IACR Communications in Cryptology
ISSN 3006-5496, Vol. 1, No. 4, 43 pages.

https://doi.org/10.62056/abe0wa3y6

# Round-Optimal Compiler for Semi-Honest to Malicious Oblivious Transfer via CIH

Varun Madathil[1], Alessandra Scafuro[2] and Tanner Verber[a, 2]

[1] Yale University, Computer Science, New Haven, USA
[2] North Carolina State University, Computer Science, Raleigh, USA

**Abstract.**
A central question in the theory of cryptography is whether we can build protocols that achieve stronger security guarantees, e.g., security against malicious adversaries, by combining building blocks that achieve much weaker security guarantees, e.g., security only against semi-honest adversaries; and with the *minimal* number of rounds. An additional focus is whether these building blocks can be used only as a black-box. Since Oblivious Transfer (OT) is the necessary and sufficient building block to securely realize any two-party (and multi-party) functionality, theoreticians often focus on proving whether maliciously secure OT can be built from a weaker notion of OT.
There is a rich body of literature that provides (black-box) compilers that build malicious OT from OTs that achieve weaker security such as semi-malicious OT and defensibly secure OT, within the minimal number of rounds. However, no round-optimal compiler exists that builds malicious OT from the weakest notion of semi-honest OT, in the plain model.
Correlation intractable hash (CIH) functions are special hash functions whose properties allow instantiating the celebrated Fiat-Shamir transform, and hence reduce the round complexity of public-coin proof systems.
In this work, we devise the first round-optimal compiler from semi-honest OT to malicious OT, by a novel application of CIH for collapsing rounds in the plain model. We provide the following contributions. First, we provide a new CIH-based round-collapsing construction for general cut-and-choose. This gadget can be used generally to prove the correctness of the evaluation of a function. Then, we use our gadget to build the first round-optimal compiler from semi-honest OT to malicious OT.
Our compiler uses the semi-honest OT protocol and the other building blocks in a black-box manner. However, for technical reasons, the underlying CIH construction requires the upper bound of the circuit size of the semi-honest OT protocol used. The need for this upper-bound makes our protocol not fully black-box, hence is incomparable with existing, fully black-box, compilers.

**Keywords:** Foundations · Round-Optimal · Compiler

## 1 Introduction

**Round-Collapsing Techniques for Proof Systems with the Random Oracle.** The Fiat-Shamir transform [FS86] is a seminal technique that allows to collapse a three-round *public-coin* proof system into one single round. Recall that a proof system is an interactive protocol between a prover $P$ and a verifier $V$, with common input a theorem $x \in L$ (for some NP-language $L$), and the prover aims to convince the verifier that the theorem is

E-mail: varun.madathil@yale.edu (Varun Madathil), ascafur@ncsu.edu (Alessandra Scafuro), tverber@ncsu.edu (Tanner Verber)

[a]Alessandra Scafuro and Tanner Verber are supported by a research grant from Horizen Labs

Received: 2024-10-08 Accepted: 2024-12-03

true, without revealing any information about the witness. Public coin means that the messages of the verifier of the proof system are simply random strings. In the Fiat-Shamir transform, every message of the verifier is replaced with the evaluation of a hash function, on input the transcript of the proof system so far. These hash evaluations can be performed by the prover locally, as a result, the prover can compute the entire transcript on their own, and the proof consists of a single string. To prove that the collapsed version of the proof system still retains its security properties (i.e., soundness and zero-knowledge) the hash function, originally, was modeled as a Random Oracle (RO) [BR93]. Since the Random Oracle is an ideal object that cannot be instantiated in practice, any concrete instantiation of this transform written in the RO model will not be provably secure, but heuristically secure. From a theoretical perspective, this was unsatisfactory and a rich line of work studied under which conditions the Fiat-Shamir round-collapsing technique could be instantiated with a cryptographic primitive in the standard model (i.e., without RO). While some works [GK03] show that this round-collapsing technique is impossible to securely instantiate with any concrete function if the interactive proof system is an argument, other works [BLV06, CCR16, KRR17, CCRR18] showed that, if the underlying hash function satisfies the additional property of conditional entropy, then the Fiat-Shamir transform can be securely instantiated. Canetti et al.[CCRR18] and Kalai et al.[KRR17] built a new primitive called correlation intractable hash functions (CIH) and show that it satisfies conditional entropy. Such constructions, however, were based on strong computational assumptions, such as subexponentially secure indistinguishability obfuscation, subexponentially secure puncturable PRFs, and input-hiding obfuscation for multi-bit point functions.

**Round-Collapsing Techniques for Proof Systems with the CIH in the CRS model.** Recently, however, there has been an exciting sequence of works building CIH from much weaker and standard assumptions, such as LWE assumption [PS19, CCH$^+$19, LV22], sub-exponential DDH [JJ21, CGJ$^+$23], and trapdoor hash functions [BKM20] (which can be built from DDH, QR, DCR, and LWE). However, all existing frameworks are designed for proof systems only, and aim at achieving the specific task of non-interactive zero-knowledge.

**Round-Optimal Compilers for Secure Computation in the Plain Model.** A central question in the theory of cryptography is to establish the *minimal number* of rounds required to realize *any* functionality securely. A further focus of such question is what is the minimal assumption that can be used to realize such a functionality, in a minimal number of rounds, and in the plain model (that is, a model that does not use setup assumptions, such as the CRS or RO). Since Oblivious Transfer[1] (OT) has been proven to be sufficient to implement any functionality, often the question is narrowed down to building a protocol that securely realizes the OT functionality in the presence of malicious adversaries (under the simulation-based paradigm), from the minimal assumption of the existence of an OT protocol that is secure only in the presence of *semi-honest* adversaries. Often, such protocols are called *compilers*, as they compile a semi-honest secure protocol into a maliciously secure protocol.

In the plain model, Katz and Ostrovsky in [KO04] proved that four rounds are necessary and sufficient to build a maliciously secure OT (under the simulation-based paradigm). Their transformation builds a malicious secure OT from the very specific assumption of certified trapdoor permutations. Since [KO04], a sequence of work showed various round-optimal compilers that build maliciously secure OT from weaker notions of OT [FMV19, MOSV22]. Furthermore, all such compilers focused on using the underlying building blocks in a *black-box* manner[2]. Recall that a protocol uses a primitive in a

[1]Oblivious Transfer [Rab05] is a two-party functionality with a sender $S$, possessing two inputs $s_0, s_1$ and a Receiver $R$ possessing a bit $b$, and the goal is for $R$ to learn $s_b$. The security requirements are $R$ learns nothing about $s_{\bar{b}}$ and $S$ learns nothing about $b$.

[2]We note that to our knowledge there are no round-optimal non-black-box compilers of semi-honest

black-box manner if it only uses the input/output interface of the primitives. In contrast, non-black-box usage is when the protocol needs to use the knowledge of the circuit of the primitives (e.g., the GMW compiler [GMW87] needs the circuit of the commitments and OT protocol to compute a zero-knowledge proof), or it uses the specific structure of the assumptions (e.g., being a group element, etc). Black-box protocols are often more efficient than non-black-box protocol [IKOS07] and from a theoretical perspective constitute a more general result.

Currently, all existing round-optimal black-box compilers build maliciously secure OT from notions of OT that are stronger than basic semi-honest OT. The compiler of Friolo et al. [FMV19] builds on top of an OT protocol that is strong-uniform[3], while Madathil et al. [MOSV22] builds on top of a defensibly-secure OT.

The question of whether we can build a maliciously secure OT protocol from black-box use of semi-honest secure OT, with the minimal number of rounds, in the plain model is still open.

**This work: Round-Optimal Semi-honest to Malicious OT Compiler using CIH in the Plain model.** The main bottleneck in starting from a semi-honest protocol and upgrading to malicious security in just four rounds is that the number of rounds are insufficient to both enforce and check that a malicious receiver is honestly running the protocol, using the primitives in a black-box manner (i.e., without adding zero-knowledge proofs, which require the circuit of the primitives).

This is the reason why all known compilers start with an OT protocol that has a bit more structure (e.g., strong uniformity) or a bit more security guarantees (e.g., defensibility) than the bare-bone semi-honest OT.

The main challenge in starting with bare semi-honest security is that this notion only gives guarantees when the adversary is forced to use proper inputs *and* proper randomness. It is well known that through coin-flipping we can force a malicious adversary to use input and randomness that he cannot influence, and with cut-and-choose we can check that the adversary is using the randomness that was established in the coin-flipping. The problem is that to achieve round optimality we have only four rounds, which means that coin-flipping *and* cut-and-choose for the receiver must be concluded by the third round.

Motivated by this open problem, and inspired by the recent development of CIH based round-collapsing techniques for proof systems under standard assumptions [CCH+19], in this work we explore how we can extend and apply the techniques from CIH-based round-collapsing techniques, to general compilers, and in the plain model.

## 1.1 Our Contribution

In this paper we provide the first round-optimal compiler from semi-honest OT to malicious OT, using CIH. Our compiler uses all the cryptographic primitives in a black-box manner, which means that when running the protocol, the sender and the receiver treat all primitives as black boxes. However, for technical reasons [CGH04], the underlying CIH construction requires the upper bound of the circuit size of the semi-honest OT protocol used. This is because the CIH we use from [CCH+19][4] is built on sparse relations and the code of the functions involved is required (in the proof) to define the relation (more details will be provided in Sec. 4). The need for this upper-bound makes our protocol not fully black-box, hence is incomparable with existing, fully black-box, compilers.

Our contribution can be summarized as follows:

OT to malicious OT. However, it is folklore that if one uses the code of the underlying OT protocol one can build a compiler based on non-interactive witness indistinguishable proof systems.

[3]Strong uniformity means that the messages sent by the receiver appear computationally indistinguishable from random to a malicious sender

[4]Although, all existing CIH constructions are non-black-box in the relation.

1. The first application of CIH round-collapsing technique to cut-and-choose and in the plain model. Most known round collapsing techniques apply to proof systems [CCRR18, HL18, CCH+19, PS19, BKM20, LV22, CJJ21b, JJ21, JKKZ21, CJJ21a, CGJ+23]. We introduce novel techniques that allow us to apply CIH for general cut-and-choose in the plain model (without the CRS or RO), and avoid additional assumption of using public key encryption with pseudorandom public keys. Cut-and-choose is a major building block in protocols [LP07, Hai08, HIK+11, Lin13, MOSV22], hence this gadget can be used as a building block for other purposes besides the ones we are pursuing in this paper.

2. The first round-optimal compiler from semi-honest OT to malicious OT from CIH. We use our gadget to design the first cut-and-choose-based compiler from semi-honest OT to malicious in the plain model. While our compiler uses all primitives in a black-box manner, we do need an upper bound on the circuit of the underlying semi-honest OT, and our proof of security is non-black-box. This makes our construction not fully black-box.

## 1.2 Our Techniques

Recall that our goal is to build a four-round (optimal) malicious OT from a two-round semi-honest OT. Furthermore, in the protocol, we want the parties to invoke the underlying cryptographic primitives as a black-box. First, let us recall the definition of simulatable malicious OT (see Sec. 3.1) roughly requires that there exists a polynomial time simulator, that on input only the security parameter, is able to simulate the protocol transcript for a malicious receiver (resp. sender) so that it is indistinguishable from the transcript played by an honest sender (resp. receiver) who plays with the real inputs $s_0, s_1$ (resp. $b$). A simulatable protocol must fulfill two properties: (1) extraction: the protocol should be designed so that – through rewinding – the (black-box) simulator can correctly extract the input that the malicious party is using in the protocol; (2) indistinguishability: the protocol is such that a malicious party cannot distinguish if the honest party is playing with an input sampled from the input distribution, or a random input.

With our upper bound of four rounds, we have that extraction and indistinguishability with respect to a malicious receiver must be achieved in three rounds only. That is, by the end of the third round the receiver must have committed to their input and must have convinced the sender that they played it correctly.

Let us walk through a simple approach to identify the root of the challenge.

First, we define some notation. Let us denote by $\Pi^{\mathsf{sh}} = (\mathtt{OTR}, \mathtt{OTS}, \mathtt{OTD})$ a semi-honest oblivious transfer protocol, which achieves indistinguishability only. Let $\mathrm{ot}_1 = \mathtt{OTR}(b; r\text{-}ot1)$, and $\mathrm{ot}_2 = \mathtt{OTS}(s_0, s_1, \mathrm{ot}_1; r\text{-}ot2)$ be the first and second round messages, with $r\text{-}ot1$ and $r\text{-}ot2$ being the input randomness. Then, $\mathtt{OTD}$ is the output computation procedure of the receiver. Recall that semi-honest security guarantees input privacy for the sender only if the receiver follows the protocol and computes $\mathrm{ot}_1 = \mathtt{OTR}(b; r\text{-}ot1)$, using the randomness $r\text{-}ot1$ and the input $b$ provided by the "challenger." If the receiver instead chooses its randomness or input, no security is guaranteed.

Hence, when using $\Pi^{\mathsf{sh}}$ as a building block in a protocol that must withstand a malicious receiver who can arbitrarily deviate from the protocol and use arbitrary inputs, we must add a mechanism that (1) **forces the receiver to use inputs** $(b; r\text{-}ot1)$ in $\Pi^{\mathsf{sh}}$, and (2) **forces the honest execution** of $\Pi^{\mathsf{sh}}.\mathtt{OTR}(\cdot)$ with such inputs while guaranteeing the privacy of the inputs played by the receiver.

Coin flipping can be used to force the inputs used by the receiver. That is, the inputs $(b, r\text{-}ot1)$ can be generated using contributions from both $\mathsf{S}$ and $\mathsf{R}$ as $r\text{-}ot1_{\mathsf{S}}, r\text{-}ot1_{\mathsf{R}}$, and setting $r\text{-}ot1 = r\text{-}ot1_{\mathsf{S}} \oplus r\text{-}ot1_{\mathsf{R}}$ and $b = r\text{-}ot1[0]$. Coin-flipping requires at least two rounds to be completed, as it entails one message from $\mathsf{R}$ to $\mathsf{S}$ where $\mathsf{R}$ sends a commitment

$\mathsf{com\text{-}r_R} = \mathtt{COM}(r\text{-}ot1_{\mathsf{R}})$ to R's contribution to the inputs of $\mathtt{OTR}(\cdot)$, then S replies with their own contribution $r\text{-}ot1_{\mathsf{S}}$. R can then proceed and compute $\mathrm{ot}_1 = \mathtt{OTR}(b; r\text{-}ot1_{\mathsf{S}} \oplus r\text{-}ot1_{\mathsf{R}})$ in the third round, after the $r\text{-}ot1_{\mathsf{S}}$ is observed. Cut-and-choose can then be used to confirm that for most of the executions, the receiver is computing $\mathrm{ot}_1 := \mathtt{OTR}(\cdot)$ using the inputs generated by the coin-flipping, i.e., $(b, r\text{-}ot1) = (r\text{-}ot1_{\mathsf{R}}[0] \oplus r\text{-}ot1_{\mathsf{S}}[0], r\text{-}ot1_{\mathsf{R}} \oplus r\text{-}ot1_{\mathsf{S}})$. Recall that cut-and-choose consists of having a party, in this case the receiver R, run many, say $m$, executions of $\mathtt{OTR}$ in parallel (all with independent inputs) and sending $\mathrm{ot}_1^{(i)}$ for all $i \in [m]$ to S. Then S will choose a subset $A$ of them, and challenge the receiver to show the inputs and randomness used to compute $\mathrm{ot}_1^{(j)}$ for all $j \in A$. The cut-and-choose is successful if the input and randomness used in the opened sessions correspond to the output of the coin-flipping and are consistent with the honest computation of $\mathrm{ot}_1^{(j)}$.

The cut-and-choose itself requires three rounds, and since it can start only after the coin-flipping is concluded, this approach yields more than four rounds.

**Our approach: Round-collapsing Cut-and-Choose with CIH.** We take inspiration from the sequence of work aiming at collapsing the rounds in zero-knowledge proof system using a correlation intractable hash function (CIH) [CCRR18, HL18, CCH+19, BKM20].

We have the same goal of turning a three-round public-coin protocol into a one-round one, however, our setting has more constraints than [CCRR18, HL18, CCH+19, BKM20]. First, our setting is the plain model, hence we cannot borrow directly their approach of using a trusted setup to choose parameters for the protocol. Second, we aim to provide a transformation that does not use public key with pseudorandom keys, as this property has the same flavor of strongly uniform oblivious transfer, for which a transformation is already known [FMV19]. On the positive side, however, our setting is also very different from the setting of non-interactive zero-knowledge proofs where the CRS must be established once and can be reused many times by different provers. We will leverage such differences to design a new CIH-based transform where there is a one-time CRS for a *designated prover*, as we describe next.

**Our technique: new round-collapsing technique for designated provers from CIH.** First, let us recall the definition of CIH. A CIH is a hash function family $\mathcal{H}$ associated with a relation ensemble $\mathcal{R}$, that has the following property. Given a hash function $H \in \mathcal{H}$, it is hard for any PPT adversary to find an input $x^*$ such that $y = H(x^*)$ and $(x^*, y)$ are in any relation $R \in \mathcal{R}$. Canetti et al. [CCH+18] show that when the relation $\mathcal{R}$ is sparse and efficiently samplable, then there exists $\mathcal{H}$ that is correlation intractable with respect to $\mathcal{R}$ (see Theorem 3.11 [CCH+18]) Then, their goal is to collapse a three-round sigma protocol $x, r_1, r_2, r_3$ into a one message proof system, by computing $r_2 = H(x, r_1)$. The intuition is to take the three-round public-coin proof system and define a relation to pair $((x, r_1), r_2)$ such that, when $x$ is not in the language, the pair $((x, r_1), r_2)$ yield an accepting proof that violates the soundness of the proof-system. Then, for a hash family that is correlation intractable with respect to this relation, we can collapse the rounds knowing that the output $r_2$ of the hash will only allow a prover to prove a false statement with negligible probability.

We follow the blueprint of Canetti et al [CCH+19], but we provide a novel implementation with symmetric key encryption instead of public key encryption, as follows.

- **A designated prover proof system for proving correct function computation.** First, we provide a three-round proof system that allows a designated prover to show that the outputs $y$ for a function $\mathcal{F}$ are computed correctly and consistently to some public inputs. Specifically, the proof system is defined for a language $\mathcal{L}$, where the instances of the language are vectors of inputs $x = (\mathcal{F}, \{inp_1^{(i)}, \mathsf{com\text{-}inp}_2^{(i)}, y^{(i)}\}_{i\in[m]})$, where $inp_1^{(i)}$ is a plaintext input and $\mathsf{com\text{-}inp}_2^{(i)}$ is a commitment to input $inp_2^{(i)}$. Then $x \in \mathcal{L}$ if, for all but a small fraction of $i$, $y^{(i)}$

are the result of the computation of $\mathcal{F}$ on $inp^{(i)}$, where $inp^{(i)}$ is constructed using $inp_1^{(i)}$ and $inp_2^{(i)}$. Looking ahead, it is in the definition of $\mathcal{R}_{\mathcal{L}}$, the relation containing instances $x \in \mathcal{L}$ and their witnesses, that the construction becomes non-black-box, as the relation depends on the circuit of $\mathcal{F}$. The knowledge of the circuit of $\mathcal{F}$ is required only in the proof of security. In the protocols, this information is not required, and just an upper bound of the size of the function is needed.

The proof system that we show is the natural one, where the verifier challenges the prover to open $m/3$ of the first round messages and checks that the prover is being honest.

We deviate from the blueprint of Canetti et al. [CCH+19] in our CRS. Specifically, instead of containing a public key, it contains a vector of commitments to *symmetric keys*, namely, $CRS = (\mathsf{com\text{-}key}^{(1)}, \dots \mathsf{com\text{-}key}^{(m)})$, where each $\mathsf{com\text{-}key}^{(i)} = \mathtt{COM}(key^{(i)}; \rho^{(i)})$ and $key^{(i)}$ is a key for a symmetric key encryption scheme with perfect decryption. This CRS is for one-time use and can be used only by a *designated prover* who will be provided the opening of all the commitments. These keys will be used by the prover to encrypt a partial witness for each session, which serves as the first-round message. Hence, this proof system is for designated provers because the prover must know the keys committed in the CRS to compute these ciphertexts. Along with the witness for these sessions, the prover must also provide the opening to the commitment $\mathsf{com\text{-}key}^{(i)}$ in the CRS so that the verifier can confirm that the encryptions of the first round were computed correctly.

This proof system is defined for any function $\mathcal{F}$, hence it can be used as a building block for other cut-and-choose applications. To prove soundness of this system, we use the standard counting argument for cut-and-choose (Lem. 1). In our compiler, we instantiate $\mathcal{F}$ with $\mathtt{OTR}$ and $\mathtt{OTS}$ which are the receiver's and sender's functions respectively.

- **A round-collapsing transform via CIH.** Towards collapsing the rounds of the three-round proof system, we then define our relation $R^{\mathcal{F}}_{\{key^{(i)}\}_{i\in[m]}}$ with respect to this proof system. Specifically, this relation consists of proofs that violate the soundness of the three-round proof system. We then prove that the relation $R^{\mathcal{F}}_{\{key^{(i)}\}_{i\in[m]}}$ is sparse and efficiently sampleable (Lem. 2). It would be preferable to use a relation that is efficiently searchable, as opposed to sampleable, since this would allow us to use a CIH constructed from plain LWE [PS19]. However, a relation can only be searchable if each pair $(x, y) \in R$ is unique. Since our construction is based on cut and choose, it is not the case that an $x$ has a unique $y$.

  With this three-round proof system and our relation $R^{\mathcal{F}}_{\{key^{(i)}\}_{i\in[m]}}$, we are now ready to use a CIH to collapse the rounds. This proof system uses the same CRS as the three-round proof system, except that the CIH $H$ is part of the CRS as well. Here we follow the blueprint of Canetti et al. [CCH+19], using the hash function to determine the challenge message. We denote the collapsed proof system $\Sigma_{NIP}$. The soundness of our collapsed proof system $\Sigma_{NIP}$ (Lem. 4) mostly follows from the soundness of the three-round proof system, as the only change is using $H$ to compute the challenge message. We then prove that any malicious prover who can violate the soundness of the collapsed proof system can violate the correlation intractability of the hash family.

- **Making the CRS computable through coin-flipping.** The last piece of the puzzle is to address the computation of the CRS through a coin-flipping protocol, as our goal is to be in the plain model. The CRS that we use is a vector of well-formed commitments to keys, whose openings must be known by the prover. At first sight, it seems that this CRS cannot be computed via a coin-flipping, as coin-flipping can only generate a vector of random commitments, for which no one knows the openings. Another look however suggests a different approach. Instead of using coin-flipping to generate the commitments, we use coin-flipping to generate the *openings* of the commitments. From the openings, which also serve as the backdoor for the designated prover, the prover can derive the commitments, which determine the CRS. The choosing of the CIH is done by the verifier (as is done by Kalai, Rothblum, and Rothblum [KRR17]). In Fig. 6, we provide the one-round proof system combined with the CRS generation, which we call $\Sigma_{NIP}^{\mathsf{plain}}$.

