UC-secure MPC against malicious adversaries from the same concrete assumptions:

Corollary 1 (Informal). *Assuming CDH, LPN, or certain isogeny-based assumptions (notably, CSIDH [CLM+18] or CSI-FiSh [BKV19]), there exists a two-round MPC protocol for any functionality f that satisfies UC security against adaptive corruption of any subset of the parties by malicious adversaries in the CRS model.*

In summary, we show that it is feasible to construct round-optimal maliciously secure MPC in the adaptive corruption model from essentially all of the commonly used cryptographic assumptions. This essentially closes the gap between the static corruption model and the adaptive corruption model in terms of constructing round-optimal maliciously secure MPC from concrete assumptions. Figure 3 presents a high-level summary of our roadmap from r-iOT to adaptively UC-secure MPC.

[7] Unlike CDH or LPN, we do not achieve a construction of r-iOT from isogeny-based assumptions in the plain model. Achieving this seemingly requires new techniques for sampling obliviously from the "set" of an effective group action beyond those used in state-of-the-art isogeny-based cryptography.

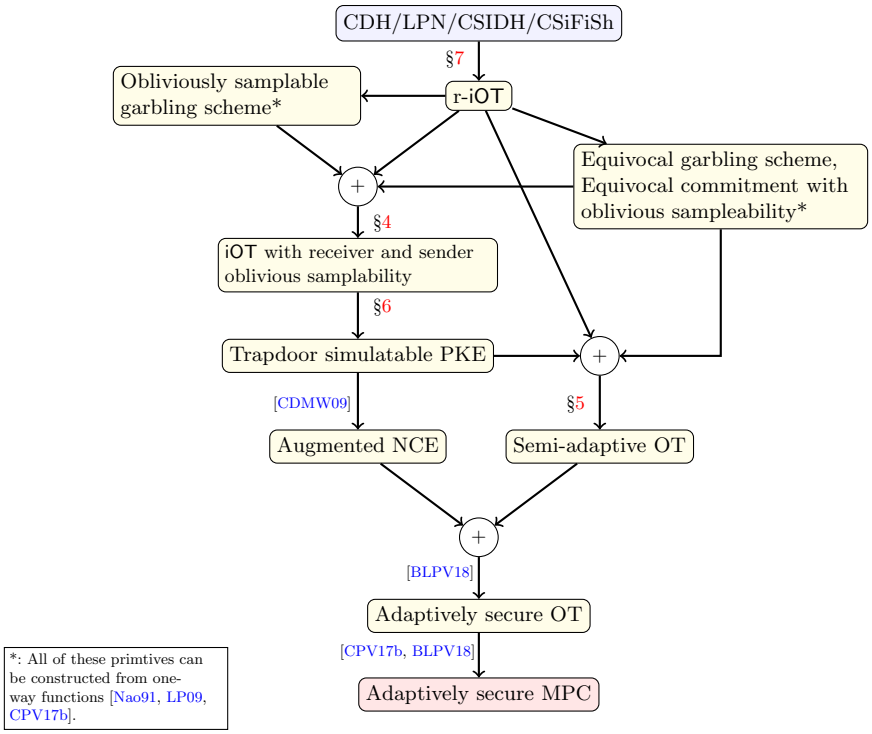

Fig. 3. An overview of our results

Additional Results. Besides our main contributions, we show some additional results that could be of independent interest. In particular, we show that any r-iOT protocol that is secure against semi-honest adversaries implies the existence of a *trapdoor-simulatable* PKE, which in turn is known to imply *non-committing encryption* (NCE) in a generic manner (in fact, it was shown in [CDMW09] that trapdoor-simulatable PKE implies an "augmented" variant of NCE). Due to its wide range of applications, NCE (and its augmented variants) have been studied by a long line of works [CFGN96, CDMW09, CPR17, YKT19, BBD+20].

Theorem 4 (Informal). *Any* r-iOT *that is secure against semi-honest adversaries implies a construction of trapdoor-simulatable PKE.*

Combined with the previous theorem on instantiations of r-iOT from concrete assumptions, and the known implication due to [CDMW09], we obtain as a corollary the *first* constructions (to our knowledge) of (augmented) NCE from LPN or certain isogeny-based assumptions:

Corollary 2 (Informal). *Assuming LPN, there exists a construction of a two round (augmented) non-committing encryption (NCE) scheme.*

Corollary 3 (Informal). *Assuming isogeny-based assumptions such as CSIDH [CLM+18] or CSI-FiSh [BKV19], there exists a construction of a two round (augmented) non-committing encryption (NCE) scheme in the CRS model.*

Outline. The rest of the paper is organized as follows. Section 2 provides a high-level overview of our compiler. Section 3 presents notations and definitions for two-round OT protocols in the CRS model. Section 4 describes our construction of two-round iOT with both receiver and sender oblivious sampleability from any two-round r-iOT protocol. Section 5 describes our construction (and proof) of semi-adaptively secure two-round OT from any two-round iOT with both receiver and sender oblivious sampleability. Combining with the result of [BLPV18] we obtain an adaptively secure MPC protocol in the CRS model. Section 6 presents

our construction of trapdoor simulatable PKE (and augmented NCE) from any two-round r-iOT protocol. Section 7 describes our concrete constructions of two-round r-iOT from isogeny-based assumptions, CDH or LPN.

2 Technical Overview

In this section, we present an overview of our proposed route to achieving two round maliciously secure MPC protocols in the adaptive corruption setting, and compare this roadmap to that in the work of Benhamouda *et al.* [BLPV18]. In particular, we present a discussion of why it is seemingly difficult to instantiate the framework of [BLPV18] from concrete assumptions such as CDH, LPN and certain isogeny-based assumptions. We then aim to elucidate how our framework relies on inherently "less powerful" primitives that can actually be instantiated from these assumptions.

The [BLPV18] Construction. As already mentioned, [BLPV18] was the first work to show how to construct round-optimal malicious-adaptively secure MPC protocols from standard computational assumptions such as DDH, QR, and LWE. Their construction uses a sequence of transformations that progressively build OT protocols with stronger security guarantees from ones with weaker security guarantees (and some additional cryptographic primitives). Here, we focus on the weakest OT protocol that is essentially the starting point of their construction. At a high level, this (two-round) OT protocol is required to satisfy the following properties:

- *Universal composability* (UC) security against a malicious receiver[8] and a semi-honest sender in the static corruption model.
- *Receiver oblivious sampleability*, which requires the possibility of *obliviously* sampling the message from the receiver to the sender (without knowing any secrets), and also the possibility of claiming that any honestly generated receiver's message was, in fact, obliviously sampled.
- *Sender oblivious sampleability*, which requires the possibility of *obliviously* sampling the message from the sender to the receiver (again without knowing any secrets), and also the possibility of claiming that any honestly generated sender's message (for random input strings) was obliviously sampled.

In [BLPV18], Benhamouda *et al.* showed how to instantiate such a two-round OT protocol in the plain/CRS model from a variety of mainstream cryptographic assumptions, including DDH, QR, and LWE.

Barriers to More Instantiations. Unfortunately, it seems inherently difficult to instantiate such a two-round OT protocol from other concrete assumptions such as CDH, LPN, and isogeny-based assumptions. We outline some reasons for why this seems to be the case for each of the assumptions below.

- The two-round OT constructions from CDH and LPN due to [DGH+20] do achieve UC security against a malicious receiver in the static corruption model. However, the final UC-secure construction is achieved through a sequence of generic transformations, and it is unclear how to argue receiver/sender oblivious sampleability for this final construction. While it could be potentially easier to argue receiver/sender oblivious sampleability for some of the weaker OT protocols at various stages of this transformation process, these weaker protocols do not satisfy UC security for a maliciously corrupt receiver.
- The two-round OT construction from the *(restricted) effective group action* framework due to [ADMP20] achieves UC security against a maliciously corrupt receiver in the static corruption model. It can also be instantiated from certain families of isogeny-based assumptions such as CSIDH [CLM+18] and CSI-FiSh [BKV19]. However, this construction does not satisfy receiver/sender oblivious sampleability; in particular, achieving such properties in a straightforward manner would require the ability to sample obliviously from the "set" of an effective group action, which is a well-known open problem in the isogeny literature and is believed to intractable from state-of-the-art isogeny techniques (see [Pet17, DMPS19, CPV20] for more details).

[8] UC security requires that the malicious party's input is extractable.

To summarize, while there do exist two-round OT protocols in the static setting with UC security against maliciously corrupt receiver from CDH, LPN, and isogeny-based assumptions, these do not satisfy the necessary oblivious sampleability properties, and hence, cannot be used to instantiate the [BLPV18] construction of adaptive MPC.

Difficulty in Mirroring Similar Techniques. We additionally argue that it is difficult to mirror the techniques used by [BLPV18] in the context of CDH, LPN, and isogeny-based assumptions. We again outline why this seems to be the case for each of the assumptions below.

- In [BLPV18], the authors use smooth projective hash proof systems with certain special oblivious sampleability properties as a starting point for their DDH-based instantiation. On one hand, we do no know how to instantiate hash proof systems from the CDH assumption; in particular, unlike DDH, CDH does not naturally support an "algebraically structured" hard membership language that is a prerequisite for any hash proof system. On the other hand, the known constructions of hash proof systems from isogenies [ADMP20] do not satisfy the special oblivious sampleability properties for the same reasons as described earlier.
- In [BLPV18], the authors use an LWE-based construction of *half-OT*, which does not translate naturally to the LPN setting. This is because the LWE-based construction in [BLPV18] crucially relies on (full-rank) lattice trapdoors (and associated techniques from [GPV08]) for which there are no known LPN-based analogues. There are also no known constructions of smooth projective hash proof systems from LPN.

To summarize, it seems hard to import the techniques used by [BLPV18] to construct adaptively secure MPC to the setting of CDH, LPN, and isogeny-based assumptions. At a high level, these difficulties arise from the fact that these concrete assumptions lack the necessary structure that is seemingly needed to support the techniques in [BLPV18]. While it might be possible to overcome of these structural limitations using novel cryptographic techniques, we opt for a different strategy - that of weakening the necessary generic assumption to build adaptive MPC to the point where instantiations from CDH, LPN, and isogenies become feasible based on a combination of known techniques.

Weakening the Generic Assumption. Our main technical contribution is showcasing a new set of techniques for realizing adaptively secure MPC from a strictly weaker (two-round) OT protocol as compared to the [BLPV18] construction. In particular, our starting point is a (two-round) OT protocol that satisfies the following properties (we refer to this primitive as r-iOT throughout the paper):

- Indistinguishability security against a sender and the receiver in the static corruption model (this is a strictly weaker requirement as compared to UC security required by the [BLPV18] construction).
- *Only* receiver oblivious sampleability; in particular, sender oblivious sampleability need not be satisfied.

We show how to bootstrap this r-iOT protocol with seemingly weak security guarantees all the way up to our final goal of UC-secure adaptive MPC. We achieve this via a sequence of transformations. At a high level, similar to [BLPV18], our transformations also progressively build OT protocols with stronger security properties from ones with weaker security properties. However, the ingredients and techniques used differ significantly. Figure 3 presents a high-level summary of our roadmap from r-iOT to adaptively UC-secure MPC. We expand more on this below.

Our Roadmap from r-iOT to Adaptively Secure MPC. The first step in our roadmap is a generic construction of (two-round) OT satisfying indistinguishability security in the static corruption model and *both* receiver and sender oblivious sampleability starting from any generic r-iOT protocol. In a bit more detail, this step can be summarized as follows:

Theorem 5 (Informal). *Assuming* r-iOT *secure against malicious adversaries in the CRS model, there exists a two-round OT protocol that satisfies indistinguishability security and* ***both*** *receiver and sender oblivious sampleability against static corruption of the sender/receiver by malicious adversaries in the CRS model.*

This transformation is detailed in Section 4. Besides r-iOT, we require three additional cryptographic primitives:

- A non-interactive commitment scheme with oblivious sampleability [Nao91].
- A garbling scheme with oblivious sampleability [LP09, HV16].
- An equivocal garbling scheme [CPV17b].

Based on known results [Nao91, LP09, HV16, CPV17b], all of these primitives can be built in a generic way from one-way functions, and hence, from any generic r-iOT protocol. In other words, we do not need any additional assumptions for this step in the roadmap.

Bootstrapping to Semi-Adaptively Secure OT. The second step in our roadmap is a construction of (two-round) *semi-adaptively* secure OT protocol from any (two-round) OT protocol satisfying indistinguishability security in the static corruption model and receiver oblivious sampleability, trapdoor simulatable PKE (PKE where both the public key and the ciphertext can be obliviously sampled) and one-way function. We build the trapdoor simulatable PKE from the two round OT with *both* receiver and sender oblivious sampleability. We provide a high-level recap of what is meant by *semi-adaptive* security. Note that we use the same notion of semi-adaptive security for OT as used in previous works [GWZ09, ABP17, BLPV18, CSW20].

Informally, an OT protocol is said to be secure against *semi-adaptive sender corruptions* if it is secure in a model where the adversary can adaptively corrupt the sender, but is restricted to corrupting the receiver only in a static manner. Similarly, an OT protocol is said to be secure against *semi-adaptive receiver corruptions* if it is secure in a model where the adversary can adaptively corrupt the receiver, but is restricted to corrupting the sender only in a static manner. Finally, an OT protocol is said to be semi-adaptively secure if it is simultaneously secure against *both* semi-adaptive sender *and* receiver corruptions.

At this point, we summarize the construction of our semi-adaptive OT in our sequence of transformations as follows:

Theorem 6 (Informal). *Assuming a two-round OT protocol that satisfies indistinguishability security and receiver oblivious sampleability against static corruption of the sender/receiver by malicious adversaries in the CRS model, and a trapdoor simulatable PKE, then there exists a two-round OT protocol that satisfies UC security against semi-adaptive corruption of the parties by malicious adversaries in the CRS model.*

This transformation is detailed in Section 5. Besides r-iOT, we also require a trapdoor simulatable PKE scheme [CDMW09] (and some other ingredients implied by any one-way function). Finally, we show in Section 6 that any r-iOT protocol that is secure against semi-honest adversaries in the CRS model implies a construction of trapdoor-simulatable PKE. We note here that the authors of [BLPV18] also presented a construction of semi-adaptively UC-secure OT as a crucial step of their framework. In comparison with their construction, our construction relies on *significantly weaker* assumptions. In particular, the construction in [BLPV18] relies on a two-round OT protocol which is receiver and sender oblivious sampleable and the receiver's choice bit can be extracted from the OT first message. On the other hand, our construction relies entirely on primitives that are implied by any indistinguishability-secure two-round OT with receiver oblivious sampleability (with no extractability and no sender oblivious sampleability requirements).

Bootstrapping to Adaptive MPC. At this point, we import the following theorem from [BLPV18] to achieve our final goal of adaptively UC-secure MPC:

Theorem 7 (Informal, [BLPV18]). *Assuming an augmented NCE scheme and a two-round OT protocol that satisfies UC security against semi-adaptive corruption of the parties by malicious adversaries in the CRS model, there exists a two-round MPC protocol for any functionality f that satisfies UC security against adaptive corruption of any subset of the parties by malicious adversaries in the CRS model.*

Note that augmented NCE can be built from any (trapdoor-) simulatable PKE scheme (this was shown in [CDMW09]), which in turn can be built from r-iOT, as outlined earlier.

Having provided an overview of our proposed roadmap for building adaptively UC-secure two-round MPC from any two-round r-iOT, we now turn to instantiating the same from a variety of concrete assumptions, including isogeny-based assumptions, as well as CDH and LPN. We provide here an overview of how we build two-round r-iOT protocols from each of these assumptions.

Instantiation from Isogeny-Based Assumptions. In Section 7.1, we show how to construct a two-round r-iOT protocol secure against malicious adversaries in the CRS setting from certain isogeny-based assumptions (notably, CSIDH [CLM+18] or CSI-FiSh [BKV19]). More specifically, we assume the existence of a secure *(restricted) effective group action* equipped with appropriate computational hardness assumptions as described in [ADMP20].

The starting point of our construction of r-iOT is the construction of iOT from any (restricted) effective group action proposed originally in [ADMP20]. This construction offers all of the features we want, except for receiver oblivious sampleability. It turns out that achieving receiver oblivious sampleability for this construction in a straightforward manner would require the ability to sample obliviously from the "set" of a (restricted) effective group action. As already mentioned, this is a well-known open problem in the isogeny literature and is likely to require fundamentally new ideas beyond state-of-the-art techniques for isogeny-based cryptography (see [Pet17, DMPS19, CPV20] for more details).

In this paper, we propose a workaround for this by settling for a weaker notion of *trapdoor oblivious sampleability* for the "set" of a (restricted) effective group action. In other words, while it hard to obliviously sample a "set" element in the plain model, one can obliviously sample a "set" element given a specially designed trapdoor (corresponding to some public CRS). This is the core idea behind our construction of r-iOT from (restricted) effective group action. In view of the inherent restrictions outlined earlier, our workaround only allows us to achieve an r-iOT construction in the CRS model (and not in the plain model). However, in the CRS model, we achieve the desired notion of indistinguishability secure against malicious sender/receiver, while also achieving receiver oblivious sampleability.

Instantiation from CDH or LPN. In Section 7.2, we show that assuming CDH or LPN, there exists a construction of r-iOT that is secure against malicious adversaries in the CRS model. We can also extend the same results for semi-honest adversaries in the plain model and thus construct an adaptively secure semi-honest MPC from CDH and LPN in the plain model. This result follows from a simple observation on the two-round iOT constructions from CDH or LPN due to Döttling *et al.* [DGH+20].

Specifically, Döttling *et al.* showed that iOT can be constructed from a weaker notion of OT called elementary OT, and they demonstrated instantiations of two-round elementary OT from CDH or LPN. The generic transformation of [DGH+20] is done in two steps: (1) they first show how to build iOT from an intermediate primitive called search OT via parallel repetition, (2) they show how to construct search OT from elementary OT. At this point, we rely on the following (relatively straightforward) observations:

- The transformation preserves *receiver* oblivious sampleability at both steps. In other words, if the initial (two-round) elementary OT satisfies receiver oblivious sampleability, then the final iOT resulting from this transformation is, in fact, an r-iOT.
- The elementary OT constructions from CDH or LPN described in [DGH+20] already satisfy receiver oblivious sampleability.

Combining the aforementioned observations, we get constructions of r-iOT assuming CDH or LPN, as desired. We refer the reader to Section 7.2 for more details surrounding these observations and the corresponding implications. We believe that the simplicity of these observations essentially highlights the benefits of basing our construction on a much weaker notion of OT that follows more naturally from a wider variety of cryptographic assumptions. We hope that our roadmap makes it similarly easier to achieve adaptive MPC protocols from newer assumptions in the future.

3 Preliminaries

In this section, we present some core preliminaries that are integral to our constructions. We defer many definitions and other background with which we expect most readers to be familiar to the full version of our paper.

3.1 Notations

We denote by $a \leftarrow D$ a uniform sampling of an element a from a distribution D. The set of elements $\{1, \ldots, n\}$ is represented by $[n]$. We denote polylog(a) and poly(b) as polynomials in $\log a$ and b respectively. We denote a probabilistic polynomial time algorithm as PPT. We denote the computational security parameter by κ. We denote a negligible function in κ as $\mathsf{neg}(\kappa)$. When a party S gets corrupted we denote it by S^*. Our security proofs are in the Universal Composability (UC) framework of [Can01]. We refer to the original paper for details. We denote computational and statistical indistinguishability by $\overset{c}{\approx}$ and $\overset{s}{\approx}$ respectively. We abbreviate "common reference string" as CRS. Unless otherwise specified, our constructions and proofs are in "local" CRS model. This happens to be the same CRS model in which the prior work due to Benhamouda *et al.* [BLPV18] showed constructions of adaptive MPC protocols with security against malicious adversaries. We refer to Section A for the security model of adaptively secure MPC.

3.2 Two-Message Oblivious Transfer in the CRS Model

In this section, we formally define a two-message oblivious transfer (OT) protocol in the common reference string (CRS) model. We then define two security notions for such an OT protocol, namely universal composability (UC) security and a weaker notion of indistinguishability-based security. We first focus on security against static corruptions by a malicious adversary. Subsequently, we discuss different levels of adaptive security.

A two-message OT protocol in the CRS model is a tuple of four algorithms of the form $\mathsf{OT} = (\mathsf{Setup}, \mathsf{OTR}_1, \mathsf{OTS}, \mathsf{OTR}_2)$ described below:

- $\mathsf{Setup}(1^\kappa)$: Takes as input the security parameter κ and outputs a CRS string crs and a trapdoor td. [9]
- $\mathsf{OTR}_1(\mathsf{crs}, b \in \{0,1\})$: Takes as input the crs and a bit $b \in \{0,1\}$, and outputs the receiver's message $\mathsf{M_R}$ and the receiver's internal state st.
- $\mathsf{OTS}(\mathsf{crs}, \mathsf{M_R}, m_0, m_1)$: Takes as input the crs, the receiver's message $\mathsf{M_R}$, a pair of input strings (m_0, m_1), and outputs the sender's message $\mathsf{M_S}$.
- $\mathsf{OTR}_2(\mathsf{crs}, \mathsf{M_S}, b, \mathsf{st})$: Takes as input the crs, the sender's message $\mathsf{M_S}$, a bit b, and receiver's internal state st, and outputs a message string m'.

Correctness. A two-message OT protocol in the CRS model is said to be correct if for any $b \in \{0,1\}$ and any (m_0, m_1), letting $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{Setup}(1^\kappa)$ and $(\mathsf{M_R}, \mathsf{st}) \leftarrow \mathsf{OTR}_1(\mathsf{crs}, b)$, the following holds with overwhelming probability:

$$\mathsf{OTR}_2(\mathsf{crs}, \mathsf{OTS}(\mathsf{crs}, \mathsf{M_R}, m_0, m_1), b, \mathsf{st}) = m_b.$$

Corruption Models. We consider the following (progressively non-decreasing in strength) adversarial models against any two-message OT protocol:

- *Static Corruption:* The adversary corrupts the parties at the onset of the protocol.
- *Semi-Adaptive Corruption:* The adversary corrupts one party (either the receiver or the sender) adaptively (at any point before/during/after the protocol) and the other party statically at the beginning of the protocol.
- *Adaptive Corruption:* The adversary corrupts both parties adaptively (at any point before/during/after the protocol). This scenario covers the previous corruption cases.

Indistinguishability-Based Security. We also consider a weaker notion of indistinguishability-based security against malicious adversaries in the static corruption setting. This notion is adopted directly from [DGH+20]. A two-message OT protocol $\mathsf{iOT} = (\mathsf{Setup}, \mathsf{iOTR}_1, \mathsf{iOTS}, \mathsf{iOTR}_2)$ satisfies indistinguishability-based security if the following properties hold:

[9] For standard two-message OT protocols, the setup algorithm need not output a trapdoor td, but we include it for certain security properties described subsequently.

Receiver's Indistinguishability Security. Formally, receiver's indistinguishability security requires that the following holds for any $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{Setup}(1^\kappa)$:

$$(\mathsf{crs}, \mathsf{iOTR}_1(\mathsf{crs}, 0)) \overset{c}{\approx} (\mathsf{crs}, \mathsf{iOTR}_1(\mathsf{crs}, 1)).$$

Sender's Indistinguishability Security. Sender's indistinguishability security is defined in [DGH+20] via an experiment $\mathsf{Exp}_{\mathsf{iOT}}^{\mathsf{crs},r,w,b}(\mathcal{A})$ between a non-uniform PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ and a challenger, where the experiment is parameterized by some honestly generated crs, random coins $r \in \{0,1\}^\kappa$, an integer n representing the bitwise length of messages, a bit $w \in \{0,1\}$, and a bit $b \in \{0,1\}$:

$\underline{\mathsf{Exp}_{\mathsf{iOT}}^{\mathsf{crs},r,w,b}(\mathcal{A})}$:

1. Run $(m_0, m_1, \mathsf{M_R}, \mathsf{st}) \leftarrow \mathcal{A}_1(1^\kappa, \mathsf{crs})$.
2. If $b = 0$, compute $\mathsf{M_S} \leftarrow \mathsf{iOTS}(\mathsf{crs}, \mathsf{M_R}, (m_0, m_1))$.
3. If $b = 1$, compute $\mathsf{M_S} \leftarrow \mathsf{iOTS}(\mathsf{crs}, \mathsf{M_R}, (m_0', m_1'))$ where $m_w' \leftarrow \{0,1\}^n$ and $m_{1-w}' := m_{1-w}$.
4. Output $s \leftarrow \mathcal{A}_2(\mathsf{st}, \mathsf{M_S})$.

For a given $(\mathsf{crs}, r, w \in \{0,1\})$, we define the advantage $\mathcal{A}_{\mathsf{iOT}}^{\mathsf{crs},r,w}(\mathcal{A})$ as:

$$\mathsf{Adv}_{\mathsf{iOT}}^{\mathsf{crs},r,w}(\mathcal{A}) = |\Pr[\mathsf{Exp}_{\mathsf{iOT}}^{\mathsf{crs},r,w,0}(\mathcal{A}) = 1] - \Pr[\mathsf{Exp}_{\mathsf{iOT}}^{\mathsf{crs},r,w,1}(\mathcal{A}) = 1].$$

We say that iOT satisfies sender's indistinguishability security if for any PPT adversary $\mathcal{A}$, $\mathsf{Adv}_{\mathsf{iOT}}^{\mathsf{crs},r,w}(\mathcal{A})$ is negligible in κ for at least one $w \in \{0,1\}$, where the probability is taken over $\mathsf{crs} = \mathsf{Setup}(1^\kappa)$ and $r \leftarrow \{0,1\}^\kappa$.[10]

3.3 iOT with Oblivious Sampleability

We also consider notions of *oblivious sampleability* for indistinguishability-secure two-message OT protocols in the CRS model. An iOT protocol of the form $\mathsf{iOT} = (\mathsf{Setup}, \mathsf{iOTR}_1, \mathsf{iOTS}, \mathsf{iOTR}_2)$ is said to satisfy oblivious sampleability if it supports additional "oblivious sampling" algorithms - $(\widetilde{\mathsf{iOTR}}, \widetilde{\mathsf{iOTS}})$ and the corresponding "randomness inversion" algorithms - $(\widetilde{\mathsf{iOTR}_{\mathsf{Inv}}}, \widetilde{\mathsf{iOTS}_{\mathsf{Inv}}})$ defined as:

- $\widetilde{\mathsf{iOTR}}(\mathsf{crs}; r)$: Outputs an obliviously sampled receiver's message $\mathsf{M_R}$.
- $\widetilde{\mathsf{iOTS}}(\mathsf{crs}, \mathsf{M_R}, w, m_{1-w}; r)$ Outputs an obliviously sampled sender's message $\mathsf{M_S}$ where sender's OT protocol message for choice bit w is obliviously sampled and sender's message for choice bit $1-w$ is m_{1-w}. If $w = m_{1-w} = \bot$, then the sender samples OT protocol messages for both choice bits 0 and 1 obliviously.
- $\widetilde{\mathsf{iOTR}_{\mathsf{Inv}}}(\mathsf{crs}, \mathsf{M_R}, \mathsf{td}, r)$: Outputs randomness $\tilde{r}$ corresponding to an honestly generated receiver message $\mathsf{M_R}$ where $\mathsf{M_R}$ is claimed to be obliviously generated.
- $\widetilde{\mathsf{iOTS}_{\mathsf{Inv}}}(\mathsf{crs}, w, \mathsf{M_S}, \mathsf{td}, r)$: Outputs randomness $\tilde{r}$ corresponding to an honestly generated sender message $\mathsf{M_S}$ where branch w is claimed to be obliviously sampled. If $w = \bot$, then both branches are claimed to be obliviously sampled.

We say that the iOT is obliviously sampleable if it satisfies *both* receiver and sender oblivious sampleability, as defined below.

Receiver Oblivious Sampleability: For any bit $b \in \{0,1\}$, an obliviously sampled receiver's message should be indistinguishable from an honestly generated one, even given the sampling randomness. More formally, we require that for any $(\mathsf{crs}, \mathsf{td}) = \mathsf{Setup}(1^\kappa)$ and any bit $b \in \{0,1\}$, we have $(\mathsf{crs}, \mathsf{M_R}, \widehat{r}) \overset{c}{\approx} (\mathsf{crs}, \widetilde{\mathsf{M_R}}, \widetilde{r})$, where for uniformly random coins $r, \widetilde{r} \leftarrow \{0,1\}^\kappa$, we have

$$\mathsf{M_R} = \mathsf{iOTR}(\mathsf{crs}, b; r), \quad \widehat{r} = \widetilde{\mathsf{iOTR}_{\mathsf{Inv}}}(\mathsf{crs}, \mathsf{M_R}, \mathsf{td}, r), \quad \widetilde{\mathsf{M_R}} = \mathsf{iOTR}(\mathsf{crs}, b; \widetilde{r}).$$

[10] This is slightly different from the traditional notion of sender's indistinguishability security for two-message OT; we refer to [DGH+20] for more details.

Sender Oblivious Sampleability: We also require that a corrupt receiver cannot infer whether the sender's message (corresponding to the bit w which is not chosen by the receiver) was obliviously sampled or generated honestly. We consider an adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ participating in an experiment $\mathsf{Exp}^{\mathsf{crs},r,w,b}_{\widetilde{\mathsf{iOT}}}(\mathcal{A})$, indexed by a crs, random coins $r \in \{0,1\}^\kappa$, a bit $w \in \{0,1\}$ and a bit $b \in \{0,1\}$:

$\underline{\mathsf{Exp}^{\mathsf{crs},r,w,b}_{\widetilde{\mathsf{iOT}}}(\mathcal{A})}$ (for branch w):

- Run $(m_0, m_1, \mathsf{M_R}, \mathsf{st}) \leftarrow \mathcal{A}_1(1^\kappa, \mathsf{crs}; r)$.
- If $b = 0$, sample randomness $\tilde{r}$ and compute $\mathsf{M_S} \leftarrow \widetilde{\mathsf{iOTS}}(\mathsf{crs}, \mathsf{M_R}, w, m_{1-w}; \tilde{r})$.
- If $b = 1$, sample randomness $\widehat{r}$, compute $\mathsf{M_S} \leftarrow \mathsf{iOTS}(\mathsf{crs}, \mathsf{M_R}, (m_0, m_1); \widehat{r})$ and $\tilde{r} = \widetilde{\mathsf{iOTS}}_{\mathsf{Inv}}(\mathsf{crs}, w, \mathsf{M_S}, \mathsf{td}, \widehat{r})$.
- Compute and output $s \leftarrow \mathcal{A}_2(\mathsf{st}, \tilde{r}, \mathsf{M_S})$.

