Efficient, Round-optimal, Universally-Composable Oblivious Transfer and Commitment Scheme with Adaptive Security

Megha Byali, Arpita Patra, Divya Ravi and Pratik Sarkar

Indian Institute of Science, India
{megha,arpita,divyar,pratiks}@iisc.ac.in

Abstract. Adaptive security embodies one of the strongest notions of security that allows an adversary to corrupt parties at any point during protocol execution and gain access to its internal state. Since it models real-life situations such as "hacking", adaptively-secure multiparty computation (MPC) protocols are desirable. Such protocols demand primitives such as oblivious transfer (OT) and commitment schemes that are adaptively-secure as building blocks. Efficient realisations of these primitives have been found to be challenging when no erasures is assumed. In this paper, we provide efficient constructions for these primitives that are Universally-Composable.

- *Adaptively-Secure Oblivious Transfer.* We present the first *round optimal* adaptively-secure OT based on the 2-round static OT protocol of *Peikert* et al. (Crypto 2008). Our protocol is in the programmable random oracle (PRO) model. It incurs a minimal communication overhead of one κ bit string and computational overhead of 5 random oracle queries over its static counterpart, where κ is the security parameter. Additionally, we present a construction of adaptively-secure 1-out-of-N OT by extending the result of *Naor* et al. (Journal of Cryptology 2005) that transforms $\log \mathsf{N}$ copies of 1-out-of-2 OTs to one 1-out-of-N OT. Based on PRO assumption, we prove that the transformation is adaptively-secure at the expense of $\mathcal{O}(\log \mathsf{N})$ exponentiations whereas, the existing state-of-the-art protocols for adaptively-secure 1-out-of-N OT incur at least $\mathcal{O}(\mathsf{N})$ exponentiations. Interestingly, it can be established that our transformation continues to be adaptively-secure, despite replacing the adaptively-secure 1-out-of-2 OTs in the above result with statically-secure OTs, that support equivocation of receiver's view irrespective of equivocation of sender's view.
- *Adaptively-Secure Commitment Scheme.* We provide a *round optimal* non-interactive commitment scheme (NICOM) based on the observable random oracle (ORO) assumption in the CRS model. Our construction incurs communication of 4κ bit strings and computation of 4 exponentiations and 2 random oracle queries for committing to an arbitrary length message. Additionally, we present a statically-secure scheme for one-time generation of CRS that can be reused for multiple commitments. This eliminates the need of a trusted CRS setup for the commitment scheme, thereby reducing the assumptions solely to ORO. The static version of our NICOM finds applications in secure two-party computation (2PC) protocols that adopt offline-online paradigm, where the CRS can be generated in the offline phase.

1 Introduction

Secure multiparty computation (MPC) [Yao82, GMW87] is an area of cryptography that deals with multiple parties who wish to compute a joint function of their private inputs such that the goals of MPC i.e. correctness and privacy of inputs remain intact. Various security notions of MPC are defined in accordance with different types of adversaries. The area of MPC achieving security against a malicious adversary causing static corruptions has been the center of attention in the recent past. Here, static corruption refers to the model in which the adversary corrupts the subset of parties at the outset of the protocol. Most widely known protocols of [Lin13, HKK+14, RR16, MR17], consider malicious security against static corruptions. Although static security is of interest, it is desirable to achieve security against adaptive adversary. Adaptive security is a stronger corruption model that allows the adversary to choose which parties to corrupt during the protocol execution and models real-life situations in a more comprehensive way. For instance, it captures the event of "hacking", where a hacker can illegally capitalize on the system and corrupt any workstation while protocols are in execution. Adaptive security is further classified based on secure erasures of the memory. An adaptively-secure protocol assuming erasure allows secure erasure of a workstation's internal memory once it is corrupted by a hacker. [Can01] argued that security relying on erasures often leads to problems and is impractical, especially for real-life systems. It requires an inherent trust assumption on the part of a workstation that it will erase its memory upon being corrupted. Hence, adaptive security without erasures (referred to simply as *adaptive security* throughout the paper) is preferable as it correctly models real world "hacking" attacks. However, the current literature of MPC dealing with adaptive adversaries is less explored compared to static security since it turns out to be considerably more challenging. In this paper, we explore the less traveled path of dealing with adaptive adversaries in the Universal Composability (UC) model of [Can01, CLOS02]. We aim at obtaining efficient, adaptively-secure primitives which are instrumental in construction of adaptively-secure MPC protocols. We focus on two such primitives in particular: Oblivious Transfer (OT) and Commitment schemes.