Note that a malicious prover might lie in the commitments that were generated as a part of the coin-flipping. However, recall that in our proof system, the prover must open a large fraction of the first round messages, which must be validated against the CRS that the prover claimed to be the result of the coin-flipping. If the prover changes too many commitments in the CRS, they will be caught.

**Putting it all together.** The one-round designated prover proof system for correct function evaluation that we developed can now be used in combination with the two-round coin-flipping. The receiver is asked to provide a proof that the output of function $\mathcal{F} = \mathtt{OTR}$, on the inputs, $\{inp_1^{(i)}, \mathsf{com\text{-}inp}_2^{(i)}\}$, appearing in the transcript of the coin-flipping, is correctly computed. Note that $\mathtt{OTR}$ is used as a black-box by the sender and the receiver. But the hash family from which the hash function is picked is determined by the size of the circuit of $\mathtt{OTR}$.

We have focused on the receiver, but the same approach is mirrored on the sender side, with only one difference. We leverage the observation (made in [MOSV22]) that any two-round semi-honest OT is always private against a malicious sender. Hence, on the sender side, we do not require a coin-flipping for the inputs to $\mathtt{OTS}$. The detailed OT protocol is described in Sec. 5. In the following, we describe the protocol with very broad strokes:

- **Round 1.** $\mathsf{R} \rightarrow \mathsf{S}$: $\mathsf{R}$ prepares and sends commitments $\mathsf{com\text{-}crs\text{-}prvr}_{\mathsf{R}}$ for the coin-flipping for the CRS for her proof system and commitments $\mathsf{com\text{-}r}_{\mathsf{R}}^{(i)}$ to randomness $r\text{-}ot1_{\mathsf{R}}^{(i)}$ for the coin-flipping to build the inputs to $\mathtt{OTR}$.

- **Round 2.** $\mathsf{S} \rightarrow \mathsf{R}$ : The sender sends the randomness $crs\text{-}vrfr_{\mathsf{R}}$ which includes $\{r\text{-}key_V^{(i)}\}_{i\in[m]}$ for the coin-flipping of the receiver's CRS, and the randomness $r\text{-}ot1_{\mathsf{S}}^{(i)}$ for the coin-flipping for the input to $\mathtt{OTR}$. $\mathsf{S}$ also chooses the CIH $H_k^{\mathsf{R}} \leftarrow \mathcal{H}$ to be used by $\mathsf{R}$.

  Next $\mathsf{S}$ initiates a coin-flipping for their own CRS and commits to the inputs they will use to compute the sender's message via $\mathtt{OTS}$. $\mathsf{S}$ computes commitments $\mathsf{com\text{-}crs\text{-}prvr}_{\mathsf{S}}$ for the CRS and $\mathsf{com\text{-}r}_{\mathsf{S}}^{(i)}$ for their input $r\text{-}ot2^{(i)} = k_0^{(i)} \| k_1^{(i)} \| r\text{-}ot2^{*(i)}$ for $i \in [m]$ in the $\mathtt{OTS}$ algorithm and sends them to $\mathsf{R}$.

- **Round 3.** $\mathsf{R} \rightarrow \mathsf{S}$: The coin-flipping for $\mathsf{R}$ is concluded, and $\mathsf{R}$ learns the openings $key_{\mathsf{R}}^{(i)}, \rho_{\mathsf{R}}^{(i)}$ (computed as $r\text{-}key_P^{(i)} \oplus r\text{-}key_V^{(i)}$) from which she can derive the commitments to be included in the CRS she will be using in the proof $\mathbf{CRS}_{\mathsf{R}} = (\{\mathsf{com\text{-}key}_{\mathsf{R}}^{(i)}\}_{i\in[m]}, H_k^{\mathsf{R}})$. $\mathsf{R}$ also learns the inputs $r\text{-}ot1^{(i)}$ to be used to run $\mathtt{OTR}$ and obtain outputs $\mathrm{ot}_1^{(i)}$ and proves using our one-round proof system $\Sigma_{NIP}^{\mathsf{plain}}$ that each $\mathrm{ot}_1^{(i)}$ was computed as $\mathtt{OTR}(r\text{-}ot1_{\mathsf{S}}^{(i)}[0] \oplus r\text{-}ot1_{\mathsf{R}}^{(i)}[0]; r\text{-}ot1_{\mathsf{S}}^{(i)} \oplus r\text{-}ot1_{\mathsf{R}}^{(i)})$. That is, prove that $\mathrm{ot}_1^{(i)}$ is the output of $\mathtt{OTR}$ on input the result of the coin-flipping, and sends the proof to the $\mathsf{S}$.

Then $\mathsf{R}$ participates in the coin-flipping for $\mathsf{S}$'s CRS, sending $crs\text{-}vrfr_{\mathsf{R}}$ to $\mathsf{S}$. Along with this, $\mathsf{R}$ chooses the hash function $H_k^{\mathsf{S}}$ that $\mathsf{S}$ must use in their proof system and sends $H_k^{\mathsf{S}}$.

Finally, $\mathsf{R}$ computes "adjustment bits". Because we are performing many OT sessions while $\mathsf{R}$ has as input a single bit, we cannot use $\mathsf{R}$'s input bit for each session. Otherwise, $\mathsf{S}$ would learn the bit from an opened session. Instead, $\mathsf{R}$ plays with random bits and now sends adjustment bits to ensure that each session results in $\mathsf{R}$ learning the correct string.

- **Round 4.** $\mathsf{S} \to \mathsf{R}$ In the last round, $\mathsf{S}$ finishes the coin-flipping to obtain $\mathbf{CRS}_{\mathsf{S}} = (\{\mathsf{com\text{-}key}_{\mathsf{S}}^{(i)}\}_{i\in[m]}, H_k^{\mathsf{S}})$. $\mathsf{S}$ then computes $\mathrm{ot}_2^{(i)}$ and proves via $\Sigma_{NIP}^{\mathsf{plain}}$ that each $\mathrm{ot}_2^{(i)}$ was computed as $\mathrm{ot}_2^{(i)} = \mathtt{OTS}(k_0^{(i)}, k_1^{(i)}, \mathrm{ot}_1^{(i)}; r\text{-}ot2^{*(i)})$, where $r\text{-}ot2^{(i)} = k_0^{(i)} \| k_1^{(i)} \| r\text{-}ot2^{*(i)}$ was committed in round two. $\mathsf{S}$ then sends the proof to $\mathsf{R}$.

  Now $\mathsf{S}$ computes shares $s_0^{(i)}$ and $s_1^{(i)}$ of their input strings $s_0, s_1$, and encrypts each share using one-time-pad keys $k_0^{(i)}, k_1^{(i)}$, adjusting the positions according to the adjustment bits, which were the inputs to $\mathtt{OTS}$. The sender sends each ciphertext to $\mathsf{R}$.

- **Output Computation.** $\mathsf{R}$ uses $\mathrm{ot}_2^{(i)}$ as input to the output computation function $\mathtt{OTD}$ to learn one of the keys $k_0^{(i)}, k_1^{(i)}$. Then, using these keys, $\mathsf{R}$ is able to decrypt shares of the input strings and reconstruct the final output $s_b$.

## 2 Related Work

**Correlation-Intractable Hash Functions.** Correlation-intractable hash functions (CIH) were first used to instantiate the Fiat-Shamir transform in two concurrent works: Canetti, Chen, and Reyzin [CCR16] and Kalai, Rothblum, and Rothblum [KRR17]. [CCR16] constructed a CIH from subexponentially secure iO, subexponentially secure puncturable PRFs, and input hiding obfuscation for multi-bit point functions. [KRR17] used CIH to collapse a constant round public-coin proof to a two-round system.

Canetti et al. [CCRR18] construct a CIH from strong KDM-secure encryption and use it to transfrom public-coin HVZK proof system to a non-interactive ZK (NIZK) in the CRS model. Holmgren and Lombardi [HL18] construct a CIH from subexponential iO and exponentially secure OWFs to achieve NIZK in the CRS model. These works rely on heavy assumptions. Subsequent works aimed to construct CIH from simpler assumptions. Canetti et al. [CCH+19] provided a CIH from almost optimal search-LWE and one from circular secure FHE. They used this CIH to construct a publicly verifiable SNARG by collapsing the round of the GKR [GKR08] interactive proof.

Peikert and Shiehian [PS19] construct a CIH from plain LWE and use it to construct a NIZK. Holmgren, Lobardi, and Rothblum [HLR21] later showed that by coupling the hash function of Peikert and Shiehian [PS19] or of Canneti et al. [CCH+19] with a derandomization procedure to apply the Fiat-Shamir transform to GMW, or, more generally, any commit-challenge-response proof[5]. Lombardi and Vaikuntanathan [LV22] present a CIH from shift hiding shiftable functions[6]. Brakerski, Koppula, and Mour [BKM20] consider CIHs for approximable relations, which they build using trapdoor hash functions (which can be built from DDH, QR, DCR, and LWE) and consequently construct a NIZK. Two concurrent works [BFJ+20, GJJM20] use these strategies to construct the first statistical

[5]There is an additional requirement that the commit-challenge-response proof have their commitments instantiated with a public key encryption scheme.

[6]Shift hiding shiftable functions [PS18] are a type of constrained PRFs, where a party can compute secret keys that allow a party to evaluate only at authorized points, while all other points remain pseudorandom.

**Table 1:** Comparison of Our Work with Existing OT Compilers

| Work | OT from | Black-box | Plain Model | Round optimal |
|---|---|---|---|---|
| [KO04] | Certified TDPs | ✗ | ✓ | ✓ |
| [HIK+11] | Semi-honest OT | ✓ | ✓ | ✗ |
| [ORS15] | Certified TDPs | ✓ | ✓ | ✓ |
| [DGH+20] | Elementary OT | ✗ | ✗ | ✓ |
| [FMV19] | Strongly-uniform OT | ✓ | ✓ | ✓ |
| [CCG+21] | TDPs | ✓ | ✓ | ✓ |
| [IKSS22a] | Semi-honest OT | ✓ | ✗ | ✓ |
| [MOSV22] | Defensible OT | ✓ | ✓ | ✓ |
| [MOSV22] | Semi-honest OT | ✓ | ✗ | ✓ |
| This work | Semi-honest OT | ✗ | ✓ | ✓ |

two message public coin witness indistinguishable (statistical ZAP) arguments. Both works modify the NIZK approach by using an extractable statistically hiding commitment scheme. Jain and Jin [JJ21] construct their CIH from sub-exponential DDH, and use this CIH to construct a NIZK and a statistical ZAP. Finally, Choudhuri, Jain, and Jin [CJJ21b] use CIH to construct SNARGs for P, and Choudhuri et al. [CGJ+23] later extend this to SNARGs for P and Batch NP.

**Round-Optimal Compilers for Secure Computation.** There is a large body of literature on how to build cryptographic tasks (e.g., MPC [Kil88, IKLP06, CDMW09, PW09, Wee10, Goy11, LP12, KMO14, ORS15], zero-knowledge proofs [PW09, Goy11, GOSV14], non-malleable commitments [PW09, Wee10, Goy11, GLOV12]) using weaker primitives in a black-box manner.

Since our focus is on (round-optimal) compilers we shall focus only on related work for round-optimal black-box constructions of Oblivious Transfer and secure computation, from weaker building blocks.

Ostrovsky, Richelson, and Scafuro show four-round OT from certified trapdoor permutation [ORS15], improved by Choudhuri et al. [CCG+21], who do not require the permutation to be certifiable. In [FMV19] Friolo, Masny, and Venturi, show a four-round OT from strongly-uniform OT (i.e., the first message of the receiver must be indistinguishable from a random string). Such a result can be seen as building four-round OT from PKE with pseudorandom keys. Such assumptions are less general than semi-honest OT, which we use in this paper, although our compiler uses CIH (which are incomparable with PKE with pseudorandom keys). Madathil et al. in [MOSV22] build malicious OT from two-round defensible OT (defensible means that, while a malicious receiver can cheat in the protocol and learn both inputs of the sender without being detected, it should be hard to later convince the sender that it behaved honestly). In the multiparty setting, Ishai et al. [IKSS21] show various round-optimal black-box transformations of round-optimal MPC protocols. Their transformations rely on PKE with pseudorandom keys and on semi-malicious OT. Semi-malicious security [AJL+12] means that security is guaranteed against a passive adversary, even if the adversary is allowed to choose the randomness maliciously. Although mildly, semi-malicious security is stronger than semi-honest security. More recently, Ishai et al [IKSS23] show a new transformation that uses a sub-exponential 2-round OT secure against unbounded malicious receivers, to build a 4-round MPC protocol. Still in the MPC setting, [COSW23] Ciampi et al, also explores lower bounds for round-optimal transformation for specific inputless functionalities. Their transformation builds on OT that is private (but not simulatable) against malicious adversaries.

Finally, in the CRS model, Ishai at al. [IKSS22b] show a two-round OT from two-round malicious private OT. The same authors show a two-round OT protocol from a semi-honest

**Ideal Functionality $\mathcal{F}_{\mathsf{OT}}$:**

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

**Figure 1:** Ideal functionality for oblivious transfer

OT protocol, in the Random Oracle Model. We note that, their use of the RO is not only for collapsing the rounds of the protocol, but also to allow the simulator to extract the inputs of the parties, as well as programming the output of the RO in the proof. Therefore, even allowing for more rounds, the CIH cannot be directly used to instantiate the RO in [IKSS22b]'s construction, but new techniques are required. Our results are in the plain model.

In Table 1, we present a succinct comparison of our compiler with existing OT Compilers. We compare compilers that construct malicious secure simulatable OT from weaker primitives. We emphasize that our work is non-black-box because the CIH instantiations will require an upper bound of the size of the circuit (semi-honest OT in our work).

## 3 Preliminaries

**Notation:** We denote our security parameter by $\lambda \in \mathbb{N}$, which we treat as an implicit parameter. For a number $n \in \mathbb{N}$, let $[n] = \{1, \ldots, n\}$. We denote by $x^{(i)} \in X$ the $i$-th element of the set $X$. For a bit string $bits$, we use $bits[i]$ to refer to the $i$th bit. We show plaintext values in *italics* and hidden values (either by commitment or encryption) in sans serif. Lastly, let $\mathbf{view}^{\mathtt{A}}_{\Pi,\mathtt{B}}(x, y)$ denote the random variable representing the view of party $\mathtt{A}$ using input $x$ after executing the two-party interactive protocol $\Pi$ with party $\mathtt{B}$ with input $y$. This view contains $\mathtt{A}$'s input, randomness, and messages received.

### 3.1 Oblivious Transfer

We present the definition of semi-honest OT [MOSV22].

The ideal OT functionality definition is presented in Fig 1.

**Definition 1.** Let $\Pi = \langle \mathsf{S}(s_0, s_1), \mathsf{R}(b)\rangle$ be a non-trivial OT protocol. We say that $\Pi$ is **private for random inputs against semi-honest receivers** if the following holds:

$$\{\mathbf{view}^{\mathsf{R}}_{\Pi,\mathsf{S}}((b), (s_0, s_1)), s_{1-b}\}_b \approx \{\mathbf{view}^{\mathsf{R}}_{\Pi,\mathsf{S}}((b), (s_0, s_1)), s'\}_b$$

Where $s_0, s_1, s' \xleftarrow{\$} \{0,1\}^\lambda$. We say that $\Pi$ is **private for random inputs against semi-honest senders** if the following holds:

$$\{\mathbf{view}^{\mathsf{S}}_{\Pi,\mathsf{R}}((s_0, s_1), (0))\}_{s_0,s_1} \approx \{\mathbf{view}^{\mathsf{S}}_{\Pi,\mathsf{R}}((s_0, s_1), (1))\}_{s_0,s_1}$$

Where $s_0, s_1 \in \{0,1\}^\lambda$

Note that Madathil et al.[MOSV22] showed that any two-round OT protocol with privacy against semi-honest senders is already private against malicious senders.

**Simulatable OT** Simulatable OT considers security in the real/ideal world paradigm. In this paradigm, an attacker corrupts either the sender $\mathsf{S}$ or receiver $\mathsf{R}$ (denoted $\mathsf{S}^*$ or $\mathsf{R}^*$ respectively when corrupt), and participates in the OT protocol. The protocol is executed in either the real world or the ideal world, where an ideal functionality $\mathcal{F}_{OT}$ (Fig. 1 [MOSV22]) performs the computation. $\mathbf{real}_{\Pi,\mathsf{S}^*(z)}((s_0,s_1),(b))$ (respectively $\mathbf{real}_{\Pi,\mathsf{R}^*(z)}((s_0,s_1),(b))$) is the view, and consequently the output, of the adversary after executing the OT protocol $\Pi$ in the real world as a corrupt sender (respectively receiver), where $z$ is the auxiliary input of the adversary. Likewise, $\mathbf{ideal}_{\Pi,\mathsf{Sim}^{\mathsf{S}^*(z)}}((s_0,s_1),(b))$ (respectively $\mathbf{ideal}_{\Pi,\mathsf{Sim}^{\mathsf{R}^*(z)}}((s_0,s_1),b))$ is the view simulated by the ideal adversary $\mathsf{Sim}$.

**Definition 2.** The protocol $\Pi = (\mathsf{S},\mathsf{R})$ **securely realizes** $\mathcal{F}_{OT}$ if

- For every non-uniform, PPT malicious sender $\mathsf{S}^*$ there exists a non-uniform, PPT simulator $\mathsf{Sim}$ such that

$$\{\mathbf{real}_{\Pi,\mathsf{S}^*(z)}((s_0,s_1),(b))\}_{\lambda,s_0,s_1,b,z} \approx \{\mathbf{ideal}_{\Pi,\mathsf{Sim}^{\mathsf{S}^*(z)}}((s_0,s_1),(b))\}_{\lambda,s_0,s_1,b,z}$$

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

- For every non-uniform, PPT malicious receiver $\mathsf{R}^*$ there exists a non-uniform, PPT simulator $\mathsf{Sim}$ such that

$$\{\mathbf{real}_{\Pi,\mathsf{R}^*(z)}((s_0,s_1),(b))\}_{\lambda,s_0,s_1,b,z} \approx \{\mathbf{ideal}_{\Pi,\mathsf{Sim}^{\mathsf{R}^*(z)}}((s_0,s_1),(b))\}_{\lambda,s_0,s_1,b,z}$$

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

## 3.2 Commitment Scheme

A commitment scheme allows a party to commit to a value and later reveal the committed value. A commitment scheme consists of one polynomial-time algorithm:

- $\mathtt{COM}(x\ ;\ r) = \mathsf{com}$: Takes as input a value $x$ and randomness $r$ and outputs a commitment $\mathsf{com}$ to $x$. The commitment can be opened by revealing the value $x$ and the randomness $r$. Any party with knowledge of these values can check that $\mathsf{com} = \mathtt{COM}(x\ ;\ r)$.

The two properties we require of our commitment scheme are statistical binding and computational hiding.

**Definition 3.** (Hiding of commitments) A commitment scheme for message space $\mathcal{M}$ is **statistically (resp. computationally) hiding** if for all $m_0, m_1 \in \mathcal{M}$ it holds the $\{\mathtt{COM}(m_0; U_\lambda)\}_{\lambda\in\mathbb{N}}$ is statistically (resp. computationally) indistinguishable from $\{\mathtt{COM}(m_1; U_\lambda)\}_{\lambda\in\mathbb{N}}$, where $U_\lambda$ denotes the uniform distribution over $\{0,1\}^\lambda$.

**Definition 4.** (Binding of commitments) A commitment scheme is **computationally binding** if for all PPT adversaries $\mathcal{A}$ there is a negligible function $\mathsf{negl} : \mathbb{N} \to [0,1]$ such that:

$$Pr[\mathtt{COM}(m;r) = \mathtt{COM}(m';r') \wedge m \neq m' : ((m,r),(m',r') \leftarrow \mathcal{A}(1^\lambda))] < \mathsf{negl}(\lambda)$$

We say that $\mathtt{COM}$ is **statistically binding** if $\mathcal{A}$ is unbounded and the above probability holds.

In this work, we occasionally specify the randomness being used to commit. Specifically $\mathsf{com} = \mathtt{COM}(m\ ;\ r)$ means that $\mathsf{com}$ is a commitment to the message $m$ using randomness $r$. However, if the randomness is not specified, as in $\mathsf{com} = \mathtt{COM}(m)$, then we let the second half of $m$ be the randomness used to compute the commitment.

## 3.3 Symmetric Key Encryption

A symmetric key encryption scheme $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ is a tuple of algorithms defined as follows:

- $\mathsf{KeyGen}(1^\lambda) = key$: On input the security parameter outputs a key $key$
- $\mathsf{Enc}(key, m) = c$: On input a key $key$ and a message $m$ outputs a ciphertext $c$ of $m$
- $\mathsf{Dec}(key, c) = m$: On input a key $key$ and a ciphertext $c = \mathsf{Enc}(key, m)$, outputs the original message $m$

We require that $\mathcal{E}$ is secure against chosen plaintext attacks and that keys are random.

**Definition 5.** (IND-CPA security with random keys) An encryption scheme $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ is **IND-CPA secure** if

$$|Pr[\mathcal{A}(c^*) = 1 \ : \ c^* = \mathsf{Enc}(key, m_0)] - Pr[\mathcal{A}(c^*) = 1 \ : \ c^* = \mathsf{Enc}(key, m_1)]| < \mathsf{negl}(\lambda)$$

Where $key \leftarrow \mathsf{KeyGen}(1^\lambda)$, $\mathsf{KeyGen}(1^\lambda)$ is indistinguishable from the uniform distribution, and $\mathcal{A}(1^\lambda) \to (m_0, m_1)$

Finally, we require that $\mathcal{E}$ has perfect decryption correctness. This property, similar to the binding of a commitment scheme, states that given a ciphertext and key, this ciphertext can only be decrypted to one value.

**Definition 6.** (Perfect decryption correctness) An encryption scheme $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ has **perfect decryption correctness** if

$$Pr[\mathsf{Enc}(key, m_0 \ ; \ r_0) = \mathsf{Enc}(key, m_1 \ ; \ r_1)] = 0$$

For $m_0 \neq m_1$ or $r_0 \neq r_1$ where $r_0, r_1$ is the randomness used to encrypt

## 3.4 Correlation Intractability

A hash family $\mathcal{H} = \{H : K \times X \to Y\}$ is correlation intractable (Def. 7) with respect to a relation ensemble $\mathcal{R} = \{R \subseteq X \times Y\}$ if it is hard for an adversary, given $H_k \in \mathcal{H}$ to output $x \in X$ such that $(x, H_k(x)) \in R$ [CGH98, CGH04]. That is, an adversary should not be able to find an input $x$ to the hash function such that the output of the hash function satisfies some relation. We present the formal definition in Def. 7.