Define the advantage of $\mathcal{A}$ as

$$\mathsf{Adv}^{\mathsf{crs},r,w}_{\widetilde{\mathsf{iOT}}}(\mathcal{A}) = |\Pr[\mathsf{Exp}^{\mathsf{crs},r,w,0}_{\widetilde{\mathsf{iOT}}}(\mathcal{A}) = 1] - \Pr[\mathsf{Exp}^{\mathsf{crs},r,w,1}_{\widetilde{\mathsf{iOT}}}(\mathcal{A}) = 1]|.$$

We say that iOT satisfies sender oblivious sampleability for branch w if for any PPT adversary $\mathcal{A}$ $\mathsf{Adv}^{\mathsf{crs},r,w}_{\widetilde{\mathsf{iOT}}}(\mathcal{A})$ is negligible in κ, where $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{Setup}(1^\kappa)$ and $r \leftarrow \{0,1\}^\kappa$.

OT with receiver Sampleability r-iOT: We denote by $\mathsf{r\text{-}iOT} = (\mathsf{Setup}, \mathsf{r\text{-}iOTR}_1, \mathsf{r\text{-}iOTS}, \mathsf{r\text{-}iOTR}_2, \widetilde{\mathsf{r\text{-}iOTR}}, \widetilde{\mathsf{r\text{-}iOTR}}_{\mathsf{Inv}})$, an indistinguishability-secure two-message OT in the CRS model that satisfies receiver oblivious sampleability but *not necessarily* sender oblivious sampleability. Such an OT protocol only needs to support "receiver oblivious sampling" algorithm $\widetilde{\mathsf{r\text{-}iOTR}}$ (where $\widetilde{\mathsf{r\text{-}iOTR}}$ is defined similar to $\widetilde{\mathsf{iOTR}}$) and the corresponding "randomness inversion" algorithm $\widetilde{\mathsf{r\text{-}iOTR}}_{\mathsf{Inv}}$ (where $\widetilde{\mathsf{r\text{-}iOTR}}_{\mathsf{Inv}}$ is defined similar to $\widetilde{\mathsf{iOTR}}_{\mathsf{Inv}}$) for the receiver.

3.4 UC Security of OT in the Static Corruption Setting

For UC security of OT against malicious adversaries in the static corruption setting, we directly use Canetti's UC security framework for static corruptions [Can01]. We note here that the UC security framework is only relevant to our construction of semi-adaptively UC-secure two-message OT detailed in Section 5.

Fig. 4. The ideal functionality $\mathcal{F}_{\mathsf{OT}}$ for Oblivious Transfer

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

$\mathcal{F}_{\mathsf{OT}}$ interacts with an ideal sender S and an ideal receiver R as follows:

- On input $(\mathsf{Choose}, \mathsf{rec}, \mathsf{sid}, b)$ from R where $b \in \{0,1\}$; if no message of the form $(\mathsf{rec}, \mathsf{sid}, b)$ has been recorded in the memory, store $(\mathsf{rec}, \mathsf{sid}, b)$ and send $(\mathsf{rec}, \mathsf{sid})$ to S.
- On input $(\mathsf{Transfer}, \mathsf{sen}, \mathsf{sid}, (m_0, m_1))$ from S with $m_0, m_1 \in \{0,1\}^n$, if no message of the form $(\mathsf{sen}, \mathsf{sid}, (m_0, m_1))$ is recorded and a message of the form $(\mathsf{rec}, \mathsf{sid}, b)$ is stored, send $(\mathsf{sent}, \mathsf{sid}, m_b)$ to R and $(\mathsf{sent}, \mathsf{sid})$ to S. Ignore future messages with the same sid.

Following the standard notation associated with Canetti's UC security framework [Can01], we use $\mathcal{Z}$ to denote the underlying environment. For a real protocol π and an adversary $\mathcal{A}$, we use $\mathrm{EXEC}_{\pi,\mathcal{A},\mathcal{Z}}$ to denote the real-world ensemble. Also, for an ideal functionality $\mathcal{F}$ and an adversary $\mathcal{S}$, we use $\mathrm{IDEAL}_{\mathcal{F},\mathcal{S},\mathcal{Z}}$ to denote the corresponding ideal-world ensemble.

The Ideal Functionality for OT. The ideal functionality $\mathcal{F}_{\mathsf{OT}}$ for any OT protocol is described in Figure 4. We adopt this description essentially verbatim from prior works [CLOS02, PVW08, DGH+20].

Fig. 5. The ideal functionality $\mathcal{F}_{\mathsf{CRS}}^{\mathcal{D}}$

$\mathcal{F}_{\mathsf{CRS}}^{\mathcal{D}}$

$\mathcal{F}_{\mathsf{CRS}}^{\mathcal{D}}$ (parameterized by a distribution $\mathcal{D}$) is run by parties $P_1, \ldots, P_n$ and an adversary $\mathcal{S}$ as follows:

- Whenever receiving a message of the form (sid, P_i, P_j) from a party P_i, sample $\mathsf{crs} \leftarrow \mathcal{D}$ and send $(\mathsf{sid}, \mathsf{crs})$ to the party P_i and send $(\mathsf{sid}, \mathsf{crs}, P_i, P_j)$ to the adversary $\mathcal{S}$.
- Whenever receiving the message of the form (sid, P_i, P_j) from a party P_j, send $(\mathsf{sid}, \mathsf{crs})$ to both the party P_j and the adversary $\mathcal{S}$.

The Ideal Functionality for CRS. Since we describe OT protocols in the CRS model, we also describe the ideal functionality $\mathcal{F}_{\mathsf{CRS}}^{\mathcal{D}}$ (parameterized by a distribution $\mathcal{D}$ from which the CRS string is sampled) in Figure 5. Again, we adopt this description essentially verbatim from prior works [CR03, PVW08, DGH+20].

Receiver's UC Security. We say that an OT protocol π_{OT} satisfies receiver's UC security if for any (malicious) adversary $\mathcal{A}$ corrupting the sender, there exists a simulator $\mathcal{S}$ such that for all environments $\mathcal{Z}$, we have:

$$\mathrm{EXEC}_{\pi_{\mathsf{OT}},\mathcal{A},\mathcal{Z}} \overset{c}{\approx} \mathrm{IDEAL}_{\mathcal{F}_{\mathsf{OT}},\mathcal{S},\mathcal{Z}},$$

where the ideal OT functionality $\mathcal{F}_{\mathsf{OT}}$ is as described in Figure 4.

Sender's UC Security. We say that an OT protocol π_{OT} satisfies sender's UC security if for any (malicious) adversary $\mathcal{A}$ corrupting the receiver, there exists a simulator $\mathcal{S}$ such that for all environments $\mathcal{Z}$, we have:

$$\mathrm{EXEC}_{\pi_{\mathsf{OT}},\mathcal{A},\mathcal{Z}} \overset{c}{\approx} \mathrm{IDEAL}_{\mathcal{F}_{\mathsf{OT}},\mathcal{S},\mathcal{Z}},$$

where the ideal OT functionality $\mathcal{F}_{\mathsf{OT}}$ is again as described in Figure 4.

Note that in the above descriptions, we adopt the same style of security definitions as was used in prior work [CLOS02, PVW08, DGH+20]. We refer to Section A for the security model of adaptively secure MPC.

3.5 Garbling Schemes

A garbling scheme [Yao86, CPV17b] is a tuple $\mathsf{Garble} = (\mathsf{Gb}, \mathsf{En}, \mathsf{Ev})$, described as follows:

- $\mathsf{Gb}\,(1^\kappa, \mathcal{C}) \rightarrow (\mathrm{GC}, \mathsf{Keys})$: A randomized algorithm which takes as input the security parameter and a circuit $\mathcal{C} : \{0,1\}^n \rightarrow \{0,1\}^m$ and outputs a tuple of strings $(\mathrm{GC}, \mathsf{Keys})$, where GC is the garbled circuit and Keys denotes the input-wire labels.
- $\mathsf{En}\,(x, \mathsf{Keys}) = \mathsf{X}$: a deterministic algorithm that outputs the garbled input X corresponding to input x.
- $\mathsf{Ev}\,(\mathrm{GC}, \mathsf{X}) = y$: A deterministic algorithm which evaluates garbled circuit GC on garbled input X and outputs y.

We borrow this definition and the associated notations from the work of [CPV17b]. The garbling scheme used in our protocols needs to satisfy standard properties such as correctness and privacy (we refer to [Yao86, BHR12, CPV17b] for the definitions). We additionally borrow two extra properties from [CPV17b] for our garbling schemes: namely, oblivious sampleability and equivocability, which we define here.

Definition 1. Perfect Correctness: *A garbling scheme* Garble *is perfectly correct if for all input lengths* $n \leq \mathsf{poly}(\kappa)$*, circuits* $\mathcal{C} : \{0,1\}^n \rightarrow \{0,1\}^m$ *and inputs* $x \in \{0,1\}^n$*, the following holds:*

$$\Pr\left[\mathsf{Ev}(\mathrm{GC}, \mathsf{En}(\mathit{Keys}, x)) \neq \mathcal{C}(x) : (\mathrm{GC}, \mathit{Keys}) \leftarrow \mathsf{Gb}(1^\kappa, \mathcal{C})\right] = 1.$$

Definition 2. Privacy [BHR12] *A garbling scheme* Garble *is private if for all input lengths* $n \leq \mathsf{poly}(\kappa)$*, circuits* $\mathcal{C} : \{0,1\}^n \rightarrow \{0,1\}^m$*, there exists a PPT simulator* $\mathcal{S}$ *such that for all inputs* $x \in \{0,1\}^n$*, for all probabilistic polynomial-time adversaries* $\mathcal{A}$*, the following two distributions are computationally indistinguishable:*

– $\text{REAL}(\mathcal{C}, x)$: *run* $(\text{GC}, \mathsf{Keys}) \leftarrow \mathsf{Gb}(1^\kappa, \mathcal{C})$, *and output* $(\text{GC}, \mathsf{En}(x, \mathsf{Keys}))$.
– $\text{IDEAL}_{\mathcal{S}}(\mathcal{C}, \mathcal{C}(x))$: *Compute* $(\text{GC}', \mathsf{X}, \mathsf{st}) \leftarrow \mathcal{S}_{GC}(\mathcal{C}, \mathcal{C}(x))$, *output* (GC', X).

Oblivious Sampleability [CPV17b]. Oblivious sampleability allows the garbler to obliviously sample a garbled circuit and a garbled input given the output, without the knowledge of the input keys Keys. It also enables an honestly computed garbled circuit to be claimed as obliviously sampled. A garbling scheme $\mathsf{Garble} = (\mathsf{Gb}, \mathsf{En}, \mathsf{Ev})$ is said to satisfy oblivious sampleability if there exist PPT algorithms $\widetilde{\mathsf{Gb}}$ and $\mathsf{Gb}_{\mathsf{Inv}}$ defined as:

– $\widetilde{\mathsf{Gb}}(1^\kappa, \mathcal{C}, y) \to (\widetilde{\text{GC}}, \widetilde{\mathsf{X}})$: A randomized algorithm that outputs an obliviously sampled garbled circuit $\widetilde{\text{GC}}$ and obliviously sampled wire labels $\widetilde{\mathsf{X}}$ such that evaluating $\widetilde{\text{GC}}$ on $\widetilde{\mathsf{X}}$ would yield y as output,
– $\mathsf{Gb}_{\mathsf{Inv}}(r, \mathsf{Keys}, x) \to \widehat{r}$: A randomness inversion algorithm that given some randomness of garbling r, input-wire labels Keys, and an input x, outputs some random coins $\widehat{r}$,

such that for any polynomial-time circuit $\mathcal{C}$ and for all input output pairs (x, y) such that $\mathcal{C}(x) = y$ it holds that

$$(\mathsf{Gb}_{\mathsf{Inv}}(r, \mathsf{Keys}, x), \text{GC}, \mathsf{X}) \overset{c}{\approx} (\widetilde{r}, \widetilde{\text{GC}}, \widetilde{\mathsf{X}}),$$

where for random coins $r, \widetilde{r} \leftarrow \{0,1\}^\kappa$, we have

$$(\text{GC}, \mathsf{Keys}) = \mathsf{Gb}(1^\kappa, \mathcal{C}; r), \quad \mathsf{X} = \mathsf{En}(x, \mathsf{Keys}), \quad (\widetilde{\text{GC}}, \widetilde{\mathsf{X}}) = \widetilde{\mathsf{Gb}}(1^\kappa, \mathcal{C}, y; \widetilde{r}).$$

The point-and-permute variant of Yao's garbling scheme by [BMR90] based on one-way functions satisfies oblivious sampling property.

Equivocal Garbling [CPV17b]. Finally, we require the garbled circuit to be equivocal [CPV17b]. It allows a privacy simulator $\mathcal{S}_{\text{GC}}$ to generate a fake garbled circuit $\widetilde{\text{GC}}$ and fake input wire labels $\widetilde{\mathsf{X}}$ that always evaluate to a fixed output. Later, the simulator can open $(\widetilde{\text{GC}}, \widetilde{\mathsf{X}})$ to a particular input x by providing consistent randomness used in the garbling process. We define this as follows: a garbling scheme $\mathsf{Garble} = (\mathsf{Gb}, \mathsf{En}, \mathsf{Ev})$ is said to be equivocal if there exists a pair of PPT algorithms $(\mathcal{S}^1_{\text{GC}}, \mathcal{S}^2_{\text{GC}})$, such that any PPT adversary $\mathcal{A}$ wins the following game with at most negligible advantage:

1. $\mathcal{A}$ gives a circuit $\mathcal{C}$ and an input x to the challenger.
2. The challenger samples a bit $b \leftarrow \{0,1\}$.
   – If $b = 0$: It computes $(\text{GC}, \mathsf{Keys}) \leftarrow \mathsf{Gb}(\mathcal{C}; r)$ and $\mathsf{X} \leftarrow \mathsf{En}(x, \mathsf{Keys})$.
   – If $b = 1$: It sets $y = C(x)$. It runs the simulator $(\text{GC}, \mathsf{X}, \mathsf{st}) \leftarrow \mathcal{S}^1_{\text{GC}}(\mathcal{C}, y)$. It runs the simulator $(\mathsf{Keys}, r) \leftarrow \mathcal{S}^2_{\text{GC}}(\mathsf{st}, x)$.
3. The challenger sends $(\text{GC}, \mathsf{X}, \mathsf{Keys}, r)$ to the adversary $\mathcal{A}$.
4. The adversary outputs a bit b'.

The adversary wins if $b = b'$. For the privacy game, $\mathcal{S}^1_{\mathsf{GC}}$ plays the role of privacy simulator in Def. 2 for the equivocal garbling scheme.

The garbling scheme of [CPV17b] based on one-way function satisfies equivocal property. Moreover, the wire labels of [CPV17b] are uniformly distributed.

3.6 Trapdoor Simulatable PKE.

We recall the definition of trapdoor simulatable PKE from [CDMW09]. A trapdoor simulatable PKE scheme in the CRS model is a tuple of the form $(\mathsf{Setup}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{oGen}, \mathsf{oEnc})$, where the tuple $(\mathsf{Setup}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ is a standard PKE scheme that is augmented with oblivious sampling algorithms $(\mathsf{oGen}, \mathsf{oEnc})$ and randomness inverting algorithms $(\mathsf{rGen}, \mathsf{rEnc})$. The trapdoor of the setup string allows generating a public key (resp. a ciphertext) honestly and then claiming that the public key (resp. a ciphertext) was obliviously sampled using the rGen (resp. the rEnc) algorithm. Formally, we require that for any message $m \in \{0,1\}^\ell$, letting $(\mathsf{crs}, \mathsf{td}) = \mathsf{Setup}(1^\kappa)$,

$$(\mathsf{pk}, c, \widehat{r}_G, \widehat{r}_E) \overset{c}{\approx} (\widetilde{\mathsf{pk}}, \widetilde{c}, \widetilde{r}_G, \widetilde{r}_E),$$

where for random coins $r_G, r_E, \widetilde{r}_G, \widetilde{r}_E \leftarrow \{0,1\}^\kappa$, we have $(\mathsf{pk}, \mathsf{sk}) = \mathsf{Gen}(\mathsf{crs}; r_G)$, $c = \mathsf{Enc}(\mathsf{crs}, \mathsf{pk}, m; r_E)$, $\widetilde{\mathsf{pk}} = \mathsf{oGen}(\mathsf{crs}; \widetilde{r}_G)$, $\widetilde{c} = \mathsf{oEnc}(\mathsf{crs}, \widetilde{\mathsf{pk}}; \widetilde{r}_E)$, and

$$\widehat{r}_G = \mathsf{rGen}(\mathsf{crs}, r_G, \mathsf{td}), \quad \widehat{r}_E = \mathsf{rEnc}(\mathsf{crs}, r_G, r_E, m, \mathsf{td}).$$

3.7 Non-Committing Encryption in the crs model.

A non-committing encryption in the setup model consists of four algorithms $\mathsf{NCE} = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$. It is a public key encryption scheme which allows a simulator to encrypt a plaintext in the presence of an adaptive adversary. Given a trapdoor, the simulator (on behalf of the honest party) can produce some dummy ciphertext c without the knowledge of any plaintext m. Later when the honest party gets corrupted and the simulator produces matching randomness (or decryption key) s.t. c decrypts to m. More formally, it is defined as follows.

Definition 3. (Non-Committing Encryption). *A non-committing (bit) encryption scheme (NCE) consists of a tuple* $(\mathsf{Setup}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{NCE.Dec}, \mathcal{S})$ *where* $(\mathsf{Setup}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *is an IND-CPA public key encryption scheme in the* crs *model and* $\mathcal{S}$ *is the simulation satisfying the following property: for* $b \in \{0,1\}$ *the following distributions are computationally indistinguishable:*

$$\{(\mathsf{pk}, c, r_G, r_E) : (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(\mathsf{crs}, r_G), c = \mathsf{Enc}(\mathsf{crs}, \mathsf{pk}, b; r_E)\}_{\kappa,b} \overset{c}{\approx}$$

$$\{(\mathsf{pk}, c, r_G^b, r_E^b) : (\mathsf{pk}, c, r_G^0, r_E^0, r_G^1, r_E^1) \leftarrow \mathcal{S}(\mathsf{crs}, \mathsf{td})\}_{\kappa,b},$$

where $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{Setup}(1^\kappa)$.

Definition 4. (Augmented Non-Committing Encryption). *An augmented* NCE *scheme consists of a tuple of algorithms* $(\mathsf{Setup}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \mathcal{S}, \mathsf{Gen}_{\mathsf{Obl}}, \mathsf{Gen}_{\mathsf{Inv}})$ *where* $(\mathsf{Setup}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \mathcal{S})$ *is an* NCE *in the* crs *model and:*

- *Oblivious Sampling:* $\mathsf{Gen}_{\mathsf{Obl}}(1^\kappa)$ *obliviously generates a public key* pk *(without knowing the associated secret key* sk*.*
- *Inverse Key Sampling:* $\mathsf{Gen}_{\mathsf{Inv}}(\mathsf{pk})$ *explains the randomness for the key* pk *satisfying the obliviousness property. Given the trapdoor* td *of the* crs*, the following distributions are indistinguishable:*

$$\{(\mathsf{pk}, r) : \mathsf{pk} \leftarrow \mathsf{NCE.Gen}_{\mathsf{Obl}}(\mathsf{crs}; r)\}_\kappa \approx$$

$$\{(\mathsf{pk}, r') : (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{NCE.Gen}(\mathsf{crs}; r_G); r' \leftarrow \mathsf{NCE.Gen}_{\mathsf{Inv}}(\mathsf{crs}, r_G, \mathsf{pk}, \mathsf{td})\}_\kappa,$$

where $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{Setup}(1^\kappa)$.

3.8 Cryptographic Background

Min-entropy and Leftover Hash Lemma. For a discrete random variable Z with sample space Ω, its *min-entropy* is defined as

$$H_\infty(Z) = \min_{\omega \in \Omega} \{-\log \Pr[Z = \omega]\}.$$

For two random variables Y and Z, we use $H_\infty(Z|Y)$ to denote the min-entropy of Z conditioned on Y. We will use the following lemma, which is a simplified version of the leftover hash lemma.

Lemma 1. *Let* $\{H_s : \mathcal{Z} \to Y\}_{s \in \mathcal{S}}$ *be a family of pairwise independent hash functions, and assume that* Z *and* S *be discrete random variables over* $\mathcal{Z}$ *and* $\mathcal{S}$*, respectively. If* $H_\infty(Z) > \log|Y| + 2\log(\varepsilon^{-1})$ *we have*

$$\Delta[(S, H_S(Z)), (S, U)] \leq \varepsilon,$$

where Δ *denotes the statistical distance and* U *denotes the uniform distribution over the set* Y*.*

We will also use the following corollary of the leftover hash lemma.

Lemma 2. *Let* G *be an (additive) finite abelian group such that* $|G| = \lambda^{\omega(1)}$*. Let* n *be an integer such that* $n > \log|G| + \omega(\log(\lambda))$*. If* $\mathbf{g} \leftarrow G^n$ *and* $\mathbf{s} \leftarrow \{0,1\}^n$*, we have*

$$(\mathbf{g}, \sum_{i=1}^{n} s_i \cdot g_i) \overset{s}{\approx} (\mathbf{g}, u),$$

where $u \leftarrow G$ *is a uniformly chosen element from* G*.*

One-Way Function. Let P, X and Y be sets indexed by the parameter λ, and let $\mathcal{D}_P$ and $\mathcal{D}_X$ be distributions on P and X respectively. A (D_P, D_X)-OWF family is a family of efficient computable functions $\{f_{\mathsf{pp}}(\cdot) : X \to Y\}_{\mathsf{pp} \in P}$ such that for all PPT adversaries $\mathcal{A}$ we have

$$\Pr[f_{\mathsf{pp}}(\mathcal{A}(\mathsf{pp}, f_{\mathsf{pp}}(x))) = f_{\mathsf{pp}}(x)] \leq \mathsf{negl}(\lambda),$$

where $\mathsf{pp} \leftarrow \mathcal{D}_P$ and $x \leftarrow \mathcal{D}_X$. If $\mathcal{D}_P$ and $\mathcal{D}_X$ are uniform distributions, then we simply speak of an OWF family.

Weak Pseudorandom Permutation. Let K and X be sets indexed by λ, and let $\mathcal{D}_K$ and $\mathcal{D}_X$ be distributions on K and X respectively. A $(\mathcal{D}_K, \mathcal{D}_X)$-weak PRP (wPRP) is a family of efficiently computable permutations $\{F(k, \cdot) : X \to X\}_{k \in K}$ such that for all PPT adversaries $\mathcal{A}$ we have

$$\left|\Pr[\mathcal{A}^{F_k^{\$}}(1^\lambda) = 1] - \Pr[\mathcal{A}^{\pi^{\$}}(1^\lambda) = 1]\right| \leq \mathsf{negl}(\lambda),$$

where $k \leftarrow \mathcal{D}_k$, $\pi \leftarrow S_X$ and $\pi^{\$}$ is the randomized oracle that samples $x \leftarrow \mathcal{D}_X$ and outputs $(x, \pi(x))$. If $\mathcal{D}_K$ and $\mathcal{D}_S$ are uniform distributions, then we simply speak of a wPRP family.

3.9 Cryptographic Assumptions

Definition 5. (Computational Diffie-Hellman Assumption). *We say that the CDH assumption holds in a group $\mathbb{G}$ if for any PPT adversary $\mathcal{A}$,*

$$\Pr[\mathcal{A}(g, h, T) = Z] = \mathit{neg}(\kappa).$$

holds, where $h, T \leftarrow \mathbb{G}$, and $T = g^t$, $Z = h^t$.

For our LPN-based construction, we rely on a variant of LPN assumption with noise rate $n^{-\varepsilon}$ for any constant $\varepsilon > 1/2$. In addition, we draw the secret from error (Bernoulli) distribution, which is known to be equivalent to standard LPN assumption [ACPS09, DGH+20]. We formally define this variant of LPN as follows:

Definition 6. (Learning Parity with Noise Assumption). *Let $n = poly(\lambda)$, and let $\varepsilon < 1/2$ be a fixed constant. In addition, let $\mathcal{B}_p$ be Bernoulli distribution which outputs 1 with probability $\rho = n^{-\varepsilon}$. If $\mathbf{A} \leftarrow \mathbb{Z}_2^{n \times n}$, $\mathbf{s} \leftarrow \mathcal{B}_\rho^n$, and $\mathbf{s} \leftarrow \mathcal{B}_\rho^n$, then the LPN assumption states that*

$$(\mathbf{A}, \mathbf{As} + \mathbf{e}) \overset{c}{\approx} (\mathbf{A}, \mathbf{As} + \mathbf{u}),$$

where $\mathbf{u} \leftarrow \mathbb{Z}_2^n$.

3.10 Commitment Schemes

We define an equivocal commitment scheme $\mathsf{Com} = (\mathsf{Setup}, \mathsf{Com}, \mathsf{Ver}, \mathsf{Equiv})$ as follows:

Definition 7. ***(Correctness)*** *Com is a correct commitment scheme if the following holds true*

$$\Pr\left[\mathit{Ver}(m, c, \mathit{crs}, r) = 1 | (\mathit{crs}, \mathit{td}) \leftarrow \mathit{Setup}(1^\kappa), c \leftarrow \mathit{Com}(m, \mathit{crs}; r)\right] = 1$$

Definition 8. ***(Binding)*** *Com is computationally binding scheme if the following holds true for all PPT adversary $\mathcal{A}$*

$$\Pr\left[(m_0, r_0, m_1, r_1) \leftarrow \mathcal{A}(\mathit{crs}) | (\mathit{crs}, \mathit{td}) \leftarrow \mathit{Setup}(1^\kappa),\right.$$

$$\left.\mathit{Com}(m_0; r_0) = \mathit{Com}(m_1; r_1)\right] \leq \mathit{neg}(\kappa)$$

Definition 9. ***(Hiding)*** *Com is computationally hiding scheme if the following holds true for all PPT adversary* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$.

$$\Pr\left[b == b' | (\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{Setup}(1^\kappa), (m_0, m_1, \mathsf{st}) \leftarrow \mathcal{A}_1(\mathsf{crs}), b \leftarrow_r \{0,1\},\right.$$

$$\left.(c, d) \leftarrow \mathsf{Com}(m_b), b' \leftarrow \mathcal{A}_2(c; \mathsf{st})\right] \leq \frac{1}{2} + \mathsf{neg}(\kappa)$$

Definition 10. ***(Equivocal)*** *Com is equivocal if it has a PPT algorithm Equiv s.t. the following holds true for all PPT adversary* $\mathcal{A}$ *and all message pairs* (m_0, m_1).

$$\Big|\Pr\left[\mathcal{A}(c, r) = 1 | (\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{Setup}(1^\kappa), m \leftarrow \mathcal{A}(\mathsf{crs}), c = \mathsf{Com}(\mathsf{crs}, m; r)\right]$$

$$-\Pr\left[\mathcal{A}(c, r) = 1 | (\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{Setup}(1^\kappa), m \leftarrow \mathcal{A}(\mathsf{crs}), c = \mathsf{Com}(\mathsf{crs}, m'; r'),\right.$$

$$\left.r = \mathsf{Equiv}(m, r, c', \mathsf{td})\right]\Big| \leq \mathsf{neg}(\kappa), \text{ for } m \neq m'$$

A commitment scheme with a sampling algorithm $\widetilde{\mathsf{Com}}$ and randomness inversion algorithm $\widetilde{\mathsf{Com}}_{\mathsf{Inv}}$ satisfies oblivious commitment sampling property as follows:

Definition 11. ***(Oblivious Sampling)*** *Com satisfies oblivious sampling if the following holds true for all PPT adversary* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$.

$$\big|\Pr[\mathcal{A}_2(\mathsf{st}, c, \tilde{r}) = 1 | (m, \mathsf{st}) \leftarrow \mathcal{A}_1(\mathsf{crs}), c = \widetilde{\mathsf{Com}}(\mathsf{crs}; \tilde{r})] - \Pr[\mathcal{A}_2(\mathsf{st}, c, \tilde{r}) = 1|$$

$$(m, \mathsf{st}) \leftarrow \mathcal{A}_1(\mathsf{crs}), c = \mathsf{Com}(\mathsf{crs}, m; r), \tilde{r} \leftarrow \widetilde{\mathsf{Com}}_{\mathsf{Inv}}(\mathsf{crs}, m, r, \mathsf{td})]\big| < \mathsf{neg}(\kappa),$$

where $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{Setup}(1^\kappa)$.

3.11 Cryptographic Group Actions

In this section we recall the definitions of cryptographic group actions from [ADMP20].

Definition 12. (Group Action) *A group* G *is said to* act on *a set* X *if there is a map* $\star : G \times X \to X$ *that satisfies the following two properties:*

1. *Identity: If* e *is the identity element of* G*, then for any* $x \in X$*, we have* $e \star x = x$.
2. *Compatibility: For any* $g, h \in G$ *and any* $x \in X$*, we have* $(gh) \star x = g \star (h \star x)$.

The authors of [ADMP20] use the abbreviated notation $(G, X, \star)$ to denote a group action.

Remark 1. If $(G, X, \star)$ is a group action, for any $g \in G$ the map $\pi_g : x \mapsto g \star x$ defines a permutation of X.