In the literature, OT has been regarded as the fundamental primitive [Rab81, BCR86, Kil88, NP05, IPS08], known to be complete for MPC [Kil88]. Following that, many flavors of OT such as 1-out-of-2 OT [PVW08, CO15], 1-out-of-N OT [NP05], k-out-of-N OT [GH08], have been explored in past. In its most basic form, a 1-out-of-2 OT consists of two parties, sender S and a receiver R. S has input messages say m_0, m_1 and R has a choice bit b. At the end of the protocol, R obtains message m_b corresponding to his choice bit and nothing else. S remains oblivious to the message obtained by R. Another important primitive in the MPC literature is commitment scheme. Informally, we describe a commitment scheme as follows: The sender S commits to a message m in a commitment c and sends c to the receiver R in the COMMIT phase. In the DECOMMIT phase, R learns the message m, along with some decommitment information, such that R is convinced that indeed m was committed in the COMMIT phase. UC secure commitment scheme has been proven to be a powerful tool. It implies key exchange [DG03] and more general two-party computation (2PC) and MPC [CLOS02]. Moreover, the seminal work of [CF01] proved that UC secure commitments are non-malleable in na-

ture. In this work, we explore round-optimal OT and commitment protocols which are adaptively-secure.

1.1 Related Work

The extensive use of fundamental building blocks of MPC has led to substantial work towards attaining efficient primitives. Since our work primarily focuses on universally composable OT and commitment schemes, we outline only the relevant literature below.

Oblivious Transfer. The literature of OT is vast and quite diverse in terms of assumptions and security. We highlight few works that are closely related to ours. Firstly, in the standalone model, the works of [NP01, AIR01, HK12, Lin08] are statically-secure against malicious adversary. Secondly, in the UC model, [CLOS02] proposed the first UC secure OT protocol based on general assumptions. Their work includes construction of both static and adaptively UC-secure OTs. Despite being inefficient, [CLOS02] motivated research towards obtaining OTs in the UC model.

In the setting of static security, [GMY04] presented a constant round committed bit-OT under Decisional Diffie Hellman (DDH) and RSA assumptions. The work of [JS07] proposed a four round, UC-secure protocol under the Decisional Composite Residue (DCR) assumption. [HK07] provided the first round-optimal protocol that is UC-secure, assuming common reference string (CRS). This was followed by the seminal work of Peikert *et al*. [PVW08], that provided a general framework for round optimal UC-secure OT protocols along with efficient instantiations based on DDH, Quadratic Residue and LWE assumptions in the CRS model. We denote the popular DDH based construction of the [PVW08] paper as PVW protocol in rest of the paper.

In the setting of adaptive security, [GWZ09] followed the compiler approach to transform the PVW OT protocol into an adaptively-secure OT in the CRS model. Their protocol incurred an overhead of $\mathcal{O}(n)$ exponentiations, where n is the size of sender's input messages to the OT. Concurrently, the papers of [CDMW09a, CDMW09b] proposed theoretical constructions of adaptively-secure OT. [CDMW09b] follows a compiler approach for transforming a protocol that is secure against a semi-honest adaptive adversary into one that is secure against a malicious adaptive adversary. [CDMW09a] presented an optimized non-committing encryption (NCE) scheme based on trapdoor simulatable cryptosystem. [CKWZ13] presented a framework for adaptively-secure OT with erasures in the global CRS model. They provide instantiations of protocols under various assumptions such as DLIN, Symmetric External Diffie Hellman (SXDH), DDH and DCR. However, their protocols are not round optimal and achieve adaptive security at the cost of significant overhead in communication and computation compared to the PVW protocol. [CKWZ13] also provided two explicit constructions (Appendix A of [CKWZ13]) of [GWZ09] framework [GWZ09] under DDH and Decisional Linear (DLIN) assumptions. These instantiations are adaptively-secure with erasures, with computational overhead reduced to constant number of exponentiations. Recently, [BCG17] proposed a 3-round OT protocol in the CRS model assuming erasures.