**Definition 7.** Given a family $\mathcal{H} = \{H : K \times X \to Y\}$ of hash functions and a relation ensemble $\mathcal{R} = \{R \subseteq X \times Y\}$, $\mathcal{H}$ is said to be **correlation intractable** with respect to $\mathcal{R}$ if for all polynomial time $\mathcal{A}$, $\Pr[(x, H_k(x)) \in R] \leq \mathsf{negl}(\lambda)$, when $k \leftarrow K$ and $\mathcal{A}(k) \to x$.

Further, it is necessary that the relation $R \in \mathcal{R}$ be *sparse* (Def. 8) and *efficiently sampleable* (Def. 9). As shown by Canetti et al. (see Theorem 3.11 [CCH+18]) these properties imply the existence of a CIH for that relation. For a relation to be sparse, it means that, given $x \in X$, there should be a restricted number of $y \in Y$ such that $(x, y) \in R$. We present the definition of sparsity from [CCH+19] below:

**Definition 8.** A relation $\mathcal{R} = \{R \subseteq X \times Y\}$ is said to be $\rho(\cdot)$ **-sparse** if for $\lambda \in \mathbb{N}$ and any $x \in X$ we have

$$\Pr[(x, y) \in R] \leq \rho(\lambda)$$

when $\rho(\cdot)$ is negligible, we simply say that $\mathcal{R}$ is **sparse**.

**Algorithm**: $\Sigma.\mathtt{SetUp}(1^\lambda)$

1. For $i \in [m]$ : Compute $key^{(i)} \leftarrow \mathsf{KeyGen}(1^\lambda)$. Sample $\rho^{(i)} \overset{\$}{\leftarrow} \{0,1\}^\lambda$. Compute $\mathsf{com\text{-}key}^{(i)} = \mathtt{COM}(key^{(i)} \ ; \ \rho^{(i)})$
2. Output $\{\mathsf{com\text{-}key}^{(i)}\}_{i\in[m]}$ as the CRS, and send $(key^{(i)} \ ; \ \rho^{(i)})$ to the prover

**Figure 2:** Set-Up Algorithm of $\Sigma$

Efficient sampleability means that there exists a polynomial-sized circuit that, given $x$, can approximate the distribution $\{z \in Y : (x, z) \in R\}$ within some error $\epsilon$, assuming that this distribution is non-empty.

**Definition 9.** A relation ensemble $\mathcal{R} = \{R \subseteq X \times Y\}$ is **non-uniformly $\epsilon$-approximately sampleable** if there exists a polynomial time circuit ensemble $\{\mathbf{Samp}\}$ such that for every $(x, y) \in R$ the distribution $\mathbf{Samp}(x)$ multiplicatively $\epsilon$-approximates the uniform distribution on the (by assumption, non-empty) set $\{y \in Y : (x, y) \in R\}$

Lastly, we need the CIH to have *approximate average-case programmability* (Def. 10) [CCH+19]. This property states that, given $x$ and a uniformly random $y$, there is a way to efficiently sample a hash function $H \in \mathcal{H}$ such that $H(x) = y$.

**Definition 10.** We say that a hash family $\mathcal{H}$ has **approximate average-case programmability** if there exists an efficient sampling algorithm **Samp** such that for all fixed $x$, the distribution $\{H \leftarrow \mathbf{Samp}(x, y)\}$ where $y$ is uniformly random is statistically indistinguishable from the distribution $H \leftarrow \mathcal{H}$. In other words, there exists a sampling algorithm $\mathbf{Samp}(1^\lambda, x, y)$ that samples from the conditional distribution $k \leftarrow \mathcal{H}.\mathsf{KeyGen}(1^\lambda)|H(k, x) = y$.

Note that without the programmability property, we would have such a CIH from plain LWE [HLR21]. However, the programmability is what allows the simulators for our OT protocol to extract the input of the malicious parties and is therefore necessary.

## 4 Proof of Correct Function Evaluation

In this section, we present a designated prover proof system for proving that a set of values $\{y^{(i)}\}_{i\in[m]}$ are the output of a function $\mathcal{F}$ on a specific input $\{inp^{(i)}\}_{i\in[m]}$. Each $inp^{(i)}$ is computed using a plaintext input $inp_1^{(i)}$ that is part of the statement, and a hidden input $inp_2^{(i)}$ that is part of the witness and committed in the statement. We follow the cut-and-choose paradigm [LP07], where the first message contains encryptions of the witnesses, and the second round (challenge) message determines which encryptions to open. This proof system is in the CRS model, where our CRS contains a set of commitments $\{\mathsf{com\text{-}key}^{(i)}\}_{i\in[m]}$ to the symmetric encryption keys, and the prover receives the opening $\{(key^{(i)}, \rho^{(i)})\}_{i\in[m]}$ to these commitments from the same party who computes the CRS. This means that only the party who receives these openings can compute a verifying proof, hence the term designated prover. We present the set-up algorithm of our proof system in Fig. 2.

In the language of this proof system, $\mathcal{L}$, we have the statement $x = (\mathcal{F}, \{inp_1^{(i)}, \mathsf{com\text{-}inp}_2^{(i)}, y^{(i)}\}_{i\in[m]})$ where $\mathcal{F}$ is the function, $inp_1^{(i)}$ is some plaintext input, $\mathsf{com\text{-}inp}_2^{(i)}$ is a commitment to some input $inp_2^{(i)}$, and $y^{(i)}$ is the output $y^{(i)} = \mathcal{F}(inp^{(i)})$ where $inp^{(i)}$ is constructed using $inp_1^{(i)}$ and $inp_2^{(i)}$. Next, we have the witness $w = (\{inp^{(i)}, inp_2^{(i)}, \gamma^{(i)}\}_{i\in[m]})$

**Proof System:**

$\Sigma = \langle \mathsf{P}(x, w \ ; \ \{key^{(i)}, \rho^{(i)}\}_{i\in[m]}), \mathsf{V}(x) \ ; \ \{\textsf{com-key}^{(i)}\}_{i\in[m]} \rangle$

1. $\mathsf{P}$ computes $c^{(i)} = \mathsf{Enc}(key^{(i)}, (inp^{(i)}, inp_2^{(i)}, \gamma^{(i)}, \rho^{(i)}); \zeta^{(i)})_{i\in[m]}$ and send $(\{c^{(i)}\}_{i\in[m]})$.
2. $\mathsf{V}$ : Upon receipt of $(\{c^{(i)}\}_{i\in[m]})$ from $\mathsf{P}$, choose $e \xleftarrow{\$} \{0,1\}^m$ such that $m/3$ bits of $e$ are 1. Send $(e)$ to $\mathsf{P}$.
3. $\mathsf{P}$ : Upon receipt of $(e)$ from $\mathsf{V}$. If there are not $m/3$ 1s in $e$, abort. Let $z^{(i)} = (inp^{(i)}, inp_2^{(i)}, \gamma^{(i)}, \zeta^{(i)}, key^{(i)}, \rho^{(i)})$ such that $e[i] = 1$. Send $(\{z^{(i)}\})$ to $\mathsf{V}$
4. $\mathsf{V}$ (verification): Upon receipt of $(\{z^{(i)}\})$ from $\mathsf{P}$. If $\textsf{com-key}^{(i)} \neq \texttt{COM}(key^{(i)} \ ; \ \rho^{(i)})$, $\textsf{com-inp}_2^{(i)} \neq \texttt{COM}(inp_2^{(i)}; \gamma^{(i)})$, $y^{(i)} \neq \mathcal{F}(inp^{(i)})$, or $\texttt{Check}_{\mathcal{F}}(inp_1^{(i)}, inp_2^{(i)}, inp^{(i)}) = 0$ for any $z^{(i)} = (inp^{(i)}, inp_2^{(i)}, \gamma^{(i)}, \zeta^{(i)}, key^{(i)}, \rho^{(i)})$ output 0. Else output 1

**Figure 3:** Three-Round Proof of Function Output for receiver via Cut-and-Choose

where, for at least $8m/9$ of $i \in [m]$, $inp^{(i)}$ is the input to the function built using $inp_1^{(i)}$ and $inp_2^{(i)}$, $inp_2^{(i)}$ is the input committed in $\textsf{com-inp}_2^{(i)}$, $\gamma^{(i)}$ is the randomness used to compute the commitment $\textsf{com-inp}_2^{(i)}$, and the external predicate $\texttt{Check}_{\mathcal{F}}$ confirms that $inp^{(i)}$ is constructed correctly. This predicate is dependent on the function and must be computable in polynomial time.

$$\mathcal{L} = \{x = (\mathcal{F}, \{inp_1^{(i)}, \textsf{com-inp}_2^{(i)}, y^{(i)}\}_{i\in[m]}) \mid \exists \ w = (\{inp^{(i)}, inp_2^{(i)}, \gamma^{(i)}\}_{i\in[m]})$$
$$\text{where } \textsf{com-inp}_2^{(i)} = \texttt{COM}(inp_2^{(i)}; \gamma^{(i)}), y^{(i)} = \mathcal{F}(inp^{(i)}), \texttt{Check}_{\mathcal{F}}(inp_1^{(i)}, inp_2^{(i)}, inp^{(i)}) = 1$$
$$\text{for} > 8m/9 \text{ of } i \in [m]\}$$

A formal description of the proof system $\Sigma$ is presented in Fig. 3.
Now we present our lemma on the soundness of $\Sigma$.

**Lemma 1.** *If $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ is a symmetric key encryption scheme with perfect decryption correctness, $\mathcal{F}$ is a deterministic function, and* COM *is a statistically binding commitment scheme, then $\Sigma$ is a sound proof system*

*Proof.* Assume towards a contradiction that $\Sigma$ is not sound. That means that there exists an adversary $\mathcal{A}$ that can prove a statement $x = (\mathcal{F}, \{inp_1^{(i)}, \textsf{com-inp}_2^{(i)}, y^{(i)}\}_{i\in[m]}) \notin \mathcal{L}$ with non-negligible probability $p(\lambda)$.

Because COM is a statistically binding commitment, we know that $\mathcal{A}$ cannot open $\textsf{com-inp}_2^{(i)}$ to anything other than $(inp_2^{(i)} \ ; \ \gamma^{(i)})$ or $\textsf{com-key}^{(i)}$ to anything other than $(key^{(i)} \ ; \ \rho^{(i)})$.

Next, because $\mathcal{F}$ is a deterministic function, we know that there is only one $inp^{(i)}$ such that $\mathcal{F}(inp^{(i)}) = y^{(i)}$.

Further, because we assume that $\mathcal{E}$ has perfect decryption correctness, we know that $\mathcal{A}$ cannot open a ciphertext to any value other than the value that was encrypted.

Thus the only way for $\mathcal{A}$ to win is if cut-and-choose fails. That is, if the $m/3$ values $\{z^{(i)}\}$ are all correct, but there are less than $8m/9$ correct values in the witness $w$. Note that our analysis here is identical to the analysis of cut-and-choose in [MOSV22], we restate it here for completeness. Let $j \in [m]$ be the number of incorrect values in $w$. Let $\mathbf{Bad}_0$ be the event that $j$ exceeds or equals $m/9$. We need to show $Pr[\mathbf{Bad}_0] < \mathsf{negl}(\lambda)$. This can be done by summing the probability of $\mathbf{Bad}_0$ over all $j$.

First note that if $j > 2m/3$, then $\mathcal{A}$ cannot pass cut-and-choose as there are not enough correct values to open. Next note that if $j < m/9$, $\mathcal{A}$ has not reached the set limit on the number of bad values. Therefore we need only take the sum from $m/9$ to $2m/3$.

Finally, since $\mathcal{A}$ must pass cut-and-choose, $Pr[\mathbf{Bad}_0]$ for a specific $j$ can be computed by taking the ratio of the number of ways to select only good values over the number of ways to select any values. Therefore we have:

$$Pr[\mathbf{Bad}_0] = \sum_{j=m/9}^{2m/3} \frac{\binom{m-j}{m/3}}{\binom{m}{m/3}} \tag{1}$$

$$= \frac{\sum_{j=0}^{8m/9} \binom{j}{m/3}}{\binom{m}{m/3}} \tag{2}$$

$$= \frac{\binom{8m/9+1}{m/3+1}}{\binom{m}{m/3}} \tag{3}$$

$$= \frac{(8m/9+1)!(m/3)!(2m/3)!}{m!(m/3+1)!(5m/9)!} \tag{4}$$

$$= \frac{(2m/3)(2m/3-1)\dots(5m/9+1)}{(m)(m-1)\dots(8m/9+2)(m/3+1)} \tag{5}$$

$$= \frac{\prod_{j=5m/9+1}^{2m/3} j}{(m/3+1)\prod_{j=8m/9+2}^{m} j} \tag{6}$$

$$= \frac{(8(m/9)+1)}{(3(m/9)+1)} \frac{\prod_{j=5(m/9)+1}^{6(m/9)} j}{\prod_{j=8(m/9)+1}^{9(m/9)} j} \tag{7}$$

$$= \frac{(8(m/9)+1)}{(3(m/9)+1)} \frac{\prod_{j=1}^{(m/9)} 5(m/9)+j}{\prod_{j=1}^{(m/9)} 8(m/9)+j} \tag{8}$$

$$= \frac{(8(m/9)+1)}{(3(m/9)+1)} \prod_{j=1}^{m/9} \frac{5(m/9)+j}{8(m/9)+j} \tag{9}$$

In the above, Eq. 2 follows from shifting the indices of the summation. Eq. 3 follows from the column-sum property of binomial coefficients. Eqs. 4, 5, and 6 are based on the definition of binomial coefficients. Eq. 7 we simply multiply both the numerator and denominator by $8m/9+1$. Eq. 8 follows from shifting the indices of the products.

In the equations above we have that $1 \leq j \leq (m/9)$ we know that $\frac{8(m/9)+1}{3(m/9)+1} \leq 8/3$ and $\prod_{j=1}^{m/9} \frac{5(m/9)+j}{8(m/9)+j} \leq (2/3)^{m/9}$. Therefore we have $Pr[\mathbf{Bad}_0] \leq (8/3)(2/3)^{m/9}$, which is negligible due to the choice of $m = O(\lambda)$[7].

[7] Madathil et al. [MOSV22] show that this can be reduced to a tighter bound using Stirling's formula

Thus we know that cut-and-choose fails negligibly, and therefore $\Sigma$ is sound. $\square$

**The Relation.** Given this three-round cut-and-choose-based proof system $\Sigma$, we now collapse the rounds using a CIH. To do this, we first define the relation $R^{\mathcal{F}}_{\{key^{(i)}\}_{i\in[m]}}$ that we need the hash family to be correlation intractable with respect to (Def. 7). Then, we will prove that the relation is both sparse (Def. 8) and sampleable (Def. 9).

Our relation is similar to the relation of Canetti et al. [CCH+19], however we use symmetric key encryption and have the additional requirement that the input $inp^{(i)}$ is constructed using the value $inp_1^{(i)}$ in the statement, and the value $inp_2^{(i)}$ in the witness (which was committed to in the statement via $\mathsf{com\text{-}inp}_2^{(i)}$). Because of this requirement on the input $inp^{(i)}$, we also encrypt the opening $(inp_2^{(i)}\ ;\ \gamma^{(i)})$ to the commitment $\mathsf{com\text{-}inp}_2^{(i)}$ in the third round to retain sampleability of the relation. The shape of $inp^{(i)}$ is function dependent, and therefore we defer this check to an external predicate $\mathtt{Check}_{\mathcal{F}}$. Moreover, the predicate $\mathtt{Check}$ validates the commitments, the encryptions, and the output of the function $\mathcal{F}$.

$$
\begin{gathered}
R^{\mathcal{F}}_{\{key^{(i)}\}_{i\in[m]}} = \{((\{\mathsf{com\text{-}key}^{(i)}\}_{i\in[m]}, x = (\mathcal{F}, \{inp_1^{(i)}, \mathsf{com\text{-}inp}_2^{(i)}, y^{(i)}\}_{i\in[m]}),\\
\{c^{(i)}\}_{i\in[m]}), e)\ :\ x \notin \mathcal{L}, \mathtt{Check}(\{\mathsf{com\text{-}key}^{(i)}\}_{i\in[m]}x, \{c^{(i)}\}_{i\in[m]}, e, \{z^{(i)}\}_{e[i]=1}) = 1\\
\text{and } \mathtt{Check}_{\mathcal{F}}(inp_1^{(i)}, inp_2^{(i)}, inp^{(i)}) = 1 \text{ for at least } 8m/9 \text{ of } i \in [m] \text{ where}\\
(inp^{(i)}, inp_2^{(i)}, \gamma^{(i)}, \rho^{(i)}) = \mathsf{Dec}(key^{(i)}, c^{(i)})\}
\end{gathered}
$$

**Lemma 2.** *$R^{\mathcal{F}}_{\{key^{(i)}\}_{i\in[m]}}$ is sparse (Def. 8) and non-uniformly efficiently sampleable (Def. 9) for every key in support of key $\leftarrow \mathsf{KeyGen}(1^\lambda)$*

*Proof.* Our proof is nearly identical to the proof of Lemma 7.5 by Canetti et al. [CCH+18]. $R^{\mathcal{F}}_{\{key^{(i)}\}_{i\in[m]}}$ is sparse due to the soundness of $\Sigma$. For any statement $x \notin \mathcal{L}$, an adversary has probability at most $(8/3)(2/3)^{m/9} + \mathsf{negl}(\lambda)$ of providing a verifying proof. Therefore, given this statement $x \notin \mathcal{L}$, for every first message $\{c^{(i)}\}_{i\in[m]}$ we know there are as many second messages $e$ such that the relation is satisfied. To prove efficient sampleability, note that one can compute $(inp^{(i)}, inp_2^{(i)}, \gamma^{(i)}, \rho^{(i)}) = \mathsf{Dec}(key^{(i)}, c^{(i)})$ for each $i \in [m]$ given $x = (\mathcal{F}, \{inp_1^{(i)}, \mathsf{com\text{-}inp}_2^{(i)}, y^{(i)}\}_{i\in[m]}), \{c^{(i)}\}$, and $\{key^{(i)}\}_{i\in[m]}$. Then compute $\mathsf{com\text{-}key}^{(i)} = \mathtt{COM}(key^{(i)}; \rho^{(i)})$ for $i \in [m]$. Finally, choose $e$ such that $((\{\mathsf{com\text{-}key}^{(i)}\}_{i\in[m]}, \mathcal{F}, \{inp_1^{(i)}, \mathsf{com\text{-}inp}_2^{(i)}, y^{(i)}\}_{i\in[m]}), \{c^{(i)}\}_{i\in[m]}), e) \in R^{\mathcal{F}}_{\{key^{(i)}\}_{i\in[m]}}$ $\square$

**Collapsing the Rounds.** Now we give our non-interactive cut-and-choose based proof system $\Sigma_{NIP}$ for the same language $\mathcal{L}$ by collapsing the rounds of $\Sigma$ via CIH family $\mathcal{H}$. The statement and witness are the same. The set-up algorithm is nearly identical, except that it now samples and includes the CIH $H_k \in \mathcal{H}$ in the CRS. This hash family is chosen based on the relation. More specifically, the hash function's description depends on the circuit's size representing the relation.

Recall, that the hash function needs to output a binary string with $m/3$ ones and $2m/3$ zeros on expectation. We outline and analyze below how this can be accomplished with rejection sampling. Let the output of $H_k$ be a random binary string of length $N$. Now, we modify the code of $H_k$ as follows.

**The Algorithm**

1. Treat the $N$-bit string as $\frac{N}{2}$ independent bit pairs.
2. Ignore the *bad* pairs that are $(0,0)$ and keep only the *good* pairs that are $(1,1), (1,0), (0,1)$.
3. From these good pairs, produce output bits according to the following rule:

- $(1, 1) \rightarrow 1$
- $(0, 1)$ or $(1, 0) \rightarrow 0$

4. Stop once $m$ outputs have been collected.

**Analysis** For any single pair, the probability that it is good is $\frac{3}{4}$ and the probability it is bad is $\frac{1}{4}$. Hence the expected number of good pairs in $\frac{N}{2}$ draws is

$$\frac{N}{2} \times \frac{3}{4} = \frac{3N}{8}$$

Now using the Chernoff bound we can estimate that if $N \geq \frac{8m}{3}$, the number of good pairs is $m$ with high probability. More specifically,

$$\Pr[\#\text{ Good pairs among } \frac{N}{2} \text{ pairs} \leq (1-\delta) \cdot \frac{3N}{8}] \leq e^{-c\delta^2 \cdot N}$$

for some absolute constant $c > 0$.

Therefore, if $N$ is set as $N > \frac{8m}{3N(1-\delta)}$, the probability of failure is exponentially small. We treat this algorithm as implicit, and instead assume that the hash function outputs strings of length $m$ where $m/3$ bits are one.

We give a formal description of the round-collapsed proof system in Fig. 4.

**Proof System:**

$\Sigma_{NIP} = \langle \mathsf{P}(x, w\ ;\ \{key^{(i)}, \rho^{(i)}\}_{i\in[m]}), \mathsf{V}(x)\ ;\ (H_k, \{\textsf{com-key}^{(i)}\}_{i\in[m]}) \rangle$

1. $\mathsf{P}$ : Compute $c^{(i)} = \mathcal{E}.\mathsf{Enc}(key^{(i)}, (inp^{(i)}, inp_2^{(i)}, \gamma^{(i)}); \zeta^{(i)})$ for $i \in [m]$. Compute $e = H_k(\mathcal{F}, \{\textsf{com-key}^{(i)}, inp_1^{(i)}, \textsf{com-inp}_2^{(i)}, y^{(i)}, c^{(i)}\}_{i\in[m]})$. If there are more than $m/3$ 1s in $e$, abort. Let $z^{(i)} = (inp^{(i)}, inp_2^{(i)}, \gamma^{(i)}, \zeta^{(i)}, key^{(i)}, \rho^{(i)})$ such that $e[i] = 1$. Send $(\{c^{(i)}\}_{i\in[m]}, e, \{z^{(i)}\}_{e[i]=1})$ to $\mathsf{V}$.
2. $\mathsf{V}$ (verification): Upon receipt of $(\{c^{(i)}\}_{i\in[m]}, e, \{z^{(i)}\}_{e[i]=1})$: If $H_k(\mathcal{F}, \{\textsf{com-key}^{(i)}, inp_1^{(i)}, \textsf{com-inp}_2^{(i)}, y^{(i)}, c^{(i)}\}_{i\in[m]}) \neq e$ or $\texttt{Check}(\{\textsf{com-key}^{(i)}\}_{i\in[m]}, x, \{c^{(i)}\}_{i\in[m]}, e, \{z^{(i)}\}_{e[i]=1}) \neq 1$ output 0. If $\texttt{Check}_{\mathcal{F}}(inp_1^{(i)}, inp_2^{(i)}, inp^{(i)}) = 0$ for any $(inp^{(i)}, inp_2^{(i)}, \gamma^{(i)}, \zeta^{(i)}) = z^{(i)}$ output 0. Else output 1.