Properties of group actions. The authors of [ADMP20] consider group actions that satisfy one or more of the following properties:

1. Transitive: A group action $(G, X, \star)$ is said to be *transitive* if for every $x_1, x_2 \in X$, there exists a group element $g \in G$ such that $x_2 = g \star x_1$. For such a transitive group action, the set X is called a *homogeneous space* for G.
2. Faithful: A group action $(G, X, \star)$ is said to be *faithful* if for each group element $g \in G$, either g is the identity element or there exists a set element $x \in X$ such that $x \neq g \star x$.
3. Free: A group action $(G, X, \star)$ is said to be *free* if for each group element $g \in G$, g is the identity element if and only if there exists some set element $x \in X$ such that $x = g \star x$.
4. Regular: A group action $(G, X, \star)$ is said to be *regular* if it is *both* free *and* transitive. For such a regular group action, the set X is called a *principal homogeneous space* for the group G, or a G*-torsor*.

Remark 2. Typically group action-based cryptography has focused on regular actions. If a group action is regular, then for any $x \in X$, the map $f_x : g \mapsto g \star x$ defines a bijection between G and X; in particular, if G (or X) is finite, then we must have $|G| = |X|$.

Effective Group Action The authors of [ADMP20] define an effective group action (EGA) as follows.

Definition 13. (Effective Group Action) *A group action $(G, X, \star)$ is* effective *if the following properties are satisfied:*

1. *The group G is finite and there exist efficient (PPT) algorithms for:*
   (a) *Membership testing, i.e., to decide if a given bit string represents a valid group element in G.*
   (b) *Equality testing, i.e., to decide if two bit strings represent the same group element in G.*
   (c) *Sampling, i.e., to sample an element g from a distribution $\mathcal{D}_G$ on G. In this paper, We consider distributions that are statistically close to uniform.*
   (d) *Operation, i.e., to compute gh for any $g, h \in G$.*
   (e) *Inversion, i.e., to compute g^{-1} for any $g \in G$.*
2. *The set X is finite and there exist efficient algorithms for:*
   (a) *Membership testing, i.e., to decide if a bit string represents a valid set element.*
   (b) *Unique representation, i.e., given any arbitrary set element $x \in X$, compute a string $\hat{x}$ that canonically represents x.*
3. *There exists a distinguished element $x_0 \in X$, called the* origin, *such that its bit-string representation is known.*
4. *There exists an efficient algorithm that given (some bit-string representations of) any $g \in G$ and any $x \in X$, outputs $g \star x$.*

Computational assumptions. The authors of [ADMP20] define certain computational assumptions pertaining to group actions.

Definition 14. (One-Way Group Action) *A group action $(G, X, \star)$ is $(\mathcal{D}_X, \mathcal{D}_G)$-one-way if the family of efficiently computable functions $\{f_x : G \to X\}_{x \in X}$ is $(\mathcal{D}_X, \mathcal{D}_G)$-one-way, where $f_x : g \mapsto g \star x$, and $\mathcal{D}_X$, $\mathcal{D}_G$ are distributions on X, G respectively.*

Definition 15. (Weak Pseudorandom Group Action) *A group action $(G, X, \star)$ is $(\mathcal{D}_G, \mathcal{D}_X)$-weakly pseudorandom if the family of efficiently computable permutations $\{\pi_g : X \to X\}_{g \in G}$ is $(\mathcal{D}_G, \mathcal{D}_X)$-weakly pseudorandom, where $\pi_g : x \mapsto g \star x$, and $\mathcal{D}_X$, $\mathcal{D}_G$ are distributions on X, G respectively.*

In each of the definitions above, if $\mathcal{D}_G$ is a probability distribution on G and $\mathcal{D}_X$ is the distribution induced on X by taking $g \leftarrow \mathcal{D}_G$ and outputting $g \star x_0$, then we simply write $\mathcal{D}_G$-OW group action for $(\mathcal{D}_X, \mathcal{D}_G)$-OW group action, and similarly for weak unpredictable/pseudorandom group actions. If both distributions are uniform (or statistically close to uniform), we omit them.

Restricted Effective Group Action The authors of [ADMP20] pointed out that while EGA is a useful abstraction, sometimes it is too powerful in comparison to what is achievable in practice. A *Restricted Effective Group Action* (REGA) is a weakening of EGA, where we can only evaluate the action of a generating set of small cardinality.

Definition 16. (Restricted Effective Group Action) *Let $(G, X, \star)$ be a group action and let $\mathbf{g} = (g_1, \ldots, g_n)$ be a (not necessarily minimal) generating set for G. The action is said to be* $\mathbf{g}$-restricted effective, *if the following properties are satisfied:*

- *G is finite and $n = poly(\log(|G|))$.*
- *The set X is finite and there exist efficient algorithms for:*
  1. *Membership testing, i.e., to decide if a bit string represents a valid set element.*
  2. *Unique representation, i.e., to compute a string $\hat{x}$ that canonically represents any given set element $x \in X$.*
- *There exists a distinguished element $x_0 \in X$, called the* origin, *such that its bit-string representation is known.*
- *There exists an efficient algorithm that given any $i \in [n]$ and any bit string representation of $x \in X$, outputs $g_i \star x$ and $g_i^{-1} \star x$.*

Although an REGA is limited to evaluations of the form $g_i \star x$, this is actually enough to evaluate the action of many, and potentially all elements of G without even needing axioms on the effectivity of G.

A *word on* $(g_1, \ldots, g_n)$ is a finite sequence $\sigma \in \{g_1, \ldots, g_n, g_1^{-1}, \ldots, g_n^{-1}\}^*$, to which we canonically associate an element of G by

$$\sigma = \sigma_1 \sigma_2 \cdots \sigma_\ell \mapsto \prod_{i=1}^{\ell} \sigma_i.$$

By hypothesis, any element of G can be represented by a word on $\mathbf{g}$, however this representation may not be unique, nor equality needs to be efficiently testable. From the definition of a $\mathbf{g}$-REGA, it is clear that the action on $x \in X$ of any word of polynomial length on $\mathbf{g}$ can be computed in polynomial time.

When G is abelian, words on $\mathbf{g}$ can be rewritten as vectors in Z^n, canonically mapped to G by

$$(a_1, \ldots, a_n) \mapsto \prod_{i=1}^{n} g_i^{a_i}.$$

It follows from the axioms of REGA that the action of a vector $\mathbf{a} \in \mathsf{Z}^n$ can be efficiently evaluated on any $x \in X$ as long as $\|\mathbf{a}\|$ is polynomial in $\log(|G|)$, where $\|\cdot\|$ is any L^p-norm.

Protocols built on REGA will need to sample elements from G that are statistically close to uniform and for which the group action is efficiently computable. Prior works suggest sampling from a distribution $\mathcal{D}_G$ on the words on $\mathbf{g}$ in the non-abelian case, or from a distribution on vectors in Z^n in the abelian case. Classic choices in the latter case are balls of fixed radius in L^∞-norm [CLM+18], in L^1-norm [NOTT20], in weighted infinity norms [Sto12, MR18], or discrete Gaussian distributions [DG19]. The latter is plausibly sufficient for applications that require group elements to be sampled from distributions statistically close to uniform [DG19].

4 iOT with Oblivious Sampleability from r-iOT

In this section, we present the first generic construction in our overall framework: we show how to build a two-message iOT protocol in the CRS model with *both* oblivious sender and receiver sampleability given a two-message r-iOT protocol (iOT with receiver oblivious sampleability but *not necessarily* sender oblivious sampleability). For simplicity of exposition, we describe the construction in the CRS model against malicious corruptions; the corresponding construction in the plain model against semi-honest corruptions follows analogously.

4.1 Overview and Intuition

We construct a two-message iOT protocol with oblivious sender and receiver sampleability in the CRS model given the following ingredients: (1) a two-message r-iOT protocol in the CRS model which satisfies receiver oblivious sampleability, (2) an equivocal garbling scheme, (3) an obliviously sampleable garbling scheme, and (4) an obliviously sampleable commitment scheme (in the CRS model). The latter three schemes are implied by one-way functions (and hence by r-iOT).

A First Attempt. We describe below an initial attempt to build iOT with receiver and sender oblivious sampleability from r-iOT. This simple construction additionally uses a standard garbling scheme and an obliviously sampleable non-interactive commitment scheme. Additionally, let $\mathcal{C}[\beta, c](r, m)$ denote a circuit that is hardwired with a bit $\beta \in \{0, 1\}$ and a commitment c. It takes as input some randomness r and a message m, and outputs m if c is valid commitment to β using randomness r. Otherwise, it outputs $\perp$ (the circuit $\mathcal{C}$ is also hardwired with the CRS string for the commitment scheme, but we avoid mentioning this explicitly for simplicity of presentation.).

- iOTR_1: The receiver uses the commitment scheme to create a commitment $c = \mathsf{Com}(b; r)$ to its input choice bit b under randomness r. The receiver transmits this commitment c to the sender. Then the receiver runs two sets of r-iOT protocols corresponding to bit 0 and bit 1. For bit b, the receiver uses the r-iOT protocol to send one r-iOT-receiver message corresponding to each bit of the randomness r (in

parallel). For bit $1-b$, the receiver samples $\ell = |r|$ number of r-iOT protocol messages obliviously. The receiver sends the two sets of ℓ protocol messages of r-iOT to the sender.

- iOTS: The sender uses the commitment c from the receiver to create two circuits $\mathcal{C}_{0,c}(\cdot, m_0)$ and $\mathcal{C}_{1,c}(\cdot, m_1)$ as described earlier, where m_0 and m_1 are the sender input messages. It then garbles these circuits using the garbling scheme to create a pair of garbled circuits GC_0 and GC_1, along with the corresponding wire labels for their input bits.

  The sender sends across GC_0 and GC_1 to the receiver. The sender also sends the wire labels for m_0 and m_1 corresponding to GC_0 and GC_1 respectively. In parallel, the sender uses the underlying r-iOT scheme and the two sets of $\ell = |r|$ r-iOT messages from the receiver to generate two sets of ℓ number of r-iOT-sender messages. The sender inputs to the r-iOT are the wire labels of the garbled circuits.

  Let us denote the sets as the 0th set and the 1th set corresponding to GC_0 and GC_1. The 0th (resp. 1th) set corresponds to the ℓ bits of the commitment randomness of the receiver corresponding to c being a commitment to 0 (resp 1). For the ith OT in the 0th set, the sender considers its inputs as the garbled circuit wire labels (for GC_0) corresponding to the ith bit of the receiver randomness, which is provided as input to the garbled circuit. Similarly, for the ith OT in the 1th set, the sender considers its inputs as the garbled circuit wire labels (for GC_1) corresponding to the ith bit of the receiver randomness, which is provided as input to the garbled circuit. The sender sends the two sets of ℓ number of r-iOT sender messages to the receiver.

- iOTR_2: The receiver considers the bth set of r-iOT sender messages. It uses the r-iOT-sender messages to recover the wire labels corresponding to its randomness string r for garbled circuits GC_b. It then evaluates GC_b on r by using the corresponding wire labels to recover the message m_b.

Correctness. It can be observed that correctness trivially holds, since an honest receiver obtains the wire labels (of GC_b) corresponding to randomness r from the bth set of r-iOT sender messages. Then it evaluates GC_b to obtain m_b. Evaluation is successful since the garbled verifies that the commitment $c = \mathsf{Com}(b; r)$ is correct given randomness r and bit b, and if verification succeeds then it outputs m_b to the receiver.

Receiver Security. Arguing receiver's indistinguishability security is relatively straightforward. Informally, the commitment c computationally hides the receiver's choice bit b and the r-iOT messages sent by the receiver computationally hide the receiver's randomness string r.

Sender Security. To argue sender's indistinguishability security, we need to establish (at a high level) that the receiver learns no information about m_{1-b}. To see why this is the case, observe that the only information about m_{1-b} that the receiver could learn is from the garbled circuit GC_{1-b}. However, the receiver cannot evaluate GC_{1-b} to anything other than $\perp$ since: (1) it cannot prove that c is a commitment to $(1-b)$ under randomness some r' (this follows from the binding property of the commitment scheme), and (2) it cannot recover any input labels to GC_{1-b} other than those corresponding to r' (due to the sender privacy of the underlying r-iOT protocol). At this point, we can invoke the privacy of the garbling scheme itself to argue that the receiver learns no information about the message m_{1-b}.

Oblivious Receiver Sampleability. The above protocol satisfies oblivious receiver sampleability since the commitment is obliviously sampleable and the r-iOT messages are obliviously receiver sampleable.

Barriers to Sender Oblivious Sampleability. The above approach fails to give us sender's oblivious sampleability as explain next. Note that the sender's message has two parts - the garbled circuits $(\mathrm{GC}_0, \mathrm{GC}_1)$, and the two sets of ℓ number of r-iOT-sender messages. Using an *obliviously sampleable garbling scheme* naturally allows oblivious sampleability for the first part of the sender's message, i.e. the garbled circuits GC_0 and GC_1. However, it is not clear if the second part of the sender's message can be obliviously sampled since the underlying r-iOT protocol does not necessarily support sender oblivious sampleability.

Recall the *sender's oblivious sampleability* security definition as defined in Definition 3.3. In this game, the challenger must provide the adversary with the randomness $\tilde{r}$ used in the $\widetilde{\mathsf{iOTS}}$ algorithm. Since we have an obliviously sampleable garbling scheme available to us by assumption, the garbled circuits $(\mathrm{GC}_0, \mathrm{GC}_1)$

can be obliviously sampled. However, the r-iOT sender messages are not obliviously sampleable, since r-iOT does not give sender's oblivious sampleability by definition.

As a result, we can construct adversary for the sender's oblivious sampleability game that can distinguish between an honestly generated sender message from an obliviously sampled one. Since the adversary has access to the oblivious sampling randomness $\tilde{r}$, it can obtain all of the wire labels from the r-iOT sender messages in the sampling game. It can then evaluate the garbled circuits on whatever inputs it wants using all the wire labels and as a result distinguish an honestly generated garbled circuit from an obliviously sampled one, as the evaluation of the "obliviously sampled" garbled circuit will be unlikely to be correct.

4.2 Our Protocol

We address this issue by using *two separate sets* of garbled circuits. The first set of garbled circuits are created using a garbling scheme Garble' that is *obliviously sampleable* [LP09], while the second set of garbled circuits are created using a garbling scheme Garble that is *equivocal* [CPV17b]. More concretely, we use the following circuits (hardwired with the CRS string for the commitment scheme):

- Let $C[\beta, c](r, p)$ denote a circuit that is hardwired with a bit $\beta \in \{0,1\}$ and a commitment c. It takes as input some randomness r and a string p, and outputs p if c is valid commitment to the bit β using randomness r. Otherwise, it outputs $\perp$.
- Let $\mathcal{C}'(c', s, m)$ denote a circuit that takes as input a commitment c', some randomness s and a message m, and outputs m if c' is valid commitment to 0 using randomness s. Otherwise, it outputs $\perp$.

Note. The circuits $\mathcal{C}'$ and $\mathcal{C}$ are both also hardwired with the CRS string for the commitment scheme, but we again avoid mentioning this explicitly for simplicity of presentation.

The Receiver's Message. In our modified protocol, the receiver message to the sender is still generated as in the simple protocol described earlier. In other words, the receiver uses the commitment scheme to create a commitment c to its input choice bit b under randomness r. The receiver transmits this commitment c to the sender. Then the receiver also uses the underlying r-iOT protocol to compute two sets of $\ell(= |r|)$ number of r-iOT-receiver message corresponding to each bit of the randomness r (in parallel). The receiver runs two sets of r-iOT protocols corresponding to bit 0 and bit 1. For bit b, the receiver uses the r-iOT protocol to send one r-iOT-receiver message corresponding to each bit of the randomness r (in parallel). For bit $1 - b$, the receiver samples ℓ number of r-iOT protocol messages obliviously. The receiver sends the two sets of ℓ protocol messages of r-iOT to the sender.

The Sender's Message. The key alteration is in how the sender's message to the receiver is generated. The sender proceeds as follows:

- The sender creates two commitments $c'_0 = \mathsf{Com}(0; s_0)$ and $c'_1 = \mathsf{Com}(0; s_1)$ - both of which are commitments to 0 under randomness s_0 and s_1 respectively.
- The sender then uses the *obliviously sampleable* garbling scheme Garble' to create two (independent) garbled circuits GC'_0 and GC'_1 for the circuit $\mathcal{C}'$.
- Corresponding to GC'_0, the sender computes p_0 – a string representing the set of wire-labels in GC'_0 corresponding to the input c'_0. Similarly, corresponding to GC'_1, the sender computes p_1 – a string representing the set of wire-labels corresponding to c'_1 in GC'_1.
- For $\beta \in \{0,1\}$, the sender sets V_β as the wire labels in GC'_β corresponding to $(s_\beta || m_\beta)$. It also represents p_β as the garbled sets p_β as the garbled input for c'_β corresponding to GC'_β.
- At this point, the sender does something very similar to flavor to its counterpart in the simpler protocol described earlier. In particular, it uses the commitment c from the receiver and the p_0 and p_1 strings from the previous step to create two circuits $\mathcal{C}_{0,c}(\cdot, p_0)$ and $\mathcal{C}_{1,c}(\cdot, p_1)$. It then garbles these circuits using the *equivocal* garbling scheme Garble to create a pair of garbled circuits GC_0 and GC_1, along with the corresponding wire labels for their input bits.

Fig. 6. Constructing iOT with Oblivious Sampleability from r-iOT

π_{iOT}

- **Public Inputs:** $\mathsf{crs}_{\mathsf{iOT}} = (\mathsf{crs}_{\mathsf{r\text{-}iOT}}, \mathsf{crs}_{\mathsf{com}})$ where $\mathsf{crs}_{\mathsf{r\text{-}iOT}}$ and $\mathsf{crs}_{\mathsf{com}}$ are the setup strings of r-iOT and Com respectively.
- **Circuits:** Circuit $\mathcal{C}[c, \mathsf{crs}_{\mathsf{com}}, \beta](r, p) = p$ if $c = \mathsf{Com}(\mathsf{crs}_{\mathsf{com}}, \beta; r)$, else $\mathcal{C}$ outputs $\perp$. Circuit $\mathcal{C}'[\mathsf{crs}_{\mathsf{com}}](c', s, m) = m$ if $c' = \mathsf{Com}(\mathsf{crs}_{\mathsf{com}}, 0; s)$, else it outputs $\perp$.
- **Private Inputs:** S has input bits (m_0, m_1) where $m_0, m_1 \in \{0,1\}$; R has input choice bit b.
- **Primitives:** Let $\pi_{\mathsf{r\text{-}iOT}} = (\mathsf{r\text{-}iOTR}_1, \mathsf{r\text{-}iOTS}, \mathsf{r\text{-}iOTR}_2, \widetilde{\mathsf{r\text{-}iOTR}})$ denote a receiver obliviously sampleable indistinguishable OT. $(\mathsf{Com}, \widetilde{\mathsf{Com}})$ is an obliviously sampleable commitment scheme. $\mathsf{Garble} = (\mathsf{Gb}, \mathsf{En}, \mathsf{Ev}, \mathcal{S}_{\mathsf{GC}})$ is an equivocal garbling scheme. $\mathsf{Garble}' = (\mathsf{Gb}', \mathsf{En}', \mathsf{Ev}', \widetilde{\mathsf{Gb}}')$ is an obliviously sampleable garbling scheme.

$\mathbf{iOTR}_1(\mathsf{crs}_{\mathsf{iOT}}, b)$:

- R commits to b using randomness r as $c = \mathsf{Com}(b; r)$. Let $|r| = \ell$.
- R computes $\pi_{\mathsf{r\text{-}iOT}}$ receiver messages as $\{\gamma_{0,i}, \gamma_{1,i}\}$ where $\gamma_{b,i} = \mathsf{r\text{-}iOTR}_1(\mathsf{crs}_{\mathsf{r\text{-}iOT}}, r_i)$ and $\gamma_{1-b,i} \leftarrow \widetilde{\mathsf{r\text{-}iOTR}}(\mathsf{crs}_{\mathsf{r\text{-}iOT}})$ for $i \in [\ell]$.
- R sends $\mathsf{M}_{\mathsf{R}} = \left(c, \{\gamma_{0,i}, \gamma_{1,i}\}_{i\in[\ell]}\right)$ as the receiver's message.

$\mathbf{iOTS}(\mathsf{crs}_{\mathsf{iOT}}, \mathsf{M}_{\mathsf{R}}, (m_0, m_1))$:
S runs the following algorithm for $\beta \in \{0,1\}$:

- S computes $c'_\beta = \mathsf{Com}(0; s_\beta)$. S garbles $\left(\mathrm{GC}'_\beta, \mathsf{Keys}'_\beta\right) = \mathsf{Gb}'(1^\kappa, \mathcal{C}'[\mathsf{crs}_{\mathsf{com}}]; r_{\beta,\mathrm{GC}'})$. S computes $\{\mathsf{Y}_{\beta,i}\}_{i\in[t+\ell+1]} = \mathsf{En}'(c'_\beta || s_\beta || m_\beta, \mathsf{Keys}'_\beta)$.
- The sender sets $\mathsf{V}_\beta = \{\mathsf{Y}_{\beta,i}\}_{i\in[t,t+\ell]}$, where $\{\mathsf{Y}_{\beta,i}\}_{i\in[t,t+\ell]}$ is the garbled input for $(s_\beta||m_\beta)$ in GC'_β. The sender sets $p_\beta = \{\mathsf{Y}_{\beta,i}\}_{i\in[t]}$ as the garbled input for c'_β corresponding to GC'_β.
- S garbles another garbled circuit for circuit $\mathcal{C}$ as $(\mathrm{GC}_\beta, \mathsf{Keys}_\beta) = (\mathrm{GC}_\beta, \{\mathsf{X}^0_{\beta,i}, \mathsf{X}^1_{\beta,i}\}_{i\in[\ell+t\kappa]}) = \mathsf{Gb}(1^\kappa, \mathcal{C}[c, \mathsf{crs}_{\mathsf{com}}, \beta]; r_{\beta,\mathrm{GC}})$. Let $|p_\beta| = t\kappa$ from the previous step. Sender sets $\mathsf{X}^p_\beta = \mathsf{En}(p_\beta, \{\mathsf{X}^0_{\beta,i}, \mathsf{X}^1_{\beta,i}\}_{i\in[\ell,\ell+t\kappa]})$ as the garbled input for p_β corresponding to GC_β.
- S computes $\pi_{\mathsf{r\text{-}iOT}}$ sender messages for receiver's input r in GC_β as $\tau_{\beta,i} = \mathsf{r\text{-}iOTS}(\mathsf{crs}_{\mathsf{r\text{-}iOT}}, \gamma_{\beta,i}, (\mathsf{X}^0_{\beta,i}, \mathsf{X}^1_{\beta,i}); r_{\beta,\mathsf{r\text{-}iOT}})$ for $i \in [\ell]$. It corresponds to the wire labels in GC_β corresponding to the bits of the commitment randomness r sampled by the receiver for constructing commitment $c = \mathsf{Com}(\mathsf{crs}_{\mathsf{com}}, \beta; r)$.

S sends $\mathsf{M}_s = \{\mathsf{M}_{\mathsf{S},\beta}\}_{\beta\in\{0,1\}} = \{\mathrm{GC}_\beta, \mathrm{GC}'_\beta, \{\tau_{\beta,i}\}_{i\in[\ell]}, \mathsf{X}^p_\beta, \mathsf{V}_\beta\}_{\beta\in\{0,1\}}$ and sets $r_{\mathsf{S}} = (r_0||r_1)$ where $r_\beta = (r_{\beta,\mathrm{GC}}, r_{\beta,\mathrm{GC}'}, r_{\beta,\mathsf{r\text{-}iOT}})$ for $\beta \in \{0,1\}$.

$\mathbf{iOTR}_2(\mathsf{crs}_{\mathsf{iOT}}, \mathsf{M}_{\mathsf{S}}, b)$:

- R computes the wire labels corresponding to commitment randomness r in GC_b as $\mathsf{X}_i = \mathsf{r\text{-}iOTR}_2(\mathsf{crs}_{\mathsf{r\text{-}iOT}}, \tau_{b,i})$ for $i \in [\ell]$. R evaluates GC_b on $\{\mathsf{X}_i\}_{i\in[\ell]}$ and X^p_b to receive garbled input U for c'_b corresponding to GC' as follows:

$$\mathsf{U} = \mathsf{Ev}(\mathrm{GC}_b, \{\mathsf{X}_i\}_{i\in[\ell]} || \mathsf{X}^p_b),$$

  where $|\mathsf{U}| = t\kappa$ and $\mathsf{U} = p_b$ corresponds to the wire labels for the commitment c'_b in GC'_b.
- R computes m_b by evaluating GC'_b on U and V_b as follows:

$$m_b = \mathsf{Ev}'(\mathrm{GC}', \mathsf{U}, \mathsf{V}).$$

– As in the earlier protocol, the sender sends across GC_0 and GC_1 to the receiver. The sender sends the wire labels, denoted as X^p_0 and X^p_1, for p_0 and p_1 corresponding to GC_0 and GC_1 respectively. Next, the sender computes the r-iOT sender messages similar to the previous simple protocol. The sender inputs to the r-iOT are the wire labels of the garbled circuits GC_0 and GC_1 corresponding to the receiver's input randomness for the receiver's commitment c.

  More formally, the sender uses the underlying r-iOT scheme and the two sets of ℓ number of r-iOT receiver messages from the receiver to generate two sets of ℓ number of r-iOT-sender messages. Let us denote the sets as the 0th set and the 1th set corresponding to GC_0 and GC_1. The 0th set corresponds to the ℓ bits of the commitment randomness of the receiver corresponding for commitment c. For the ith OT in the 0th set, the sender encrypts the garbled circuit wire labels (for GC_0) corresponding to the ith bit of

the receiver randomness, which is provided as input to the garbled circuit. Similarly, for the ith OT in the 1th set, the sender encrypts the garbled circuit wire labels (for GC_1) corresponding to the ith bit of the receiver randomness, which is provided as input to the garbled circuit. The sender sends the two sets of ℓ number of $\mathsf{r\text{-}iOT}$ sender messages to the receiver.

The sender also sends V_0 and V_1 to the receiver, which are the wire labels for inputs $(s_0||m_0)$ and $(s_1||m_1)$ in GC'_0 and GC'_1 respectively. Additionally, the sender also sends across the two garbled circuits GC'_0 and GC'_1 to the receiver.

Receiver's Message-Recovery. As in the simpler protocol described earlier, the receiver uses the $\mathsf{r\text{-}iOT}$-sender messages to recover the wire labels corresponding to its randomness string r for garbled circuit GC_b. It then evaluates GC_b on r by using the corresponding obtained wire labels from the $\mathsf{r\text{-}iOT}$ protocols and X_b^p string. However, unlike the simple protocol, it does not immediately recover the message m_b. Instead, it recovers the intermediate string $\mathsf{U} = p_b$. Recall, however, that $\mathsf{U} = p_b$ is nothing but a string representation of the wire-labels corresponding to the garbled circuit GC'_b on c'_b. Combined with the string V_b, $\mathsf{U}||\mathsf{V}_b$ is the garbling of (c'_b, s_b, m_b) for GC'_b. Hence, the receiver can evaluate the garbled circuit GC'_b using these wire-labels to recover the message m_b.

The complete solution is described formally in Figure 6. The oblivious sampling and randomness inversion algorithms are described in Figure 7.

Receiver Oblivious Sampleability. The receiver obliviously computes $\widetilde{\mathsf{iOTR}}$ by obliviously sampling the commitment c and obliviously sampling the $\mathsf{r\text{-}iOTR}$ messages using $\widetilde{\mathsf{r\text{-}iOTR}}$ algorithm. To argue oblivious sampleability, we need an inversion algorithm $\widehat{\mathsf{iOTR}_{\mathsf{Inv}}}$ that given an honestly generated M_R as the receiver's message and the corresponding randomness r_R computes the oblivious sampling randomness $\widehat{r_\mathsf{R}}$ such that the honestly generated receiver message and $\widehat{r_\mathsf{R}}$ is indistinguishable from obliviously sampled M_R and the sampling randomness. The inversion algorithm receives the commitment randomness and the $\mathsf{r\text{-}iOTR}$ receiver randomness as input corresponding to honestly generated commitment c and $\mathsf{r\text{-}iOTR}$ messages. The inversion algorithm invokes the inversion algorithms $\widehat{\mathsf{Com}_{\mathsf{Inv}}}$ (of commitment scheme) and $\widehat{\mathsf{r\text{-}iOTR}_{\mathsf{Inv}}}$ (for inverting randomness for $\mathsf{r\text{-}iOTR}$) to generate the sampling randomness as $\widehat{r_\mathsf{R}} = (\widehat{r_{\mathsf{Com}}}||\widehat{r_{\mathsf{r\text{-}iOT}}})$.

Sender Oblivious Sampleability. We now argue that the modified construction also achieves sender oblivious sampleability. To obliviously sample, i.e. $\widetilde{\mathsf{iOTS}}$, a sender message for the branch $w = (1 - b)$, the sender obliviously samples GC'_w (recall that GC'_w is generated using an obliviously sampleable garbling scheme Garble') to receive the obliviously sampled wire labels $\widetilde{\mathsf{Y}}$. The sender uses these wire labels to compute $\widetilde{\mathsf{V}}_w$ and $\widetilde{p_w}$ using $\widetilde{\mathsf{Y}}$. The sender garbles GC_w as per the "real" garbling scheme and computes $\widetilde{\mathsf{X}}^p$ by encoding $\widetilde{p_w}$. The $\mathsf{r\text{-}iOT}$ messages are computed using the wire labels of GC_w. Formal protocol details can be found in Fig. 7. To argue oblivious sampleability, we need an inversion algorithm $\widehat{\mathsf{iOTS}_{\mathsf{Inv}}}$ that given an honestly generated M_S as the sender's message and the corresponding randomness r_S computes the oblivious sampling randomness $\widehat{r_\mathsf{S}}$ such that the honestly generated receiver message and $\widehat{r_\mathsf{S}}$ is indistinguishable from obliviously sampled M_R and the sampling randomness r_S. The corresponding inversion algorithm takes as input the randomness used for correctly constructing GC_w, GC'_w and $\mathsf{r\text{-}iOT}$ messages. The simulator can now rely on the oblivious sampleability of the garbling scheme Garble' to claim that GC'_w was, in fact, obliviously sampled. The randomness for the honestly generated $\mathsf{r\text{-}iOT}$ messages and GC_w is provided as the sampling randomness. This is indistinguishable from an obliviously sampled sender message since in both cases GC_w evaluates to $\perp$. At this point, we rely on the equivocal property of the garbling scheme Garble to argue that these two cases are indistinguishable since the inputs of GC_w are predetermined from receiver's OT message. This holds true even when the sampling adversary gets all the input wire labels for GC_w from the $\mathsf{r\text{-}iOT}$ randomness. This is the fundamental reason why we added the extra "layer" of garbling to our protocol. In the formal proof, this argument is a bit more technically involved: we need to also rely on distinguisher dependent simulation techniques [JKKR17, DGH+20].