The work of [CO15], the "simplest OT" protocol explored OT constructions in the programmable random oracle (PRO) model, without assuming CRS, to provide a 3-round protocol that outperforms all other UC-secure OT protocols in terms of

both communication and computation. The authors of [CO15] claim that their construction achieves adaptive security. However, a number of bugs have been identified [GIR17, BDD+17, HL17] recently in their static security proof thereby rendering the protocol of [CO15] insecure in the UC model. Consequently, the problem of attaining round-optimal, efficient and adaptively-secure OT that continued to remain open, is affirmatively answered in this paper. Concurrent to our work, [BDD+17, HL17] claim to achieve adaptive security in the UC model under the PRO assumption. However, in Section 3, we give a justification that while these protocols are not UC-secure. Table 1 summarizes the literature on adaptively-secure OT protocols and our result. We do not include [GWZ09, CDMW09a, CDMW09b] in the comparison since they require $\mathcal{O}(n)$ exponentiations, when sender's input message is of n bits. While the other protocols in the table require constant number of exponentiations.

Table 1. Comparison among UC secure Oblivious Transfer Protocols

Protocol	Communication (κ-bit strings / Group elements)	Computation				Rounds	Assumptions	Setup	Security
		Sender		Receiver					
		SKE	PKE	SKE	PKE				
[GWZ09] + [Lin11][a]	51	2	34	2	31	8	DDH	CRS	Adaptive with erasures
[GWZ09] + [FLM11][b]	83	3	26	3	72	4	DLIN	CRS	Adaptive with erasures
[CKWZ13]	59	3	$\geq$14	2	$\geq$27	3	DLIN	CRS	Adaptive with erasures
[CKWZ13]	43	3	$\geq$8	2	$\geq$15	3	SXDH	CRS	Adaptive with erasures
[CKWZ13]	35	4	19	3	37	4	DDH	CRS	Adaptive with erasures
[CKWZ13]	28	4	13	3	26	4	DCR	CRS	Adaptive with erasures
[ABB+13]	15	2	13	1	11	3	SXDH	CRS	Adaptive with erasures
[BCG17]	10	4	18	4	9	3	SXDH	CRS	Adaptive with erasures
[PVW08]	6	-	8	-	3	2	DDH	CRS	Static
Our scheme	7	3	8	2	3	2	DDH	PRO	Adaptive

Notations:
SKE - symmetric key encryptions, PKE - exponentiations,
DDH - Decisional Diffie Hellman, DLIN - Decisional Linear, SXDH - symmetric external Diffie Hellman,
CDH - Computational Diffie Hellman, DCR - Decisional Composite Residuosity, PRO - programmable random oracle
a The commitment scheme used for instantiation is of [Lin11].
b The commitment scheme used for instantiation is of [FLM11].

Commitment Schemes. The study of UC secure commitment schemes was initiated by the seminal work of [CF01]. It was followed by the works of [CLOS02, DG03, HM04, Lin11, Fuj16] and many more. We highlight some of the notable works in the relevant literature below.

The contributions of [DG03, Lin11, GIKW14, Fuj16, FLM11, CJS14] based on hardness assumptions such as DDH, DLIN and Discrete Log (DLP) are interactive (involve either an interactive COMMIT or DECOMMIT phase) in nature. In contrast, the contributions of [CLOS02, CF01, HM04, FLM11, NFT09] present non-interactive commitment (NICOM) schemes where both COMMIT and DECOMMIT phases are non-interactive.

The offline-online paradigm also forms an interesting flavour of NICOM schemes. In this setting, the notable works include [CDD+15, DDGN14]. These schemes consist of an input independent setup phase for preprocessing of data. The cost of preprocessing is amortized over multiple commitments.