**Figure 4:** One Round Proof of Function Output via Cut-and-Choose

Next, we define the predicate `Check` (Fig. 5) used by $\Sigma_{NIP}$ to verify that the commitments, the encryptions, and the function output are all computed correctly for the opened values of the proof $\pi = (\{c^{(i)}\}_{i\in[m]}, e, \{z^{(i)}\}_{c^{(i)}=1})$.

**Predicate:** $\mathtt{Check}(\{\textsf{com-key}^{(i)}\}_{i\in[m]}x, \{c^{(i)}\}_{i\in[m]}, e, \{z^{(i)}\}_{e[i]=1})$

1. For $i$ such that $e[i] = 1$:
   (a) Parse $z^{(i)} = (inp^{(i)}, inp_2^{(i)}, \gamma^{(i)}, \zeta^{(i)}, key^{(i)}, \rho^{(i)})$.
   (b) If $c^{(i)} \neq \mathcal{E}.\mathsf{Enc}(key^{(i)}, (inp^{(i)}, inp_2^{(i)}, \gamma^{(i)}, \rho^{(i)}) \ ; \ \zeta^{(i)})$, or $\textsf{com-key}^{(i)} \neq \mathtt{COM}(key^{(i)} \ ; \ \rho^{(i)})$, or $\textsf{com-inp}_2^{(i)} \neq \mathtt{COM}(inp_2^{(i)}; \gamma^{(i)})$, or $y^{(i)} \neq \mathcal{F}(inp^{(i)})$ for $y^{(i)} \in x$, output 0. Else Output 1

**Figure 5:** Input Check Predicate of $\Sigma_{NIP}$

Finally, we prove that our round collapsed proof system $\Sigma_{NIP}$ is sound.

**Lemma 3.** *If* $\mathtt{COM}$ *is a statistically binding commitment scheme,* $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *is a symmetric key encryption scheme with perfect decryption correctness, and* $H$ *is a correlation intractable hash function with respect to relation* $R^{\mathcal{F}}_{\{key^{(i)}\}_{i\in[m]}}$*, then* $\Sigma_{NIP}$ *is an adaptive sound proof system.*

*Proof.* By the proof of Lem. 1, we know that the only way an adversary can gain a non-negligible advantage in breaking soundness is through the use of $H$ to generate the challenge message $e$.

Towards a contradiction, assume that $\Sigma_{NIP}$ is not sound. Then $\exists$ an adversary $\mathcal{A}(H_k, \{\textsf{com-key}^{(i)}, key^{(i)}, \rho^{(i)}\}_{i\in[m]})$ that can construct a proof $\pi = (\{c^{(i)}\}_{i\in[m]}, e, \{z^{(i)}\}_{e[i]=1})$ for statement $x$ such that $x \notin \mathcal{L}$ but $\mathsf{V}(\{\textsf{com-key}^{(i)}\}_{i\in[m]}, x, \pi) = 1$ with non-negligible probability $p(\lambda)$. Since we prove adaptive soundness, $\mathcal{A}$ will receive the public parameters as input first, then choose the statement $x$. We will then construct an adversary $\mathcal{B}$ such that $\mathcal{B}$ can violate the correlation intractability of $\mathcal{H}$. Define $\mathcal{B}$ as follows:

$\underline{\mathcal{B}(H_k):}$

1. Compute $key^{(i)} \leftarrow \mathsf{KeyGen}(1^\lambda)$ and $\rho^{(i)} \xleftarrow{\$} \{0,1\}^\lambda$ for $i \in [m]$
2. Compute $\textsf{com-key}^{(i)} = \mathtt{COM}(key^{(i)}, \rho^{(i)})$ for $i \in [m]$
3. Activate $\mathcal{A}(H_k, \{\textsf{com-key}^{(i)}, key^{(i)}, \rho^{(i)}\}_{i\in[m]}) = (x, \pi = (\{c^{(i)}\}_{i\in[m]}, e, \{z^{(i)}\}_{e[i]=1}))$
4. Output $(\mathcal{F}, \{\textsf{com-key}^{(i)}, inp_1^{(i)}, \textsf{com-inp}_2^{(i)}, y^{(i)}, c^{(i)}\}_{i\in[m]})$ as the hash function input

We know that $\mathsf{V}(x,\pi) = 1$ for $x \notin \mathcal{L}$ with non-negligible probability. For this to be true, it must be the case that $\mathtt{Check}(\mathsf{pk}, x, \{c^{(i)}\}_{i\in[m]}, e, \{z^{(i)}\}_{e[i]=1}) = 1$ and $\mathtt{Check}_{\mathcal{F}}(inp_1^{(i)}, inp_2^{(i)}, inp^{(i)}) = 1$ for at least $8m/9$ of $i \in [m]$. This is exactly the requirements of the relation $R^{\mathcal{F}}_{\{key^{(i)}\}_{i\in[m]}}$. Therefore,

$$\Pr_{\substack{H_k \leftarrow \mathcal{H} \\ \mathcal{B}(H_k) \to (\{\textsf{com-key}^{(i)}\}_{i\in[m]}, x, \{c^{(i)}\}_{i\in[m]})}} \left[((\{\textsf{com-key}^{(i)}\}_{i\in[m]}, x, \{c^{(i)}\}_{i\in[m]}), e) \in R^{\mathcal{F}}_{\{key^{(i)}\}_{i\in[m]}}\right]$$

$$= \Pr_{\mathcal{A}(H_k, \{\textsf{com-key}^{(i)}, key^{(i)}, \rho^{(i)}\}_{i\in[m]}) \to (x,\pi)} \left[x \notin \mathcal{L} \text{ and } \mathsf{V}(x,\pi) = 1\right] = p(\lambda)$$

for non-negligible $p(\lambda)$. Thus we have found an adversary that violates the correlation intractability of $H$ with non-negligible probability, and $\Sigma_{NIP}$ is a sound proof system. □

**Towards the Plain Model.** Finally we give our protocol for CRS generation between the prover and verifier. In this CRS generation, the verifier chooses the CIH that the prover must use (as is done in [KRR17]). We specify the protocol with four algorithms $\Sigma_{NIP}^{\mathsf{plain}} = \langle \mathsf{P} = (\mathsf{P}_0, \mathsf{P}_1), \mathsf{V} = (\mathsf{V}_0, \mathsf{V}_1)\rangle$. The randomized algorithm $\mathsf{P}_0$ takes as input the security parameter $\lambda$ outputs a string $\mathsf{com\text{-}crs\text{-}prvr}$ and auxiliary information $crs\text{-}prvr$. The randomized algorithm $\mathsf{V}_0$ takes as input the security parameter and outputs a string $crs\text{-}vrfr$. The randomized algorithm $\mathsf{P}_1$ takes as input $crs\text{-}prvr, crs\text{-}vrfr, x, w$ outputs a proof $\pi$ and the **CRS** The deterministic algorithm $\mathsf{V}_1$ takes as input the $\pi$, **CRS** and outputs a bit 1. In this CRS generation, $\mathsf{P}$ begins $M > m$ sessions of coin flipping, and $\mathsf{V}$ uses the string $sel$ to determine which sessions to continue with. This is necessary later, in the proof of security of our OT protocol, to allow us to rewind and set the CRS. We present a concrete protocol, $\Sigma_{NIP}^{\mathsf{plain}}$, in Fig. 6.

**Lemma 4.** *If* $\mathtt{COM}$ *is a statistically binding commitment scheme and* $\Sigma_{NIP}$ *is an adaptive sound proof system, then* $\Sigma_{NIP}^{\mathsf{plain}}$ *is an adaptive sound proof system.*

*Proof.* By the proof of Lem. 3, we know that the only way an adversary can gain a non-negligible advantage in breaking soundness is through the generation of the CRS.

Towards a contradiction, assume that $\Sigma_{NIP}^{\mathsf{plain}}$ is not an adaptive sound proof system. Then there exists an adversary $\mathcal{A}(x, w)$ that can construct a proof $\pi = (\{c^{(i)}\}_{i\in[m]}, e, \{z^{(i)}\}_{e[i]=1})$ for statement $x$ such that $x \notin \mathcal{L}$ but $\mathsf{V}(\{\mathsf{com\text{-}key}^{(i)}\}_{i\in[m]}, x, \pi) = 1$ with non-negligible probability $p(\lambda)$.

Because $\mathtt{COM}$ is a statistically binding commitment scheme, we know that $\mathcal{A}$ cannot succeed by opening $\mathsf{com\text{-}r\text{-}key}^{(i)}$ to anything other than $r\text{-}key_P^{(i)}$.

Then, the only way for $\mathcal{A}$ to succeed is for cut-and-choose to fail. That is, if the $m/3$ opened commitments $\mathsf{com\text{-}key}^{(i)}$ open to the correct $key^{(i)}, \rho^{(i)})$, but there are less than $8m/9$ honest commitments in the CRS.

By the proof of Lem. 1, we know that the probability this occurs is at most $(8/3)(2/3)^{m/9}$ which is negligible due to our choice of $m = O(\lambda)$

□

## 5 Four-Round Malicious OT from Semi-Honest OT

In this section, we present our compiler that transforms any 2-round semi-honest oblivious transfer to a round-optimal simulatable oblivious transfer protocol.

### 5.1 Protocol Description

Let $\Pi^{\mathsf{sh}} = (\mathtt{OTR}, \mathtt{OTS}, \mathtt{OTD})$ be a 2 round semi-honest OT protocol. We transform $\Pi^{\mathsf{sh}}$ into a four-round OT protocol $\Pi$ in Fig. 7. Here we provide a high-level description of our protocol.
**Round 1:**

1. *The proof system for receiver*: The receiver executes the $\mathsf{P}_0$ algorithm of the $\Sigma_{NIP}^{\mathsf{plain}}$ protocol. Recall that this is the first step of the coin flipping for the generation of the CRS.

2. *Inputs to OT for receiver*: The receiver also commits to $m$ independently sampled random strings (denoted $r\text{-}ot1_{\mathsf{R}}^{(i)}$) that will determine the input to the first message function $\mathtt{OTR}$ for each session of a semi-honest OT protocol.

**Prover**$(x, w)$

$x = (\mathcal{F}, \{inp_1^{(i)}, \textsf{com-inp}_2^{(i)}, y^{(i)}\}_{i\in[m]})$

$w = (\{inp^{(i)}, inp_2^{(i)}, \gamma^{(i)}\}_{i\in[m]})$

**Verifier**$(x)$

$x = (\mathcal{F}, \{inp_1^{(i)}, \textsf{com-inp}_2^{(i)}, y^{(i)}\}_{i\in[m]})$

---

$\underline{\mathsf{P}_0(\lambda)}$ :

- For $i \in [M]$
  - $r\text{-}key_P^{(i)} = \leftarrow \{0,1\}^\lambda$
  - $\textsf{com-r-key}^{(i)} \leftarrow \texttt{COM}(r\text{-}key_P^{(i)})$
- $\textsf{com-crs-prvr} = \{\textsf{com-r-key}^{(i)}\}_{i\in[M]}$
- $crs\text{-}prvr = \{r\text{-}key_P^{(i)}\}_{i\in[M]}$

$\xrightarrow{\textsf{com-crs-prvr}}$

$\underline{\mathsf{V}_0(\textsf{com-crs-prvr})}$

- $sel \leftarrow \{0,1\}^M$ such that $\sum_i sel[i] = m$
- $r\text{-}key_V^{(i)} \leftarrow \{0,1\}^\lambda$ for $i \in [m]$
- $H_k \leftarrow \mathcal{H}$
- $crs\text{-}vrfr = (H_k, sel, \{r\text{-}key_V^{(i)}\}_{i\in[m]})$

$\xleftarrow{crs\text{-}vrfr}$

$\underline{\mathsf{P}_1(crs\text{-}prvr, crs\text{-}vrfr, x, w)}$

- $r\text{-}key^{(i)} \| \rho^{(i)} = r\text{-}key_P^{(j)} \oplus r\text{-}key_V^{(i)}$ where $j$ is the $i$-th index where $sel[j] = 1$
- $key^{(i)} \leftarrow \textsf{KeyGen}(r\text{-}key^{(i)})$
- $\textsf{com-key}^{(i)} = \texttt{COM}(key^{(i)} \ ; \ \rho^{(i)})$
- $\mathbf{CRS} = \{\textsf{com-key}^{(i)}\}_{i\in[m]}$
- $c^{(i)} = \mathcal{E}.\textsf{Enc}(key^{(i)}, (inp^{(i)}, inp_2^{(i)}, \gamma^{(i)}); \zeta^{(i)})$ for $i \in [m]$
- $e = H_k(\mathcal{F}, \{\textsf{com-key}^{(i)}, inp_1^{(i)}, \textsf{com-inp}_2^{(i)}, y^{(i)}, c^{(i)}\}_{i\in[m]})$
- If there are more than $m/3$ 1s in $e$, abort
- $z^{(i)} = (inp^{(i)}, inp_2^{(i)}, \gamma^{(i)}, \zeta^{(i)}, key^{(i)}, \rho^{(i)})$ such that $e[i] = 1$
- $\pi = (x, w, c^{(i)}, e, z^{(i)})$

$\xrightarrow{\mathbf{CRS}, \pi}$

$\underline{\mathsf{V}_1(\mathbf{CRS}, \pi)}$

$H_k(\mathcal{F}, \{\textsf{com-key}^{(i)}, inp_1^{(i)}, \textsf{com-inp}_2^{(i)}, y^{(i)}, c^{(i)}\}_{i\in[m]}) = e$

- $\texttt{Check}(\{\textsf{com-key}^{(i)}\}_{i\in[m]}, x, \{c^{(i)}\}_{i\in[m]}, e, \{z^{(i)}\}_{e[i]=1})$
- $\texttt{Check}_{\mathcal{F}}(inp_1^{(i)}, inp_2^{(i)}, inp^{(i)})$ $\forall (inp^{(i)}, inp_2^{(i)}, \gamma^{(i)}, \zeta^{(i)}) = z^{(i)}$.
- $\forall key^{(i)}, \rho^{(i)} \in z^{(i)} \textsf{com-r-key}^{(i)} = \texttt{COM}_{\textsf{stat-hide}}(key^{(i)} \oplus r\text{-}key_V^{(i)})$
- If any checks fail output $0$, else $1$

**Figure 6:** $\Sigma_{NIP}^{\textsf{plain}}$ Proof system with CRS generation using coin-flipping

**Round 2:**

1. *The proof system for receiver*: The sender executes algorithm $\mathsf{V}_0$ of $\Sigma_{NIP}^{\mathsf{plain}}$. Recall that this algorithm samples randomness that will be used in the coin-flipping to determine the CRS for the receiver.

2. *The proof system for the sender*: Similar to the receiver in round 1, the sender executes $\mathsf{P}_0$ of $\Sigma_{NIP}^{\mathsf{plain}}$ to begin the generation of their own CRS.

3. *Input to OT for receiver* Next, for each session $i \in [m]$, the sender samples a random string (denoted $r\text{-}ot1_{\mathsf{S}}^{(i)}$). Looking ahead, the receiver input to the first round message function $\mathtt{OTR}$ of the semi-honest OT protocol in session $i$ will be determined by $r\text{-}ot1_{\mathsf{R}}^{(i)} \oplus r\text{-}ot1_{\mathsf{S}}^{(i)}$.

4. *Input to OT for sender*: The sender also commits to $m$ random strings (denoted $r\text{-}ot2^{(i)}$) that will be the sender input to the second message function $\mathtt{OTS}$ of the 2 round semi-honest OT protocol.

**Round 3:**

1. *Compute* $\mathrm{ot}_1$ *messages*: The receiver computes the first message of the OT protocol in each of the sessions. The input bit is the first bit of $r\text{-}ot1_{\mathsf{R}}^{(i)} \oplus r\text{-}ot1_{\mathsf{S}}^{(i)}$ and the randomness for the OT is determined by the rest of the bits.

2. *Execute the proof system*: The receiver then executes the $\mathsf{P}_1$ algorithm of $\Sigma_{NIP}^{\mathsf{plain}}$. Recall that this first computes the CRS for the proof system, and then computes a proof that each $\mathrm{ot}_1^{(i)}$ is computed correctly. In the proof system, the hash function determines which of the indices must be opened. These indices are denoted as $A$.

3. *Adjustment bits for encryption*: For each session in $[m] \setminus A$ the receiver sends an adjustment bit $d^{(i)} = b^{(i)} \oplus b$, where $b^{(i)}$ is the random bit used as input in session $i$ by the receiver. This informs the sender how to encrypt the input strings in the last round.

4. *Proof system for the sender*: The receiver runs the $\mathsf{V}_0$ algorithm of $\Sigma_{NIP}^{\mathsf{plain}}$ that computes the randomness that will be used to determine the CRS used by the sender's proof system.

**Round 4:**

1. *Verify the receiver's proofs*: The sender executes the $\mathsf{V}_1$ algorithm of $\Sigma_{NIP}^{\mathsf{plain}}$. This checks that the CRS for the proof system used by the receiver was computed correctly, the correct indices were opened, and the OT messages for the opened indices were computed correctly.

2. *Compute OT messages*: The sender computes the $\mathrm{ot}_2$ messages using the randomness they committed to in the second round using the $\mathtt{OTS}$ function of $\Pi^{\mathsf{sh}}$.

3. *Execute the proof system*: Similar to the receiver, the sender runs the algorithm $\mathsf{P}_1$ of $\Sigma_{NIP}^{\mathsf{plain}}$ to first compute the CRS for the sender and then prove that the $\mathrm{ot}_2$ messages are computed correctly. The indices opened by this proof are denoted as $B$. Let $Alive = [m] \setminus A \cup B$ be the unopened indices of $[m]$. These are the sessions that are still "alive" and will be used to finish the simulatable OT protocol.

4. *Encrypt shares of input for the main OT protocol*: The sender secret shares its input strings $s_0$ and $s_1$ obtaining $m/3$ shares $\{s_0^i\}_{i \in Alive}, \{s_1^i\}_{i \in Alive}$ such that each pair of shares can be assigned to a distinct alive session. Finally, the sender uses the

key $k^{(i)}_{0\oplus d^{(i)}}$ (resp. $k^{(i)}_{1\oplus d^{(i)}}$) to encrypt the share $s^i_0$ (resp. $s^i_1$) and sends the resulting ciphertexts $\mathsf{ct}_{0\oplus d^{(i)}}$ and $\mathsf{ct}_{1\oplus d^{(i)}}$ to the receiver. These keys are determined by the randomness used to compute $\mathrm{ot}_2$.

**Output Computation:**

1. *Verify sender proofs*: The receiver first executes the $\mathsf{V}_1$ algorithm of $\Sigma^{\mathsf{plain}}_{NIP}$. This checks that the CRS generated by the sender is correct, the correct indices are opened, and the OT messages for these indices were computed correctly.
2. *Output computation*: Next the receiver computes $k^{(i)}_{b^{(i)}}$ using $\mathtt{OTD}$ of $\Pi^{\mathsf{sh}}$, the output computation function of the semi-honest OT protocol. Finally, since $b^{(i)} = d^{(i)} \oplus b$, the receiver decrypts the ciphertexts $\mathsf{ct}^{(i)}_{b^{(i)}}$ using $k^{(i)}_{b^{(i)}}$ to get secret shares of $s_b$ denoted as $s^{(i)}_b$. Using any $\tau$ valid shares, where $\tau$ is the threshold of the secret sharing scheme, the receiver reconstructs the secret $s_b$.

We define the predicates $\mathtt{Check}_{\mathtt{OTR}}$ (Fig. 8) and $\mathtt{Check}_{\mathtt{OTS}}$ (Fig. 9) which are part of $\mathsf{V}_1$ algorithm run by the sender and the receiver respectively. These predicates (along with $\mathtt{Check}$) are used by the sender (resp. receiver) to verify the proof $\pi_{\mathsf{R}}$ (resp. $\pi_{\mathsf{S}}$)) in $\Sigma^{\mathsf{plain}}_{NIP}$. The theorem below states the security of our compiler.

**Theorem 1.** *Let* $(\mathtt{OTR}, \mathtt{OTS}, \mathtt{OTD})$ *be a semi-honest secure OT protocol,* $\mathtt{COM}$ *be a computationally hiding commitment scheme,* $\mathtt{COM}_{\mathsf{stat\text{-}hide}}$ *be a statistically hiding commitment scheme,* $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *be a CPA secure encryption scheme,* $\mathcal{H}$ *be a family of correlation intractable hash functions with approximate average case programmability, and* $(\mathtt{Share}, \mathtt{Reconstruct})$ *be a statistically private secret sharing scheme. Then for parameters* $m, M, \tau$*, such that* $m = O(\lambda), M = O(poly(m, \lambda)), \tau = 2m/9$*, the protocol* $\Pi = (S, R)$ *presented in Fig. 7 securely realizes the* $\mathcal{F}_{OT}$ *functionality.*

To prove our Thm. 1, we consider the case of a malicious receiver and the case of a malicious sender. We simulate the protocol in both cases, then prove through a series of hybrids that the simulation is indistinguishable from the real-world protocol $\Pi_{\mathsf{OT}}$.

## 5.2 Simulator for malicious receivers

In Fig. 10 we present the simulator with oracle access to a malicious receiver $R^*$.

Upon receiving the first round message from the malicious receiver the simulator plays as an honest sender and sends the corresponding second round messages to the receiver. Upon receiving the third round message from the receiver, which includes the $\mathrm{ot}^{(i)}_1$ messages, the CRS $\mathbf{CRS}_{\mathsf{R}}$, and the proof $\pi_{\mathsf{R}}$ that proves that the $\mathrm{ot}^{(i)}_1$ messages were computed correctly, the simulator first checks that the proof verifies. If not the simulator aborts. Recall that the proof includes $m/3$ indices (denoted $A$) for which the receiver sends the inputs to the $\mathrm{ot}^{(i)}_1$ messages.

The simulator then computes the fourth round message as an honest sender and learns the indices that will be selected for the sender's cut and choose (denoted $B$). Now the goal of the simulator is to extract the inputs to the $\mathrm{ot}^{(i)}_1$ messages sent by the receiver that are in the remaining "alive" sessions ($Alive = [m] \setminus (A \cup B)$).