We prove that our OT protocol in Fig. 6 satisfies receiver and sender privacy, and we prove that the protocol satisfies receiver and sender oblivious sampleability (via the oblivious sampling and randomness inversion algorithms in Fig. 7) by proving Thm. 8.

Fig. 7. Oblivious Sampling and Randomness Inversion Algorithms for iOT

$\widetilde{\mathsf{iOTR}}(\mathsf{crs}_{\mathsf{iOT}})$:
R computes $c \leftarrow \widetilde{\mathsf{Com}}(\mathsf{crs}_{\mathsf{com}}; r_{\mathsf{Com}})$ and $\gamma_{\beta,i} \leftarrow \widetilde{\mathsf{r\text{-}iOTR}}(\mathsf{crs}_{\mathsf{r\text{-}iOT}}; r_{\mathsf{r\text{-}iOT},\beta,i})$ for $\beta \in \{0,1\}, i \in [\ell]$. Set $r_{\mathsf{r\text{-}iOT}} = \{r_{\mathsf{r\text{-}iOT},\beta,i}\}_{\beta\in\{0,1\},i\in[\ell]}$. R returns $r_{\mathsf{R}} = (r_{\mathsf{Com}}||r_{\mathsf{r\text{-}iOT}})$ as the sampling randomness.

$\widetilde{\mathsf{iOTR}}_{\mathsf{Inv}}(\mathsf{crs}, \mathsf{M}_{\mathsf{R}}, \mathsf{td} = (\mathsf{td}_{\mathsf{Com}}, \mathsf{td}_{\mathsf{r\text{-}iOT}}), r_{\mathsf{R}})$
Denote $r_{\mathsf{R}} = (r_{\mathsf{Com}}||r_{\mathsf{r\text{-}iOT}}||b||r)$ as the randomness for an honestly generated $\mathsf{M}_{\mathsf{R}} = \mathsf{iOTR}_1(\mathsf{crs}_{\mathsf{iOT}}, b; r_{\mathsf{R}})$. Obtain $\widehat{r_{\mathsf{Com}}} = \widetilde{\mathsf{Com}}_{\mathsf{Inv}}(\mathsf{crs}_{\mathsf{com}}, b, r_{\mathsf{Com}}, \mathsf{td}_{\mathsf{Com}})$ and $\widehat{r_{\mathsf{r\text{-}iOT}}} = \{\mathsf{r\text{-}i}\widetilde{\mathsf{OTR}}_{\mathsf{Inv}}(\mathsf{crs}_{\mathsf{r\text{-}iOT}}, \gamma_{\beta,i}, \mathsf{td}_{\mathsf{r\text{-}iOT}}, r_{\mathsf{r\text{-}iOT},\beta,i})\}_{\beta\in\{0,1\},i\in[\ell]}$. Return $\widehat{r}_{\mathsf{R}} = (\widehat{r_{\mathsf{Com}}}||\widehat{r_{\mathsf{r\text{-}iOT}}})$ as the randomness.

$\widetilde{\mathsf{iOTS}}(\mathsf{crs}_{\mathsf{iOT}}, \mathsf{M}_{\mathsf{R}}, w, m_{1-w}; r_{\mathsf{S}})$:
- S obliviously samples garbled circuit as $(\widetilde{\mathrm{GC}}'_w, \widetilde{\mathsf{Y}}) = \widetilde{\mathsf{Gb}}'(1^\kappa, \mathcal{C}'[\mathsf{crs}_{\mathsf{com}}], \bot; r_{w,\mathrm{GC}'})$ using randomness $r_{w,\mathrm{GC}'}$. The sender sets $\widetilde{p_w} = \{\widetilde{\mathsf{Y}}_{w,i}\}_{i\in[t]}$ and $\widetilde{\mathsf{V}}_w = \{\widetilde{\mathsf{Y}}_{w,i}\}_{i\in[t,t+\ell]}$.
- S correctly garbles GC_w as $(\mathrm{GC}_w, \mathsf{Keys}_w) = (\mathrm{GC}_w, \{\mathsf{X}^0_{w,i}, \mathsf{X}^1_{w,i}\}_{i\in[\ell+t\kappa]}) = \mathsf{Gb}(1^\kappa, \mathcal{C}[c, \mathsf{crs}_{\mathsf{com}}, b]; r_{w,\mathrm{GC}})$ using randomness $r_{w,\mathrm{GC}}$. Sender sets $\widetilde{\mathsf{X}}^p_w = \mathsf{En}(\widetilde{p_w}, \{\mathsf{X}^0_{w,i}, \mathsf{X}^1_{w,i}\}_{i\in[\ell,\ell+t\kappa]})$ as the garbled input for $\widetilde{p_w}$ in GC_w.
- S computes $\tau_{w,i}$ messages correctly as follows. S computes $\tau_{w,i} = \mathsf{r\text{-}iOTS}(\mathsf{crs}_{\mathsf{r\text{-}iOT}}, \gamma_{w,i}, (\mathsf{X}^0_{w,i}, \mathsf{X}^1_{w,i}); r_{w,\mathsf{r\text{-}iOT},i})$ using randomness $r_{w,\mathsf{r\text{-}iOT},i}$ for $i \in [\ell]$. Denote $r_{w,\mathsf{r\text{-}iOT}} = \{r_{w,\mathsf{r\text{-}iOT},i}\}_{i\in[\ell]}$.
- If $m_{1-w} \neq \bot$, S computes $\mathsf{M}_{\mathsf{S},1-w}$ correctly corresponding to message m_{1-w} using randomness r_{1-w}. Else, S repeats the above steps for branch $1-w$ similar to branch w.
- S sets $\mathsf{M}_{s,w} = (\mathrm{GC}_w, \widetilde{\mathrm{GC}}'_w, \{\tau_{w,i}\}_{i\in[\ell]}, \widetilde{\mathsf{X}}^p_w, \widetilde{\mathsf{V}}_w)$. Set r_w as the garbling randomness for GC_w, $\widetilde{\mathrm{GC}}'_w$ and $\{\tau_{w,i}\}_{i\in[\ell]}$ as $r_w = (r_{w,\mathrm{GC}}||r_{w,\mathrm{GC}'}||r_{w,\mathsf{r\text{-}iOT}})$.
- Return $\mathsf{M}_{\mathsf{S}} = (\mathsf{M}_{\mathsf{S},0}, \mathsf{M}_{\mathsf{S},1})$ as the sender OT message. Set the sender randomness as $r_{\mathsf{S}} = (r_0||r_1)$.

$\widetilde{\mathsf{iOTS}}_{\mathsf{Inv}}(\mathsf{crs}, w, \mathsf{M}_{\mathsf{S}}, \mathsf{td} = \bot, r_{\mathsf{S}})$ *(Inverts randomness for branch w)*
- Denote $r_{\mathsf{S}} = (r_0||r_1)$ and set $\widehat{r_{1-w}} = r_{1-w}$ as the randomness for an honestly generated M_{S} message.
- r_w contains the garbling randomness for GC_w, GC'_w and $\{\tau_{w,i}\}_{i\in[\ell]}$ as $r_w = (r_{w,\mathrm{GC}}||r_{w,\mathrm{GC}'}||r_{w,\mathsf{r\text{-}iOT}})$. Set $\widehat{r_{w,\mathrm{GC}}} = r_{w,\mathrm{GC}}$, $\widehat{r_{w,\mathrm{GC}'}} = \mathsf{Gb}_{\mathsf{Inv}}(r_{w,\mathrm{GC}'}, \mathsf{Keys}'_w, 0^{t+\ell+1})$, where Keys' is the encoding information of GC' when GC' was honestly garbled. Compute $\widehat{r_w} = (\widehat{r_{w,\mathrm{GC}}}||\widehat{r_{w,\mathrm{GC}'}}||r_{w,\mathsf{r\text{-}iOT}})$.
- Output $\widehat{r_{\mathsf{S}}} = (\widehat{r_0}||\widehat{r_1})$ and claim that $\mathsf{X}^p_w, \{\mathsf{Y}_{w,i}\}_{i\in[t,t+\ell]}$ were randomly sampled.

$\widetilde{\mathsf{iOTS}}_{\mathsf{Inv}}(\mathsf{crs}, \bot, \mathsf{M}_{\mathsf{S}}, \mathsf{td} = \bot, r_{\mathsf{S}})$ *(Inverts randomness for both branches $w \in \{0,1\}$)*
- Denote $r_{\mathsf{S}} = (r_0||r_1)$.
- For $w \in \{0,1\}$: r_w contains the garbling randomness for GC_w, GC'_w and $\{\tau_{w,i}\}_{i\in[\ell]}$ as $r_w = (r_{w,\mathrm{GC}}||r_{w,\mathrm{GC}'}||r_{w,\mathsf{r\text{-}iOT}})$. Set $\widehat{r_{w,\mathrm{GC}}} = r_{w,\mathrm{GC}}$, $\widehat{r_{w,\mathrm{GC}'}} = \mathsf{Gb}_{\mathsf{Inv}}(r_{w,\mathrm{GC}'}, \mathsf{Keys}'_w, 0^{t+\ell+1})$, where Keys' is the encoding information of GC' when GC' was honestly garbled. Compute $\widehat{r_w} = (\widehat{r_{w,\mathrm{GC}}}||\widehat{r_{w,\mathrm{GC}'}}||r_{w,\mathsf{r\text{-}iOT}})$.
- Output $\widehat{r_{\mathsf{S}}} = (\widehat{r_0}||\widehat{r_1})$ and claim that $\mathsf{X}^p_w, \{\mathsf{Y}_{w,i}\}_{i\in[t,t+\ell]}$ were randomly sampled for $w \in \{0,1\}$.

Theorem 8. *Assuming that: (1) $\pi_{\mathsf{r\text{-}iOT}}$ is a two-message r-iOT protocol satisfying sender privacy and oblivious receiver sampleability in the $\mathsf{crs}_{\mathsf{iOT}}$ model, (2) Com is an obliviously sampleable commitment scheme, (3) Garble is an equivocal garbling scheme, and (4) Garble′ is an obliviously sampleable garbling scheme, π_{iOT} is a two-message iOT protocol with sender and receiver oblivious sampleability in the CRS model.*

Proof. We prove Theorem 8 by showing receiver privacy, sender privacy, receiver oblivious sampleability and sender oblivious sampleability as follows. The corresponding algorithms can be found in Fig. 7.

Receiver Privacy. Receiver privacy of iOT follows from the hiding property of Com and oblivious receiver sampleability of r-iOT. We prove the following via a sequence of hybrids.

$$\{\mathsf{Com}(0;r), \{\gamma_{0,i}, \gamma_{1,i}\} : \gamma_{0,i} \leftarrow \mathsf{r\text{-}iOTR}_1(\mathsf{crs}_{\mathsf{r\text{-}iOT}}, r_i), \gamma_{1,i} \leftarrow \widetilde{\mathsf{r\text{-}iOTR}}(\mathsf{crs}_{\mathsf{r\text{-}iOT}})\}_{i\in[\ell]} \overset{c}{\approx}$$

$$\{\mathsf{Com}(1;r), \{\gamma_{0,i}, \gamma_{1,i}\} : \gamma_{1,i} \leftarrow \mathsf{r\text{-}iOTR}_1(\mathsf{crs}_{\mathsf{r\text{-}iOT}}, r_i), \gamma_{0,i} \leftarrow \widetilde{\mathsf{r\text{-}iOTR}}(\mathsf{crs}_{\mathsf{r\text{-}iOT}})\}_{i\in[\ell]}$$

Our hybrids and indistinguishability argument are as follows:

- $\mathtt{Hyb}_0 : \mathsf{M}_{\mathsf{R}} = \{\mathsf{Com}(0;r), \{\gamma_{0,i}, \gamma_{1,i}\} : \gamma_{0,i} \leftarrow \mathsf{r\text{-}iOTR}_1(\mathsf{crs}_{\mathsf{r\text{-}iOT}}, r_i), \gamma_{1,i} \leftarrow \widetilde{\mathsf{r\text{-}iOTR}}(\mathsf{crs}_{\mathsf{r\text{-}iOT}})\}_{i\in[\ell]}$. This is the receiver OT message M_{R} with choice bit 0 as input.

- $\mathtt{Hyb}_1$: $\mathsf{M_R} = \{\mathsf{Com}(0;r), \{\gamma_{0,i}, \gamma_{1,i}\} : \gamma_{0,i} \leftarrow \widetilde{\mathsf{r\text{-}iOTR}}(\mathsf{crs_{r\text{-}iOT}})\}_{i\in[\ell]}, \gamma_{1,i} \leftarrow \widetilde{\mathsf{r\text{-}iOTR}}(\mathsf{crs_{r\text{-}iOT}})\}_{i\in[\ell]}$. This is same as $\mathtt{Hyb}_0$, except $\gamma_{0,i}$ OT messages are obliviously sampled. An adversary distinguishing between $\mathtt{Hyb}_0$ and $\mathtt{Hyb}_1$ can be used to break oblivious receiver sampleability of $\mathsf{r\text{-}iOT}$ since $\gamma_{0,i}$ is honestly generated in $\mathtt{Hyb}_0$ and $\gamma_{0,i}$ is obliviously sampled in $\mathtt{Hyb}_1$.
- $\mathtt{Hyb}_2$: $\mathsf{M_R} = \{\mathsf{Com}(1;r), \{\gamma_{0,i}, \gamma_{1,i}\} : \gamma_{0,i} \leftarrow \widetilde{\mathsf{r\text{-}iOTR}}(\mathsf{crs_{r\text{-}iOT}})\}_{i\in[\ell]}, \gamma_{1,i} \leftarrow \widetilde{\mathsf{r\text{-}iOTR}}(\mathsf{crs_{r\text{-}iOT}})\}_{i\in[\ell]}$. This is same as $\mathtt{Hyb}_1$, except c is a commitment to 1 instead of commitment to 0. Indistinguishability follows from hiding property of the commitment scheme.
- $\mathtt{Hyb}_3$: $\mathsf{M_R} = \{\mathsf{Com}(1;r), \{\gamma_{0,i}, \gamma_{1,i}\} : \gamma_{1,i} \leftarrow \mathsf{r\text{-}iOTR}_1(\mathsf{crs_{r\text{-}iOT}}, r_i), \gamma_{0,i} \leftarrow \widetilde{\mathsf{r\text{-}iOTR}}(\mathsf{crs_{r\text{-}iOT}})\}_{i\in[\ell]}$. This is same as $\mathtt{Hyb}_2$, except $\gamma_{1,i}$ OT messages are correctly generated using r_i as input choice bits where r is the commitment randomness. This is the receiver OT message $\mathsf{M_R}$ with choice bit 1 as input. An adversary distinguishing between $\mathtt{Hyb}_2$ and $\mathtt{Hyb}_3$ can be used to break oblivious receiver sampleability of $\mathsf{r\text{-}iOT}$ since $\gamma_{1,i}$ is obliviously sampled in $\mathtt{Hyb}_2$ and $\gamma_{1,i}$ is honestly generated in $\mathtt{Hyb}_3$.

Sender Privacy. Sender privacy of iOT follows from the binding of the commitment scheme, sender privacy of $\mathsf{r\text{-}iOT}$, and the privacy of GC and GC′. If the adversary's choice bit is $1-w$ then the commitment c should be a valid commitment to $1-w$ using randomness r_{Com} that needs to be provided as input to $\mathsf{r\text{-}iOTR}$ by the receiver during the computation of $\gamma_{1-w,i}$. If c is also a valid commitment to w using randomness r'_{Com} and the receiver provides this as randomness to $\mathsf{r\text{-}iOTR}$ during the computation of $\gamma_{w,i}$ then the receiver breaks the binding property of the commitment scheme. By relying on distinguisher dependent simulation techniques, the reduction can the two openings - $(1-w, r_{\mathsf{Com}})$ and (w, r'_{Com}), of c and break the binding property. Hence, we can safely rely on the binding property of the commitment scheme to argue that (w, r'_{Com}) will be an invalid opening of c. As a result, GC_w will always evaluate to $\perp$ and as a result GC'_w would always evaluate to $\perp$ since the receiver fails to get the correct wire labels (corresponding to GC_w) for c'_w. This happens irrespective of whether GC'_w is evaluated on $m_w = 0$ or $m_w = 1$. Thus, indistinguishability follows from the privacy of the garbling scheme. We also require sender privacy of $\mathsf{r\text{-}iOT}$ since a corrupt receiver should not get both input wire labels for any input wire of GC_w. For adversary's chosen bit $(1-w) \in \{0,1\}$, we demonstrate the following:

$$\mathsf{iOTS}(\mathsf{crs_{iOT}}, \mathsf{M_R}, (m_0, m_1); r) \approx_c \mathsf{iOTS}(\mathsf{crs_{iOT}}, \mathsf{M_R}, (m'_0, m'_1); r'),$$

where $m'_{1-w} = m_{1-w}$, $m_w = 0$ and $m'_w = 1$. This is shown via a sequence of hybrids:

- $\mathtt{Hyb}_0$: Sender's message is $\mathsf{M_S} = \mathsf{iOTS}(\mathsf{crs_{iOT}}, \mathsf{M_R}, (m_0, m_1); r)$.
- $\mathtt{Hyb}_1$: Same as $\mathtt{Hyb}_0$, except the reduction extracts r_{Com} and r'_{Com} from the $\gamma_{1-w,i}$ and $\gamma_{w,i}$ messages as receiver input choice bits for the randomness using distinguisher dependent simulation techniques. If both $(1-w, r_{\mathsf{Com}})$ and (w, r'_{Com}) are valid openings of c then the reduction aborts. Else, the reduction computes $\mathsf{M_S} = \mathsf{iOTS}(\mathsf{crs_{iOT}}, \mathsf{M_R}, (m_0, m_1); r)$ correctly following the sender algorithm. An adversary distinguishes between $\mathtt{Hyb}_0$ and $\mathtt{Hyb}_1$ if it breaks the binding property of the commitment scheme.
- $\mathtt{Hyb}_2$: Same as $\mathtt{Hyb}_1$, except $\tau_{w,i} = \mathsf{r\text{-}iOT}(\mathsf{crs_{iOT}}, \gamma_{w,i}, (\mathsf{X}^0_{w,i}, \mathsf{X}^0_{w,i}))$. Indistinguishability follows from sender privacy of $\mathsf{r\text{-}iOT}$ since in both hybrids GC_w evaluates to $\perp$ (and as a result GC'_w also evaluates to $\perp$), given $\mathsf{X}^0_{w,i}$ or $\mathsf{X}^1_{w,i}$ (but not both), and the only difference lies in the $\mathsf{r\text{-}iOT}$ sender messages.
- $\mathtt{Hyb}_3$: Same as $\mathtt{Hyb}_2$, except GC_w is computed by invoking the privacy simulator for the garbling scheme with output $\perp$. The $\mathsf{r\text{-}iOT}$ sender messages are set similar to previous hybrid, i.e. $\tau_{w,i} = \mathsf{r\text{-}iOT}(\mathsf{crs_{iOT}}, \gamma_{w,i}, (\mathsf{X}_{w,i}, \mathsf{X}_{w,i}))$, where the privacy simulator $\mathcal{S}^1_{\mathsf{GC}}$ (of garbling scheme Gb) returns X_w as the wire labels on being invoked as $(\mathrm{GC}_w, \mathsf{X}_w, \mathsf{st}) \leftarrow \mathcal{S}^1_{\mathsf{GC}}(\mathcal{C}[c, \mathsf{crs_{com}}, \beta], \perp)$. Indistinguishability follows from the privacy of garbling scheme for GC_w since in both hybrids GC_w outputs $\perp$.
- $\mathtt{Hyb}_4$: Same as $\mathtt{Hyb}_3$, except GC'_w is computed using the privacy simulator $\mathcal{S}_{\mathsf{GC}}$ (of Gb') with output $\perp$. The privacy simulator $\mathcal{S}_{\mathsf{GC}}$ (of garbling scheme Gb') returns Y_w as the wire labels on being invoked as $(\mathrm{GC}'_w, \mathsf{Y}_w, \mathsf{st}') \leftarrow \mathcal{S}_{\mathsf{GC}}(\mathcal{C}'[\mathsf{crs_{com}}], \perp)$. The sender sets $\mathsf{V}_w = \{\mathsf{Y}_{w,i}\}_{i\in[t,t+\ell]}$ and $p_w = \{\mathsf{Y}_w\}_{i\in[t]}$. The rest of the sender protocol is same as $\mathtt{Hyb}_3$. Note that in $\mathtt{Hyb}_3$ the garbled circuit GC'_w was computed correctly and $p_w||\mathsf{V}_w$ was a valid encoding of $(c'_w||s_w||m_w)$ in GC'_w. Meanwhile in $\mathtt{Hyb}_4$ the receiver obtains simulated wire labels. GC'_w in both hybrids evaluates to $\perp$. As a result, if an adversarial receiver distinguishes between the two hybrids then it breaks the privacy of the garbling scheme Gb'.

– $\mathtt{Hyb}_5$: Same as $\mathtt{Hyb}_4$, except GC'_w is correctly garbled and the sender encodes $(c'_w||s_w||m'_w)$ as $p_w||\mathsf{V}_w$ using the encoding information generated during the garbling process. Indistinguishability between $\mathtt{Hyb}_4$ and $\mathtt{Hyb}_5$ follows from privacy of garbling scheme Gb' since in both hybrids GC'_w evaluates to $\perp$.

– $\mathtt{Hyb}_6$: Same as $\mathtt{Hyb}_5$, except GC_w is correctly garbled and the r-iOT sender messages are set as $\tau_{w,i} = \mathsf{r\text{-}iOT}(\mathsf{crs}_{\mathsf{iOT}}, \gamma_{w,i}, (\mathsf{X}^0_{w,i}, \mathsf{X}^0_{w,i}))$. Indistinguishability follows from privacy of GC since in both hybrids the receiver evaluates GC_w to $\perp$.

– $\mathtt{Hyb}_7$: Sender's message is $\mathsf{M_S} = \mathsf{iOTS}(\mathsf{crs}_{\mathsf{iOT}}, \mathsf{M_R}, (m'_0, m'_1))$. Indistinguishability follows from sender privacy of r-iOT since in $\mathtt{Hyb}_6$ the r-iOT sender messages are $\tau_{w,i} = \mathsf{r\text{-}iOT}(\mathsf{crs}_{\mathsf{iOT}}, \gamma_{w,i}, (\mathsf{X}^0_{w,i}, \mathsf{X}^0_{w,i}))$ and in $\mathtt{Hyb}_7$ the r-iOT sender messages are $\tau_{w,i} = \mathsf{r\text{-}iOT}(\mathsf{crs}_{\mathsf{iOT}}, \gamma_{w,i}, (\mathsf{X}^0_{w,i}, \mathsf{X}^1_{w,i}))$.

Receiver Oblivious Sampleability. To argue receiver oblivious sampleability the adversary needs to be provided with the obliviously sampled/honestly generated receiver message and the sampling randomness/inverted randomness for the receiver message $\mathsf{M_R}$. Receiver oblivious sampleability of iOT follows from the oblivious sampling property of Com and receiver oblivious sampleability of r-iOT. We demonstrate this via a sequence of hybrids:

– $\mathtt{Hyb}_0$: This hybrid corresponds to the receiver's view by running the oblivious receiver sampling algorithm $\widetilde{\mathsf{iOTR}}$. Compute $c = \widetilde{\mathsf{Com}}(\mathsf{crs}_{\mathsf{com}}; r_{\mathsf{Com}})$ and $\gamma_{b,i} \leftarrow \widetilde{\mathsf{r\text{-}iOTR}}(\mathsf{crs}_{\mathsf{r\text{-}iOT}}; r_{\mathsf{r\text{-}iOT}})$ for $b \in \{0,1\}, i \in [\ell]$. Return $r = (r_{\mathsf{Com}}||r_{\mathsf{r\text{-}iOT}})$ as the sampling randomness.

– $\mathtt{Hyb}_1$: Compute $c = \mathsf{Com}(\mathsf{crs}_{\mathsf{com}}, b; r_{\mathsf{Com}})$ and $\gamma_{\beta,i} \leftarrow \widetilde{\mathsf{r\text{-}iOTR}}(\mathsf{crs}_{\mathsf{r\text{-}iOT}}; r_{\mathsf{r\text{-}iOT}})$ for $\beta \in \{0,1\}, i \in [\ell]$. Compute commitment sampling randomness as $\widehat{r_{\mathsf{Com}}} = \widetilde{\mathsf{Com}}_{\mathsf{Inv}}(\mathsf{crs}_{\mathsf{com}}, b, r_{\mathsf{Com}}, \mathsf{td}_{\mathsf{Com}})$ and return $(\widehat{r_{\mathsf{Com}}}||r_{\mathsf{r\text{-}iOT}})$ as the sampling randomness. The two hybrids are indistinguishable due to oblivious sampleability of Com.

– $\mathtt{Hyb}_2$: This hybrid corresponds to the receiver's view by running the receiver algorithm iOTR_1 on choice bit b and generating the sampling randomness by running the inversion algorithm $\widetilde{\mathsf{iOTR}}_{\mathsf{Inv}}$. Compute $c = \mathsf{Com}(\mathsf{crs}_{\mathsf{com}}, b; r_{\mathsf{Com}})$. Construct r-iOT receiver messages as $\gamma_{b,i} = \mathsf{r\text{-}iOTR}_1(\mathsf{crs}_{\mathsf{r\text{-}iOT}}, r_{\mathsf{Com},i})$ and $\gamma_{\bar{b},i} \leftarrow \widetilde{\mathsf{r\text{-}iOTR}}(\mathsf{crs}_{\mathsf{r\text{-}iOT}})$ for $i \in [\ell]$ using $r_{\mathsf{r\text{-}iOT}}$ as randomness. Compute sampling randomness $\widehat{r_{\mathsf{R}}} = (\widehat{r_{\mathsf{Com}}}||\widehat{r_{\mathsf{r\text{-}iOT}}})$ using $\widetilde{\mathsf{iOTR}}_{\mathsf{Inv}}$ algorithm. Indistinguishability between the two hybrids follows from the receiver sampleability of r-iOT.

Sender Oblivious Sampleability. To argue sender oblivious sampleability the adversary needs to be provided with the obliviously sampled/honestly generated sender message and the sampling randomness/inverted randomness for the sender message $\mathsf{M_S}$. An obliviously sampled $\mathsf{M_S}$ contains $(\mathsf{M}_{\mathsf{S},0}, \mathsf{M}_{\mathsf{S},1})$. $\mathcal{A}$ provides (m_0, m_1) and $\mathsf{M}_{\mathsf{S},1-w}$ is correctly generated using m_{1-w} where $1-w$ is the choice bit of R. We will argue that an honestly generated $\mathsf{M}_{\mathsf{S},w}$ can be claimed as obliviously generated based on the equivocal property of GC_w and the oblivious sampling property of GC'_w. Let us concentrate on the structure of an obliviously sampled $\mathsf{M}_{\mathsf{S},w}$. It contains an honestly generated GC_w, an obliviously generated GC'_w and the r-iOT messages $\{\tau_{w,i}\}_{i\in[\ell]}$ are correctly constructed using the input wire labels of GC_w. But the wire labels for c'_w in GC'_w are not provided correctly in GC_w. As a result the receiver cannot obtain the correct input wire labels(corresponding to c'_w) for GC'_w since GC_w always evaluates to $\perp$ (since the receiver's commitment c is a commitment to $1-w$). This holds even if the adversarial receiver is provided with the sampling randomness for GC due to equivocal property of GC_w and oblivious sampleability of GC'_w. An honestly generated $\mathsf{M_S}$ would contain an honestly generated $\mathsf{M}_{\mathsf{S},w}$ which encrypts sender's message m_w. However, in this case the adversarial receiver also fails to obtain the correct input wire labels(corresponding to c'_w) for GC'_w since GC_w always evaluates to $\perp$. As a result, we show sender oblivious sampleability. The previous argument assumes that the commitment c is binding and it is a valid commitment to $1-w$. The formal argument for sender oblivious sampleability is more complex and we it is proven through a sequence of hybrids shown below. We only describe changes in $\mathsf{M}_{\mathsf{S},w}$ since $\mathsf{M}_{\mathsf{S},1-w}$ remains same in every hybrid.

– $\mathtt{Hyb}_0$: $\mathsf{M_S}$ is obliviously sampled using randomness r_{S}. Provide r_{S} as the sampling randomness.