To this end, the simulator starts a rewind thread, where the simulator rewinds the receiver to the beginning of round 2. In each iteration, the simulator first randomly samples a bit string $e_{\mathsf{R}} \xleftarrow{\$} \{0,1\}^\lambda$ such that $m/3$ of the bits are 1. Using the approximate average-case programmability (Def. 10) of the hash family $\mathcal{H}$, the simulator samples a hash function $H_k$ such that on the inputs used by the receiver in the main thread, the hash function $H$ outputs $e_{\mathsf{R}}$. The simulator then sends this hash function along with the rest of the round 2 message that is honestly computed to the malicious receiver. Upon receiving

**Sender**$(s_0, s_1)$ **Receiver**$(b)$

**Receiver:**

$(\textsf{com-crs-prvr}_{\mathsf{R}}, \textit{crs-prvr}_{\mathsf{R}}) \leftarrow \mathsf{P}_0(\lambda)$

$\textit{r-ot}1_{\mathsf{R}}^{(i)} \xleftarrow{\$} \{0,1\}^{\lambda}$ for $i \in [m]$

For $i \in [m]$ : $\textsf{com-r}_{\mathsf{R}}^{(i)} \leftarrow \texttt{COM}(\textit{r-ot}1_{\mathsf{R}}^{(i)}; \gamma_{\mathsf{R}}^{(i)})$

$\textit{msg}_1 = (\textsf{com-crs-prvr}_{\mathsf{R}}, \textsf{com-r}_{\mathsf{R}}^{(i)}\}_{i \in [m]})$

$\xleftarrow{\textit{msg}_1}$

**Sender:**

$\textit{crs-vrfr}_{\mathsf{R}} \leftarrow \mathsf{V}_0(\textsf{com-crs-prvr}_{\mathsf{R}})$

$(\textsf{com-crs-prvr}_{\mathsf{S}}, \textit{crs-prvr}_{\mathsf{S}}) \leftarrow \mathsf{P}_0(\lambda)$

For $i \in [m]$ :

- $\textit{r-ot}1_{\mathsf{S}}^{(i)} \xleftarrow{\$} \{0,1\}^{\lambda}, \textit{r-ot}2^{(i)} \xleftarrow{\$} \{0,1\}^{3\lambda}$
- $\textsf{com-r}_{\mathsf{S}}^{(i)} \leftarrow \texttt{COM}(\textit{r-ot}2^{(i)} \ ; \ \gamma_{\mathsf{S}}^{(i)})$

$\textit{msg}_2 = (\textit{crs-vrfr}_{\mathsf{R}}, \textsf{com-crs-prvr}_{\mathsf{S}},$
$\{\textit{r-ot}1_{\mathsf{S}}^{(i)}, \textsf{com-r}_{\mathsf{S}}^{(i)}\}_{i \in [m]})$

$\xrightarrow{\textit{msg}_2}$

**Receiver:**

$\textit{r-ot}1^{(i)} = \textit{r-ot}1_{\mathsf{R}}^{(i)} \oplus \textit{r-ot}1_{\mathsf{S}}^{(i)}$ for $i \in [m]$

$b^{(i)} = \textit{r-ot}1^{(i)}[0]$

$\text{ot}_1^{(i)} = \texttt{OTR}(b^{(i)} \ ; \ \textit{r-ot}1^{(i)})$ for $i \in [m]$

$x_{\mathsf{R}} = (\texttt{OTR}, \{\textit{r-ot}1_{\mathsf{S}}^{(i)}, \textsf{com-r}_{\mathsf{R}}^{(i)}, \text{ot}_1^{(i)}\}_{i \in [m]})$

$w_{\mathsf{R}} = (\{\textit{r-ot}1^{(i)}, \textit{r-ot}1_{\mathsf{R}}^{(i)}, \gamma_{\mathsf{R}}^{(i)}\}_{i \in [m]})$

$\mathbf{CRS}_{\mathsf{R}}, \pi_{\mathsf{R}} = \mathsf{P}_1(\textit{crs-prvr}_{\mathsf{R}}, \textit{crs-vrfr}_{\mathsf{R}}, x_{\mathsf{R}}, w_{\mathsf{R}})$

$\{d^{(i)} = b \oplus b^{(i)}\}_{i \in [m] \setminus A}; A = \{i\}_{e[i]=1},$

$\textit{crs-vrfr}_{\mathsf{S}} \leftarrow \mathsf{V}_0(\textsf{com-crs-prvr}_{\mathsf{S}})$

$\textit{msg}_3 = (\mathbf{CRS}_{\mathsf{R}}, \pi_{\mathsf{R}}, \textit{crs-vrfr}_{\mathsf{S}}, \{\text{ot}_1^{(i)}\}_{i \in [m]},$
$\{d^{(i)}\}_{i \in [m] \setminus A})$

$\xleftarrow{\textit{msg}_3}$

**Sender:**

Check $\mathsf{V}_1(\mathbf{CRS}_{\mathsf{R}}, \pi_{\mathsf{R}}) = 1,$ else abort

Parse $\textit{r-ot}2^{(i)} = k_0^{(i)} \| k_1^{(i)} \| \textit{r-ot}2^{*(i)}$

$\text{ot}_2^{(i)} = \texttt{OTS}(k_0^{(i)}, k_1^{(i)}, \text{ot}_1^{(i)} \ ; \ \textit{r-ot}2^{*(i)})$

$x_{\mathsf{S}} = (\texttt{OTS}, \{\text{ot}_1^{(i)}, \textsf{com-r}_{\mathsf{S}}^{(i)}, \text{ot}_2^{(i)}\}_{i \in [m]})$

$w_{\mathsf{S}} = (\{(k_0^{(i)}, k_1^{(i)}, \text{ot}_1^{(i)}, \textit{r-ot}2^{*(i)}), \textit{r-ot}2^{(i)},$
$\gamma_{\mathsf{S}}^{(i)}\}_{i \in [m] \setminus A})$

$\mathbf{CRS}_{\mathsf{S}}, \pi_{\mathsf{S}} \leftarrow \mathsf{P}_1(\textit{crs-prvr}_{\mathsf{S}}, \textit{crs-vrfr}_{\mathsf{S}}, x_{\mathsf{S}}, w_{\mathsf{S}})$

$B = \{i\}_{e_{\mathsf{S}}[i]=1}; \textit{Alive} = [m] \setminus (A \cup B)$

$(s_0^{(i)})_{i \in [|\textit{Alive}|]} \leftarrow \texttt{Share}(s_0, \tau),$

$(s_1^{(i)})_{i \in [|\textit{Alive}|]} \leftarrow \texttt{Share}(s_1, \tau)$

$\textsf{ct}_{0 \oplus d^{(i)}}^{(i)} = k_{0 \oplus d^{(i)}}^{(i)} \oplus s_0^{(i)}$ for $i \in \textit{Alive}$

$\textsf{ct}_{1 \oplus d^{(i)}}^{(i)} = k_{1 \oplus d^{(i)}}^{(i)} \oplus s_1^{(i)}$ for $i \in \textit{Alive}$

$\textit{msg}_4 = (\{\text{ot}_2^{(i)}\}_{i \in \textit{Alive}}, \mathbf{CRS}_{\mathsf{S}}, \pi_{\mathsf{S}},$
$\{\textsf{ct}_0^{(i)}, \textsf{ct}_1^{(i)}\}_{i \in \textit{Alive}})$

$\xrightarrow{\textit{msg}_4}$

**Receiver:**

Check $\mathsf{V}_1(\mathbf{CRS}_{\mathsf{S}}, \pi_{\mathsf{S}}) = 1,$ else abort

$k_b^{(i)} = \texttt{OTD}(b^{(i)}, \textit{r-ot}1^{(i)}, \text{ot}_2^{(i)})$ for $i \in \textit{Alive}$

$s_b^{(i)} = k_{b \oplus d^{(i)}}^{(i)} \oplus \textsf{ct}_{b \oplus d^{(i)}}^{(i)}$

$s_b = \texttt{Reconstruct}(\{s_b^{(i)}\})$

**Figure 7:** Four Round Oblivious Transfer Protocol Π

**Predicate:** $\texttt{Check}_{\texttt{OTR}}(inp_1, inp_2, inp)$

1. If $inp \neq inp_1 \oplus inp_2$ output 0. Else output 1.

**Figure 8:** Preducate for Checking Input to OTR

**Predicate:** $\texttt{Check}_{\texttt{OTS}}(inp_1, inp_2, inp)$

1. Parse $k_0 \| k_1 \| r_2 = inp_2$
2. If $inp = (k_0, k_1, inp_1, r_2)$ output 1, else output 0

**Figure 9:** Predicate for Checking Input to OTS

the openings of the $\mathrm{ot}_1^{(i)}$ messages that correspond to $e_{\mathsf{R}}[i] = 1$, the simulator stores each valid opening it receives for the $\mathrm{ot}_1^{(i)}$ messages that correspond to the sessions in *Alive*.

The simulator repeats this until it receives at least $2m/9$ openings from the sessions in *Alive*. Now the simulator exits the rewind thread and computes bits $\hat{b}^{(i)} = d^{(i)} \oplus b^{(i)}$. The input of the malicious receiver is then set as the bit $b$ that appears at least $m/9$ times in the extracted $\hat{b}^{(i)}$ bits. The simulator then sends this bit to the $\mathcal{F}_{OT}$ functionality and receives back the string $s_b$. It then samples an arbitrary string $s_{1-b} \xleftarrow{\$} \{0,1\}^\lambda$ and simulates the rest of the protocol as an honest sender would with the strings $s_b$ and $s_{1-b}$.

First, we prove that $\mathsf{Sim}_{\mathsf{R}^*}$ runs in polynomial time.

**Lemma 5.** $\mathsf{Sim}_{\mathsf{R}^*}$ *runs in expected time polynomial in* $\lambda$ *and* $m$

*Proof.* Note that outside of the rewind thread, all actions performed by $\mathsf{Sim}_{\mathsf{R}^*}$ in the main thread are polynomial-time. Further, all aborts in the main thread are at the same point and for the same reason that a real-world sender would abort. Assume that the simulator does not abort with probability $p \in (0,1)$ in the main thread. Within a rewind iteration note, the simulator only changes how the hash function $H_k^{\mathsf{R}}$ is sampled. Now, since $H_k^{\mathsf{R}}$ is sampled efficiently and is statistically indistinguishable from a randomly sampled hash function, each rewind iteration runs in polynomial time and does not abort with probability $p$. Thus we only need to present a bound on the number of rewinds that occur.

The goal of rewinding is to receive enough bits in *Alive*, specifically $2m/9$, such that the input bit $b$ can be reconstructed. Upon receiving an opening for an index $i \in Alive$ for the first time, if the opening is such that it computes the $\mathrm{ot}_1^{(i)}$ message in the main thread, then add the bit $b^{(i)}$ to *Bits*. Thus the number of rewinds corresponds to covering $2m/9$ indices of *Alive*. This is a variation of the coupon collector's problem. The coupon collector's problem asks, given $m$ items with equal probability of selection, how many selections with replacement must be performed before all $m$ have been chosen at least once. The coupon collector's problem models the worst-case scenario of our simulator, where at most only one new index is selected per rewind.

Consider the $N$ to be the number of rewinds required to gather $2m/9$ of the indices of *Alive*. Then we have $N = n_1 + \ldots n_{2m/9}$ where $n_i$ is the number of rewinds needed to obtain the $i$th new index. The probability of selecting the $i$th new index is $p_i = \frac{(2m/9)-i+1}{m}$.

**Main Thread:**

- Upon receiving $(\mathsf{com\text{-}crs\text{-}prvr_R}, \mathsf{com\text{-}r}_\mathsf{R}^{(i)}\}_{i\in[m]})$ from $\mathsf{R}^*$
  - Compute $crs\text{-}vrfr_\mathsf{R}, \mathsf{com\text{-}crs\text{-}prvr_S}, \{r\text{-}ot1_\mathsf{S}^{(i)}, \mathsf{com\text{-}r}_\mathsf{S}^{(i)}\}_{i\in[m]}$ as an honest sender would, and send to $\mathsf{R}^*$
- Upon receiving $(\mathbf{CRS}_\mathsf{R}, \pi_\mathsf{R}, crs\text{-}vrfr_\mathsf{S}, \{\mathrm{ot}_1^{(i)}\}_{i\in[m]}, \{d^{(i)}\}_{i\in[m]\setminus A})$ from $\mathsf{R}^*$
  - If $\mathsf{V}_1(\mathbf{CRS}_\mathsf{R}, \pi_\mathsf{R}) \neq 1$ abort
  - Set $Bits = \emptyset$, $ctr = 0$, and let $Alive := [m] \setminus (B \cup A)$

**Rewind Thread:**

- $\boxed{\text{Choose } e_\mathsf{R} \xleftarrow{\$} \{0,1\}^m, \text{ such that } e_\mathsf{R} \text{ contains } m/3 \text{ ones, and } H_k^\mathsf{R} \leftarrow \mathbf{Samp}((\mathtt{OTR}, \{\mathsf{com\text{-}key}_\mathsf{R}^{(i)}, r\text{-}ot1_{\mathsf{S}^{(i)}}, \mathsf{com\text{-}r}_\mathsf{R}^{(i)}, \mathrm{ot}_1^{(i)}, \{c_\mathsf{R}^{(i)}\}_{i\in[m]}), e_\mathsf{R}) \text{ where } \{c_\mathsf{R}^{(i)}\}_{i\in[m]} \in \pi_\mathsf{R}, \text{ and compute } crs\text{-}vrfr_\mathsf{R} \text{ using } H_k^\mathsf{R}}$
- Send $(crs\text{-}vrfr_\mathsf{R}, \mathsf{com\text{-}crs\text{-}prvr_S}, \{r\text{-}ot1_\mathsf{S}^{(i)}, \mathsf{com\text{-}r}_\mathsf{S}^{(i)}\}_{i\in[m]})$ to $\mathsf{R}^*$ and receive $(\mathbf{CRS}_\mathsf{R}, \pi_\mathsf{R}, crs\text{-}vrfr_\mathsf{S}, \{\mathrm{ot}_1^{(i)}\}_{i\in[m]}, \{d^{(i)}\}_{i\in[m]\setminus A}))$
  - If $\mathsf{V}_1(\mathbf{CRS}_\mathsf{R}, \pi_\mathsf{R}) \neq 1$ abort
  - For every $i \in Alive$ that was not observed in a previous rewind:
    - Parse $r\text{-}ot1^{(i)} = b^{(i)} \| r\text{-}ot1^{*(i)}$
    - If $\mathrm{ot}_1^{(i)} = \mathtt{OTR}(b^{(i)} \; ; \; r\text{-}ot1^{*(i)})$ add $b^{(i)}$ to $Bits$
  - $\boxed{\text{Set } ctr = ctr + 1 \text{ if } ctr = 2^\lambda \text{ abort, else if } |Bits| < 2m/9, \text{ go to beginning of rewind thread, else proceed}}$

  - For each $b^{(i)} \in Bits$ compute $\hat{b}^{(i)} = b^{(i)} \oplus d^{(i)}$
  - Let $b$ be the bit $\hat{b}^{(i)}$ that appears more than $\tau/2$ times.
  - $\boxed{\boxed{\text{Forward } b \text{ to } \mathcal{F}_{OT} \text{ obtaining } s_b \in \{0,1\}^\lambda. \text{ Sample } s_{1-b} \xleftarrow{\$} \{0,1\}^\lambda}}$
  - Simulate the rest of the protocol as an honest sender using $s_b$ and $s_{1-b}$

**Figure 10:** $\mathsf{Sim}_{\mathsf{R}^*}$: The Simulator of $\Pi$ for a Malicious Receiver $\mathsf{R}^*$. $\boxed{\mathbf{Hyb}_1}$, $\boxed{\boxed{\mathbf{Hyb}_2, \mathbf{Hyb}_3}}$

Therefore, we have that the expected value of $N$ is:

$$E(N) = E(n_1, \dots n_{2m/9}) = E(n_1) + \dots E(n_{2m/9}) = \frac{1}{p_1} + \dots + \frac{1}{p_{2m/9}}$$
$$= \frac{m}{2m/9} + \frac{m}{2m/9 - 1} + \dots \frac{m}{1}$$
$$= m(\frac{1}{2m/9} + \frac{1}{2m/9 - 1} + \dots + 1) = m \cdot \mathrm{H}_{2m/9}$$

Where $\mathrm{H}_{2m/9}$ is the $2m/9$th harmonic number, which be approximated to $\mathrm{H}_{2m/9} \approx \ln(2m/9) + \gamma + 1/(2n) + \sum_{k=1}^{\infty} \frac{B_{2k}}{2k(2m/9)^{2k}}$ for the Euler-Mascheroni constant $\gamma \approx 0.57722$ and Bernouli numbers $B_{2k}$. Therefore we need approximately $m\ln(2m/9)+m\gamma+O(1/m) = O(m\ln(m))$ rewinds to cover $2m/9$ of the indices in *Alive*. Since the receiver continues with probability $p$ in each rewind iteration, the expected number of rewinds is $O(m\ln(m))/p$.

We can thus bound the expected running time of the simulator as:

$$poly(\lambda, m) \cdot p \cdot O(m \ln m)/p = poly(\lambda, m) \cdot O(m \ln m) = poly(\lambda, m)$$

For our choice of $m = O(\lambda)$, this is a polynomial in the security parameter and this concludes our analysis.

□

Before proving indistinguishability, we present a helper lemma for the proof system $\Sigma_{NIP}$. Specifically, the proof system is defined with a CRS generated by a trusted party, however, our OT protocol takes two rounds to generate the CRS. First, we define what it means to break soundness in the context of our OT protocol. We prove that if $\Sigma_{NIP}$ is sound, then our generation of the CRS and subsequent computation of the proof is also sound.

**Definition 11.** Let $\mathsf{bad}_\mathsf{R} \subset$ *Alive* be the set of indices for which $\mathtt{Check}_{\mathtt{OTR}}(r\text{-}ot1_\mathsf{S}^{(i)}, r\text{-}ot1_\mathsf{R}^{(i)}, r\text{-}ot1^{(i)}) \neq 1$

**Lemma 6.** *If $\Sigma_{NIP}$ is a sound proof system, then $Pr[|\mathsf{bad}_\mathsf{R}| > m/9] \leq \mathsf{negl}(\lambda)$*

*Proof.* Towards a contradiction, assume that there exists an adversary $\mathsf{R}^*$ such that $\mathsf{Sim}_{\mathsf{R}^*}$ does not abort and $|\mathsf{bad}_\mathsf{R}| > m/9$. We can then construct a reduction $\mathcal{B}$ such that $\mathcal{B}$ can violate the soundness of $\Sigma_{NIP}$. Let $c$ be a constant and let $H_m$ be the $m$th harmonic number. Define $\mathcal{B}$ as follows:

$\mathcal{B}(\mathbf{CRS}_{\mathsf{R}}, \{key_{\mathsf{R}}^{(i)}, \rho_{\mathsf{R}}^{(i)}\}_{i\in[m]})$ :

1. Activate $\mathsf{R}^*(1^\lambda)$
2. Simulate as in $\mathsf{Sim}_{\mathsf{R}^*}$ until receiving the round 3 message $(\mathbf{CRS}_{\mathsf{R}}^*, \pi_{\mathsf{R}}, crs\text{-}vrfr_{\mathsf{S}}, \{\mathrm{ot}_1^{(i)}\}_{i\in[m]}, \{d^{(i)}\}_{i\in[m]\setminus A}))$.
3. **Rewind to set CRS**
   (a) Rewind to the beginning of round 2
   (b) For every $(key_{\mathsf{R}}^{*(i)}, \rho_{\mathsf{R}}^{*(i)}) \in z_{\mathsf{R}}^{(i)} \in \pi_{\mathsf{R}}$ that has not been observed, compute the corresponding $r\text{-}key_{\mathsf{R}}^{*(i)}$.
   (c) Set $r\text{-}key_{\mathsf{S}}^{(i)} = (key_{\mathsf{R}}^{(i)} \| \rho_{\mathsf{R}}^{(i)}) \oplus r\text{-}key_{\mathsf{R}}^{*(i)}$ and store $r\text{-}key_{\mathsf{S}}^{(i)}$ in $\mathsf{CRSKeys}$
   (d) If $|\mathsf{CRSKeys}| < m$ go to step (a).
4. Rewind to the beginning of Round 2, and use $\mathsf{CRSKeys}$ as $\{r\text{-}key_{\mathsf{S}}^{(i)}\}_{i\in[m]}$ to compute $crs\text{-}vrfr_{\mathsf{S}}$.
5. Simulate the rest of the protocol as in $\mathsf{Sim}_{\mathsf{R}^*}$, except abort if $ctr = c \cdot m \cdot H_m + 1$.
   (a) If during the rewind thread we have $|\mathsf{bad}_{\mathsf{R}}| > m/9$, output $(x, \pi_{\mathsf{R}})$ to the challenger
   (b) Else, abort.

We know that the reduction runs in strictly polynomial time, because, as we saw in the proof of Lemma 5, $m \cdot H_m$ is polynomial and therefore $c \cdot m \cdot H_m$ is polynomial. Further, by the coupon collectors problem, we have that

$$Pr[N \geq c \cdot m \cdot H_m] \leq \frac{1}{c}$$

where $N$ is the number of rewinds to collect *all* indices. Therefore, our reduction aborts with probability at most $1/c$.

Since $\mathtt{Check}_{\mathtt{OTR}}(r\text{-}ot1_{\mathsf{S}}^{(i)}, r\text{-}ot1_{\mathsf{R}}^{(i)}, r\text{-}ot1^{(i)}) \neq 1$ for more than $m/9$ of $i$, we know that $x \notin \mathcal{L}$. However, $\pi_{\mathsf{R}}$ must verify, else $\mathsf{Sim}_{\mathsf{R}^*}$ must have aborted. We have then constructed a reduction that breaks the soundness of $\Sigma_{NIP}$ with non-negligible probability, and $Pr[[|\mathsf{bad}| > m/9] \leq \mathsf{negl}(\lambda)$. □

Next we prove indistinguishability through a series of hybrids, beginning from the real-world protocol $\Pi_{\mathsf{OT}}$, making small changes until we reach the simulated protocol. Our hybrids are as follows:

- $\mathbf{Hyb}_0$ The real world protocol
- $\boxed{\mathbf{Hyb}_1}$ This is the same as $\mathbf{Hyb}_0$, except we rewind as in $\mathsf{Sim}_{\mathsf{R}^*}$
- $\boxed{\boxed{\mathbf{Hyb}_2}}$ This is the same as $\mathbf{Hyb}_1$, except that $\mathsf{ct}_{1-b^{(i)}}^{(i)}$ is computed as $\hat{k}_{1-b^{(i)}}^{(i)} \oplus s_{1-\hat{b}^{(i)}}^{(i)}$ for a randomly sampled $\hat{k}_{1-b^{(i)}}^{(i)}$
- $\boxed{\boxed{\mathbf{Hyb}_3}}$ This is the same as $\mathbf{Hyb}_2$, except that we sample $s_{1-b}$ randomly. This is exactly the simulation of the protocol for a malicious receiver $\mathsf{Sim}_{\mathsf{R}^*}$

**Lemma 7.** *If $\mathcal{H}$ has approximate average case programmability (Def. 10) and $\Sigma_{NIP}$ is a sound proof system then $\mathbf{Hyb}_0$ is indistinguishable from $\mathbf{Hyb}_1$*

*Proof.* Towards a contradiction assume that there exists an adversary $\mathsf{R}^*$ such that $\mathsf{R}^*$ can distinguish between $\mathbf{Hyb}_0$ and $\mathbf{Hyb}_1$ with non-negligible probability $p(\lambda)$

These hybrids are distinguishable only if $\mathsf{Sim}_{\mathsf{R}^*}$ aborts when $ctr = 2^\lambda$, or if the adversary can distinguish between $H_k \leftarrow \mathcal{H}$ and $H_k \leftarrow \mathbf{Samp}((\mathtt{OTR}, \{\mathsf{com\text{-}key}^{(i)}, r\text{-}ot1_{\mathsf{S}}^{(i)}, \mathsf{com\text{-}r}_{\mathsf{R}}^{(i)}, \mathrm{ot}_1^{(i)}, c_{\mathsf{R}}^{(i)}\}_{i\in[m]}), e)$.