Fig. 8. Hybrids 0-2 for arguing Sender Oblivious Sampleability

$\mathtt{Hyb}_0$: $\widetilde{\mathbf{iOTS}}(\mathsf{crs}_{\mathsf{iOT}}, \mathsf{M}_{\mathsf{R}}, w, m_{1-w}; r_{\mathsf{S}})$:

- S computes $\mathsf{M}_{\mathsf{S},1-w}$ correctly corresponding to message m_{1-w} using randomness r_{1-w}.
- S obliviously samples garbled circuit as $(\widetilde{\mathrm{GC}}'_w, \widetilde{\mathsf{Y}}) = \widetilde{\mathsf{Gb}}'(1^\kappa, \mathcal{C}'[\mathsf{crs}_{\mathsf{com}}], \perp; r_{w,\mathrm{GC}'})$ using randomness $r_{w,\mathrm{GC}'}$. The sender sets $\widetilde{p_w} = \{\widetilde{\mathsf{Y}}_{w,i}\}_{i\in[t]}$ and $\widetilde{\mathsf{V}}_w = \{\widetilde{\mathsf{Y}}_{w,i}\}_{i\in[t,t+\ell]}$.
- S correctly garbles GC_w as $(\mathrm{GC}_w, \mathsf{Keys}_w) = (\mathrm{GC}_w, \{\mathsf{X}^0_{w,i}, \mathsf{X}^1_{w,i}\}_{i\in[\ell+t\kappa]}) = \mathsf{Gb}(1^\kappa, \mathcal{C}[c, \mathsf{crs}_{\mathsf{com}}, b]; r_{w,\mathrm{GC}})$ using randomness $r_{w,\mathrm{GC}}$. Sender sets $\widetilde{\mathsf{X}}^p_w = \mathsf{En}(\widetilde{p_w}, \{\mathsf{X}^0_{w,i}, \mathsf{X}^1_{w,i}\}_{i\in[\ell,\ell+t\kappa]})$ as the garbled input for $\widetilde{p_w}$ in GC_w.
- S computes $\tau_{w,i}$ messages correctly as follows. S computes $\tau_{w,i} = \mathsf{r\text{-}iOTS}(\mathsf{crs}_{\mathsf{r\text{-}iOT}}, \gamma_{w,i}, (\mathsf{X}^0_{w,i}, \mathsf{X}^1_{w,i}); r_{w,\mathsf{r\text{-}iOT},i})$ using randomness $r_{w,\mathsf{r\text{-}iOT},i}$ for $i \in [\ell]$. Denote $r_{w,\mathsf{r\text{-}iOT}} = \{r_{w,\mathsf{r\text{-}iOT},i}\}_{i\in[\ell]}$.
- S sets $\mathsf{M}_{s,w} = (\mathrm{GC}_w, \widetilde{\mathrm{GC}}'_w, \{\tau_{w,i}\}_{i\in[\ell]}, \widetilde{\mathsf{X}}^p_w, \widetilde{\mathsf{V}}_w)$. Set r_w as the garbling randomness for GC_w, $\widetilde{\mathrm{GC}}'_w$ and $\{\tau_{w,i}\}_{i\in[\ell]}$ as $r_w = (r_{w,\mathrm{GC}}||r_{w,\mathrm{GC}'}||r_{w,\mathsf{r\text{-}iOT}})$.
- Return $\mathsf{M}_{\mathsf{S}} = (\mathsf{M}_{\mathsf{S},0}, \mathsf{M}_{\mathsf{S},1})$ as the sender OT message. Set the sender randomness as $r_{\mathsf{S}} = (r_0||r_1)$.

$\mathtt{Hyb}_1$: $\widetilde{\mathbf{iOTS}}(\mathsf{crs}_{\mathsf{iOT}}, \mathsf{M}_{\mathsf{R}}, w, m_{1-w}; r_{\mathsf{S}})$:

- The reduction (acting as S) extracts r_{Com} and r'_{Com} from the $\gamma_{1-w,i}$ and $\gamma_{w,i}$ messages as receiver input choice bits for the randomness using distinguisher dependent simulation techniques. If both $(1-w, r_{\mathsf{Com}})$ and (w, r'_{Com}) are valid openings of c then the reduction aborts.
- S computes $\mathsf{M}_{\mathsf{S},1-w}$ correctly corresponding to message m_{1-w} using randomness r_{1-w}.
- S obliviously samples garbled circuit as $(\widetilde{\mathrm{GC}}'_w, \widetilde{\mathsf{Y}}) = \widetilde{\mathsf{Gb}}'(1^\kappa, \mathcal{C}'[\mathsf{crs}_{\mathsf{com}}], \perp; r_{w,\mathrm{GC}'})$ using randomness $r_{w,\mathrm{GC}'}$. The sender sets $\widetilde{p_w} = \{\widetilde{\mathsf{Y}}_{w,i}\}_{i\in[t]}$ and $\widetilde{\mathsf{V}}_w = \{\widetilde{\mathsf{Y}}_{w,i}\}_{i\in[t,t+\ell]}$.
- S correctly garbles GC_w as $(\mathrm{GC}_w, \mathsf{Keys}_w) = (\mathrm{GC}_w, \{\mathsf{X}^0_{w,i}, \mathsf{X}^1_{w,i}\}_{i\in[\ell+t\kappa]}) = \mathsf{Gb}(1^\kappa, \mathcal{C}[c, \mathsf{crs}_{\mathsf{com}}, b]; r_{w,\mathrm{GC}})$ using randomness $r_{w,\mathrm{GC}}$. Sender sets $\widetilde{\mathsf{X}}^p_w = \mathsf{En}(\widetilde{p_w}, \{\mathsf{X}^0_{w,i}, \mathsf{X}^1_{w,i}\}_{i\in[\ell,\ell+t\kappa]})$ as the garbled input for $\widetilde{p_w}$ in GC_w.
- S computes $\tau_{w,i}$ messages correctly as follows. S computes $\tau_{w,i} = \mathsf{r\text{-}iOTS}(\mathsf{crs}_{\mathsf{r\text{-}iOT}}, \gamma_{w,i}, (\mathsf{X}^0_{w,i}, \mathsf{X}^1_{w,i}); r_{w,\mathsf{r\text{-}iOT},i})$ using randomness $r_{w,\mathsf{r\text{-}iOT},i}$ for $i \in [\ell]$. Denote $r_{w,\mathsf{r\text{-}iOT}} = \{r_{w,\mathsf{r\text{-}iOT},i}\}_{i\in[\ell]}$.
- S sets $\mathsf{M}_{s,w} = (\mathrm{GC}_w, \widetilde{\mathrm{GC}}'_w, \{\tau_{w,i}\}_{i\in[\ell]}, \widetilde{\mathsf{X}}^p_w, \widetilde{\mathsf{V}}_w)$. Set r_w as the garbling randomness for GC_w, $\widetilde{\mathrm{GC}}'_w$ and $\{\tau_{w,i}\}_{i\in[\ell]}$ as $r_w = (r_{w,\mathrm{GC}}||r_{w,\mathrm{GC}'}||r_{w,\mathsf{r\text{-}iOT}})$.
- Return $\mathsf{M}_{\mathsf{S}} = (\mathsf{M}_{\mathsf{S},0}, \mathsf{M}_{\mathsf{S},1})$ as the sender OT message. Set the sender randomness as $r_{\mathsf{S}} = (r_0||r_1)$.

$\mathtt{Hyb}_2$: $\widetilde{\mathbf{iOTS}}(\mathsf{crs}_{\mathsf{iOT}}, \mathsf{M}_{\mathsf{R}}, w, m_{1-w}; r_{\mathsf{S}})$:

- The reduction (acting as S) extracts r_{Com} and r'_{Com} from the $\gamma_{1-w,i}$ and $\gamma_{w,i}$ messages as receiver input choice bits for the randomness using distinguisher dependent simulation techniques. If both $(1-w, r_{\mathsf{Com}})$ and (w, r'_{Com}) are valid openings of c then the reduction aborts.
- S computes $\mathsf{M}_{\mathsf{S},1-w}$ correctly corresponding to message m_{1-w} using randomness r_{1-w}.
- S obliviously samples garbled circuit as $(\widetilde{\mathrm{GC}}'_w, \widetilde{\mathsf{Y}}) = \widetilde{\mathsf{Gb}}'(1^\kappa, \mathcal{C}'[\mathsf{crs}_{\mathsf{com}}], \perp; r_{w,\mathrm{GC}'})$ using randomness $r_{w,\mathrm{GC}'}$. The sender sets $\widetilde{p_w} = \{\widetilde{\mathsf{Y}}_{w,i}\}_{i\in[t]}$ and $\widetilde{\mathsf{V}}_w = \{\widetilde{\mathsf{Y}}_{w,i}\}_{i\in[t,t+\ell]}$.
- S garbles GC_w as $(\mathrm{GC}_w, \mathsf{X}_w, \mathsf{st}) \leftarrow \mathcal{S}^1_{\mathsf{GC}}(\mathcal{C}[c, \mathsf{crs}_{\mathsf{com}}, w], \perp)$ by invoking the privacy simulator $\mathcal{S}^1_{\mathsf{GC}}$ of the equivocal garbling scheme Garble. S computes $(\mathsf{Keys}_w, r_{w,\mathrm{GC}}) = (\{\mathsf{X}^0_{w,i}, \mathsf{X}^1_{w,i}\}_{i\in[\ell+t\kappa]}, \mathrm{GC}_w) \leftarrow \mathcal{S}^2_{\mathsf{GC}}(\mathsf{st}, r_{\mathsf{Com}})$ as the encoding information by invoking the privacy simulator $\mathcal{S}^2_{\mathsf{GC}}$ with input r_{Com} that is provided as input by the receiver to $\mathsf{r\text{-}iOT}$. Sender sets $\widetilde{\mathsf{X}}^p_w = \mathsf{En}(\widetilde{p_w}, \{\mathsf{X}^0_{w,i}, \mathsf{X}^1_{w,i}\}_{i\in[\ell,\ell+t\kappa]})$ as the garbled input for $\widetilde{p_w}$ in GC_w.
- S computes $\tau_{w,i}$ messages correctly as follows. S computes $\tau_{w,i} = \mathsf{r\text{-}iOTS}(\mathsf{crs}_{\mathsf{r\text{-}iOT}}, \gamma_{w,i}, (\mathsf{X}^0_{w,i}, \mathsf{X}^1_{w,i}); r_{w,\mathsf{r\text{-}iOT},i})$ using randomness $r_{w,\mathsf{r\text{-}iOT},i}$ for $i \in [\ell]$. Denote $r_{w,\mathsf{r\text{-}iOT}} = \{r_{w,\mathsf{r\text{-}iOT},i}\}_{i\in[\ell]}$.
- S sets $\mathsf{M}_{s,w} = (\mathrm{GC}_w, \widetilde{\mathrm{GC}}'_w, \{\tau_{w,i}\}_{i\in[\ell]}, \widetilde{\mathsf{X}}^p_w, \widetilde{\mathsf{V}}_w)$. Set r_w as the garbling randomness for GC_w, $\widetilde{\mathrm{GC}}'_w$ and $\{\tau_{w,i}\}_{i\in[\ell]}$ as $r_w = (r_{w,\mathrm{GC}}||r_{w,\mathrm{GC}'}||r_{w,\mathsf{r\text{-}iOT}})$.
- Return $\mathsf{M}_{\mathsf{S}} = (\mathsf{M}_{\mathsf{S},0}, \mathsf{M}_{\mathsf{S},1})$ as the sender OT message. Set the sender randomness as $r_{\mathsf{S}} = (r_0||r_1)$.

- $\mathtt{Hyb}_1$: Same as $\mathtt{Hyb}_0$, except the reduction extracts r_{Com} and r'_{Com} from the $\gamma_{1-w,i}$ and $\gamma_{w,i}$ messages as receiver input choice bits for the randomness using distinguisher dependent simulation techniques of [JKKR17, DGH+20]. If both $(1-w, r_{\mathsf{Com}})$ and (w, r'_{Com}) are valid openings of c then the reduction aborts. Else, the reduction obliviously samples M_{S} using randomness r_{S}. The reduction provides r_{S} as

Fig. 9. Hybrids 3-5 for arguing Sender Oblivious Sampleability

$\mathtt{Hyb}_3$: $\widetilde{\mathsf{iOTS}}(\mathsf{crs}_{\mathsf{iOT}}, \mathsf{M}_{\mathsf{R}}, w, (m_0, m_1); r_{\mathsf{S}})$:

- The reduction (acting as S) extracts r_{Com} and r'_{Com} from the $\gamma_{1-w,i}$ and $\gamma_{w,i}$ messages as receiver input choice bits for the randomness using distinguisher dependent simulation techniques. If both $(1-w, r_{\mathsf{Com}})$ and (w, r'_{Com}) are valid openings of c then the reduction aborts.
- S computes $\mathsf{M}_{\mathsf{S},1-w}$ correctly corresponding to message m_{1-w} using randomness r_{1-w}.
- S computes $c'_w = \mathsf{Com}(0; s_w)$. S garbles $(\mathrm{GC}'_w, \mathsf{Keys}'_w) = \mathsf{Gb}'(1^\kappa, \mathcal{C}'[\mathsf{crs}_{\mathsf{com}}])$. S computes $\{\mathsf{Y}_{w,i}\}_{i\in[t+\ell+1]} = \mathsf{En}'(c'_w||s_w||m_w, \mathsf{Keys}'_w)$. The sender sets $\mathsf{V}_w = \{\mathsf{Y}_{w,i}\}_{i\in[t,t+\ell]}$, where $\{\mathsf{Y}_{w,i}\}_{i\in[t,t+\ell]}$ is the garbled input for $(s_w||m_w)$ in GC'_w. The sender sets $p_w = \{\mathsf{Y}_{\beta,i}\}_{i\in[t]}$ as the garbled input for c'_w corresponding to GC'_w.
- S garbles GC_w as $(\mathrm{GC}_w, \mathsf{X}_w, \mathsf{st}) \leftarrow \mathcal{S}^1_{\mathsf{GC}}(\mathcal{C}[c, \mathsf{crs}_{\mathsf{com}}, w], \perp)$ by invoking the privacy simulator $\mathcal{S}^1_{\mathsf{GC}}$ of the equivocal garbling scheme Garble. S computes $(\mathsf{Keys}_w, r_{w,\mathrm{GC}}) = (\{\mathsf{X}^0_{w,i}, \mathsf{X}^1_{w,i}\}_{i\in[\ell+t\kappa]}, \mathrm{GC}_w) \leftarrow \mathcal{S}^2_{\mathsf{GC}}(\mathsf{st}, r_{\mathsf{Com}})$ as the encoding information by invoking the privacy simulator $\mathcal{S}^2_{\mathsf{GC}}$ with input r_{Com} that is provided as input by the receiver to r-iOT. Sender sets $\overline{X^p_w} = \mathsf{En}(p_w, \{\mathsf{X}^0_{w,i}, \mathsf{X}^1_{w,i}\}_{i\in[\ell,\ell+t\kappa]})$ as the garbled input for p_w in GC_w.
- S computes $\tau_{w,i}$ messages correctly as follows. S computes $\tau_{w,i} = \mathsf{r}$-$\mathsf{iOTS}(\mathsf{crs}_{\mathsf{r}\text{-}\mathsf{iOT}}, \gamma_{w,i}, (\mathsf{X}^0_{w,i}, \mathsf{X}^1_{w,i}); r_{w,\mathsf{r}\text{-}\mathsf{iOT},i})$ using randomness $r_{w,\mathsf{r}\text{-}\mathsf{iOT},i}$ for $i \in [\ell]$. Denote $r_{w,\mathsf{r}\text{-}\mathsf{iOT}} = \{r_{w,\mathsf{r}\text{-}\mathsf{iOT},i}\}_{i\in[\ell]}$.
- S sets $\mathsf{M}_{s,w} = (\mathrm{GC}_w, \widetilde{\mathrm{GC}}'_w, \{\tau_{w,i}\}_{i\in[\ell]}, \overline{\mathsf{X}^p_w}, \mathsf{V}_w)$. Set r_w as the garbling randomness for GC_w, $\widetilde{\mathrm{GC}}'_w$ and $\{\tau_{w,i}\}_{i\in[\ell]}$ as $r_w = (r_{w,\mathrm{GC}}||r_{w,\mathrm{GC}'}||r_{w,\mathsf{r}\text{-}\mathsf{iOT}})$.
- Return $\mathsf{M}_{\mathsf{S}} = (\mathsf{M}_{\mathsf{S},0}, \mathsf{M}_{\mathsf{S},1})$ as the sender OT message. Set the sender randomness as $r_{\mathsf{S}} = (r_0||r_1)$. Return the sender's sampling randomness as $\widehat{r_{\mathsf{S}}}$.

$\mathtt{Hyb}_4$: $\widetilde{\mathsf{iOTS}}(\mathsf{crs}_{\mathsf{iOT}}, \mathsf{M}_{\mathsf{R}}, w, (m_0, m_1); r_{\mathsf{S}})$:

- The reduction (acting as S) extracts r_{Com} and r'_{Com} from the $\gamma_{1-w,i}$ and $\gamma_{w,i}$ messages as receiver input choice bits for the randomness using distinguisher dependent simulation techniques. If both $(1-w, r_{\mathsf{Com}})$ and (w, r'_{Com}) are valid openings of c then the reduction aborts.
- S computes $\mathsf{M}_{\mathsf{S},1-w}$ correctly corresponding to message m_{1-w} using randomness r_{1-w}.
- S computes $c'_w = \mathsf{Com}(0; s_w)$. S garbles $(\mathrm{GC}'_w, \mathsf{Keys}'_w) = \mathsf{Gb}'(1^\kappa, \mathcal{C}'[\mathsf{crs}_{\mathsf{com}}])$. S computes $\{\mathsf{Y}_{w,i}\}_{i\in[t+\ell+1]} = \mathsf{En}'(c'_w||s_w||m_w, \mathsf{Keys}'_w)$. The sender sets $\mathsf{V}_w = \{\mathsf{Y}_{w,i}\}_{i\in[t,t+\ell]}$, where $\{\mathsf{Y}_{w,i}\}_{i\in[t,t+\ell]}$ is the garbled input for $(s_w||m_w)$ in GC'_w. The sender sets $p_w = \{\mathsf{Y}_{\beta,i}\}_{i\in[t]}$ as the garbled input for c'_w corresponding to GC'_w.
- S correctly garbles GC_w as $(\mathrm{GC}_w, \mathsf{Keys}_w) = (\mathrm{GC}_w, \{\mathsf{X}^0_{w,i}, \mathsf{X}^1_{w,i}\}_{i\in[\ell+t\kappa]}) = \mathsf{Gb}(1^\kappa, \mathcal{C}[c, \mathsf{crs}_{\mathsf{com}}, b]; r_{w,\mathrm{GC}})$ using randomness $r_{w,\mathrm{GC}}$. Sender sets $\mathsf{X}^p_w = \mathsf{En}(p_w, \{\mathsf{X}^0_{w,i}, \mathsf{X}^1_{w,i}\}_{i\in[\ell,\ell+t\kappa]})$ as the garbled input for p_w in GC_w.
- S computes $\tau_{w,i}$ messages correctly as follows. S computes $\tau_{w,i} = \mathsf{r}$-$\mathsf{iOTS}(\mathsf{crs}_{\mathsf{r}\text{-}\mathsf{iOT}}, \gamma_{w,i}, (\mathsf{X}^0_{w,i}, \mathsf{X}^1_{w,i}); r_{w,\mathsf{r}\text{-}\mathsf{iOT},i})$ using randomness $r_{w,\mathsf{r}\text{-}\mathsf{iOT},i}$ for $i \in [\ell]$. Denote $r_{w,\mathsf{r}\text{-}\mathsf{iOT}} = \{r_{w,\mathsf{r}\text{-}\mathsf{iOT},i}\}_{i\in[\ell]}$.
- S sets $\mathsf{M}_{s,w} = (\mathrm{GC}_w, \widetilde{\mathrm{GC}}'_w, \{\tau_{w,i}\}_{i\in[\ell]}, \mathsf{X}^p_w, \mathsf{V}_w)$. Set r_w as the garbling randomness for GC_w, $\widetilde{\mathrm{GC}}'_w$ and $\{\tau_{w,i}\}_{i\in[\ell]}$ as $r_w = (r_{w,\mathrm{GC}}||r_{w,\mathrm{GC}'}||r_{w,\mathsf{r}\text{-}\mathsf{iOT}})$.
- Return $\mathsf{M}_{\mathsf{S}} = (\mathsf{M}_{\mathsf{S},0}, \mathsf{M}_{\mathsf{S},1})$ as the sender OT message. Set the sender randomness as $r_{\mathsf{S}} = (r_0||r_1)$. Return the sender's sampling randomness as $\widehat{r_{\mathsf{S}}} = \widetilde{\mathsf{iOTS}}_{\mathsf{Inv}}(\mathsf{crs}, w, \mathsf{M}_{\mathsf{S}}, \mathsf{td} = \perp, r_{\mathsf{S}})$

$\mathtt{Hyb}_5$: $\widetilde{\mathsf{iOTS}}(\mathsf{crs}_{\mathsf{iOT}}, \mathsf{M}_{\mathsf{R}}, w, (m_0, m_1); r_{\mathsf{S}})$:

- Compute $(\mathsf{M}_{\mathsf{S}}, r_{\mathsf{S}}) = \mathsf{iOTS}(\mathsf{crs}_{\mathsf{iOT}}, \mathsf{M}_{\mathsf{R}}, (m_0, m_1))$. Compute sender's sampling randomness as $\widehat{r_{\mathsf{S}}} = \widetilde{\mathsf{iOTS}}_{\mathsf{Inv}}(\mathsf{crs}_{\mathsf{iOT}}, w, \mathsf{M}_{\mathsf{S}}, \perp, r_{\mathsf{S}})$.
- Return M_{S} as the sender OT message and $\widehat{r_{\mathsf{S}}}$ as the sender's sampling randomness.

the sampling randomness. An adversary distinguishes between $\mathtt{Hyb}_0$ and $\mathtt{Hyb}_1$ if it breaks the binding property of the commitment scheme.

- $\mathtt{Hyb}_2$: Same as $\mathtt{Hyb}_1$, except the reduction extracts r_{Com} as the input of the receiver to the wth instance of r-iOTR protocol. GC_w is computed by invoking $\mathcal{S}^1_{\mathsf{GC}}$ using output $\perp$. Then Keys_w is generated by invoking $\mathcal{S}^2_{\mathsf{GC}}$ using input r_{Com}. We use the distinguisher dependent simulation technique of [JKKR17, DGH+20] to find receiver's choice bits in $\{\gamma_{w,i}\}_{i\in[\ell]}$. The sampling randomness is provided similar to $\mathtt{Hyb}_0$ except the sampling randomness for GC_w is the one provided by the privacy simulator $\mathcal{S}^2_{\mathsf{GC}}$ of the equivocal garbling scheme Garble. Indistinguishability follows due to equivocal property of Garble since in both

hybrids GC_w evaluates to $\perp$ even when the adversary knows the garbling randomness for GC_w. As a result, GC'_w also evaluates to $\perp$.

- $\mathtt{Hyb}_3$: Same as $\mathtt{Hyb}_2$, except the algorithm receives (m_0, m_1) as input corresponding to the sender's input. The sender algorithm correctly computes c'_w and garbles GC'_w correctly. Indistinguishability follows from oblivious sampling property of Garble' since in both hybrids: 1) GC'_w evaluates to $\perp$ even when the adversary knows the garbling randomness for GC'_w, 2) in $\mathtt{Hyb}_2$, $\widetilde{\mathsf{V}}_w$ and $\widetilde{p_w}$ was generated as a challenge by the oblivious garbling algorithm $\widetilde{\mathsf{Gb}}'$, whereas in $\mathtt{Hyb}_3$, V_w and p_w are generated as a challenge by the honest garbling algorithm Gb'. An adversary distinguishing between the two hybrids breaks the oblivious garbling property of Garble'.
- $\mathtt{Hyb}_4$: Same as $\mathtt{Hyb}_3$, except GC_w is correctly garbled following the honest sender algorithm. The sampling randomness is computed by running the inversion algorithm $\widetilde{\mathsf{iOTS}_{\mathsf{Inv}}}$ on the sender randomness. Indistinguishability follows from the equivocal property of the garbling scheme since in both $\mathtt{Hyb}_3$ and $\mathtt{Hyb}_4$ GC_w evaluates to $\perp$. The indistinguishability argument is same as $\mathtt{Hyb}_1 \approx_c \mathtt{Hyb}_2$.
- $\mathtt{Hyb}_5$: This is same as $\mathtt{Hyb}_4$, except the reduction does not extract the commitment randomness from M_{R} in $\mathtt{Hyb}_5$. This is same as computing M_{S} using iOTS algorithm with (m_0, m_1) as inputs and r_{S} as randomness, then computing the sampling randomness by running the inversion algorithm $\widetilde{\mathsf{iOTS}_{\mathsf{Inv}}}$ on r_{S} and M_{S}. An adversary distinguishing between $\mathtt{Hyb}_4$ and $\mathtt{Hyb}_5$ needs to break the binding property of the commitment scheme.

□

5 Semi-Adaptive OT from r-iOT with Oblivious Sampleability

In this section, we show how to build a semi-adaptively simulation-secure two-message OT protocol starting from a two-message $\mathsf{r\text{-}iOT}$ protocol with receiver sampleability as follows.

5.1 Overview and Intuition

To generate the receiver OT message, the receiver uses the equivocal commitment scheme to create a commitment c to its choice bit b under some appropriately sampled randomness r. Next, the receiver creates a set of encryptions (e_0, e_1). We need two encryptions instead of one to enable semi-adaptive security (which is discussed later on). The encryption e_b encrypts the commitment randomness r under the trapdoor simulatable PKE scheme using some appropriately sampled randomness s (we explain the intuition for this step subsequently). Meanwhile, $e_{\bar{b}}$ is obliviously sampled. The receiver also creates a set of (parallel) $\mathsf{r\text{-}iOT}$-receiver messages with the bits of r and s as input. The receiver sends across to the sender the commitment c, the encryptions (e_0, e_1) and the $\mathsf{r\text{-}iOT}$-receiver messages.

Upon receiving the receiver's first message, the sender in the semi-adaptive OT protocol uses its input strings m_0 and m_1 to create two circuits. Based on the value of m_β, the garbled circuit GC_β is created as follows for $\beta \in \{0, 1\}$:

- If $m_\beta = 0$ the garbled circuit GC_β is garbling of the following circuit $\mathcal{C}$:

$$\mathcal{C}[c, \mathsf{crs}_{\mathsf{com}}, \mathsf{crs}_{\mathsf{pk}}, \beta, e, \mathsf{pk}](r, s) = 0 \text{ if } c = \mathsf{Com}(\mathsf{crs}_{\mathsf{com}}, b; r) \wedge e = \mathsf{Enc}(\mathsf{crs}_{\mathsf{pk}}, \mathsf{pk}, r; s);\ \text{else } \mathcal{C} \text{ outputs } \perp$$

  Circuit $\mathcal{C}$ is hardwired with the setup strings $\mathsf{crs}_{\mathsf{com}}, \mathsf{crs}_{\mathsf{pk}}$, the public key pk, a bit β, the receiver's commitment c and the receiver's commitment-encryption e_β; each circuit takes as input some randomness r and some randomness s and outputs 0 if all of the following conditions are satisfied: (a) c is a valid commitment to β under randomness r, (b) e_β is a valid encryption of r under randomness s. Otherwise it outputs $\perp$.
- Else, if $m_\beta = 1$, then S computes a simulated garbled circuit by invoking the privacy simulator $\mathcal{S}^1_{\mathsf{GC}}$ of the equivocal garbling scheme with output as $\perp$. To compute the input wire labels $\mathsf{Keys}_\beta = \{\mathsf{X}^0_{\beta,i}, \mathsf{X}^1_{\beta,i}\}_{i\in[\ell]}$, the sender invokes the second privacy simulator $\mathcal{S}^2_{\mathsf{GC}}$ with input as 0^ℓ.

Fig. 10. Semi-Adaptively Simulation-Secure Oblivious Transfer

π_{saOT}

- **Public Inputs:** $\mathsf{crs}_{\mathsf{OT}} = (\mathsf{crs}_{\mathsf{r\text{-}iOT}}, \mathsf{crs}_{\mathsf{com}})$ where $\mathsf{crs}_{\mathsf{r\text{-}iOT}}$ and $\mathsf{crs}_{\mathsf{com}}$ are the setup strings of r-iOT and Com, respectively. Circuit $\mathcal{C}[c, \mathsf{crs}_{\mathsf{com}}, \mathsf{crs}_{\mathsf{pk}}, \beta, e, \mathsf{pk}](r, s) = 0$ if $c = \mathsf{Com}(\mathsf{crs}_{\mathsf{com}}, b; r) \wedge e = \mathsf{Enc}(\mathsf{crs}_{\mathsf{pk}}, \mathsf{pk}, r; s)$; otherwise $\mathcal{C}$ outputs $\perp$.
- **Private Inputs:** S has input bits (m_0, m_1) where $m_0, m_1 \in \{0,1\}$; R has input choice bit b.
- **Primitives:** Let $\pi_{\mathsf{r\text{-}iOT}} = (\mathsf{r\text{-}iOTR}_1, \mathsf{r\text{-}iOTS}, \mathsf{r\text{-}iOTR}_2, \widetilde{\mathsf{r\text{-}iOTR}})$ denote an r-iOT with receiver oblivious sampling. Com is an equivocal commitment scheme. pk is the public key of a trapdoor simulatable PKE. $\mathsf{Garble} = (\mathsf{Gb}, \mathsf{En}, \mathsf{Ev}, \mathcal{S}^1_{\mathsf{GC}}, \mathcal{S}^2_{\mathsf{GC}})$ be an equivocal garbling scheme.

$\mathbf{OTR}_1(\mathsf{crs}_{\mathsf{OT}}, b)$:

- R commits to b using randomness r as $c = \mathsf{Com}(\mathsf{crs}_{\mathsf{com}}, b; r)$.
- R encrypts r using randomness s as $e_b = \mathsf{Enc}(\mathsf{crs}_{\mathsf{pk}}, \mathsf{pk}, r; s)$. R samples $e_{\bar{b}} \leftarrow \mathsf{oEnc}(\mathsf{crs}_{\mathsf{pk}}, \mathsf{pk})$ obliviously. Let $t = (r||s)$ denote the commitment and encryption randomness, where $|t| = \ell$.
- R computes $\{\gamma_{0,i}, \gamma_{1,i}\}$ where $\gamma_{b,i} = \mathsf{r\text{-}iOTR}_1(\mathsf{crs}_{\mathsf{r\text{-}iOT}}, t_i)$ and $\gamma_{\bar{b},i} \leftarrow \widetilde{\mathsf{r\text{-}iOTR}}(\mathsf{crs}_{\mathsf{r\text{-}iOT}})$ for $i \in [\ell]$.
- R sends $\mathsf{M}_{\mathsf{R}} = \big(c, e_0, e_1, \{\gamma_{0,i}, \gamma_{1,i}\}_{i\in[\ell]}\big)$ as the receiver's OT message.