First, we prove that $\mathsf{Sim}_{\mathsf{R}^*}$ aborts with negligible probability. We consider three events $\mathsf{bad}_1$, $\mathsf{bad}_2$, and $\mathsf{bad}_3$, defined as follows, which are the only events that would cause $ctr \geq 2^\lambda$:

- $\mathsf{bad}_1$ : $|Bits| < 2m/9$ after $\mathsf{poly}(\lambda)$ rewind attempts
- $\mathsf{bad}_2$ : The proof $\pi_{\mathsf{R}}$ verifies, but $\mathtt{Check}_{\mathtt{OTR}}(r\text{-}ot1_{\mathsf{S}}^{(i)}, r\text{-}ot1_{\mathsf{R}}^{(i)}, r\text{-}ot1^{(i)}) \neq 1$ for more than $m/9$ of the opened values in the rewind thread
- $\mathsf{bad}_3$ : $\mathsf{R}^*$ never responds in the rewind thread

We know that $\mathsf{bad}_1$ occurs with negligible probability since $\mathsf{Sim}_{\mathsf{R}^*}$ runs in expected polynomial time as shown in Lem. 5. Further, we know that $\mathsf{bad}_2$ happens with negligible probability by the proof of Lem. 6.

Lastly, assume that $\mathsf{bad}_3$ occurs with non-negligible probability. The only remaining difference between $\mathbf{Hyb}_1$ and $\mathbf{Hyb}_0$ is in the rewind thread where $H_k \leftarrow \mathbf{Samp}((\mathtt{OTR}, \{\mathsf{com\text{-}key}^{(i)}, r\text{-}ot1_{\mathsf{S}^{(i)}}, \mathsf{com\text{-}r}_{\mathsf{R}}^{(i)}, \mathrm{ot}_1^{(i)}, c_{\mathsf{R}}^{(i)}\}_{i\in[m]}), e)$ for random $e$ where $\{c^{(i)}\}_{i\in[m]} \in \pi_{\mathsf{R}}$. Therefore, $\mathsf{R}^*$ must be able to distinguish between $H_k \leftarrow \mathbf{Samp}((\mathtt{OTR}, \{\mathsf{com\text{-}key}^{(i)}, r\text{-}ot1_{\mathsf{S}^{(i)}}, \mathsf{com\text{-}r}_{\mathsf{R}}^{(i)}, \mathrm{ot}_1^{(i)}, c_{\mathsf{R}}^{(i)}\}_{i\in[m]}), e)$ and $H_k \leftarrow \mathcal{H}$.

We know that this is not possible, because by the definition of approximate average case programmability (Def. 10) the two distributions are statistically indistinguishable. Therefore event $\mathsf{bad}_3$ cannot occur with probability greater than $\mathsf{negl}(\lambda)$.

Thus we have that each $\mathsf{bad}_i$ occurs with probability at most $\mathsf{negl}(\lambda)$, and have shown that $\mathbf{Hyb}_0$ and $\mathbf{Hyb}_1$ are indistinguishable. □

**Lemma 8.** *If $(\mathtt{OTR}, \mathtt{OTS}, \mathtt{OTD})$ is a secure semi-honest oblivious transfer protocol (Def. 1), $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ is a CPA-secure encryption scheme, and $\mathtt{COM}$ is a computationally hiding commitment scheme, then $\mathbf{Hyb}_1$ is indistinguishable from $\mathbf{Hyb}_2$*

*Proof.* Recall the difference between $\mathbf{Hyb}_1$ and $\mathbf{Hyb}_2$ is that the ciphertexts are computed using a randomly sampled $\hat{k}_{1-b^{(i)}}^{(i)}$.

The proof proceeds by a series of hybrids. For an index $j \in [m]$, consider the hybrid $\mathbf{Hyb}_{1,j}$ where for all values $b^{(i)} \in Bits$ where $i \leq j$ the ciphertext $\mathsf{ct}_{1-b^{(i)}}^{(i)} = k_{1-b^{(i)}}^{(i)} \oplus s_{1-\hat{b}^{(i)}}^{(i)}$ and for $i > j$ we have $\mathsf{ct}_{1-b^{(i)}}^{(i)} = \hat{k}_{1-b^{(i)}}^{(i)} \oplus s_{1-\hat{b}^{(i)}}^{(i)}$ where $\hat{k}_{1-b^{(i)}}^{(i)} \xleftarrow{\$} \{0,1\}^\lambda$.

Note that $\mathbf{Hyb}_{1,0}$ is equivalent to $\mathbf{Hyb}_2$, since $\hat{k}_{1-b^{(i)}}^{(i)}$ is used for all $b^{(i)} \in Bits$ and $\mathbf{Hyb}_{1,m}$ is equivalent to $\mathbf{Hyb}_1$ since $k_{1-b^{(i)}}^{(i)}$ is used for all $b^{(i)} \in Bits$.

Suppose towards a contradiction that $\mathsf{R}^*$ can distinguish between $\mathbf{Hyb}_{1,j-1}$ and $\mathbf{Hyb}_{1,j}$ by determining that the value committed in $\mathsf{com\text{-}r}_{\mathsf{S}}$ was not used to compute the key $k_{1-b^{(j)}}^{(j)}$. We then construct a reduction $\mathcal{B}$ that can violate the hiding of $\mathtt{COM}$. Define $\mathcal{B}$ as follows:

$\underline{\mathcal{B}(1^\lambda):}$

```
1. Activate $\mathsf{R}^*$ and simulate as in $\mathbf{Hyb}_{1,j-1}$ up to round 1
2. Query the challenger with $(0, r\text{-}ot2^{(j)})$ and receive $\mathsf{com}^*$
3. Continue to simulate $\mathbf{Hyb}_{1,j-1}$, except:
    (a) Abort if $ctr = c \cdot m \cdot H_m + 1$
    (b) Use $\mathsf{com}^*$ in place of $\mathsf{com\text{-}r}_{\mathsf{S}}^{(j)}$
    (c) If $j \in A \cup B$, abort
4. Output whatever $\mathsf{R}^*$ outputs
```

$\mathcal{B}$ aborts when $j \in A \cup B$, however this only happens with probability 2/3. Further $\mathcal{B}$ aborts if $ctr = c \cdot m \cdot H_m + 1$, where $c$ is a constant and $H_m$ is the $m$th harmonic number, however this occurs with probability at most $1/c$. If $\mathsf{com}^*$ is a commitment to $r\text{-}ot2^{(j)}$, then this is exactly $\mathbf{Hyb}_{1,j}$, as the value committed is used to compute the key $k_{1-b^{(j)}}^{(j)}$. If instead $\mathsf{com}^*$ is a commitment to 0, then this is exactly $\mathbf{Hyb}_{1,j-1}$, as the value committed in $\mathsf{com}^*$ is independent of the key $\hat{k}_{1-b^{(j)}}^{(j)}$. Therefore we have found an adversary that violates the hiding of $\mathtt{COM}$ with non-negligible probability and have a contradiction.

By the same argument, $\mathsf{R}^*$ cannot distinguish through $\mathsf{com\text{-}key}_{\mathsf{S}}^{(j)}$ in $\mathbf{CRS}_{\mathsf{S}}$ or $\mathsf{com\text{-}r\text{-}key}_{\mathsf{S}}^{(j)}$ in $\mathsf{com\text{-}crs\text{-}prvr}_{\mathsf{S}}$, else we would have a nearly identical reduction.

Suppose instead that $\mathsf{R}^*$ can distinguish through $c_{\mathsf{S}}^{(j)} \in \pi_{\mathsf{S}}$. We can then construct a reduction $\mathcal{A}$ that violates the CPA security of the encryption scheme $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$. Define $\mathcal{A}$ as follows:

$\underline{\mathcal{A}(1^\lambda):}$

```
1. Activate $\mathsf{R}^*(1^\lambda)$ and simulate as in $\mathbf{Hyb}_{1,j-1}$ until the key $key^{(j)}$ is computed
2. Query the challenger with $(0, ((k_0^{(j)}, k_1^{(j)}, \mathrm{ot}_1^{(j)}, r\text{-}ot2^{*(j)}), r\text{-}ot2^{(j)}, \gamma_{\mathsf{S}}^{(j)}))$ and receive $c^*$
3. Continue simulating as in $\mathbf{Hyb}_{1,j-1}$, except use $c^*$ in place of $c_{\mathsf{S}}^*$ and abort if $ctr = c \cdot m \cdot H_m + 1$
4. Abort if $j \in A \cup B$
5. Output whatever $\mathsf{R}^*$ outputs
```

Again note that although $\mathcal{A}$ aborts when $j \in A \cup B$, this only happens with probability 2/3 and $ctr = c \cdot m \cdot H_m + 1$ with probability at most $1/c$. If $c^*$ is an encryption of $((k_0^{(j)}, k_1^{(j)}, \mathrm{ot}_1^{(j)}, r\text{-}ot2^{*(j)}), r\text{-}ot2^{(j)}, \gamma_{\mathsf{S}}^{(j)})$, then this is exactly $\mathbf{Hyb}_{1,j-1}$ as $k_{1-b^{(j)}}^{(j)}$ is encrypted. If $c^*$ is an encryption of 0, then this is exactly $\mathbf{Hyb}_{1,j}$, as the key $\hat{k}_{1-b^{(j)}}^{(j)}$ used to encrypt the ciphertext $\mathsf{ct}_{1-b^{(j)}}^{(j)}$ is independent from the value encrypted in $c^*$. Therefore we have found an adversary $\mathcal{A}$ that violates the CPA security of $\mathcal{E}$ and have reached our contradiction.

Now consider the event where $B$ (determined by $e_{\mathsf{S}}$) is chosen such that $j \notin A \cup B$ (that is, the $j$th session is not chosen to be opened) and the bit $b^{(j)}$ observed in the rewinding is in the set $Bits$. Note that if this is not true, then $\mathbf{Hyb}_{1,j-1}$ and $\mathbf{Hyb}_{1,j}$ are identical, as $\mathsf{ct}_{1-b^{(j)}}^{(j)}$ is not part of the final message and therefore not in the view of the adversary. This means an adversary can only distinguish between $\mathbf{Hyb}_{1,j-1}$ and $\mathbf{Hyb}_{1,j}$ when $j \notin A \cup B$ occurs. That is, for any distinguisher $\mathcal{D}^*$:

$$\begin{aligned}|Pr[\mathcal{D}^*(\mathbf{Hyb}_{1,j-1}(\lambda, s_0, s_1, b)) = 1] - Pr[\mathcal{D}^*(\mathbf{Hyb}_{1,j}(\lambda, s_0, s_1, b)) = 1]| \\ = |Pr[\mathcal{D}^*(\mathbf{Hyb}_{1,j-1}(\lambda, s_0, s_1, b)) = 1 \wedge j \notin B \cup A] \\ -Pr[\mathcal{D}^*(\mathbf{Hyb}_{1,j}(\lambda, s_0, s_1, b)) = 1 \wedge j \notin B \cup A]|\end{aligned}$$

Towards a contradiction, assume that there exists an index $j \in [m]$ and a PPT adversary $\mathsf{R}^*$ such that $\mathsf{R}^*$ can distinguish between $\mathbf{Hyb}_{1,j-1}$ and $\mathbf{Hyb}_{1,j}$ with non-negligible probability $p(\lambda)$. We construct an adversary $\mathsf{R}'$ such that $\mathsf{R}'$ breaks privacy of the semi-honest two-round OT protocol $(\mathtt{OTR}, \mathtt{OTS}, \mathtt{OTD})$.

Define $\mathsf{R}'$ as follows:

```
R'(r-ot1^{*(j)}):
1. Activate $\mathsf{R}^*$ and execute $\mathbf{Hyb}_{1,j-1}$ honestly until receiving $(\mathbf{CRS}_{\mathsf{R}}, \pi_{\mathsf{R}}, crs\text{-}vrfr_{\mathsf{S}}, \{\mathrm{ot}_1^{(i)}\}_{i\in[m]}, \{d^{(i)}\}_{i\in[m]\setminus A})$
2. If $j \in A$: abort.
3. Else, in the rewind thread:
   (a) Sample a random $e_{\mathsf{R}}$. If $e_{\mathsf{R}}[j] = 1$, continue, else sample a different $e_{\mathsf{R}}$.
   (b) Compute $H_k^{\mathsf{R}} \leftarrow \mathbf{Samp}((\mathtt{OTR}, \{\mathsf{com\text{-}key}^{(i)}, r\text{-}ot1_{\mathsf{S}}^{(i)}, \mathsf{com\text{-}r}_{\mathsf{R}}^{(i)}, \mathrm{ot}_1^{(i)}, c_{\mathsf{R}}^{(i)}\}_{i\in[m]}), e_{\mathsf{R}})$ where $\{c_{\mathsf{R}}^{(i)}\}_{i\in[m]} \in \pi_{\mathsf{R}}$ and use $H_k^{\mathsf{R}}$ when computing $crs\text{-}vrfr_{\mathsf{S}}$.
   (c) Receive the opening $r\text{-}ot1_{\mathsf{R}}^{(j)}$ to $\mathsf{com\text{-}r}_{\mathsf{R}}^{(j)}$
   (d) Sample a random $e_{\mathsf{R}}$ such that $e_{\mathsf{R}}[j] = 0$. If not, sample again. Rewind and compute $H_k^{\mathsf{R}} \leftarrow \mathbf{Samp}((\mathtt{OTR}, \{\mathsf{com\text{-}key}^{(i)}, r\text{-}ot1_{\mathsf{S}}^{(i)}, \mathsf{com\text{-}r}_{\mathsf{R}}^{(i)}, \mathrm{ot}_1^{(i)}, c_{\mathsf{R}}^{(i)}\}_{i\in[m]}), e_{\mathsf{R}})$ where $\{c^{(i)}\}_{i\in[m]} \in \pi_{\mathsf{R}}$ and use $H_k^{\mathsf{R}}$ when computing $crs\text{-}vrfr_{\mathsf{S}}$.
   (e) Use $r\text{-}ot1_{\mathsf{S}}^{*(j)} = r\text{-}ot1_{\mathsf{R}}^{(j)} \oplus r\text{-}ot1^{*(j)}$
   (f) Upon receiving $(\mathbf{CRS}_{\mathsf{R}}, \pi_{\mathsf{R}}, crs\text{-}vrfr_{\mathsf{S}}, \{\mathrm{ot}_1^{(i)}\}_{i\in[m]}, \{d^{(i)}\}_{i\in[m]\setminus A})$
   (g) Use this thread as the main thread and continue rewinding as in $\mathbf{Hyb}_{1,j-1}$, but abort if $ctr = c \cdot m \cdot H_m + 1$
4. Forward $\mathrm{ot}_1^{(j)}$ to the challenger and receive $\mathrm{ot}_2^*$
5. Compute $k^* = \mathtt{OTD}(b^{(j)}, r\text{-}ot1^{(j)}, \mathrm{ot}_2^*)$, where $b^{(j)}$ is learned through rewinding as in $\mathbf{Hyb}_{1,j-1}$
6. If $j \in A \cup B$ abort
7. Compute $\mathsf{ct}_{1-b^{(j)}}^{(j)} = k^* \oplus s_{1-\hat{b}^{(j)}}^{(j)}$ and send the final message to $\mathsf{R}^*$
8. Output whatever $\mathsf{R}^*$ outputs
```

In the first rewind, $\mathsf{R}'$ is able to force $\mathsf{R}^*$ to use $r\text{-}ot1^{*(j)}$ to compute $\mathrm{ot}_1^{(j)}$, which is the randomness expected by the challenger. If $j \in B \cup A$, then $\mathsf{R}'$ aborts. However, this only happens with probability 2/3.

In the case where $j \in \mathit{Alive}$ (that is, $j \notin A \cup B$), then the string $k^*$ is either $k_{1-b^{(j)}}^{(j)}$ or random $\hat{k}_{1-b^{(j)}}^{(j)}$ by the definition of semi-honest OT (Def. 1). Therefore, $\mathsf{R}'$ perfectly imitates $\mathbf{Hyb}_{1,j-1}$ in the case where $k^* = k^{(j)}_{1-b^{(j)}}$ and perfectly imitates $\mathbf{Hyb}_{1,j}$ in the case where $k^* = \hat{k}^{(j)}_{1-b^{(j)}}$ and shares the same advantage.

So we have constructed a receiver $\mathsf{R}'$ that can violate the privacy of semi-honest OT with non-negligible probability $\frac{1}{3} \cdot p(\lambda)$. This contradicts our assumption that $(\mathtt{OTR}, \mathtt{OTS}, \mathtt{OTD})$ is a semi-honest secure OT protocol, and therefore $\mathbf{Hyb}_{1,j-1}$ and $\mathbf{Hyb}_{1,j}$ are indistinguishable.

Since $\mathbf{Hyb}_{1,0}$ is equivalent to $\mathbf{Hyb}_2$ and $\mathbf{Hyb}_{1,m}$ is equivalent to $\mathbf{Hyb}_1$, and we have shown that adjacent hybrids $\mathbf{Hyb}_{1,j-1}$ and $\mathbf{Hyb}_{1,j}$ are indistinguishable, we know that $\mathbf{Hyb}_2$ is indistinguishable from $\mathbf{Hyb}_1$. □

**Lemma 9.** *If* $(\mathtt{Share}, \mathtt{Reconstruct})$ *is a statistically private secret sharing scheme, then* $\mathbf{Hyb}_2$ *is indistinguishable from* $\mathbf{Hyb}_3$

*Proof.* The two hybrids only differ in how the ciphertexts $\mathsf{ct}^{(i)}_{1-b^{(i)}}$ encrypting $s_{1-b}$ are computed for the last message. Now we consider two cases:

1. Both 0 and 1 appear more than $\tau/2$ times among the $\hat{b}^{(i)}$

2. Bit $b$ appears more than $\tau/2$ times among the $\hat{b^{(i)}}$, and bit $1-b$ appears at most $\tau/2$ times

**Case 1** We show that in this case, $\mathsf{R}^*$ cannot reconstruct either $s_0$ or $s_1$. This proves that the two hybrids are indistinguishable, as $\mathsf{R}^*$ does not learn either string.

Assume that 0 appears $\tau/2 + n_0$ times and 1 appears $\tau/2 + n_1$ times. Assume the worst case, where $\mathsf{R}^*$ learns both shares for every other session. That is, for $m/3 - (\tau/2 + n_0) - (\tau/2 + n_1)$ sessions, $\mathsf{R}^*$ learns both $s^{(i)}_0$ and $s^{(i)}_1$.

So we have $(\tau/2 + n_0) + m/3 - (\tau/2 + n_0) - (\tau/2 + n_1) = m/3 - \tau/2 - n_1$ shares of $s_0$ and $(\tau/2 + n_1) + m/3 - (\tau/2 + n_0) - (\tau/2 + n_1) = m/3 - \tau/2 - n_0$ shares of $s_1$. Recall that $\tau$, the threshold of our secret sharing scheme, is $2m/9$. Therefore we have $2m/9 - n_1$ shares of $s_0$ and $2m/9 - n_0$ shares of $s_1$, and $\mathsf{R}^*$ cannot reconstruct either string.

**Case 2** Let $\tau/2 + n_b$ be the number of times the bit $b$ appears, and $n_{1-b} \in [1, \tau/2]$ be the number of times the bit $1-b$ appears. Again assume the worst case where $\mathsf{R}^*$ learns both shares in every other session. So for $m/3 - (\tau/2 + n_b) - n_{1-b}$ sessions, $\mathsf{R}^*$ learns both $s^{(i)}_0$ and $s^{(i)}_1$.

Therefore $\mathsf{R}^*$ learns $\tau/2 + n_b + m/3 - (\tau/2 + n_b) - n_{1-b} = m/3 - n1 - b$ shares of $s_b$, and $n_{1-b} + m/3 - (\tau/2 + n_b) - n_{1-b} = m/3 - \tau/2 - n_b = 2m/9 - n_b$ shares of $s_{1-b}$. As in case 1, $2m/9 - n_b$ is less than our secret sharing threshold, and not enough shares for $\mathsf{R}^*$ to learn $s_{1-b}$.

We have shown that in both cases, $\mathsf{R}^*$ either learns neither string, or only $s_b$. Therefore $\mathbf{Hyb}_3$ is indistinguishable from $\mathbf{Hyb}_2$.

□

## 5.3 Simulation for malicious senders

Next, in Fig. 11 we present our simulator $\mathsf{Sim}_{\mathsf{S}^*}$ for the case of a malicious sender.

The simulator computes the first round message as an honest receiver and sends it to the sender. Upon receiving the second round message from the sender, the simulator then computes the $\mathbf{CRS}_{\mathsf{R}}$ and the proof $\pi_{\mathsf{R}}$ as an honest receiver would. Note that at this point, the simulator knows which indices (denoted $A$) are to be opened on behalf of the receiver. The simulator then randomly samples adjustment bits $d^{(i)}$ for $i \in [m] \setminus A$. This is in contrast with the real-world receiver where the receiver knows its input bit $b$ and computes $d^{(i)} = b^{(i)} \oplus b$, where $b^{(i)}$ are the inputs used in the computed $\mathrm{ot}^{(i)}_1$ messages.