$\mathbf{OTS}(\mathsf{crs}_{\mathsf{OT}}, \mathsf{M}_{\mathsf{R}}, (m_0, m_1))$:
S runs the following algorithm for $\beta \in \{0,1\}$:

- If $m_\beta = 0$ then S computes the garbled circuit and input encoding as $(\mathrm{GC}_\beta, \{\mathsf{X}^0_{\beta,i}, \mathsf{X}^1_{\beta,i}\}_{i\in[\ell]}) = \mathsf{Gb}(1^\kappa, \mathcal{C}[c, \mathsf{crs}_{\mathsf{com}}, \mathsf{crs}_{\mathsf{pk}}, \beta, e_\beta, \mathsf{pk}]; r_{\beta,\mathrm{GC}})$.
- If $m_\beta = 1$ then S computes a simulated garbled circuit as $(\mathrm{GC}_\beta, \widetilde{\mathsf{X}}, \mathsf{st}_\beta) \leftarrow \mathcal{S}^1_{\mathsf{GC}}(\mathcal{C}[c, \mathsf{crs}_{\mathsf{com}}, \mathsf{crs}_{\mathsf{pk}}, \beta, e_\beta, \mathsf{pk}], \perp)$. S computes the encoding information as $(\mathsf{Keys}_\beta, r_{\beta,\mathrm{GC}}) \leftarrow \mathcal{S}^2_{\mathsf{GC}}(\mathsf{st}_\beta, 0^\ell)$. S parses $\mathsf{Keys}_\beta = \{\mathsf{X}^0_{\beta,i}, \mathsf{X}^1_{\beta,i}\}_{i\in[\ell]}$.
- S computes the r-iOT sender messages as $\tau_{\beta,i} = \mathsf{r\text{-}iOTS}(\mathsf{crs}_{\mathsf{r\text{-}iOT}}, \gamma_{\beta,i}, (\mathsf{X}^0_{\beta,i}, \mathsf{X}^1_{\beta,i}); r_{\beta,\mathsf{r\text{-}iOT},i})$ for $i \in [\ell]$.

S sends $\mathsf{M}_{\mathsf{S}} = \{\mathrm{GC}_0, \{\tau_{0,i}\}_{i\in[\ell]}, \mathrm{GC}_1, \{\tau_{1,i}\}_{i\in[\ell]}\}$ as the sender's OT message

$\mathbf{OTR}_2(\mathsf{crs}_{\mathsf{OT}}, \mathsf{M}_{\mathsf{S}}, b)$:

- R computes the wire labels corresponding to commitment and encryption randomness t in GC_b as $\mathsf{Y}_i = \mathsf{r\text{-}iOTR}_2(\mathsf{crs}_{\mathsf{r\text{-}iOT}}, \tau_{b,i}, b)$ for $i \in [\ell]$.
- R outputs $m_b = 0$ if $\mathsf{Ev}(\mathrm{GC}, \{\mathsf{Y}_i\}_{i\in[\ell]}) = 0$ else R outputs $m_b = 1$.

The sender finally sends across GC_0 and GC_1 to the receiver. In parallel, the sender uses the r-iOT messages from the receiver to generate one r-iOT-sender message for each pair of wire labels, and also sends all of these to the receiver.

The receiver uses the r-iOT-sender messages to recover the wire labels corresponding to its randomness strings (r, s) for both garbled circuits GC_0 and GC_1. It then evaluates GC_b on r and s by using the corresponding wire labels to recover the correct message m_b (it sets m_b to 0 if the GC_b evaluates to 0; otherwise, it sets m_b to 1).

Our Protocol Figure 10 presents a detailed description of our semi-adaptively simulation-secure two-message OT protocol π_{saOT} in the CRS model from the following ingredients: (1) a two-message r-iOT protocol $\pi_{\mathsf{r\text{-}iOT}}$ with receiver oblivious sampleability in the CRS model, (2) a trapdoor simulatable PKE, (3) an equivocal garbling scheme Garble, and (4) an equivocal commitment scheme Com (in the CRS model). We state the following theorem.

Theorem 9. *Assuming that: (1) $\pi_{\mathsf{r\text{-}iOT}}$ is a two-message r-iOT protocol with receiver oblivious sampleability in the CRS model, (2) Com is an equivocal commitment scheme, (3) Garble is an equivocal garbling scheme, (4) the PKE scheme is trapdoor simulatable, π_{saOT} is simulation-secure in the CRS-model against semi-adaptive malicious corruption of parties.*

Proof.

Security against statically corrupt sender. The commitment c computationally hides the receiver's choice bit b because the encryption (e_0, e_1) computationally hides the receiver's randomness string r used for the

commitment, and the r-iOT messages sent by the receiver computationally hide the receiver's randomness string s for encryption.

A corrupt sender's messages can be extracted by a simulator $\mathcal{S}$ (playing the role of a simulated receiver). The simulator $\mathcal{S}$ constructs the commitment c in equivocal mode, i.e. $c = \mathsf{Com}(0; r_0) = \mathsf{Com}(1; r_1)$. The encryptions are set as follows - e_0 is an encryption of r_0 under randomness s_0 and e_1 is an encryption of r_1 under randomness s_1. $\mathcal{S}$ runs the two set of r-iOT messages correctly with input choice bits $t_0 = (r_0||s_0)$ and $t_1 = (r_1||s_1)$. Upon obtaining sender's OT message, the simulator decrypts input wire labels for both GC_0 and GC_1. $\mathcal{S}$ evaluates GC_0 and GC_1 to extract m_0 and m_1 respectively.

Security against statically corrupt receiver. Next, we describe a simulator that extracts a corrupt receiver's input. The receiver's input can be extracted using the secret key associated with the public key in the crs. The simulator decrypts e_0 and e_1 to obtain candidate randomness r_0 and r_1. It then checks whether $c = \mathsf{Com}(0; r_0)$ or $c = \mathsf{Com}(1; r_1)$. If both conditions are satisfied then the corrupt receiver has broken the binding property of the commitment scheme. Otherwise, the receiver's choice bit can be uniquely extracted. We argue that a corrupt receiver learns no information about $m_{\bar{b}}$. To see why this is the case, observe that the only information about $m_{\bar{b}}$ that the receiver could learn is from the garbled circuit $\mathrm{GC}_{\bar{b}}$. However, the receiver cannot evaluate $\mathrm{GC}_{\bar{b}}$ to anything other than $\perp$ since: (1) it cannot prove that c is a commitment to $\bar{b}$ under randomness r (this follows from the binding property of the commitment scheme), (2) it cannot prove that $e_{\bar{b}}$ decrypts to anything other than the commitment randomness r (this follows from the correctness of decryption for the PKE scheme), and (3) it cannot recover any input labels to $\mathrm{GC}_{\bar{b}}$ other than those corresponding to r (due to the sender privacy of the underlying r-iOT protocol). At this point, we invoke the privacy argument of the garbling scheme to argue that the receiver learns no information about the message $m_{\bar{b}}$. Sender privacy follows from the privacy of the garbling scheme, binding of the commitment scheme and the sender privacy of r-iOT. A corrupt receiver cannot obtain both wire labels for any input wire of a garbled circuit due to sender privacy of r-iOT. Given this argument holds, an honestly generated garbled circuit $\mathrm{GC}_{\bar{b}}$ (when $m_{\bar{b}=0}$) is indistinguishable from a simulated one (when $m_{\bar{b}=1}$) since in both cases the receiver evaluates $\mathrm{GC}_{\bar{b}}$ to $\perp$. This completes out description of static security.

□

Overview of Semi-adaptive Simulation-Security. Let us denote the set of OT messages for the bth branch (resp. $\bar{b}$th branch) as the bth set(resp. $\bar{b}$th set). Semi-adaptive simulation security considers two corruption scenarios: 1) the receiver gets corrupted post execution and the sender is statically corrupt, or 2) the receiver is statically corrupt and the sender gets corrupted post execution. In either of the cases, the simulator plays the role of the honest party which gets corrupted post-execution. The simulator needs to extract the input of the statically corrupt party. Also, when the honest party gets corrupted post execution, the simulator obtains the input of the honest party. The simulator needs to show randomness for the party such that the randomness is consistent with the party's input. We consider two corruption cases:

1. We first consider the case where the receiver gets corrupted post execution and the sender is statically corrupt. The simulator constructs the receiver OT message as described above. When the receiver gets corrupted post-execution the simulator shows randomness for the construction of e_b and claims that $c = \mathsf{Com}(b; r_b)$. It also claims that $e_{\bar{b}}$ and the r-iOT sender messages for the $\bar{b}$th set were obliviously sampled. Indistinguishability follows due to the equivocal property of the commitment scheme, the oblivious ciphertext sampleability of the encryption scheme, and the receiver sampleability of r-iOT.
2. Next we consider the case where the sender gets corrupted post-execution and the receiver is statically corrupted. In this setting the simulator $\mathcal{S}$ extracts the choice bit b from the receiver's OT message. The simulator invokes the OT functionality $\mathcal{F}_{\mathsf{OT}}$ with b to obtain m_b. $\mathcal{S}$ constructs GC_b and the r-iOT sender messages for the bth set correctly. $\mathcal{S}$ also constructs $\mathrm{GC}_{\bar{b}}$ and r-iOT sender messages for the $\bar{b}$th set correctly as if $m_{\bar{b}} = 0$. This helps to equivocate the sender's view if $m_{\bar{b}}$ turns out to be 1 when the sender gets corrupted post-execution. We know that the evaluation of $\mathrm{GC}_{\bar{b}}$ always yields $\perp$ since c is not a valid commitment to $\bar{b}$. If the simulator is required to show randomness for $m_{\bar{b}} = 1$ then the simulator claims that $\mathrm{GC}_{\bar{b}}$ was generated using the privacy simulator of the garbling scheme. This is indistinguishable from the real world execution due to the equivocal property of the garbling scheme.

We prove semi-adaptive security of π_{saOT} by proving Theorem. 9 by considering two cases of corruptions for our security proof as follows.

Fig. 11. Simulation against statically corrupted S and adaptively corrupted R

Simulating $\mathbf{OTR}_1$:

- $\mathcal{S}$ constructs commitment $c = \mathsf{Com}(0; r_0) = \mathsf{Com}(1; r_1)$ in equivocal mode using trapdoor $\mathsf{td}_{\mathsf{Com}}$ of $\mathsf{crs}_{\mathsf{com}}$.
- For $b \in \{0,1\}$, $\mathcal{S}$ encrypts r_b using randomness s_b as $e_b = \mathsf{Enc}(\mathsf{pk}, r_b; s_b)$. Set $t^b = (r_b || s_b)$.
- R computes $\{\gamma_{0,i}, \gamma_{1,i}\}$ where $\gamma_{b,i} = \mathsf{r\text{-}iOTR}_1(\mathsf{crs}_{\mathsf{r\text{-}iOT}}, t^b_i; u_{i,b})$ for $b \in \{0,1\}, i \in [\ell]$.
- R sends $\mathsf{M}_{\mathsf{R}} = \left(c, e_0, e_1, \{\gamma_{0,i}, \gamma_{1,i}\}_{i\in[\ell]}\right)$ as the receiver's OT message.

$\mathbf{OTS}(\mathsf{crs}_{\mathsf{OT}}, \mathsf{M}_{\mathsf{R}}, (m_0, m_1))$:
S^* sends $\mathsf{M}_{\mathsf{S}} = \{\mathrm{GC}_0, \{\tau_{0,i}\}_{i\in[\ell]}, \mathrm{GC}_1, \{\tau_{1,i}\}_{i\in[\ell]}\}$ as the sender's OT message

Simulating $\mathbf{OTR}_2$:

- For $b \in \{0,1\}$, $\mathcal{S}$ computes the wire labels corresponding to commitment and encryption randomness t^b in GC_b as $\mathsf{Y}_i = \mathsf{r\text{-}iOTR}_2(\mathsf{crs}_{\mathsf{r\text{-}iOT}}, \tau_{b,i}, b)$ for $i \in [\ell]$. $\mathcal{S}$ extracts $m_b = 0$ if $\mathsf{Ev}(\mathrm{GC}, \{\mathsf{Y}_i\}_{i\in[\ell]}) = 0$ else it sets $m_b = 1$.
- $\mathcal{S}$ invokes $\mathcal{F}_{\mathsf{OT}}$ functionality with input (m_0, m_1) and completes simulation.

Simulating Post-execution corruption of R:
$\mathcal{S}$ obtains receiver's choice bit b when R gets corrupted. $\mathcal{S}$ opens the randomness for commitment as (b, r_b) and encryption e_b as s_b. $\mathcal{S}$ opens randomness $u_{i,b}$ for the $\gamma_{b,i}$ messages corresponding to choice bit b. $\mathcal{S}$ claims that $e_{\bar{b}}$ and $\gamma_{\bar{b},i}$ were obliviously sampled by providing sampling randomness as $\mathsf{rGen}(\mathsf{crs}_{\mathsf{pk}}, s_{\bar{b}}, \mathsf{td}_{\mathsf{pk}})$ and $\widetilde{\mathsf{r\text{-}iOTR}}_{\mathsf{Inv}}(\mathsf{crs}_{\mathsf{r\text{-}iOT}}, \gamma_{\bar{b},i}, \mathsf{td}_{\mathsf{r\text{-}iOT}}, u_{i,\bar{b}})$ where $\mathsf{td}_{\mathsf{pk}}$ and $\mathsf{td}_{\mathsf{r\text{-}iOT}}$ are the trapdoors of $\mathsf{crs}_{\mathsf{pk}}$ and $\mathsf{crs}_{\mathsf{r\text{-}iOT}}$ respectively.

S^* is statically corrupted and R is post-execution corrupted The simulator plays the role of the honest receiver and constructs c in the equivocal mode such that it can decrypt both sender messages (m_0, m_1) from M_{S}. It constructs e_0 and e_1 honestly using the commitment randomness for bits 0 and 1. The $\mathsf{r\text{-}iOT}$ messages are also constructed honestly corresponding to bit 0 and 1. When the receiver gets corrupted and the simulator obtains input choice bit b, the simulator opens the randomness corresponding to bit b and claims that $e_{\bar{b}}$ and the $\mathsf{r\text{-}iOT}$ messages $(\gamma_{\bar{b},i})$ corresponding to $\bar{b}$ were obliviously sampled. Indistinguishability follows due to equivocal property of Com and oblivious sampleability of PKE and receiver oblivious sampleability of $\mathsf{r\text{-}iOT}$. Details of simulation can be found in Fig. 11. We provide the hybrids and argue indistinguishability as follows:

- $\mathtt{Hyb}_0$: Real world execution of the protocol.
- $\mathtt{Hyb}_1$: Same as $\mathtt{Hyb}_0$, except $c = \mathsf{Com}(0; r_0) = \mathsf{Com}(1; r_1)$ is constructed in the equivocal mode with randomness r_0 and r_1 respectively. Indistinguishability follows due to equivocal property of Com.
- $\mathtt{Hyb}_2$: Same as $\mathtt{Hyb}_1$, except $e_0 = \mathsf{Enc}(\mathsf{crs}_{\mathsf{pk}}, \mathsf{pk}, r_0)$ and $e_1 = \mathsf{Enc}(\mathsf{crs}_{\mathsf{pk}}, \mathsf{pk}, r_1)$. When R gets post adaptively corrupted, $\mathcal{S}$ claims that $e_{\bar{b}}$ was obliviously sampled. Indistinguishability follows due to trapdoor sampleability of PKE.
- $\mathtt{Hyb}_3$: Same as $\mathtt{Hyb}_2$, except $\gamma_{\bar{b},i}$ is constructed honestly following simulation algorithm. When R gets post adaptively corrupted, $\mathcal{S}$ claims that $\gamma_{\bar{b},i}$ was obliviously sampled. Indistinguishability follows due to receiver oblivious sampleability of $\mathsf{r\text{-}iOT}$. This is the ideal world execution of the protocol.

R^* is statically corrupted and S is post-execution corrupted The simulator plays the role of the honest sender and extracts the corrupt receiver's committed bit b as follows: Decrypts both e_0 and e_1 to obtain randomness r_0 and r_1, if both r_0 and r_1 are valid then $\mathcal{S}$ aborts else it sets $b = \sigma$ where r_σ is the valid randomness. A corrupt R^* cannot construct c in equivocal mode due to the binding property. After extracting b, $\mathcal{S}$ invokes $\mathcal{F}_{\mathsf{OT}}$ with b to obtain m_b. GC_b and $\tau_{b,i}$ are correctly constructed. Meanwhile, $\mathrm{GC}_{\bar{b}}$ and $\tau_{\bar{b},i}$ are constructed such that $m_{\bar{b}} = 0$. This takes care of sender simulation when sender gets corrupted post-execution. In such a case if $m_{\bar{b}} = 1$ then $\mathcal{S}$ claims that $\mathrm{GC}_{\bar{b}}$ was generated using the privacy simulator of the garbling scheme. The garbled circuit $\mathrm{GC}_{\bar{b}}$ outputs $\perp$ in both worlds since $c \neq \mathsf{Com}(\bar{b}; r_{\bar{b}})$. As a result, the adversary cannot distinguish between the real and ideal world due to equivocal property of GC. Details of simulation can be found in Fig. 12. We provide the hybrids and argue indistinguishability as follows:

- $\mathtt{Hyb}_0$: Real world execution of the protocol.
- $\mathtt{Hyb}_1$: Same as $\mathtt{Hyb}_0$, except $\mathcal{S}$ decrypts r_0 and r_1 and the reduction aborts if $c = \mathsf{Com}(0; r_0) = \mathsf{Com}(1; r_1)$. The adversary distinguishes between the two hybrids if it breaks the binding property of the commitment

Fig. 12. Simulation against statically corrupted R and adaptively corrupted S

$\mathbf{OTR}_1(\mathsf{crs}_{\mathsf{OT}}, b)$:
R^* sends $\mathsf{M}_\mathsf{R} = (c, e_0, e_1, \{\gamma_{0,i}, \gamma_{1,i}\}_{i\in[\ell]})$ as the receiver's OT message.

Simulating OTS:

- $\mathcal{S}$ decrypts $r_0 = \mathsf{Dec}(\mathsf{sk}, e_0)$ and $r_1 = \mathsf{Dec}(\mathsf{sk}, e_1)$ and proceeds as following:
  - *If* $c = \mathit{Com}(0; r_0) = \mathit{Com}(1; r_1)$ then $\mathcal{S}$ aborts.
  - *Else if* $c \neq \mathit{Com}(0; r_0)$ *and* $c \neq \mathit{Com}(1; r_1)$ then $\mathcal{S}$ sets $b = \perp$.
  - *Else,* $c = \mathit{Com}(\sigma; r_\sigma)$ for $\sigma \in \{0, 1\}$ and $\mathcal{S}$ sets extracted receiver's input as $b = \sigma$.
- $\mathcal{S}$ invokes $\mathcal{F}_{\mathsf{OT}}$ with b to obtain m_b.
- If $b \neq \perp$, then the simulator performs the following:
  - $\mathcal{S}$ constructs GC_b and $\{\tau_{b,i}\}_{i\in[\ell]}$ as per the honest sender algorithm using randomness $r_{b,\mathrm{GC}}$ and $\{r_{b,\mathsf{r\text{-}iOT},i}\}_{i\in[\ell]}$ respectively.
  - $\mathcal{S}$ sets $m_{\bar{b}} = 0$ and constructs $(\mathrm{GC}_{\bar{b}}, \{\mathsf{X}^0_{\bar{b},i}, \mathsf{X}^1_{\bar{b},i}\}_{i\in[\ell]}) = (\mathrm{GC}_{\bar{b}}, \mathsf{Keys}_{\bar{b}}) \leftarrow \mathsf{Gb}(r_{\bar{b},\mathrm{GC}}, C[c, \mathsf{crs}_{\mathsf{com}}, \mathsf{crs}_{\mathsf{pk}}, \bar{b}, e_{\bar{b}}, \mathsf{pk}])$ using randomness $r_{\bar{b},\mathrm{GC}}$.
  - $\mathcal{S}$ computes the $\mathsf{r\text{-}iOT}$ sender messages as $\tau_{\bar{b},i} = \mathsf{r\text{-}iOTS}(\mathsf{crs}_{\mathsf{r\text{-}iOT}}, \gamma_{\bar{b},i}, (\mathsf{X}^0_{\bar{b},i}, \mathsf{X}^1_{\bar{b},i}); r_{\bar{b},\mathsf{r\text{-}iOT},i})$ for $i \in [\ell]$ using randomness $r_{\bar{b},\mathsf{r\text{-}iOT},i}$.
- If $b == \perp$, then the simulator performs the following for $\beta \in \{0, 1\}$:
  - $\mathcal{S}$ sets $m_\beta = 0$ and constructs $(\mathrm{GC}_\beta, \{\mathsf{X}^0_{\beta,i}, \mathsf{X}^1_{\beta,i}\}_{i\in[\ell]}) = (\mathrm{GC}_\beta, \mathsf{Keys}_\beta) \leftarrow \mathsf{Gb}(r_{\beta,\mathrm{GC}}, C[c, \mathsf{crs}_{\mathsf{com}}, \mathsf{crs}_{\mathsf{pk}}, \beta, e_\beta, \mathsf{pk}])$ using randomness $r_{\beta,\mathrm{GC}}$.
  - $\mathcal{S}$ computes the $\mathsf{r\text{-}iOT}$ sender messages as $\tau_{\beta,i} = \mathsf{r\text{-}iOTS}(\mathsf{crs}_{\mathsf{r\text{-}iOT}}, \gamma_{\beta,i}, (\mathsf{X}^0_{\beta,i}, \mathsf{X}^1_{\beta,i}); r_{\beta,\mathsf{r\text{-}iOT},i})$ for $i \in [\ell]$ using randomness $r_{\beta,\mathsf{r\text{-}iOT},i}$.

$\mathcal{S}$ sends $\mathsf{M}_\mathsf{S} = \{\mathrm{GC}_0, \{\tau_{0,i}\}_{i\in[\ell]}, \mathrm{GC}_1, \{\tau_{1,i}\}_{i\in[\ell]}\}$ as the sender's OT message.

$\mathbf{OTR}_2(\mathsf{crs}_{\mathsf{OT}}, \mathsf{M}_\mathsf{S}, b)$:
R^* performs its own adversarial algorithm.

Simulating Post-execution corruption of S:
$\mathcal{S}$ obtains sender messages (m_0, m_1) when sender gets adaptively corrupted. $\mathcal{S}$ simulates the sender's randomness as follows based on the extracted bit b as follows.

- If $b \neq \perp$ then $\mathcal{S}$ performs the following:
  - For m_b, $\mathcal{S}$ opens the randomness for GC_b as $r_{b,\mathrm{GC}}$.
  - If the adversary provided $m_{\bar{b}} = 0$, then $\mathcal{S}$ also opens the randomness for $\mathrm{GC}_{\bar{b}}$ as $r_{\bar{b},\mathrm{GC}}$ and claims that it was honestly garbled.
  - If the adversary provided $m_{\bar{b}} = 1$, then $\mathcal{S}$ claims that $\mathrm{GC}_{\bar{b}}$ was garbled by running the $\mathcal{S}^1_{\mathsf{GC}}$ and $\mathcal{S}^2_{\mathsf{GC}}$ algorithms using randomness $r_{\bar{b},\mathrm{GC}}$.
- If $b == \perp$, then $\mathcal{S}$ performs the following for $\beta \in \{0, 1\}$:
  - If the adversary provided $m_\beta = 0$, then $\mathcal{S}$ also opens the randomness for GC_β as $r_{\beta,\mathrm{GC}}$ and claims that it was honestly garbled.
  - If the adversary provided $m_\beta = 1$, then $\mathcal{S}$ claims that GC_β was garbled by running the $\mathcal{S}^1_{\mathsf{GC}}$ and $\mathcal{S}^2_{\mathsf{GC}}$ algorithms using randomness $r_{\beta,\mathrm{GC}}$.

The simulator provides $\{r_{0,\mathsf{r\text{-}iOT},i}\}_{i\in[\ell]}$ and $\{r_{1,\mathsf{r\text{-}iOT},i}\}_{i\in[\ell]}$ as the sampling randomness for the $\mathsf{r\text{-}iOT}$ sender messages.

scheme by constructing c in equivocal mode. In such a case, the reduction return $(0, r_0)$ and $(1, r_1)$ as the openings to the commitment.

- $\mathtt{Hyb}_2$: Same as $\mathtt{Hyb}_1$, except $\mathcal{S}$ extracts (using distinguisher dependent simulation [JKKR17, DGH+20]) the inputs of R to $\{\gamma_{0,i}\}_{i\in[\ell]}$ and $\{\gamma_{1,i}\}_{i\in[\ell]}$ as $(r'_0||s'_0)$ and $(r'_1||s'_1)$ respectively. $\mathcal{S}$ sets $b = \perp$ if the malicious receiver didn't provide the decommitments of c as OT inputs. More specifically, $\mathcal{S}$ sets $b = \perp$ if the following conditions hold:

$$(c \neq \mathsf{Com}(\mathsf{crs}_{\mathsf{com}}, 0; r'_0) \vee e_0 \neq \mathsf{Enc}(\mathsf{crs}_{\mathsf{pk}}, \mathsf{pk}, r'_0; s'_0)) \wedge (c \neq \mathsf{Com}(\mathsf{crs}_{\mathsf{com}}, b; r'_1) \vee e_1 \neq \mathsf{Enc}(\mathsf{crs}_{\mathsf{pk}}, \mathsf{pk}, r'_1; s'_1))$$

If $b == \perp$, then $\mathcal{S}$ assumes $m_0 = m_1 = 1$ and generates GC_0 and GC_1 by invoking the privacy simulator on input $(r'_0||s'_0)$ and $(r'_1||s'_1)$ respectively. The $\mathsf{r\text{-}iOT}$ sender messages are computed using the input

wire labels of GC_0 and GC_1 following the honest sender algorithm. Upon post-execution corruption of the sender, the simulator obtains m_0 and m_1. If $b == \perp$, then for $\beta \in \{0,1\}$, if $m_\beta == 1$, then the simulator shows the randomness for m_β. If $m_\beta == 0$, then the simulator claims that GC_β was correctly generated. The simulator follows the honest sender algorithm for showing the r-iOT sender randomness. Indistinguishability follows from the r-iOT sender privacy and equivocal property of the garbling scheme since in $\mathtt{Hyb}_1$, GC_0 and GC_1 encrypt the actual sender inputs (m_0, m_1), whereas in $\mathtt{Hyb}_2$, GC_0 and GC_1 encrypt $m_0 = m_1 = 1$. An adversary distinguishing between the two hybrids must either break the OT sender privacy to obtain the GC wire labels, evaluate the garbled circuits and distinguish between the hybrids based on the evaluated output, or the adversary can distinguish between the garbled circuits of both hybrids based on the randomness and the input wire labels (obtained after post-execution corruption of sender) and break equivocal garbling property.

- $\mathtt{Hyb}_3$: Same as $\mathtt{Hyb}_2$, except $\mathcal{S}$ sets $b = \sigma$ where $c = \mathsf{Com}(\sigma, r_\sigma)$ (unless it was set as $\perp$ based on the algorithm in $\mathtt{Hyb}_2$). $\mathcal{S}$ invokes $\mathcal{F}_{\mathsf{OT}}$ with b to obtain m_b. $\mathcal{S}$ constructs GC_b and $\{\tau_{b,i}\}_{i\in[\ell]}$ honestly based on m_b. $\mathcal{S}$ constructs $\mathrm{GC}_{\bar{b}}$ considering $m_{\bar{b}} = 0$ by running the honest garbling algorithm using randomness $r_{\bar{b},\mathrm{GC}}$. Then, $\mathcal{S}$ constructs $\{\tau_{\bar{b},i}\}_{i\in[\ell]}$ by using the input wire labels of GC_{1-b} as input. When S gets corrupted post-execution and if $m_{\bar{b}} = 0$ then the simulator opens the randomness and claims the algorithm as it has computed above. If $m_{\bar{b}} = 1$, then the simulator claims that $\mathrm{GC}_{\bar{b}}$ was generated using the privacy simulator of the garbling scheme and the simulator provides the corresponding randomness as $r_{\bar{b},\mathrm{GC}}$. Indistinguishability follows from equivocal garbling since in $\mathtt{Hyb}_3$ $\mathrm{GC}_{\bar{b}}$ is correctly generated even when $m_{\bar{b}} == 1$, but in $\mathtt{Hyb}_3$ $\mathrm{GC}_{\bar{b}}$ is simulated and in both hybrids $\mathrm{GC}_{\bar{b}}$ evaluates to $\perp$ since c is not a valid commitment to bit $\bar{b}$.
- $\mathtt{Hyb}_4$: Same as $\mathtt{Hyb}_3$, except the simulator does not perform the OT input extraction (using distinguisher dependent simulation) of $\{\tau_{0,i}\}_{i\in[\ell]}$ and $\{\tau_{1,i}\}_{i\in[\ell]}$ as $(r'_0||s'_0)$ and $(r'_1||s'_1)$ respectively. The simulator only sets $b == \perp$, if $c \neq \mathsf{Com}(0; r_0)$ and $c \neq \mathsf{Com}(1; r_1)$ where r_0 and r_1 are decrypted from e_0 and e_1 respectively. This is the ideal world execution of the protocol and the details can be found in Fig. 12. Indistinguishability follows from OT sender privacy since in $\mathtt{Hyb}_3$ the receiver can still compute the correct output if e_0 (or e_1) encrypts a valid commitment randomness r_0 (or r_1) while providing a different input $r'_0 \neq r_0$ (or $r'_1 \neq r_1$) in the OT as input. Meanwhile, in $\mathtt{Hyb}_4$, the receiver will always fail to compute the correct output if it provides a bad opening $r'_0 \neq r_0$ (or $r'_1 \neq r_1$) in the OT as input. However, the receiver needs to break OT sender privacy to distinguish between the two hybrids.

6 Trapdoor Simulatable PKE from r-iOT

In this section, we show that any (two-message) r-iOT protocol implies a trapdoor simulatable PKE. The work of [CDMW09] constructed a two-round augmented NCE protocol from any trapdoor simulatable PKE scheme. This implies that any (two-message) r-iOT protocol implies a two-round augmented NCE protocol.

We actually show that any (two-message) iOT protocol satisfying *both* receiver *and* sender oblivious sampleability implies a trapdoor simulatable PKE. Since we already showed in Section 4 that any (two-message) r-iOT protocol implies that a (two-message) iOT protocol satisfying both receiver and sender oblivious sampleability, this yields our desired result.

Our Construction. Let $\mathsf{iOT} = (\mathsf{Setup}_{\mathsf{iOT}}, \mathsf{iOTR}_1, \mathsf{iOTS}, \mathsf{iOTR}_2)$ be an indistinguishability based OT. We construct a trapdoor simulatable PKE as follows:

- $\mathsf{Setup}(1^\kappa)$: Sample and output $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{Setup}_{\mathsf{iOT}}(1^\kappa)$.
- $\mathsf{Gen}(\mathsf{crs})$: Sample $\mathsf{M}_\mathsf{R} = \mathsf{iOTR}_1(\mathsf{crs}, 0; \mathsf{rr}_\mathsf{R})$ for uniformly sampled receiver randomness rr_R. Output $(\mathsf{pk}, \mathsf{sk}) = (\mathsf{M}_\mathsf{R}, \mathsf{rr}_\mathsf{R})$.
- $\mathsf{Enc}(\mathsf{crs}, \mathsf{pk} = \mathsf{M}_\mathsf{R}, m)$: Sample $m' \leftarrow \{0,1\}$ and generate the OT sender message $\mathsf{M}_\mathsf{S} \leftarrow \mathsf{iOTS}(\mathsf{crs}, \mathsf{M}_\mathsf{R}, (m, m'))$. Output the ciphertext $\mathsf{ct} = \mathsf{M}_\mathsf{S}$.
- $\mathsf{Dec}(\mathsf{crs}, \mathsf{sk}, \mathsf{ct} = \mathsf{M}_\mathsf{S})$: Output $m' = \mathsf{iOTR}_2(\mathsf{crs}, \mathsf{sk}, \mathsf{M}_\mathsf{S})$.

Additionally, suppose that iOT is equipped with the oblivious sampling algorithms - $(\widetilde{\mathsf{iOTR}}, \widetilde{\mathsf{iOTS}})$ for the receiver and sender, and the corresponding inverting algorithms - $(\widetilde{\mathsf{iOTR}_{\mathsf{Inv}}}, \widetilde{\mathsf{iOTS}_{\mathsf{Inv}}})$. We design the trapdoor simulatable PKE to have oblivious sampling algorithms $(\mathsf{oGen}, \mathsf{oEnc})$ and randomness inverting algorithms $(\mathsf{rGen}, \mathsf{rEnc})$ defined as follows:

- $\mathsf{oGen}(\mathsf{crs}; \widetilde{\mathsf{rr}}_G)$: Sample $\widetilde{\mathsf{M}}_\mathsf{R} = \widetilde{\mathsf{iOTR}}(\mathsf{crs}; \widetilde{\mathsf{rr}}_G)$ and output $\widetilde{\mathsf{pk}} = \widetilde{\mathsf{M}}_\mathsf{R}$.
- $\mathsf{oEnc}(\mathsf{crs}, \widetilde{\mathsf{pk}}; \widetilde{\mathsf{rr}}_E)$: Sample $m' \leftarrow \{0,1\}$ and $\widetilde{\mathsf{M}}_\mathsf{S} = \widetilde{\mathsf{iOTS}}(\mathsf{crs}, \widetilde{\mathsf{M}_\mathsf{R}}, 0, m'; \widetilde{\mathsf{rr}}_E)$. Output $\widetilde{\mathsf{ct}} = \widetilde{\mathsf{M}}_\mathsf{S}$.
- $\mathsf{rGen}(\mathsf{crs}, \mathsf{rr}_G, \mathsf{td})$: Generate $\mathsf{M}_\mathsf{R} = \mathsf{iOTR}_1(\mathsf{crs}, 0; \mathsf{rr}_G)$ and output oblivious sampling randomness for key generation
$$\widehat{\mathsf{rr}}_G = \widetilde{\mathsf{iOTR}_{\mathsf{Inv}}}(\mathsf{crs}, \mathsf{M}_\mathsf{R}, \mathsf{td}, \mathsf{rr}_G).$$
- $\mathsf{rEnc}(\mathsf{crs}, m, \mathsf{rr}_G, \mathsf{rr}_E, \mathsf{td})$: Generate the following:
$$\mathsf{M}_\mathsf{R} = \mathsf{iOTR}_1(\mathsf{crs}, 0; \mathsf{rr}_G), \quad \mathsf{M}_\mathsf{S} = \mathsf{iOTS}(\mathsf{crs}, \mathsf{M}_\mathsf{R}, (m, m'); \mathsf{rr}_E),$$
and output oblivious sampling randomness for encryption
$$\widehat{\mathsf{rr}}_E = \widetilde{\mathsf{iOTS}_{\mathsf{Inv}}}(\mathsf{crs}, 0, \mathsf{M}_\mathsf{S}, \mathsf{td}, \mathsf{rr}_E).$$

Correctness of decryption follows immediately from the correctness of the underlying iOT scheme.

Theorem 10. *Our construction of trapdoor simulatable PKE is IND-CPA secure assuming that iOT satisfies indistinguishability security against a semi-honest sender and a semi-honest receiver. Our construction of trapdoor simulatable PKE satisfies trapdoor oblivious sampleability and randomness inversion assuming that iOT satisfies oblivious receiver and sender sampleability.*

Proof. At a high level, ensuring oblivious sampleability (correspondingly randomness inversion) of the public key and ciphertexts in the resulting trapdoor simulatable PKE are relatively straightforward; one can simply reuse the receiver and sender oblivious sampling (correspondingly randomness inversion) algorithms provided by the iOT for obliviously sampling (correspondingly, inverting the randomness of) the public key and the ciphertext, respectively. Formally, we require that for any message $m \in \{0,1\}^\ell$, the following holds:

$$(\widehat{r}_G, \widehat{r}_E, \mathsf{pk}, \mathsf{ct}) \overset{c}{\approx} (\widetilde{r}_G, \widetilde{r}_E, \widetilde{\mathsf{pk}}, \widetilde{\mathsf{ct}})$$

where $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{Setup}(1^\kappa)$ and let

$$\begin{aligned} (\mathsf{pk}, \mathsf{sk}) &= \mathsf{Gen}(\mathsf{crs}; \mathsf{rr}_G) \quad , \quad \mathsf{ct} = \mathsf{Enc}(\mathsf{crs}, \mathsf{pk}, m; \mathsf{rr}_E), \\ \widehat{\mathsf{rr}}_G &= \mathsf{rGen}(\mathsf{crs}, \mathsf{rr}_G, \mathsf{td}) \quad , \quad \widehat{\mathsf{rr}}_E = \mathsf{rEnc}(\mathsf{crs}, m, \mathsf{rr}_G, \mathsf{rr}_E, \mathsf{td}). \end{aligned}$$

for appropriately sampled random coins rr_G and rr_E. Also, let

$$\widetilde{\mathsf{pk}} = \mathsf{oGen}(\mathsf{crs}; \widetilde{\mathsf{rr}}_G) \quad , \quad \widetilde{\mathsf{ct}} = \mathsf{oEnc}(\mathsf{crs}, \widetilde{\mathsf{pk}}; \widetilde{\mathsf{rr}}_E),$$

for appropriately sampled random coins $\widetilde{\mathsf{rr}}_G$ and $\widetilde{\mathsf{rr}}_E$, and for $m' \leftarrow \{0,1\}$.

$\mathtt{Hyb}_0$. In this hybrid, we have the tuple $(\widehat{\mathsf{rr}}_G, \widehat{\mathsf{rr}}_E, \mathsf{pk}, \mathsf{ct})$ distributed identically to the tuple

$$(\widetilde{\mathsf{iOTR}_{\mathsf{Inv}}}(\mathsf{crs}, \mathsf{M}_\mathsf{R}, \mathsf{td}, \mathsf{rr}_G), \widetilde{\mathsf{iOTS}_{\mathsf{Inv}}}(\mathsf{crs}, 0, \mathsf{M}_\mathsf{S}, \mathsf{td}, \mathsf{rr}_E), \mathsf{M}_\mathsf{R}, \mathsf{M}_\mathsf{S})$$

where

$$\mathsf{M}_\mathsf{R} = \mathsf{iOTR}_1(\mathsf{crs}, 0; \mathsf{rr}_G), \quad \mathsf{M}_\mathsf{S} = \mathsf{iOTS}(\mathsf{crs}, \mathsf{M}_\mathsf{R}, (m, m'); \mathsf{rr}_E).$$

$\mathtt{Hyb}_1$. By invoking receiver oblivious sampleability of the underlying iOT protocol, we can claim that the aforementioned tuple in $\mathtt{Hyb}_0$ is computationally indistinguishable from the tuple

$$(\boxed{\widetilde{\mathsf{rr}}_G}, \widetilde{\mathsf{iOTS}_{\mathsf{Inv}}}(\mathsf{crs}, 0, \mathsf{M}_\mathsf{S}, \mathsf{td}, \mathsf{rr}_E), \boxed{\widetilde{\mathsf{M}}_\mathsf{R}}, \mathsf{M}_\mathsf{S})$$

where we have

$$\boxed{\widetilde{\mathsf{M}}_\mathsf{R} = \widetilde{\mathsf{iOTR}}(\mathsf{crs}; \widetilde{\mathsf{rr}}_G)}, \quad \mathsf{M}_\mathsf{S} = \mathsf{iOTS}(\mathsf{crs}, \widetilde{\mathsf{M}}_\mathsf{R}, (m, m'); \mathsf{rr}_E)$$

$\mathtt{Hyb}_2$. By invoking sender oblivious sampleability of the underlying iOT protocol, we can claim that the aforementioned tuple in $\mathtt{Hyb}_0$ is computationally indistinguishable from the tuple

$$(\widetilde{rr}_G, \boxed{\widetilde{rr}_E}, \widetilde{\mathsf{M}}_\mathsf{R}, \boxed{\widetilde{\mathsf{M}}_\mathsf{S}})$$

where we have

$$\widetilde{\mathsf{M}}_\mathsf{R} = \widetilde{\mathsf{iOTR}}(\mathsf{crs}; \widetilde{\mathsf{rr}}_G), \quad \boxed{\widetilde{\mathsf{M}}_\mathsf{S} = \widetilde{\mathsf{iOTS}}(\mathsf{crs}, \widetilde{\mathsf{M}}_\mathsf{R}, 0, m'; \widetilde{\mathsf{rr}}_E)}.$$

$\mathtt{Hyb}_3$. Finally, we note that the aforementioned tuple in $\mathtt{Hyb}_2$ is identical to the tuple

$$(\widetilde{\mathsf{rr}}_G, \widetilde{rr}_E, \boxed{\widetilde{\mathsf{pk}}}, \boxed{\widetilde{\mathsf{ct}}})$$

□

Remark. Note that in the aforementioned construction, if the underlying iOT protocol is secure in the plain model (as opposed to in the CRS model), then we immediately obtain a construction of simulatable PKE (as opposed to trapdoor simulatable PKE) in the plain model (as opposed to in the CRS model where the randomness inversion algorithms needs the CRS trapdoor).

7 Instantiations of r-iOT from Concrete Assumptions

In this section we briefly discuss our instantiations of r-iOT from isogeny-based assumptions, CDH and LPN.

7.1 Instantiation from Isogeny-based Assumptions

In this section, we show how to construct a two-message r-iOT protocol secure against malicious adversaries in the CRS model from certain isogeny-based assumptions (notably, CSIDH [CLM+18] or CSI-FiSh [BKV19]). We base our construction on the existence of a secure *(restricted) effective group action* (EGA) equipped with appropriate computational hardness assumptions as described in [ADMP20]. We then rely on known instantiations of such a group action from the aforementioned isogeny-based assumptions. There has been quite a lot of OT constructions [LGd21, BPS22, BMM+23] based on the group actions framework.

In the rest of the section, we rely on the notations and formal definitions of EGA introduced in [ADMP20]. We refer the reader to [ADMP20] and to the full version of our paper for background material on group actions and EGA. We simply state here that our construction of r-iOT from group actions relies on the existence of a weak pseusorandom EGA, which is essentially the analogue of the DDH assumption in the context of group actions. As pointed out in [ADMP20], a weak pseudorandom EGA can be instantiated from isogeny-based assumptions, such as the decisional CSIDH assumption [CLM+18] and counterpart assumption in the setting of CSI-FiSh [BKV19].

The starting point of our construction of r-iOT is the construction of iOT from any (restricted) EGA proposed originally in [ADMP20]. This construction already satisfies indistinguishability-based security against maliciously corrupted sender and the receiver in the static corruption model. The key feature that this construction does not provide is receiver oblivious sampleability.

It turns out that we could argue that this construction satisfies receiver oblivious sampleability in a straightforward manner if we had the ability to sample obliviously from the "set" of a (restricted) EGA by "hashing into" the set. However, this is a well-known open problem in the isogeny literature and is likely to require fundamentally new ideas beyond state-of-the-art techniques for isogeny-based cryptography (see [Pet17, DMPS19, CPV20] for more details).

Our Construction. Our core technical centerpiece is a workaround for this wherein we settle for a weaker notion of *trapdoor oblivious sampleability* for the "set" of a (restricted) EGA. In other words, while it is hard to obliviously sample a "set" element in the plain model, one can obliviously sample a "set" element given a specially designed trapdoor (corresponding to some public CRS). This is the core idea behind our construction of r-iOT from (restricted) EGA. In view of the inherent restrictions outlined earlier, our workaround only allows us to achieve an r-iOT construction in the CRS model (and not in the plain model). Our construction of r-iOT from any weak pseudorandom (restricted) EGA is summarized in Fig. 13. Note that the sender and receiver algorithms remain unchanged from the original iOT construction due to [ADMP20].

Fig. 13. Construction of r-iOT from weak pseudorandom EGA

Private Inputs: S has input bits (m_0, m_1); R has input choice bit b.
Primitives: $(G, X, \star)$ is a weak pseudorandom EGA with initial set element x_0 and $H : X^\ell \to \{0,1\}$ is a pairwise independent hash function.

$\mathsf{Setup}(1^\kappa)$:

- Sample and $g, h \leftarrow G$, and set $x_1 = g \star x_0$ and $x_2 = h \star x_0$.
- Sample $t \leftarrow G$ and compute $y_0 = t \star x_0, y_1 = t \star x_1$ and $y_2 = t \star x_2$.
- Set $\mathsf{crs}_{\mathsf{r\text{-}iOT}} = (x_0, x_1, x_2, y_0, y_1, y_2)$ and $\mathsf{td}_{\mathsf{r\text{-}iOT}} = (g, h, t)$.

$\mathsf{r\text{-}iOTR}_1(\mathsf{crs}_{\mathsf{r\text{-}iOT}}, b)$: Sample $s \leftarrow G$ and output $\mathsf{M_R}$ computed as follows:

$$\mathsf{M_R} = (u, v) = (s \star x_b, s \star y_b), \quad \mathsf{st} = s.$$

$\mathsf{r\text{-}iOTS}(\mathsf{crs}_{\mathsf{r\text{-}iOT}}, \mathsf{M_R}, (m_0, m_1))$: For each $\beta \in \{0,1\}$, uniformly sample

$$\mathbf{r}_\beta = (r_{\beta,1}, \ldots, r_{\beta,\ell}) \leftarrow G^\ell, \quad \mathbf{b}_\beta = (b_{\beta,1}, \ldots, b_{\beta,\ell}) \leftarrow \{0,1\}^\ell.$$

For each $i \in [\ell]$ and for each $\beta \in \{0,1\}$, compute

$$c_{\beta,i} = \begin{cases} r_{\beta,i} \star x_\beta & \text{if } b_{\beta,i} = 0 \\ r_{\beta,i} \star y_\beta & \text{if } b_{\beta,i} = 1 \end{cases}, \quad c'_{\beta,i} = \begin{cases} r_{\beta,i} \star u & \text{if } b_{\beta,i} = 0 \\ r_{\beta,i} \star v & \text{if } b_{\beta,i} = 1. \end{cases}$$

For $\beta \in \{0,1\}$, define the vectors $\mathbf{c}_\beta, \mathbf{c}'_\beta \in X^\ell$ as:

$$\mathbf{c}_\beta := (c_{\beta,1}, \ldots, c_{\beta,\ell}), \quad \mathbf{c}'_\beta := (c'_{\beta,1}, \ldots, c'_{\beta,\ell}).$$

For $\beta \in \{0,1\}$, compute $z_\beta = H(\mathbf{c}'_\beta) \oplus m_\beta$, and output the sender message

$$\mathsf{M_S} = ((\mathbf{c}_0, z_0), (\mathbf{c}_1, z_1)).$$

$\mathsf{r\text{-}iOTR}_2(\mathsf{crs}_{\mathsf{r\text{-}iOT}}, \mathsf{M_S}; \mathsf{st} = s)$: Output m_b computed as follows:

$$m_b = H(s \star c_1, \ldots, s \star c_\ell) \oplus z_b.$$

$\widetilde{\mathsf{r\text{-}iOTR}}(\mathsf{crs}_{\mathsf{r\text{-}iOT}})$: Sample $s \leftarrow G$ and output $\widetilde{\mathsf{M_R}} = (s \star x_2, s \star y_2)$.

$\widetilde{\mathsf{r\text{-}iOTR}}_{\mathsf{Inv}}(\mathsf{crs}_{\mathsf{r\text{-}iOT}}, \mathsf{M_R}, \mathsf{td}_{\mathsf{r\text{-}iOT}}, \mathsf{rr})$: Represent $\mathsf{rr} = (b||s)$ and $\mathsf{td}_{\mathsf{r\text{-}iOT}} = (g, h, t)$. If $b = 0$ then output $\hat{s} = sh^{-1}$, else output $\hat{s} = sgh^{-1}$.

Theorem 11. *Let $(G, X, \star)$ be a weak pseudorandom EGA (as introduced in [ADMP20]). The protocol in Figure 13 is an r-iOT protocol in the CRS model.*

Proof. We first show that our protocol satisfies perfect receiver privacy and then we demonstrate computational sender privacy and perfect oblivious receiver sampleability.

Perfect Receiver Privacy. The receiver's choice bit b is perfectly hidden from the point of view of a (computationally unbounded) malicious receiver, even given $\mathsf{crs}_{\mathsf{r\text{-}iOT}}$ and $\mathsf{M_R} = (u, v)$. We show this by assuming $b = 1$ (the same argument holds when $b = 0$). If receiver computes (u, v) using randomness s when $b = 1$, then the same (u, v) can be shown as a valid receiver message for $b = 0$ using randomness $s' = sg$. In particular, we have $(u, v) = (sg \star x_0, sg \star y_0)$, since $x_1 = g \star x_0$ and $y_1 = t \star x_1 = tg \star x_0 = g \star (t \star x_0) = g \star y_0$.

Computational Sender Privacy. We show the following that there must be some bit $w \in \{0,1\}$ such that

$$\mathsf{r\text{-}iOTS}(\mathsf{crs}_{\mathsf{iOT}}, \mathsf{M_R}, (m_0, m_1)) \overset{c}{\approx} \mathsf{r\text{-}iOTS}(\mathsf{crs}_{\mathsf{iOT}}, \mathsf{M_R}, (m'_0, m'_1)),$$

where $m_{1-w} = m'_{1-w}$ and $m_w \neq m'_w$. We first modify the setup string to $\mathsf{crs}'_{\mathsf{r\text{-}iOT}}$ such that $y_0 = t_0 \star x_0$ and $y_1 = t_1 \star x_1$ where $t_0 \neq t_1$. We argue that $\mathsf{crs}_{\mathsf{r\text{-}iOT}}$ and $\mathsf{crs}'_{\mathsf{r\text{-}iOT}}$ are computationally indistinguishable based on the weak pseudorandomness of EGA.

Next, we argue that under the modified CRS $\mathsf{crs}'_{\mathsf{r\text{-}iOT}}$, there must be some bit $w \in \{0,1\}$ such that $\mathsf{M_S}$ statistically hides m_w irrespective of the manner in which a malicious receiver generates the message $\mathsf{M_R}$. It allows us to move to a hybrid where the sender's message is modified to m'_w. The proof is very similar to the proof of Lemma 4.10 of [ADMP20].

Finally, we change the setup string back to $\mathsf{crs}_{\mathsf{r\text{-}iOT}}$ as in the real protocol. This switch is again computationally indistinguishable based on the weak pseudorandomness of EGA. At this point, the sender's message is distributed as $\mathsf{r\text{-}iOTS}(\mathsf{crs}_{\mathsf{iOT}}, \mathsf{M_R}, (m'_0, m'_1))$, as desired. We refer to the full version of our paper for the formal proof.

- $\mathtt{Hyb}_0$: Sender's OT message is $\mathsf{M_S} = \mathsf{r\text{-}iOTS}(\mathsf{crs}_{\mathsf{iOT}}, \mathsf{M_R}, (m_0, m_1))$ where $\mathsf{crs}_{\mathsf{r\text{-}iOT}}$ is generated as in the "real" protocol, i.e., we have
$$\mathsf{crs}_{\mathsf{r\text{-}iOT}} = (x_0, x_1, x_2, t \star x_0, t \star x_1, t \star x_2).$$
- $\mathtt{Hyb}_1$: Sender's OT message is $\mathsf{M_S} = \mathsf{r\text{-}iOTS}(\mathsf{crs}'_{\mathsf{iOT}}, \mathsf{M_R}, (m_0, m_1))$ where the CRS string is switched from $\mathsf{crs}_{\mathsf{r\text{-}iOT}}$ to $\mathsf{crs}'_{\mathsf{r\text{-}iOT}}$, where
$$\mathsf{crs}'_{\mathsf{r\text{-}iOT}} = (x_0, x_1, x_2, t_0 \star x_0, t_1 \star x_1, t_0 \star x_2)$$
for $t_0 \neq t_1$. The two hybrids are computationally indistinguishable due to the weak pseudorandomness of the EGA.
- $\mathtt{Hyb}_2$: Sender's OT message is $\mathsf{M_S} = \mathsf{r\text{-}iOTS}(\mathsf{crs}'_{\mathsf{iOT}}, \mathsf{M_R}, (m'_0, m'_1))$. We prove that this switch is statistically indistinguishable. Suppose that the malicious receiver sends to the honest sender a message of the form $\mathsf{M_R} = (u, v)$, where u and v could be *arbitrarily* created set elements (this assumption captures the fact that a maliciously corrupt receiver is allowed to behave arbitrarily during protocol execution).
We first argue that given the switch in the manner in which the CRS is generated across the hybrids $\mathtt{Hyb}_1$ and $\mathtt{Hyb}_2$, there *must* be some $w \in \{0,1\}$ such that we have
$$u = s \star x_w, \quad v = s' \star y_w,$$
for $s \neq s'$, irrespective of how the malicious receiver generated the set elements u and v. Indeed, if this was not the case, we would have an expression of the following form (without loss of generality):
$$u = s \star x_0 = \tilde{s} \star x_1, \quad v = s \star y_0 = \tilde{s} \star y_1,$$
implying that there exists some $t \in G$ such that
$$y_0 = t \star x_0, \quad y_1 = t \star x_1,$$
which immediately contradicts the manner in which the CRS was set up in its switched form across the hybrids $\mathtt{Hyb}_1$ and $\mathtt{Hyb}_2$.
We now claim that for this $w \in \{0,1\}$, the corresponding sender message m_w must be statistically hidden from the malicious receiver. Indeed, observe that given the sender message
$$\mathsf{M_S} = ((\mathbf{c}_0, z_0), (\mathbf{c}_1, z_1))$$
the malicious receiver can only infer any information about m_w from the component $(\mathbf{c}_w, z_w)$. We claim that $(\mathbf{c}_w, z_w)$ statistically hides m_w. Recall that
$$\mathbf{c}_w = (c_{w,1}, \ldots, c_{w,\ell}), \quad \mathbf{c}'_w = (c'_{w,1}, \ldots, c'_{w,\ell}), \quad z_w = H(\mathbf{c}'_w) \oplus \mathsf{m}_w,$$
where for each $i \in [\ell]$, we have
$$c_{w,i} = \begin{cases} r_{w,i} \star x_w & \text{if } b_{w,i} = 0 \\ r_{w,i} \star y_w & \text{if } b_{w,i} = 1 \end{cases}, \quad c'_{w,i} = \begin{cases} r_{w,i} \star u & \text{if } b_{w,i} = 0 \\ r_{w,i} \star v & \text{if } b_{w,i} = 1. \end{cases},$$
for some uniformly sampled vector of group elements $(r_{w,1}, \ldots, r_{w,\ell})$ and some uniformly sampled bit-string $(b_{w,1}, \ldots, b_{w,\ell})$.

Now, letting $u = s \star x_w$ and $v = s' \star y_w$ for $s \neq s'$, we get the following for each $i \in [\ell]$ and for each $w \in \{0,1\}$:

$$c'_{w,i} = \begin{cases} s \star c_{w,i} & \text{if } b_{w,i} = 0 \\ s' \star c_{w,i} & \text{if } b_{w,i} = 1 \end{cases}.$$

Since $s \neq s'$ and $(b_{w,1}, \ldots, b_{w,\ell})$ is a uniformly random bit-string (sampled by the honest sender), given $(x_0, x_1, y_0, y_1, u, v, \mathbf{c}_w)$, the vector of set elements $\mathbf{c}'_w$ has exactly ℓ bits of min-entropy, i.e.

$$H_\infty(\mathbf{c}'_w \mid (x_0, x_1, y_0, y_1, u, v, \mathbf{c}_w)) = \ell.$$

Observe that this argument crucially relies on the fact that we use independent pairs of vectors $(\mathbf{r}_w, \mathbf{b}_w)$ for each $w \in \{0,1\}$ as opposed to using the same $(\mathbf{r}, \mathbf{b})$ pair across both choice bits. In particular, given the tuple $(x_0, x_1, y_0, y_1, u, v, \mathbf{c}_w)$, the vector $\mathbf{b}_w$ corresponding to the choice bit w has ℓ bits of entropy; however, this was not the case when the same vector $\mathbf{b}$ was used for both choice bits.

In more detail, if the receiver's choice bit is β, then w must be $(1-\beta)$, and the vector $\mathbf{b}_{1-\beta}$ has ℓ bits of entropy given $(x_0, x_1, y_0, y_1, u, v, \mathbf{c}_{1-\beta})$, which ensures that $m_{1-\beta}$ is statistically hidden. On the other hand, the vector $\mathbf{b}_\beta$ is completely determined given $(x_0, x_1, y_0, y_1, u, v, \mathbf{c}_\beta)$.

Thus, our fix of using independent pairs of vectors $(\mathbf{r}_w, \mathbf{b}_w)$ for each $w \in \{0,1\}$ allows the argument to go through.

Hence, by the properties of the universal hash function H, $z_w = H(\mathbf{c}'_w)$ is statistically indistinguishable from random, and even an unbounded malicious receiver has no information about m_w (except with negligible probability).

– Hyb_3: Sender's OT message is $\mathsf{M_S} = \mathsf{r\text{-}iOTS}(\mathsf{crs}_{\mathsf{r\text{-}iOT}}, \mathsf{M_R}, (m'_0, m'_1))$ where $\mathsf{crs}_{\mathsf{r\text{-}iOT}}$ is switched back to its original form as in the "real" protocol, i.e., we have

$$\mathsf{crs}_{\mathsf{r\text{-}iOT}} = (x_0, x_1, x_2, t \star x_0, t \star x_1, t \star x_2).$$

The two hybrids are again computationally indistinguishable due to the weak pseudorandomness of the EGA.

This concludes the proof of computational sender privacy.

Perfect Receiver Oblivious Sampleability. Finally, we claim that an obliviously sampleable receiver's message is distributed identically to an honestly generated message, even given the sampling randomness. In particular:

– If the receiver's choice bit $b = 0$, then $(u, v) = (s \star x_0, s \star y_0)$ generated using randomness s can be claimed as obliviously sampled using randomness $\hat{s} = sh^{-1}$, since $(u, v) = ((sh^{-1}) \star x_2, (sh^{-1}) \star y_2)$.
– If the receiver's choice bit $b = 1$, then $(u, v) = (s \star x_1, s \star y_1)$ generated using randomness s can be claimed as obliviously sampled using randomness $\hat{s} = sgh^{-1}$, since $(u, v) = ((sgh^{-1}) \star x_2, (sgh^{-1}) \star y_2)$.

This concludes the proof of Theorem 11. □

7.2 Instantiation from CDH or LPN

To instantiate r-iOT from CDH or LPN, we rely on the iOT constructions of [DGH+20]. Specifically, Döttling *et al.* showed that iOT can be constructed from a weaker notion of OT called elementary OT, and they demonstrated instantiations of elementary OT from CDH or LPN assumption. The generic transformation of [DGH+20] is done in two steps: (1) they first show how to build iOT from an intermediate primitive called search OT via parallel repetition (which preserves receiver oblivious sampleability), (2) they show how to construct search OT from elementary OT where the receiver's message in search OT is identical to that of elementary OT.

Since the generic transformation of [DGH+20] does not affect *receiver* oblivious sampleability, it suffices to show that their elementary OT constructions from CDH or LPN *inherently* satisfy the receiver oblivious sampleability property. We refer the reader to [DGH+20] for more details.

CDH based instantiation. Let $(\mathbb{G}, q)$ be a cyclic group of order q for which the CDH assumption holds. We first recall the relevant algorithms from the CDH-based construction of elementary OT in [DGH+20].[11]

- The setup algorithm samples crs as $\mathsf{crs} := X = g^x$ where $x \leftarrow \mathbb{Z}_q$.
- In order to generate its message to the sender, the receiver samples $r \leftarrow \mathbb{Z}_q$ and it sets $\mathsf{M_R} = g^r X^{-b}$ where X is the crs and b is the choice bit.