**Main Thread:**

- Compute $(\mathsf{com\text{-}crs\text{-}prvr}_{\mathsf{R}}, \mathit{crs\text{-}prvr}_{\mathsf{R}}) \leftarrow \mathsf{P}_0(\lambda)$ as an honest receiver would:
  - Sample randomness $r\text{-}ot1_{\mathsf{R}}^{(i)} \xleftarrow{\$} \{0,1\}^{\lambda}$ for $i \in [m]$
  - Compute $\mathsf{com\text{-}r}_{\mathsf{R}}^{(i)} \leftarrow \mathtt{COM}(r\text{-}ot1_{\mathsf{R}}^{(i)}; \gamma_{\mathsf{R}}^{(i)})$ for $i \in [m]$
- Send $(\mathsf{com\text{-}crs\text{-}prvr}_{\mathsf{R}}, \mathsf{com\text{-}r}_{\mathsf{R}}^{(i)}\}_{i\in[m]})$ to $\mathsf{S}^*$ and receive $(\mathit{crs\text{-}vrfr}_{\mathsf{R}}, \mathsf{com\text{-}crs\text{-}prvr}_{\mathsf{S}}, \{r\text{-}ot1_{\mathsf{S}}^{(i)}, \mathsf{com\text{-}r}_{\mathsf{S}}^{(i)}\}_{i\in[m]})$
  - Compute $\mathbf{CRS}_{\mathsf{R}}, \{\mathrm{ot}_1^{(i)}\}_{i\in[m]}, \pi_{\mathsf{R}}, \mathit{crs\text{-}vrfr}_{\mathsf{S}}$ as an honest receiver would
  - $\boxed{\boxed{\text{Sample } d^{(i)} \xleftarrow{\$} \{0,1\} \text{ for } i \in [m]}}$

  Send $(\mathbf{CRS}_{\mathsf{R}}, \pi_{\mathsf{R}}, \mathit{crs\text{-}vrfr}_{\mathsf{S}}, \{\mathrm{ot}_1^{(i)}\}_{i\in[m]}, \{d^{(i)}\}_{i\in[m]\setminus A})$ to $\mathsf{S}^*$ and receive $(\{\mathrm{ot}_2^{(i)}\}_{i\in Alive}, \mathbf{CRS}_{\mathsf{S}}, \pi_{\mathsf{S}}, \{\mathsf{ct}_0^{(i)}, \mathsf{ct}_1^{(i)}\}_{i\in Alive})$
  - If $\mathsf{V}_1(\mathbf{CRS}_{\mathsf{S}}, \pi_{\mathsf{S}}) \neq 1$, abort
  - Else, set $Keys = \emptyset$ and $ctr = 0$ and $Alive = [m] \setminus (B \cup A)$

  **Rewind Thread**

  - $\boxed{\text{Choose } e_{\mathsf{S}} \xleftarrow{\$} \{0,1\}^m \text{ and } H_{\mathsf{S}} \leftarrow \mathbf{Samp}((\mathtt{OTS}, \{r\text{-}ot2_{\mathsf{R}}^{(i)}, \mathsf{com\text{-}r}_{\mathsf{S}}^{(i)}, \mathrm{ot}_2\}_{i\in[m]}, \{c_{\mathsf{S}}^{(i)}\}_{i\in[m]\setminus A}), e_{\mathsf{S}}) \text{ and use } H_{\mathsf{S}} \text{ to compute } \mathit{crs\text{-}vrfr}_{\mathsf{S}}}$
  - Send $(\mathbf{CRS}_{\mathsf{R}}, \pi_{\mathsf{R}}, \mathit{crs\text{-}vrfr}_{\mathsf{S}}, \{\mathrm{ot}_1^{(i)}\}_{i\in[m]}, \{d^{(i)}\}_{i\in[m]\setminus A})$ to $\mathsf{S}^*$ and receive $(\{\mathrm{ot}_2^{(i)}\}_{i\in Alive}, \mathbf{CRS}_{\mathsf{S}}, \pi_{\mathsf{S}}, \{\mathsf{ct}_0^{(i)}, \mathsf{ct}_1^{(i)}\}_{i\in Alive})$
  - If $\mathsf{V}_1(\mathbf{CRS}_{\mathsf{S}}, \pi_{\mathsf{S}}) \neq 1$, abort
  - For every $i \in Alive$ that was not observed in a previous rewind where $\mathrm{ot}_2^{(i)} = \mathtt{OTS}(k_0^{(i)}, k_1^{(i)}, \mathrm{ot}_1^{(i)}; r\text{-}ot2^{(i)})$, add $(k_0^{(i)}, k_1^{(i)})$ to $Keys$
  - $\boxed{\text{Set } ctr = ctr + 1. \text{ If } ctr = 2^{\lambda} \text{ abort, else if } |Keys| < 2m/9 \text{ go to beginning of rewind thread, else proceed}}$

- For each pair of shares $(k_0^{(i)}, k_1^{(i)}) \in Keys$
  - Compute $s_0^{(i)} = k_{0\oplus d^{(i)}} \oplus \mathsf{ct}_{0\oplus d^{(i)}}^{(i)}$ and $s_1^{(i)} = k_{1\oplus d^{(i)}} \oplus \mathsf{ct}_{1\oplus d^{(i)}}^{(i)}$ for $(k_0^{(i)}, k_1^{(i)}) \in Keys$
  - Compute $s_0 = \mathtt{Reconstruct}(s_0^{(i)})$ and $s_1 = \mathtt{Reconstruct}(s_1^{(i)})$
- Forward $(s_0, s_1)$ to $\mathcal{F}_{OT}$ and output whatever $\mathsf{S}^*$ outputs

**Figure 11:** $\mathsf{Sim}_{\mathsf{S}^*}$ The Simulation of $\Pi$ for a Malicious Sender $\mathsf{S}^*$. $\boxed{\mathbf{Hyb}_1}$, $\boxed{\boxed{\mathbf{Hyb}_2}}$

The simulator sends $\mathbf{CRS}_{\mathsf{R}}$, the proof $\pi_{\mathsf{R}}$, the $\mathrm{ot}_1^{(i)}$ messages, and the corresponding adjustment bits $d^{(i)}$ to the sender and receives back the $\mathrm{ot}_2^{(i)}$ messages, the $\mathbf{CRS}_{\mathsf{S}}$, the proof $\pi_{\mathsf{S}}$ which includes openings to some of the $\mathrm{ot}_2^{(i)}$ messages (the corresponding indices are denoted as $B$), and ciphertexts that encrypt shares of the inputs of the sender. The simulator first checks that the proofs verify, and aborts if this is not the case.

Now the simulator's objective is to extract the inputs of the $\mathrm{ot}_2^{(i)}$ messages that correspond to the $Alive = [m] \setminus (A \cup B)$ indices. To this end, the simulator starts a rewind thread where the simulator rewinds the sender to the beginning of round 3. In each rewind iteration the simulator samples a random bit string $e_{\mathsf{S}} \overset{\$}{\leftarrow} \{0,1\}^\lambda$ such that $m/3$ of the bits are 1. Recall that this bit string $e_{\mathsf{S}}$ indicates the set $B$ which is the set of indices for which the sender sends the inputs to the corresponding $\mathrm{ot}_2^{(i)}$ messages. Using the approximate average-case programmability (Def. 10) of the hash family the simulator samples a hash function $H_{\mathsf{S}}$ such that on input the sender's input from the main thread, the hash function outputs $e_{\mathsf{S}}$. The simulator then sends the round 3 message with the sampled hash function and receives back the round 4 message from the simulator.

Now for each index $i$ in $Alive$ from the main thread that was not observed before, if $e_{\mathsf{S}}[i] = 1$, the simulator receives the inputs to the $\mathrm{ot}_2^{(i)}$ messages, which includes the keys used to encrypt the shares of input strings $s_0, s_1$. The simulator first checks if the received openings are valid and stores them in a set $Keys$. The simulator rewinds the sender and continues until it receives $2m/9$ valid inputs to the $\mathrm{ot}_2^{(i)}$ messages. At this point, the simulator has extracted the keys with which the sender encrypted the shares of OT input strings in the main thread. The simulator then exits the rewind thread and computes the shares of the secret inputs denoted $s_0^{(i)}$ and $s_1^{(i)}$. The simulator then reconstructs $s_0$ and $s_1$ from these shares and sends $s_0, s_1$ to the $\mathcal{F}_{OT}$ functionality then outputs whatever the malicious sender outputs. This ends the simulation.

First we prove that $\mathsf{Sim}_{\mathsf{S}^*}$ runs in polynomial time.

**Lemma 10.** $\mathsf{Sim}_{\mathsf{S}^*}$ *runs in expected time polynomial in* $\lambda$ *and* $m$

*Proof.* All the steps of the simulator before and after the rewinding take place in strict polynomial time. And within the rewinding thread the only difference is how the hash function is sampled. Therefore an analysis similar to Lem. 5 shows that the number of rewind iterations just corresponds to sampling $m/3$ indices from $[m] \setminus A$ until $2m/9$ keys for the sessions in $Alive$ are retrieved. As before we can assume that the simulator aborts with probability $p$ in the main thread, implying the total number of rewind iterations is $O(m \ln m)/p$ which implies the expected running time for the simulator is $poly(\lambda, m)$. □

We make use of a helper lemma that the generation of the CRS preserves soundness in the case of a malicious sender.

**Definition 12.** Let $\mathsf{bad}_{\mathsf{S}} \subset Alive$ be the set of indices for which $\mathtt{Check}_{\mathtt{OTS}}((\mathrm{ot}_1^{(i)}, r\text{-}ot2_{\mathsf{R}}^{(i)}), r\text{-}ot2_{\mathsf{S}}^{(i)}, (k_0^{(i)}, k_1^{(i)}, r\text{-}ot2^{(i)})) \neq 1$

**Lemma 11.** *If* $\Sigma_{NIP}$ *is a sound proof system, then* $Pr[|\mathsf{bad}_{\mathsf{S}}| > m/9] \leq \mathsf{negl}(\lambda)$

*Proof.* This proof is very similar to the proof of Lem. 6. Towards a contradiction, assume that there exists an adversary $\mathsf{S}^*$ such that $\mathsf{Sim}_{\mathsf{S}^*}$ does not abort and $|\mathsf{bad}_{\mathsf{S}}| > m/9$. We can then construct a reduction $\mathcal{B}$ such that $\mathcal{B}$ can violate the soundness of $\Sigma_{NIP}$. Define $\mathcal{B}$ as follows:

$\mathcal{B}(\mathbf{CRS}_{\mathsf{S}}, \{key_{\mathsf{S}}^{(i)}, \rho_{\mathsf{S}}^{(i)}\}_{i\in[m]})$ :

1. Activate $\mathsf{S}^*(1^\lambda)$
2. Simulate as in $\mathsf{Sim}_{\mathsf{R}^*}$ until receiving the round 4 message $(\mathbf{CRS}^*_{\mathsf{S}}, \pi_{\mathsf{S}}, crs\text{-}vrfr_{\mathsf{R}}, \{\mathrm{ot}_2^{(i)}\}_{i\in[m]}, \{\mathsf{ct}_0^{(i)}, \mathsf{ct}_1^{(i)}\}_{i\in Alive}))$.
3. **Rewind to set CRS**
   (a) Rewind to the beginning of round 3
   (b) For every $(key_{\mathsf{S}}^{*(i)}, \rho_{\mathsf{S}}^{*(i)}) \in z_{\mathsf{S}}^{(i)} \in \pi_{\mathsf{S}}$ that has not been observed, compute the corresponding $r\text{-}key_{\mathsf{S}}^{*(i)}$.
   (c) Set $r\text{-}key_{\mathsf{R}}^{(i)} = (key_{\mathsf{S}}^{(i)} \| \rho_{\mathsf{S}}^{(i)}) \oplus r\text{-}key_{\mathsf{S}}^{*(i)}$ and store $r\text{-}key_{\mathsf{R}}^{(i)}$ in $\mathsf{CRSKeys}$
   (d) If $|\mathsf{CRSKeys}| < m$ go to step (1).
4. Rewind to the beginning of Round 2, and use $\mathsf{CRSKeys}$ as $\{r\text{-}key_{\mathsf{R}}^{(i)}\}_{i\in[m]}$ to compute $crs\text{-}vrfr_{\mathsf{R}}$.
5. Simulate the rest of the protocol as in $\mathsf{Sim}_{\mathsf{S}^*}$ but abort if $ctr = c \cdot m \cdot H_m + 1$. If during the rewind thread we have $|\mathsf{bad}_{\mathsf{S}}| > m/9$, output $(x, \pi_{\mathsf{S}})$ to the challenger
6. Else, abort.

We first show that the reduction runs in strictly polynomial time and proceeds with non-negligible probability. We know by the proof of Lemma 5 that $c \cdot m \cdot H_m + 1$ is polynomial. Further, by the coupon collectors problem, we have that $\mathcal{B}$ aborts during the rewind thread of $\mathsf{Sim}_{\mathsf{S}^*}$ with probability at most $1/c$.

Since $\mathtt{Check}_{\mathtt{OTS}}(\mathrm{ot}_1^{(i)}, r\text{-}ot2^{(i)}, (k_0^{(i)}, k_1^{(i)}, \mathrm{ot}_1^{(i)}, r\text{-}ot2^{*(i)})) \neq 1$ for more than $m/9$ of $i$, we know that $x \notin \mathcal{L}$. However, $\pi_{\mathsf{S}}$ must verify, else $\mathsf{Sim}_{\mathsf{S}^*}$ must have aborted. We have then constructed a reduction that breaks the soundness of $\Sigma_{NIP}$ with non-negligible probability, and $Pr[|\mathsf{bad}| > m/9] \leq \mathsf{negl}(\lambda)$. ☐

We prove indistinguishability through a series of hybrids, beginning from the real-world protocol, and making small changes until we reach the simulated protocol. Our hybrids are as follows:

- $\mathbf{Hyb}_0$ The real world protocol
- $\boxed{\mathbf{Hyb}_1}$ This is the same as $\mathbf{Hyb}_0$, except we rewind as in $\mathsf{Sim}_{\mathsf{S}^*}$
- $\boxed{\boxed{\mathbf{Hyb}_2}}$ This is the same as $\mathbf{Hyb}_1$, except $d^{(i)}$ is sampled randomly instead of being computed as $d^{(i)} = b \oplus b^{(i)}$

**Lemma 12.** $\mathbf{Hyb}_0$ *is indistinguishable from* $\mathbf{Hyb}_1$

*Proof.* The only difference between these two hybrids is that in $\mathbf{Hyb}_1$, we rewind as in $\mathsf{Sim}_{\mathsf{S}^*}$ and abort if there are $2^\lambda$ rewinds. We consider the following two events, which are the only events that could cause this:

- $\mathsf{bad}_4$ : The proof $\pi_{\mathsf{S}}$ verifies, but $\mathtt{Check}_{\mathtt{OTS}}(\mathrm{ot}_1^{(i)}, r\text{-}ot2^{(i)}, (k_0^{(i)}, k_1^{(i)}, \mathrm{ot}_1^{(i)}, r\text{-}ot2^{*(i)})) \neq 1$ for more than $m/9$ of the opened values in the rewind thread
- $\mathsf{bad}_5$ : $|Keys| < |Alive| - m/9$ after $\mathsf{poly}(\lambda)$ rewinds

If neither event occurs, the hybrids are indistinguishable, as the simulator rewinds $< 2^\lambda$ times and the proof is valid. By Lem. 11 we know that $\mathsf{bad}_4$ happens with negligible probability.

Suppose instead that $\mathsf{bad}_5$ occurs. Since $\mathsf{Sim}_{\mathsf{S}^*}$ runs in polynomial time, we know that this occurs with negligible probability.

Therefore $\mathbf{Hyb}_1$ is indistinguishable from $\mathbf{Hyb}_0$. □

**Lemma 13.** *If* $(\mathtt{OTR}, \mathtt{OTS}, \mathtt{OTD})$ *is a semi-honest secure OT protocol,* $\mathtt{COM}$ *is a computationally hiding commitment scheme,* $\mathtt{COM}_{\mathsf{stat\text{-}hide}}$ *is a statistically hiding commitment scheme, and* $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *is a CPA secure symmetric key encryption scheme with pseudorandom keys, then* $\mathbf{Hyb}_1$ *is indistinguishable from* $\mathbf{Hyb}_2$

*Proof.* We proceed through a series of hybrids. For an index $j \in [m]$, consider the hybrid $\mathbf{Hyb}_{1,j}$ where for $i \leq j$, $d^{(i)} = b^{(i)} \oplus b$ and for $i > j$, $d^{(i)} \xleftarrow{\$} \{0,1\}$. Note that $\mathbf{Hyb}_{1,m}$ is equivalent to $\mathbf{Hyb}_1$ as all $d^{(i)} = b^{(i)} \oplus b$. Likewise, $\mathbf{Hyb}_{1,0}$ is equivalent to $\mathbf{Hyb}_2$ as all $d^{(i)} \xleftarrow{\$} \{0,1\}$.

Towards a contradiction, suppose that there exists a PPT adversary $\mathsf{S}^*$ such that $\mathsf{S}^*$ can distinguish between the two hybrids. Consider the event where $j \notin A$, where $A$ is determined by the output of the CIH $H_k$. If $j \in A$, then the hybrids are indstinguishable as $d^{(i)}$ is never sent to $\mathsf{S}^*$. Therefore the adversary can only distinguish when $j \notin A$.

Suppose $\mathsf{S}^*$ can distinguish using the message $\mathrm{ot}_1^{(j)}$. Then, we can construct a reduction $\mathsf{S}'$ that breaks the semi-honest privacy of $(\mathtt{OTR}, \mathtt{OTS}, \mathtt{OTD})$. Note that Madathil et al. proved that any OT protocol secure against a semi-honest sender is already secure against a malicious sender (see proof of Lem. 1 [MOSV22]). Therefore, we consider $\mathsf{S}'$ to be malicious. Define $\mathsf{S}'$ as follows:

$\underline{\mathsf{S}'(1^\lambda)}$:

1. Upon receiving message $\mathrm{ot}_1^*$ from the challenger, run $\mathsf{S}^*$ and simulate as in $\mathbf{Hyb}_{1,j-1}$ using $\mathrm{ot}_1^*$ in the place of $\mathrm{ot}_1^{(j)}$ and abort if $ctr = c \cdot m \cdot H_m + 1$
2. If $j \in A \cup B$, abort.
3. Upon receiving $(\{\mathrm{ot}_2^{(i)}\}_{i \in Alive}, \mathbf{CRS}_{\mathsf{S}}, \pi_{\mathsf{S}}, \{\mathsf{ct}_0^{(i)}, \mathsf{ct}_1^{(i)}\}_{i \in Alive})$ from $\mathsf{S}^*$
   (a) Forward $\mathrm{ot}_2^{(j)}$ to the challenger
   (b) Receive challenge $b^*$
   (c) Rewind to round 3, and send $\mathsf{S}^*$ the same message, except replace $d^{(j)}$ with $b \oplus b^*$
   (d) Receive $(\{\mathrm{ot}_2^{(i)}\}_{i \in Alive}, \mathbf{CRS}_{\mathsf{S}}, \pi_{\mathsf{S}}, \{\mathsf{ct}_0^{(i)}, \mathsf{ct}_1^{(i)}\}_{i \in Alive})$ from $\mathsf{S}^*$ and output whatever $\mathsf{S}^*$ outputs

Since $c \cdot m \cdot H_m$ is polynomial, we know that $\mathsf{S}'$ runs in strictly polynomial time. There are three cases:

- If $j \in A \cup B$, $\mathsf{S}'$ aborts. However, this only happens with probability $2/3$
- $\mathsf{S}'$ exceeds $c \cdot m \cdot H_m$ rewinds and aborts. However this occurs with probability at most $1/c$
- If $j \notin A \cup B$, then either $b^* = b^{(j)}$ or $b^* \xleftarrow{\$} \{0,1\}$
  - If $b^* = b^{(j)}$, then we are in $\mathbf{Hyb}_{1,j}$, as $d^{(j)}$ is computed as $b^{(j)} \oplus b$

  – Else if $b^* \xleftarrow{\$} \{0,1\}$, then we are in $\mathbf{Hyb}_{1,j-1}$, as $d^{(j)}$ is the XOR of a random bit with $b$, which means $d^{(j)}$ is indistinguishable from a random value itself

Therefore, we have found a receiver that violates the semi-honest privacy of $(\mathtt{OTR}, \mathtt{OTS}, \mathtt{OTD})$ with non-negligible probability and have a contradiction.

Suppose instead that $\mathsf{S}^*$ distinguishes using $\textsf{com-r}_\mathsf{R}^{(j)}$ by learning the randomness used to compute $\mathrm{ot}_1^{(j)}$. We can then build a reduction $\mathcal{B}$ that violates the hiding of the commitment $\mathtt{COM}$. Define $\mathcal{B}$ as follows:

$\underline{\mathcal{B}(1^\lambda):}$

1. Query the challenger with $(0, r\text{-}ot1_\mathsf{R}^{(j)})$ and receive commitment $\mathsf{com}^*$
2. Activate $\mathsf{S}^*$ and simulate as in $\mathbf{Hyb}_{1,j-1}$ except:
   (a) Abort if $ctr = c \cdot m \cdot H_m + 1$
   (b) Use $\mathsf{com}^*$ in the place of $\textsf{com-r}_\mathsf{R}^{(j)}$
   (c) If $j \in A \cup B$, abort
3. Output whatever $\mathsf{S}^*$ outputs

We note again that $\mathcal{B}$ aborts in the case where $j \in A \cup B$, however this only happens with probability $2/3$. Further $\mathcal{B}$ aborts if more than $c \cdot m \cdot H_m$ rewinds occur, but this happens with probability at most $1/c$. If $\mathsf{com}^*$ is a commitment to $r\text{-}ot1_\mathsf{R}^{(j)}$, then this is exactly $\mathbf{Hyb}_{1,j}$ as we commit to the randomness used to compute $d^{(j)}$. If, however, $\mathsf{com}^*$ is a commitment to 0, then we are in $\mathbf{Hyb}_{1,j-1}$, as $d^{(j)}$ is computed independently of the randomness committed. Therefore we have found an adversary that breaks the hiding of $\mathtt{COM}$, and have a contradiction.

By the same argument, we know that $\mathsf{S}^*$ cannot distinguish by learning the randomness committed in $\textsf{com-r-key}_\mathsf{R}^{(j)}$, or the key committed in $\textsf{com-key}_\mathsf{R}^{(i)}$.