The receiver oblivious sampleability can be argued as follows:

- We slightly modify the setup algorithm to generate a trapdoor along with a crs. Specifically, we sample crs as $\mathsf{crs} := X = g^x$ where $x \leftarrow \mathbb{Z}_q$, and we set the trapdoor as $\mathsf{td} := x$.
- $\widetilde{\mathsf{r\text{-}iOTR}}(\mathsf{crs}; r)$: Given $\mathsf{crs} = X \in \mathbb{G}$, sample $r \leftarrow \mathbb{Z}_q$ and output $R = g^r$.
- $\widetilde{\mathsf{r\text{-}iOTR}}_{\mathsf{Inv}}(\mathsf{crs}, \mathsf{M_R}, \mathsf{td}, r)$: Given $\mathsf{crs} = X$, $\mathsf{M_R} = g^r X^{-b}$, $\mathsf{td} = x$, the inversion algorithm uses the trapdoor x to find $b \in \{0, 1\}$, which can be computed efficiently because both x and r are available. It then outputs $r - bx \in \mathbb{Z}_q$.

Based on the algorithms above, observe that

$$(g^x, g^r, r) \overset{s}{\approx} (g^x, g^{r-bx}, r - bx),$$

where x and r (in both tuples) are independent, the left tuple is generated obliviously, and the right tuple is generated honestly. It follows that the CDH-based r-iOT construction of [DGH+20] satisfies receiver oblivious sampleability, as required.

LPN based instantiation. We first recall the relevant algorithms from the LPN-based construction of elementary OT in [DGH+20].[12]

- The setup algorithm samples crs as $\mathsf{crs} := (\mathbf{A}, \mathbf{v})$ where $\mathbf{A} \leftarrow \mathbb{Z}_2^{n \times n}$ and $\mathbf{v} \leftarrow \mathbb{Z}_2^n$.
- In order to generate its message to the sender, the receiver samples two vectors $\mathbf{x} \leftarrow \mathcal{B}_\rho^n$ and $\mathbf{e} \leftarrow \mathcal{B}_\rho^n$, where $\mathcal{B}_\rho$ is Bernoulli distribution with error rate $\rho < 1/2$. It sets $\mathsf{M_R} = \mathbf{A}\mathbf{x} + \mathbf{e} + b\mathbf{v}$ where b is the choice bit.[13]

The receiver oblivious sampleability can be argued as follows:

- $\widetilde{\mathsf{r\text{-}iOTR}}(\mathsf{crs}; r)$: Given $\mathsf{crs} = (\mathbf{A}, \mathbf{v}) \in \mathbb{Z}_2^{n+1}$, sample $\mathbf{r} \leftarrow \mathbb{Z}_q$ and output $\mathbf{r}$.
- $\widetilde{\mathsf{r\text{-}iOTR}}_{\mathsf{Inv}}(\mathsf{crs}, \mathsf{M_R}, \mathsf{td}, r)$: Output $\mathsf{M_R}$ (ignore other components).

Observe that by LPN assumption we have

$$((\mathbf{A}, \mathbf{v}), \mathbf{r}, \mathbf{r}) \overset{c}{\approx} ((\mathbf{A}, \mathbf{v}), \mathbf{A}\mathbf{x} + \mathbf{e} + b\mathbf{v}, \mathbf{A}\mathbf{x} + \mathbf{e} + b\mathbf{v}),$$

where the left tuple is generated obliviously and the right tuple is generated honestly. It follows that the LPN-based r-iOT construction of [DGH+20] satisfies receiver oblivious sampleability, as required.

Acknowledgements

The work of Pratik Sarkar is supported by the DARPA SIEVE project and NSF awards 1931714, 1414119.

[11] We omit the sender's message-generation algorithm as it does not affect receiver oblivious sampleability.
[12] As before, we omit the sender's message-generation algorithm as it does not affect receiver oblivious sampleability.
[13] Notice that [DGH+20] uses a variant of LPN for which the secret comes from the noise distribution, which is known to be equivalent to the standard LPN [ACPS09].

References

AAAS+19. Gorjan Alagic, Gorjan Alagic, Jacob Alperin-Sheriff, Daniel Apon, David Cooper, Quynh Dang, Yi-Kai Liu, Carl Miller, Dustin Moody, Rene Peralta, et al. *Status report on the first round of the NIST post-quantum cryptography standardization process*. US Department of Commerce, National Institute of Standards and Technology . . . , 2019.

AASA+20. Gorjan Alagic, Jacob Alperin-Sheriff, Daniel Apon, David Cooper, Quynh Dang, John Kelsey, Yi-Kai Liu, Carl Miller, Dustin Moody, Rene Peralta, et al. Status report on the second round of the nist post-quantum cryptography standardization process. *US Department of Commerce, NIST*, 2020.

ABP17. Michel Abdalla, Fabrice Benhamouda, and David Pointcheval. Removing erasures with explainable hash proof systems. In Serge Fehr, editor, *PKC 2017, Part I*, volume 10174 of *LNCS*, pages 151–174. Springer, Heidelberg, March 2017.

ACPS09. Benny Applebaum, David Cash, Chris Peikert, and Amit Sahai. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In Shai Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 595–618. Springer, Heidelberg, August 2009.

ADMP20. Navid Alamati, Luca De Feo, Hart Montgomery, and Sikhar Patranabis. Cryptographic group actions and applications. In *ASIACRYPT 2020, Part II*, LNCS, pages 411–439. Springer, Heidelberg, December 2020.

BBD+20. Zvika Brakerski, Pedro Branco, Nico Döttling, Sanjam Garg, and Giulio Malavolta. Constant ciphertext-rate non-committing encryption from standard assumptions. In *TCC 2020, Part I*, LNCS, pages 58–87. Springer, Heidelberg, March 2020.

BD18. Zvika Brakerski and Nico Döttling. Two-message statistically sender-private OT from LWE. In Amos Beimel and Stefan Dziembowski, editors, *TCC 2018, Part II*, volume 11240 of *LNCS*, pages 370–390. Springer, Heidelberg, November 2018.

BHR12. Mihir Bellare, Viet Tung Hoang, and Phillip Rogaway. Foundations of garbled circuits. In Ting Yu, George Danezis, and Virgil D. Gligor, editors, *ACM CCS 2012*, pages 784–796. ACM Press, October 2012.

BKV19. Ward Beullens, Thorsten Kleinjung, and Frederik Vercauteren. CSI-FiSh: Efficient isogeny based signatures through class group computations. In *ASIACRYPT 2019, Part I*, LNCS, pages 227–247. Springer, Heidelberg, December 2019.

BL18. Fabrice Benhamouda and Huijia Lin. k-round multiparty computation from k-round oblivious transfer via garbled interactive circuits. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part II*, volume 10821 of *LNCS*, pages 500–532. Springer, Heidelberg, April / May 2018.

BLPV18. Fabrice Benhamouda, Huijia Lin, Antigoni Polychroniadou, and Muthuramakrishnan Venkitasubramaniam. Two-round adaptively secure multiparty computation from standard assumptions. In Amos Beimel and Stefan Dziembowski, editors, *TCC 2018, Part I*, volume 11239 of *LNCS*, pages 175–205. Springer, Heidelberg, November 2018.

BMM+23. Saikrishna Badrinarayanan, Daniel Masny, Pratyay Mukherjee, Sikhar Patranabis, Srinivasan Raghuraman, and Pratik Sarkar. Round-optimal oblivious transfer and MPC from computational CSIDH. In *Public-Key Cryptography - PKC 2023 - 26th IACR International Conference on Practice and Theory of Public-Key Cryptography, Atlanta, GA, USA, May 7-10, 2023, Proceedings, Part I*, volume 13940 of *Lecture Notes in Computer Science*, pages 376–405. Springer, 2023.

BMR90. Donald Beaver, Silvio Micali, and Phillip Rogaway. The round complexity of secure protocols (extended abstract). In *22nd ACM STOC*, pages 503–513. ACM Press, May 1990.

BPS22. Saikrishna Badrinarayanan, Sikhar Patranabis, and Pratik Sarkar. Statistical security in two-party computation revisited. In Eike Kiltz and Vinod Vaikuntanathan, editors, *Theory of Cryptography - 20th International Conference, TCC 2022, Chicago, IL, USA, November 7-10, 2022, Proceedings, Part II*, volume 13748 of *Lecture Notes in Computer Science*, pages 181–210. Springer, 2022.

Can01. Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *42nd FOCS*, pages 136–145. IEEE Computer Society Press, October 2001.

CCD+20. Megan Chen, Ran Cohen, Jack Doerner, Yashvanth Kondi, Eysa Lee, Schuyler Rosefield, and Abhi Shelat. Multiparty generation of an RSA modulus. In *CRYPTO 2020*, pages 64–93, 2020.

CDD+04. Ran Canetti, Ivan Damgård, Stefan Dziembowski, Yuval Ishai, and Tal Malkin. Adaptive versus non-adaptive security of multi-party protocols. *Journal of Cryptology*, 17(3):153–207, June 2004.

CDMW09. Seung Geol Choi, Dana Dachman-Soled, Tal Malkin, and Hoeteck Wee. Improved non-committing encryption with applications to adaptively secure protocols. In Mitsuru Matsui, editor, *ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 287–302. Springer, Heidelberg, December 2009.

CFGN96. Ran Canetti, Uriel Feige, Oded Goldreich, and Moni Naor. Adaptively secure multi-party computation. In *28th ACM STOC*, pages 639–648. ACM Press, May 1996.

CGP15. Ran Canetti, Shafi Goldwasser, and Oxana Poburinnaya. Adaptively secure two-party computation from indistinguishability obfuscation. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *TCC 2015, Part II*, volume 9015 of *LNCS*, pages 557–585. Springer, Heidelberg, March 2015.

CGPS21. Suvradip Chakraborty, Chaya Ganesh, Mahak Pancholi, and Pratik Sarkar. Reverse firewalls for adaptively secure MPC without setup. LNCS, pages 335–364. Springer, Heidelberg, 2021.

CJL+16. Lily Chen, Stephen Jordan, Yi-Kai Liu, Dustin Moody, Rene Peralta, Ray Perlner, and Daniel Smith-Tone. *Report on post-quantum cryptography*, volume 12. US Department of Commerce, National Institute of Standards and Technology, 2016.

CKWZ13. Seung Geol Choi, Jonathan Katz, Hoeteck Wee, and Hong-Sheng Zhou. Efficient, adaptively secure, and composable oblivious transfer with a single, global CRS. In Kaoru Kurosawa and Goichiro Hanaoka, editors, *PKC 2013*, volume 7778 of *LNCS*, pages 73–88. Springer, Heidelberg, February / March 2013.

CLM+18. Wouter Castryck, Tanja Lange, Chloe Martindale, Lorenz Panny, and Joost Renes. CSIDH: An efficient post-quantum commutative group action. In Thomas Peyrin and Steven Galbraith, editors, *ASIACRYPT 2018, Part III*, volume 11274 of *LNCS*, pages 395–427. Springer, Heidelberg, December 2018.

CLOS02. Ran Canetti, Yehuda Lindell, Rafail Ostrovsky, and Amit Sahai. Universally composable two-party and multi-party secure computation. In *34th ACM STOC*, pages 494–503. ACM Press, May 2002.

CPR17. Ran Canetti, Oxana Poburinnaya, and Mariana Raykova. Optimal-rate non-committing encryption. In Tsuyoshi Takagi and Thomas Peyrin, editors, *ASIACRYPT 2017, Part III*, volume 10626 of *LNCS*, pages 212–241. Springer, Heidelberg, December 2017.

CPV17a. Ran Canetti, Oxana Poburinnaya, and Muthuramakrishnan Venkitasubramaniam. Better two-round adaptive multi-party computation. In Serge Fehr, editor, *PKC 2017, Part II*, volume 10175 of *LNCS*, pages 396–427. Springer, Heidelberg, March 2017.

CPV17b. Ran Canetti, Oxana Poburinnaya, and Muthuramakrishnan Venkitasubramaniam. Equivocating yao: constant-round adaptively secure multiparty computation in the plain model. In Hamed Hatami, Pierre McKenzie, and Valerie King, editors, *49th ACM STOC*, pages 497–509. ACM Press, June 2017.

CPV20. Wouter Castryck, Lorenz Panny, and Frederik Vercauteren. Rational isogenies from irrational endomorphisms. In Vincent Rijmen and Yuval Ishai, editors, *EUROCRYPT 2020, Part II*, LNCS, pages 523–548. Springer, Heidelberg, May 2020.

CR03. Ran Canetti and Tal Rabin. Universal composition with joint state. In Dan Boneh, editor, *CRYPTO 2003*, volume 2729 of *LNCS*, pages 265–281. Springer, Heidelberg, August 2003.

CsW19. Ran Cohen, abhi shelat, and Daniel Wichs. Adaptively secure MPC with sublinear communication complexity. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part II*, volume 11693 of *LNCS*, pages 30–60. Springer, Heidelberg, August 2019.

CSW20. Ran Canetti, Pratik Sarkar, and Xiao Wang. Efficient and round-optimal oblivious transfer and commitment with adaptive security. LNCS, pages 277–308. Springer, Heidelberg, December 2020.

CSW22. Ran Canetti, Pratik Sarkar, and Xiao Wang. Triply adaptive UC NIZK. In *Advances in Cryptology - ASIACRYPT 2022 - 28th International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, December 5-9, 2022, Proceedings, Part II*, volume 13792 of *Lecture Notes in Computer Science*, pages 466–495. Springer, 2022.

DG19. Luca De Feo and Steven D. Galbraith. SeaSign: Compact isogeny signatures from class group actions. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part III*, volume 11478 of *LNCS*, pages 759–789. Springer, Heidelberg, May 2019.

DGH+20. Nico Döttling, Sanjam Garg, Mohammad Hajiabadi, Daniel Masny, and Daniel Wichs. Two-round oblivious transfer from CDH or LPN. In Vincent Rijmen and Yuval Ishai, editors, *EUROCRYPT 2020, Part II*, LNCS, pages 768–797. Springer, Heidelberg, May 2020.

DMPS19. Luca De Feo, Simon Masson, Christophe Petit, and Antonio Sanso. Verifiable delay functions from supersingular isogenies and pairings. In *ASIACRYPT 2019, Part I*, LNCS, pages 248–277. Springer, Heidelberg, December 2019.

EGL82. Shimon Even, Oded Goldreich, and Abraham Lempel. A randomized protocol for signing contracts. In David Chaum, Ronald L. Rivest, and Alan T. Sherman, editors, *CRYPTO'82*, pages 205–210. Plenum Press, New York, USA, 1982.

FMV19. Daniele Friolo, Daniel Masny, and Daniele Venturi. A black-box construction of fully-simulatable, round-optimal oblivious transfer from strongly uniform key agreement. In *TCC 2019, Part I*, LNCS, pages 111–130. Springer, Heidelberg, March 2019.

GGHR14. Sanjam Garg, Craig Gentry, Shai Halevi, and Mariana Raykova. Two-round secure MPC from indistinguishability obfuscation. In Yehuda Lindell, editor, *TCC 2014*, volume 8349 of *LNCS*, pages 74–94. Springer, Heidelberg, February 2014.

GMW87. Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or A completeness theorem for protocols with honest majority. In Alfred Aho, editor, *19th ACM STOC*, pages 218–229. ACM Press, May 1987.

GOS12. Jens Groth, Rafail Ostrovsky, and Amit Sahai. New techniques for noninteractive zero-knowledge. *J. ACM*, 59(3):11:1–11:35, 2012.

GP15. Sanjam Garg and Antigoni Polychroniadou. Two-round adaptively secure MPC from indistinguishability obfuscation. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *TCC 2015, Part II*, volume 9015 of *LNCS*, pages 614–637. Springer, Heidelberg, March 2015.

GPV08. Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In Richard E. Ladner and Cynthia Dwork, editors, *40th ACM STOC*, pages 197–206. ACM Press, May 2008.

GS12. Sanjam Garg and Amit Sahai. Adaptively secure multi-party computation with dishonest majority. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 105–123. Springer, Heidelberg, August 2012.

GS18. Sanjam Garg and Akshayaram Srinivasan. Two-round multiparty secure computation from minimal assumptions. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part II*, volume 10821 of *LNCS*, pages 468–499. Springer, Heidelberg, April / May 2018.

GWZ09. Juan A. Garay, Daniel Wichs, and Hong-Sheng Zhou. Somewhat non-committing encryption and efficient adaptively secure oblivious transfer. In Shai Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 505–523. Springer, Heidelberg, August 2009.

HK12. Shai Halevi and Yael Tauman Kalai. Smooth projective hashing and two-message oblivious transfer. *Journal of Cryptology*, 25(1):158–193, January 2012.

HV15. Carmit Hazay and Muthuramakrishnan Venkitasubramaniam. On black-box complexity of universally composable security in the CRS model. In Tetsu Iwata and Jung Hee Cheon, editors, *ASIACRYPT 2015, Part II*, volume 9453 of *LNCS*, pages 183–209. Springer, Heidelberg, November / December 2015.

HV16. Carmit Hazay and Muthuramakrishnan Venkitasubramaniam. On the power of secure two-party computation. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part II*, volume 9815 of *LNCS*, pages 397–429. Springer, Heidelberg, August 2016.

IKN+17. Mihaela Ion, Ben Kreuter, Erhan Nergiz, Sarvar Patel, Shobhit Saxena, Karn Seth, David Shanahan, and Moti Yung. Private intersection-sum protocol with applications to attributing aggregate ad conversions. Cryptology ePrint Archive, Report 2017/738, 2017. https://eprint.iacr.org/2017/738.

JKKR17. Abhishek Jain, Yael Tauman Kalai, Dakshita Khurana, and Ron Rothblum. Distinguisher-dependent simulation in two rounds and its applications. In Jonathan Katz and Hovav Shacham, editors, *CRYPTO 2017, Part II*, volume 10402 of *LNCS*, pages 158–189. Springer, Heidelberg, August 2017.

LGd21. Yi-Fu Lai, Steven D. Galbraith, and Cyprien de Saint Guilhem. Compact, efficient and UC-secure isogeny-based oblivious transfer. LNCS, pages 213–241. Springer, Heidelberg, 2021.

LGdSG21. Yi-Fu Lai, Steven D. Galbraith, and Cyprien Delpech de Saint Guilhem. Compact, efficient and uc-secure isogeny-based oblivious transfer. In Anne Canteaut and François-Xavier Standaert, editors, *EUROCRYPT 2021*, pages 213–241, 2021.

LJA+18. Andrei Lapets, Frederick Jansen, Kinan Dak Albab, Rawane Issa, Lucy Qin, Mayank Varia, and Azer Bestavros. Accessible privacy-preserving web-based data analysis for assessing and addressing economic inequalities. In *Proceedings of the 1st ACM SIGCAS Conference on Computing and Sustainable Societies*, pages 1–5, 2018.

LLW20. Huijia Lin, Tianren Liu, and Hoeteck Wee. Information-theoretic 2-round MPC without round collapsing: Adaptive security, and more. In Rafael Pass and Krzysztof Pietrzak, editors, *Theory of Cryptography - 18th International Conference, TCC 2020, Durham, NC, USA, November 16-19, 2020, Proceedings, Part II*, volume 12551 of *Lecture Notes in Computer Science*, pages 502–531. Springer, 2020.

LP09. Yehuda Lindell and Benny Pinkas. A proof of security of Yao's protocol for two-party computation. *Journal of Cryptology*, 22(2):161–188, April 2009.

MR18. Michael Meyer and Steffen Reith. A faster way to the CSIDH. In Debrup Chakraborty and Tetsu Iwata, editors, *INDOCRYPT 2018*, volume 11356 of *LNCS*, pages 137–152. Springer, Heidelberg, December 2018.

MW16. Pratyay Mukherjee and Daniel Wichs. Two round multiparty computation via multi-key FHE. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part II*, volume 9666 of *LNCS*, pages 735–763. Springer, Heidelberg, May 2016.

Nao91. Moni Naor. Bit commitment using pseudorandomness. *Journal of Cryptology*, 4(2):151–158, January 1991.

NOTT20. Kohei Nakagawa, Hiroshi Onuki, Atsushi Takayasu, and Tsuyoshi Takagi. l_1-norm ball for CSIDH: optimal strategy for choosing the secret key space. Cryptology ePrint Archive, Report 2020/181, 2020.

NP01. Moni Naor and Benny Pinkas. Efficient oblivious transfer protocols. In S. Rao Kosaraju, editor, *12th SODA*, pages 448–457. ACM-SIAM, January 2001.

Pet17. Christophe Petit. Faster algorithms for isogeny problems using torsion point images. In Tsuyoshi Takagi and Thomas Peyrin, editors, *ASIACRYPT 2017, Part II*, volume 10625 of *LNCS*, pages 330–353. Springer, Heidelberg, December 2017.

PVW08. Chris Peikert, Vinod Vaikuntanathan, and Brent Waters. A framework for efficient and composable oblivious transfer. In David Wagner, editor, *CRYPTO 2008*, volume 5157 of *LNCS*, pages 554–571. Springer, Heidelberg, August 2008.

Rab05. Michael O. Rabin. How to exchange secrets with oblivious transfer. Cryptology ePrint Archive, Report 2005/187, 2005. https://eprint.iacr.org/2005/187.

Sto12. Anton Stolbunov. Cryptographic schemes based on isogenies, 2012.

unb. Unbound security. https://www.unboundtech.com.

Yao86. Andrew Chi-Chih Yao. How to generate and exchange secrets (extended abstract). In *27th FOCS*, pages 162–167. IEEE Computer Society Press, October 1986.

YKT19. Yusuke Yoshida, Fuyuki Kitagawa, and Keisuke Tanaka. Non-committing encryption with quasi-optimal ciphertext-rate based on the DDH problem. In *ASIACRYPT 2019*, pages 128–158, 2019.

A Security Model for Adaptively Secure MPC

In this section we recall the formal definition of adaptively secure MPC protocols in the *stand-alone* setting as in [CFGN96, GS12].

Some MPC Notations. Let n denote the number of parties involved in the protocol. We assume that n is fixed. A multi-party protocol problem is cast by specifying a n-ary *functionality*, denoted by $f : (\{0,1\}^*)^n \to (\{0,1\}^*)^n$, where $f = (f_1, \cdots, f_n)$. For a input vector $\vec{x} = \{x_1, \cdots, x_n\}$ the output is a tuple of random variables denoted by $(f_1(\vec{x}), \cdots, f_n(\vec{x}))$. The i^{th} party P_i initially holds the input x_i and obtains $f_i(\vec{x})$. We also assume that all the parties hold input of equal length, i.e., $|x_i| = |x_j|$ for all $i, j \in [n]$.

Adversarial behavior. For the analysis of our protocols we consider the setting of *malicious* adversaries that can *adaptively* corrupt parties throughout the protocol execution depending on its view during the execution. We consider the definition of security in terms of the real-world and ideal-world simulation paradigm.

Real World. In the real world, the MPC protocol Π is executed by the interaction of n parties $\{P_1, \cdots P_n\}$. Each party P_i has input $x_i \in \{0,1\}^*$, random input $r_i \in \{0,1\}^*$, and the security parameter κ. Let $\mathcal{C} \subset [n]$ and $\mathcal{H} = [n] \setminus \mathcal{C}$ denote the indices of the malicious corrupted parties and honest parties in Π. Consequently, let us denote by $P_{\mathcal{C}}$ and $P_{\mathcal{H}}$ the set of maliciously corrupted and honest parties respectively. We assume that all communication is done via a *broadcast* channel. We consider the synchronous, with rushing model of computation.

At the onset of the computation the adversary $\mathcal{A}$ receives some auxiliary input denoted by z. The computation proceeds in *rounds*, with each round consisting of several *mini-rounds*. Each mini-round starts by allowing $\mathcal{A}$ to *adaptively* corrupt parties one by one. Once a party is corrupted the party's input and random input become known to $\mathcal{A}$. Next, $\mathcal{A}$ activates an uncorrupted party P_i, which has not been activated so far in this round. Upon activation, P_i receives the messages sent to it in the previous round, generates the message for this round, and the next mini-round begins. $\mathcal{A}$ also gets to learn the messages sent by P_i. Once all the uncorrupted parties are activated, $\mathcal{A}$ sends the messages on behalf of the corrupt parties that were not yet activated in this round, and the next round begins. Finally, at the end of the computation (after some pre-specified number of rounds) the parties locally generate their outputs. Each uncorrupted/honest parties output what is specified as in the protocol. The corrupt parties may output an arbitrary probabilistic polynomial-time function of the view of $\mathcal{A}$.

The overall output of the real-world experiment consists of the output of all parties at the end of the protocol, and the real world adversary view is denoted by $\mathsf{Real}_{\Pi,(\mathcal{C},\mathcal{A})}(\kappa, \vec{x}, \vec{r}, z)$. Let $\mathsf{Real}_{\Pi,(\mathcal{C},\mathcal{A})}(\kappa, \vec{x}, z)$ be the distribution of $\mathsf{Real}_{\Pi,(\mathcal{C},\mathcal{A})}(\kappa, \vec{x}, \vec{r}, z)$ when $\vec{r}$ is chosen uniformly at random. Let $\mathsf{Real}_{\Pi,(\mathcal{C},\mathcal{A})}$ denote the distribution ensemble $\{\mathsf{Real}_{\Pi,(\mathcal{C},\mathcal{A})}(\kappa, \vec{x}, z)\}_{\kappa \in \mathbb{N}, \vec{x} \in (\{0,1\}^*)^n, z \in \{0,1\}^*}$.

Fig. 14. The Ideal World Execution of an Adaptive MPC Protocol

- **Input:** Let $\vec{x} = \{x_1, \cdots, x_n\}$ denote the initial inputs for the parties. As in the real-world, the adversary/simulator $\mathcal{S}$ additionally has an auxiliary input denoted by z.
- **First corruption stage:** $\mathcal{S}$ proceeds in iterations, where in each iteration $\mathcal{S}$ may decide to corrupt some party based on the random input of $\mathcal{S}$. Once a party is corrupted its input becomes known to $\mathcal{S}$.
- **Send inputs to T:** Each honest party P_i sends its input x_i to T. The corrupted parties may either abort (send special symbol $\perp$), send correct input x_i, or send some other input x_i' ($|x_i| = |x_i'|$) to T. Let $\vec{x}' = \{x_1', \cdots, x_n'\}$ denote the input vector received by T.
- **Early abort option:** If T receives the special symbol $\perp$, it sends abort to the honest parties and the ideal execution terminates. Otherwise,
- **T sends output to adversary:** T chooses r_f uniformly at random, computes $f(\vec{x}', r_f)$ and sends it to the $\mathcal{S}$ first.
- **Adversary $\mathcal{S}$ instructs T to continue or halt:** $\mathcal{S}$ either sends continue or abort to T. In case of continue, T sends $f(\vec{x}', r_f)$ to the honest parties. Otherwise, if $\mathcal{S}$ sends abort, T sends abort to the honest parties.
- **Second corruption stage:** Upon learning the corrupted parties' outputs of the computation, $\mathcal{S}$ proceeds in another sequence of iterations, where in each iteration $\mathcal{S}$ may decide to corrupt some additional parties, based on the information gathered so far. Upon corruption, $\mathcal{S}$ learns the sees the corrupted party's input and output.
- **Output stage:** Each honest party output the output received from T, while the maliciously corrupted parties $P_{\mathcal{C}}$ output any probabilistic polynomial-time computable function of their input, the auxiliary input z, and the output received from T.
- **Post-execution corruption:** Once the outputs are generated, $\mathcal{S}$ may at any point in the protocol may decide to adaptively proceed in another sequence of iterations, where in each iteration $\mathcal{S}$ may decide to corrupt some additional party, based on the information gathered so far.

Ideal World. In the ideal world we assume the existence of an incorruptible trusted third party (TTP), with whom all the parties interact. Each party P_i gets input $x_i \in \{0,1\}^*$ and wish to evaluate $f_1(\vec{x}, r_f), \cdots, f_n(\vec{x}, r_f)$, where $r_f \leftarrow \{0,1\}^s$, and s is a value determined by the security parameter, and P_i learns $f_i(\vec{x}, r_f)$. The ideal world computation in the presence of an adaptive ideal world adversary $\mathcal{S}$ (with random input r) and the TTP T proceeds as in Figure 14.

The overall output of the ideal-world experiment consists of the output of all parties at the end of the protocol, and the ideal world adversary view is denoted by $\mathsf{Ideal}_{f,(\mathcal{C},\mathcal{S})}(\kappa, \vec{x}, \vec{r}, z)$, where $\vec{r} = (r, r_f)$. Let $\mathsf{Ideal}_{f,(\mathcal{C},\mathcal{S})}(\kappa, \vec{x}, z)$ be the denote the distribution of $\mathsf{Ideal}_{f,(\mathcal{C},\mathcal{S})}(\kappa, \vec{x}, \vec{r}, z)$ where $\vec{r}$ is chosen uniformly at random. Also, let $\mathsf{Ideal}_{f,(\mathcal{C},\mathcal{S})}$ denote the distribution ensemble

$$\{\mathsf{Ideal}_{f,(\mathcal{C},\mathcal{S})}(\kappa, \vec{x}, z)\}_{\kappa \in \mathbb{N}, \vec{x} \in (\{0,1\}^*)^n, z \in \{0,1\}^*}.$$

Now that the ideal and real world executions are defined, we put forward the notion of security for an adaptively secure multi-party protocol Π. Informally, we require that executing a protocol Π in the real world emulates the ideal process for evaluating f.

Definition 17 (Adaptive MPC Security). *Let f be any adaptive well-formed n-ary function, Π be a be a protocol for n parties. We say that Π adaptively securely evaluates f if for every real world adversary $\mathcal{A}$ there exists an ideal world adversary $\mathcal{S}$, such that $\mathsf{Real}_{\Pi,(\mathcal{C},\mathcal{A})} \approx_c \mathsf{Ideal}_{f,(\mathcal{C},\mathcal{S})}$.*