Finally, suppose that $\mathsf{S}^*$ is able to distinguish through $c_\mathsf{R}^{(j)} \in \pi_\mathsf{R}$. We then construct a reduction $\mathcal{A}$ that violates the CPA security of the encryption scheme $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$. Define $\mathcal{A}$ as follows:

$\underline{\mathcal{A}(1^\lambda):}$

1. Activate $\mathsf{S}^*$ and simulate as in $\mathbf{Hyb}_{1,j-1}$ until the key $key^{(j)}$ is computed, but abort if $ctr = c \cdot m \cdot H_m + 1$
2. Query the challenger with $(0, (r\text{-}ot1^{(j)}, r\text{-}ot1_\mathsf{R}^{(j)}, \gamma_\mathsf{R}^{(j)}))$ and receive challenge $c^*$
3. Continue simulating as in $\mathbf{Hyb}_{1,j-1}$, except use $c^*$ in place of $c_\mathsf{R}^{(j)}$ and abort if $ctr = c \cdot m \cdot H_m + 1$
4. Abort if $j \in A \cup B$
5. Output whatever $\mathsf{S}^*$ outputs

Again, note that $\mathcal{A}$ aborts in the case where $j \in A \cup B$, however, this only happens with probability $2/3$. Further, $\mathcal{A}$ aborts if the number of rewinds exceeds $c \cdot m \cdot H_m$, but this occurs with probability at most $1/c$. If $c^*$ is an encryption of $(r\text{-}ot1^{(j)}, r\text{-}ot1_\mathsf{R}^{(j)}, \gamma_\mathsf{R}^{(j)})$, then this is exactly $\mathbf{Hyb}_{1,j}$, as $d^{(j)}$ is computed using the randomness encrypted in $c_\mathsf{R}^{(j)}$. If

$c^*$ is an encryption of 0, then this is exactly $\mathbf{Hyb}_{1,j-1}$, as $d^{(j)}$ is computed independently of the value encrypted in $c^{(j)}$. Therefore we have found an adversary $\mathcal{A}$ that violates the CPA security of $\mathcal{E}$ and have reached a contradiction.

Therefore, in every case, $\mathbf{Hyb}_{1,j}$ is indistinguishable from $\mathbf{Hyb}_{1,j-1}$. As we noted, $\mathbf{Hyb}_{1,m}$ is equivalent to $\mathbf{Hyb}_1$ and $\mathbf{Hyb}_{1,0}$ is equivalent to $\mathbf{Hyb}_2$. Thus $\mathbf{Hyb}_1$ is indistinguishable from $\mathbf{Hyb}_2$ □

## References

[AJL+12] Gilad Asharov, Abhishek Jain, Adriana López-Alt, Eran Tromer, Vinod Vaikuntanathan, and Daniel Wichs. Multiparty computation with low communication, computation and interaction via threshold FHE. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology - EUROCRYPT 2012 - 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK, April 15-19, 2012. Proceedings*, volume 7237 of *Lecture Notes in Computer Science*, pages 483–501. Springer, 2012. doi:10.1007/978-3-642-29011-4_29.

[BFJ+20] Saikrishna Badrinarayanan, Rex Fernando, Aayush Jain, Dakshita Khurana, and Amit Sahai. Statistical ZAP arguments. In Anne Canteaut and Yuval Ishai, editors, *Advances in Cryptology - EUROCRYPT 2020 - 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10-14, 2020, Proceedings, Part III*, volume 12107 of *Lecture Notes in Computer Science*, pages 642–667. Springer, 2020. doi:10.1007/978-3-030-45727-3_22.

[BKM20] Zvika Brakerski, Venkata Koppula, and Tamer Mour. NIZK from LPN and trapdoor hash via correlation intractability for approximable relations. In Daniele Micciancio and Thomas Ristenpart, editors, *Advances in Cryptology - CRYPTO 2020 - 40th Annual International Cryptology Conference, CRYPTO 2020, Santa Barbara, CA, USA, August 17-21, 2020, Proceedings, Part III*, volume 12172 of *Lecture Notes in Computer Science*, pages 738–767. Springer, 2020. doi:10.1007/978-3-030-56877-1_26.

[BLV06] Boaz Barak, Yehuda Lindell, and Salil P. Vadhan. Lower bounds for non-black-box zero knowledge. *J. Comput. Syst. Sci.*, 72(2):321–391, 2006. URL: https://doi.org/10.1016/j.jcss.2005.06.010, doi:10.1016/J.JCSS.2005.06.010.

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In Dorothy E. Denning, Raymond Pyle, Ravi Ganesan, Ravi S. Sandhu, and Victoria Ashby, editors, *CCS '93, Proceedings of the 1st ACM Conference on Computer and Communications Security, Fairfax, Virginia, USA, November 3-5, 1993*, pages 62–73. ACM, 1993. doi:10.1145/168588.168596.

[CCG+21] Arka Rai Choudhuri, Michele Ciampi, Vipul Goyal, Abhishek Jain, and Rafail Ostrovsky. Oblivious transfer from trapdoor permutations in minimal rounds. In Kobbi Nissim and Brent Waters, editors, *Theory of Cryptography - 19th International Conference, TCC 2021, Raleigh, NC, USA, November 8-11, 2021, Proceedings, Part II*, volume 13043 of *Lecture Notes in Computer Science*, pages 518–549. Springer, 2021. doi:10.1007/978-3-030-90453-1_18.

[CCH+18] Ran Canetti, Yilei Chen, Justin Holmgren, Alex Lombardi, Guy N. Rothblum, and Ron D. Rothblum. Fiat-shamir from simpler assumptions. *IACR Cryptol. ePrint Arch.*, page 1004, 2018. URL: https://eprint.iacr.org/2018/1004.

[CCH+19] Ran Canetti, Yilei Chen, Justin Holmgren, Alex Lombardi, Guy N. Rothblum, Ron D. Rothblum, and Daniel Wichs. Fiat-shamir: from practice to theory. In Moses Charikar and Edith Cohen, editors, *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, Phoenix, AZ, USA, June 23-26, 2019*, pages 1082–1090. ACM, 2019. doi:10.1145/3313276.3316380.

[CCR16] Ran Canetti, Yilei Chen, and Leonid Reyzin. On the correlation intractability of obfuscated pseudorandom functions. In Eyal Kushilevitz and Tal Malkin, editors, *Theory of Cryptography - 13th International Conference, TCC 2016-A, Tel Aviv, Israel, January 10-13, 2016, Proceedings, Part I*, volume 9562 of *Lecture Notes in Computer Science*, pages 389–415. Springer, 2016. doi:10.1007/978-3-662-49096-9_17.

[CCRR18] Ran Canetti, Yilei Chen, Leonid Reyzin, and Ron D. Rothblum. Fiat-shamir and correlation intractability from strong kdm-secure encryption. In Jesper Buus Nielsen and Vincent Rijmen, editors, *Advances in Cryptology - EUROCRYPT 2018 - 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29 - May 3, 2018 Proceedings, Part I*, volume 10820 of *Lecture Notes in Computer Science*, pages 91–122. Springer, 2018. doi:10.1007/978-3-319-78381-9_4.

[CDMW09] Seung Geol Choi, Dana Dachman-Soled, Tal Malkin, and Hoeteck Wee. Simple, black-box constructions of adaptively secure protocols. In Omer Reingold, editor, *Theory of Cryptography, 6th Theory of Cryptography Conference, TCC 2009, San Francisco, CA, USA, March 15-17, 2009. Proceedings*, volume 5444 of *Lecture Notes in Computer Science*, pages 387–402. Springer, 2009. doi:10.1007/978-3-642-00457-5_23.

[CGH98] Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited (preliminary version). In Jeffrey Scott Vitter, editor, *Proceedings of the Thirtieth Annual ACM Symposium on the Theory of Computing, Dallas, Texas, USA, May 23-26, 1998*, pages 209–218. ACM, 1998. doi:10.1145/276698.276741.

[CGH04] Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited. *J. ACM*, 51(4):557–594, 2004. doi:10.1145/1008731.1008734.

[CGJ+23] Arka Rai Choudhuri, Sanjam Garg, Abhishek Jain, Zhengzhong Jin, and Jiaheng Zhang. Correlation intractability and snargs from sub-exponential DDH. In Helena Handschuh and Anna Lysyanskaya, editors, *Advances in Cryptology - CRYPTO 2023 - 43rd Annual International Cryptology Conference, CRYPTO 2023, Santa Barbara, CA, USA, August 20-24, 2023, Proceedings, Part IV*, volume 14084 of *Lecture Notes in Computer Science*, pages 635–668. Springer, 2023. doi:10.1007/978-3-031-38551-3_20.

[CJJ21a] Arka Rai Choudhuri, Abhishek Jain, and Zhengzhong Jin. Non-interactive batch arguments for NP from standard assumptions. In Tal Malkin and Chris Peikert, editors, *Advances in Cryptology - CRYPTO 2021 - 41st Annual International Cryptology Conference, CRYPTO 2021, Virtual Event, August 16-20, 2021, Proceedings, Part IV*, volume 12828 of *Lecture Notes in Computer*

*Science*, pages 394–423. Springer, 2021. doi:10.1007/978-3-030-84259-8_14.

[CJJ21b] Arka Rai Choudhuri, Abhishek Jain, and Zhengzhong Jin. Snargs for $\mathcal{P}$ from LWE. In *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021, Denver, CO, USA, February 7-10, 2022*, pages 68–79. IEEE, 2021. doi:10.1109/FOCS52979.2021.00016.

[COSW23] Michele Ciampi, Rafail Ostrovsky, Luisa Siniscalchi, and Hendrik Waldner. List oblivious transfer and applications to round-optimal black-box multiparty coin tossing. In Helena Handschuh and Anna Lysyanskaya, editors, *Advances in Cryptology - CRYPTO 2023 - 43rd Annual International Cryptology Conference, CRYPTO 2023, Santa Barbara, CA, USA, August 20-24, 2023, Proceedings, Part I*, volume 14081 of *Lecture Notes in Computer Science*, pages 459–488. Springer, 2023. doi:10.1007/978-3-031-38557-5_15.

[DGH$^+$20] Nico Döttling, Sanjam Garg, Mohammad Hajiabadi, Daniel Masny, and Daniel Wichs. Two-round oblivious transfer from CDH or LPN. In Anne Canteaut and Yuval Ishai, editors, *Advances in Cryptology - EUROCRYPT 2020 - 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10-14, 2020, Proceedings, Part II*, volume 12106 of *Lecture Notes in Computer Science*, pages 768–797. Springer, 2020. doi:10.1007/978-3-030-45724-2_26.

[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 Dennis Hofheinz and Alon Rosen, editors, *Theory of Cryptography - 17th International Conference, TCC 2019, Nuremberg, Germany, December 1-5, 2019, Proceedings, Part I*, volume 11891 of *Lecture Notes in Computer Science*, pages 111–130. Springer, 2019. doi:10.1007/978-3-030-36030-6_5.

[FS86] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *Advances in Cryptology - CRYPTO '86, Santa Barbara, California, USA, 1986, Proceedings*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194. Springer, 1986. doi:10.1007/3-540-47721-7_12.

[GJJM20] Vipul Goyal, Abhishek Jain, Zhengzhong Jin, and Giulio Malavolta. Statistical zaps and new oblivious transfer protocols. In Anne Canteaut and Yuval Ishai, editors, *Advances in Cryptology - EUROCRYPT 2020 - 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10-14, 2020, Proceedings, Part III*, volume 12107 of *Lecture Notes in Computer Science*, pages 668–699. Springer, 2020. doi:10.1007/978-3-030-45727-3_23.

[GK03] Shafi Goldwasser and Yael Tauman Kalai. On the (in)security of the fiat-shamir paradigm. In *44th Symposium on Foundations of Computer Science (FOCS 2003), 11-14 October 2003, Cambridge, MA, USA, Proceedings*, pages 102–113. IEEE Computer Society, 2003. doi:10.1109/SFCS.2003.1238185.

[GKR08] Shafi Goldwasser, Yael Tauman Kalai, and Guy N. Rothblum. Delegating computation: interactive proofs for muggles. In Cynthia Dwork, editor, *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*, pages 113–122. ACM, 2008. doi:10.1145/1374376.1374396.

[GLOV12] Vipul Goyal, Chen-Kuei Lee, Rafail Ostrovsky, and Ivan Visconti. Constructing non-malleable commitments: A black-box approach. In *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012*, pages 51–60. IEEE Computer Society, 2012. doi:10.1109/FOCS.2012.47.

[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 V. Aho, editor, *Proceedings of the 19th Annual ACM Symposium on Theory of Computing, 1987, New York, New York, USA*, pages 218–229. ACM, 1987. doi:10.1145/28395.28420.

[GOSV14] Vipul Goyal, Rafail Ostrovsky, Alessandra Scafuro, and Ivan Visconti. Black-box non-black-box zero knowledge. In David B. Shmoys, editor, *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 515–524. ACM, 2014. doi:10.1145/2591796.2591879.

[Goy11] Vipul Goyal. Constant round non-malleable protocols using one way functions. In Lance Fortnow and Salil P. Vadhan, editors, *Proceedings of the 43rd ACM Symposium on Theory of Computing, STOC 2011, San Jose, CA, USA, 6-8 June 2011*, pages 695–704. ACM, 2011. doi:10.1145/1993636.1993729.

[Hai08] Iftach Haitner. Semi-honest to malicious oblivious transfer - the black-box way. In Ran Canetti, editor, *Theory of Cryptography, Fifth Theory of Cryptography Conference, TCC 2008, New York, USA, March 19-21, 2008*, volume 4948 of *Lecture Notes in Computer Science*, pages 412–426. Springer, 2008. doi:10.1007/978-3-540-78524-8_23.

[HIK+11] Iftach Haitner, Yuval Ishai, Eyal Kushilevitz, Yehuda Lindell, and Erez Petrank. Black-box constructions of protocols for secure computation. *SIAM J. Comput.*, 40(2):225–266, 2011. doi:10.1137/100790537.

[HL18] Justin Holmgren and Alex Lombardi. Cryptographic hashing from strong one-way functions (or: One-way product functions and their applications). In Mikkel Thorup, editor, *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, pages 850–858. IEEE Computer Society, 2018. doi:10.1109/FOCS.2018.00085.

[HLR21] Justin Holmgren, Alex Lombardi, and Ron D. Rothblum. Fiat-shamir via list-recoverable codes (or: parallel repetition of GMW is not zero-knowledge). In Samir Khuller and Virginia Vassilevska Williams, editors, *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, pages 750–760. ACM, 2021. doi:10.1145/3406325.3451116.

[IKLP06] Yuval Ishai, Eyal Kushilevitz, Yehuda Lindell, and Erez Petrank. Black-box constructions for secure computation. In Jon M. Kleinberg, editor, *Proceedings of the 38th Annual ACM Symposium on Theory of Computing, Seattle, WA, USA, May 21-23, 2006*, pages 99–108. ACM, 2006. doi:10.1145/1132516.1132531.

[IKOS07] Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Zero-knowledge from secure multiparty computation. In David S. Johnson and Uriel Feige, editors, *Proceedings of the 39th Annual ACM Symposium on Theory of Computing, San Diego, California, USA, June 11-13, 2007*, pages 21–30. ACM, 2007. doi:10.1145/1250790.1250794.

[IKSS21] Yuval Ishai, Dakshita Khurana, Amit Sahai, and Akshayaram Srinivasan. On the round complexity of black-box secure MPC. In Tal Malkin and Chris Peikert, editors, *Advances in Cryptology - CRYPTO 2021 - 41st Annual International Cryptology Conference, CRYPTO 2021, Virtual Event, August 16-20, 2021, Proceedings, Part II*, volume 12826 of *Lecture Notes in Computer Science*, pages 214–243. Springer, 2021. doi:10.1007/978-3-030-84245-1_8.

[IKSS22a] Yuval Ishai, Dakshita Khurana, Amit Sahai, and Akshayaram Srinivasan. Round-optimal black-box protocol compilers. In Orr Dunkelman and Stefan Dziembowski, editors, *Advances in Cryptology - EUROCRYPT 2022 - 41st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Trondheim, Norway, May 30 - June 3, 2022, Proceedings, Part I*, volume 13275 of *Lecture Notes in Computer Science*, pages 210–240. Springer, 2022. doi:10.1007/978-3-031-06944-4_8.

[IKSS22b] Yuval Ishai, Dakshita Khurana, Amit Sahai, and Akshayaram Srinivasan. Round-optimal black-box secure computation from two-round malicious OT. 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 441–469. Springer, 2022. doi:10.1007/978-3-031-22365-5_16.

[IKSS23] Yuval Ishai, Dakshita Khurana, Amit Sahai, and Akshayaram Srinivasan. Round-optimal black-box MPC in the plain model. In Helena Handschuh and Anna Lysyanskaya, editors, *Advances in Cryptology - CRYPTO 2023 - 43rd Annual International Cryptology Conference, CRYPTO 2023, Santa Barbara, CA, USA, August 20-24, 2023, Proceedings, Part I*, volume 14081 of *Lecture Notes in Computer Science*, pages 393–426. Springer, 2023. doi:10.1007/978-3-031-38557-5_13.

[JJ21] Abhishek Jain and Zhengzhong Jin. Non-interactive zero knowledge from sub-exponential DDH. In Anne Canteaut and François-Xavier Standaert, editors, *Advances in Cryptology - EUROCRYPT 2021 - 40th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, October 17-21, 2021, Proceedings, Part I*, volume 12696 of *Lecture Notes in Computer Science*, pages 3–32. Springer, 2021. doi:10.1007/978-3-030-77870-5_1.

[JKKZ21] Ruta Jawale, Yael Tauman Kalai, Dakshita Khurana, and Rachel Yun Zhang. Snargs for bounded depth computations and PPAD hardness from sub-exponential LWE. In Samir Khuller and Virginia Vassilevska Williams, editors, *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, pages 708–721. ACM, 2021. doi:10.1145/3406325.3451055.

[Kil88] Joe Kilian. Founding cryptography on oblivious transfer. In Janos Simon, editor, *Proceedings of the 20th Annual ACM Symposium on Theory of Computing, May 2-4, 1988, Chicago, Illinois, USA*, pages 20–31. ACM, 1988. doi:10.1145/62212.62215.

[KMO14] Susumu Kiyoshima, Yoshifumi Manabe, and Tatsuaki Okamoto. Constant-round black-box construction of composable multi-party computation protocol. In Yehuda Lindell, editor, *Theory of Cryptography - 11th Theory of Cryptography Conference, TCC 2014, San Diego, CA, USA, February 24-26, 2014.*

*Proceedings*, volume 8349 of *Lecture Notes in Computer Science*, pages 343–367. Springer, 2014. doi:10.1007/978-3-642-54242-8_15.

[KO04] Jonathan Katz and Rafail Ostrovsky. Round-optimal secure two-party computation. In Matthew K. Franklin, editor, *Advances in Cryptology - CRYPTO 2004, 24th Annual International CryptologyConference, Santa Barbara, California, USA, August 15-19, 2004, Proceedings*, volume 3152 of *Lecture Notes in Computer Science*, pages 335–354. Springer, 2004. doi:10.1007/978-3-540-28628-8_21.

[KRR17] Yael Tauman Kalai, Guy N. Rothblum, and Ron D. Rothblum. From obfuscation to the security of fiat-shamir for proofs. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology - CRYPTO 2017 - 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part II*, volume 10402 of *Lecture Notes in Computer Science*, pages 224–251. Springer, 2017. doi:10.1007/978-3-319-63715-0_8.

[Lin13] Yehuda Lindell. Fast cut-and-choose based protocols for malicious and covert adversaries. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part II*, volume 8043 of *Lecture Notes in Computer Science*, pages 1–17. Springer, 2013. doi:10.1007/978-3-642-40084-1_1.

[LP07] Yehuda Lindell and Benny Pinkas. An efficient protocol for secure two-party computation in the presence of malicious adversaries. In Moni Naor, editor, *Advances in Cryptology - EUROCRYPT 2007, 26th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Barcelona, Spain, May 20-24, 2007, Proceedings*, volume 4515 of *Lecture Notes in Computer Science*, pages 52–78. Springer, 2007. doi:10.1007/978-3-540-72540-4_4.

[LP12] Huijia Lin and Rafael Pass. Black-box constructions of composable protocols without set-up. In Reihaneh Safavi-Naini and Ran Canetti, editors, *Advances in Cryptology - CRYPTO 2012 - 32nd Annual Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2012. Proceedings*, volume 7417 of *Lecture Notes in Computer Science*, pages 461–478. Springer, 2012. doi:10.1007/978-3-642-32009-5_27.

[LV22] Alex Lombardi and Vinod Vaikuntanathan. Correlation-intractable hash functions via shift-hiding. In Mark Braverman, editor, *13th Innovations in Theoretical Computer Science Conference, ITCS 2022, January 31 - February 3, 2022, Berkeley, CA, USA*, volume 215 of *LIPIcs*, pages 102:1–102:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.ITCS.2022.102.

[MOSV22] Varun Madathil, Chris Orsini, Alessandra Scafuro, and Daniele Venturi. From privacy-only to simulatable OT: black-box, round-optimal, information-theoretic. In Dana Dachman-Soled, editor, *3rd Conference on Information-Theoretic Cryptography, ITC 2022, July 5-7, 2022, Cambridge, MA, USA*, volume 230 of *LIPIcs*, pages 5:1–5:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.ITC.2022.5.

[ORS15] Rafail Ostrovsky, Silas Richelson, and Alessandra Scafuro. Round-optimal black-box two-party computation. In Rosario Gennaro and Matthew Robshaw, editors, *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part II*, volume 9216 of *Lecture Notes in Computer Science*, pages 339–358. Springer, 2015. doi:10.1007/978-3-662-48000-7_17.

[PS18] Chris Peikert and Sina Shiehian. Privately constraining and programming prfs, the LWE way. In Michel Abdalla and Ricardo Dahab, editors, *Public-Key Cryptography - PKC 2018 - 21st IACR International Conference on Practice and Theory of Public-Key Cryptography, Rio de Janeiro, Brazil, March 25-29, 2018, Proceedings, Part II*, volume 10770 of *Lecture Notes in Computer Science*, pages 675–701. Springer, 2018. doi:10.1007/978-3-319-76581-5_23.

[PS19] Chris Peikert and Sina Shiehian. Noninteractive zero knowledge for NP from (plain) learning with errors. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology - CRYPTO 2019 - 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2019, Proceedings, Part I*, volume 11692 of *Lecture Notes in Computer Science*, pages 89–114. Springer, 2019. doi:10.1007/978-3-030-26948-7_4.

[PW09] Rafael Pass and Hoeteck Wee. Black-box constructions of two-party protocols from one-way functions. In Omer Reingold, editor, *Theory of Cryptography, 6th Theory of Cryptography Conference, TCC 2009, San Francisco, CA, USA, March 15-17, 2009. Proceedings*, volume 5444 of *Lecture Notes in Computer Science*, pages 403–418. Springer, 2009. doi:10.1007/978-3-642-00457-5_24.

[Rab05] Michael O. Rabin. How to exchange secrets with oblivious transfer. *IACR Cryptol. ePrint Arch.*, page 187, 2005. URL: http://eprint.iacr.org/2005/187.

[Wee10] Hoeteck Wee. Black-box, round-efficient secure computation via non-malleability amplification. In *51th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2010, October 23-26, 2010, Las Vegas, Nevada, USA*, pages 531–540. IEEE Computer Society, 2010. doi:10.1109/FOCS.2010.87.