The literature regarding commitment schemes is concentrated mainly around static security [Lin11, BCPV13, CJS14, GIKW14, DDGN14, CDD+15, Fuj16, CDD+16]. Building commitment schemes against adaptive adversaries has been a challenging task, since it involves equivocation of the internal states of the parties in case corruption occurs. There have been few contributions in the past addressing adaptive security. Adaptively-secure schemes can be broadly classified into two categories based on their ability to erase the internal state of the party when corruption occurs. [Fuj16, BCPV13, FLM11, NFT09] proposed adaptively-secure protocols which rely on secure erasure of the party's internal state whereas the constructions of [HM04, DG03, CLOS02, CF01] achieved the same level of security without erasures.

Since our focus is on adaptively-secure NICOM schemes, we elaborate on the relevant results as follows: [CLOS02, CF01] provided schemes for bit commitments, communicating at least $\mathcal{O}(\kappa^2)$ bits for committing to a κ bit string. [HM04] provided the first efficient NICOM for an arbitrary length message in the RO model and involves communication of constant number of bits for commitment. Programmability of RO is used to attain the property of equivocation in [HM04]. From the literature we can observe that, there is immense scope of improvement in terms of efficiency and assumptions in the adaptive world and our paper presents a construction of an adaptively-secure NICOM in the CRS model assuming ORO. Table 2 consolidates the comparison of various UC secure commitment schemes alongside our protocol.

Table 2. Comparison among UC secure commitment schemes

Protocols	Message Size (bits)	Communication (κ-bit strings / Group elements)	Rounds (Commit/ Decommit)	Assumptions	Setup	Security
[Lin11]	κ	14	1/4	DDH + CRHF	CRS	Static
[Fuj16]	κ	10	1/3	DDH + CRHF	CRS	Static
[BCPV13]	κ	12	1/3	DDH+CRHF	CRS	Static
[CDD+16]	κ	$1 + o(1)$	5/1	OT	CRS	Static
[CDD+15]	κ	$\geq 9 +$ $\mathcal{O}(\kappa^2)$ (one-time)	1/1 + 5 (one-time)	OT	CRS	Static
[CJS14]	poly(κ)	7	2/3	DLP	ORO	Static
Our Scheme	poly(κ)	$4 +$ $\mathcal{O}(\mu\lvert\mathcal{C}\rvert)$ (one-time)	1/1 + 4 (one-time)	DLP	ORO	Static
[FLM11]	κ	21	1/1	DLIN + CRHF	CRS	Adaptive with erasures
[NFT09]	κ	7	1/1	DDH + sEUF-OT	CRS (Non-reusable)	Adaptive with erasures
[Fuj16]	κ	10	3/1	DDH + CRHF	CRS	Adaptive with erasures
[BCPV13]	κ	14	3/1	DDH+CRHF	CRS	Adaptive with erasures
[CF01]	κ	$\mathcal{O}(\kappa)$	1/1	DDH + UOWHF	CRS	Adaptive
[CLOS02]	κ	$\mathcal{O}(\kappa)$	1/1	TDP	CRS	Adaptive
[DG03]	κ	48	3/1	DDH + CRHF	CRS	Adaptive
[HM04]	poly(κ)	5	1/1	OWF	PRO	Adaptive
Our Scheme	poly(κ)	4	1/1	DLP	CRS + ORO	Adaptive

Notations:
DDH - Decisional Diffie Hellman, CRHF - collision resistant hash function, DLP - Discrete Log Problem, DLIN - Decisional Linear, sEUF-OT - strongly unforgeable one-times signature, TDP - trapdoor permutations, UOWHF - universal one-way hash functions, OWF - one-way functions, ORO - observable random oracle, PRO - programmable random oracle, circuit $\mathcal{C}$ computes g^x
Note : The protocol of [CJS14] requires a trapdoor commitment scheme which has been instantiated with Pedersen commitment

1.2 Our Results

In this work, we focus on optimizing the round complexity while attaining adaptive security for two widely used primitives: Oblivious Transfer and Commitment Scheme. A well-defined transformation from 1-out-of-2 OTs to obtain 1-out-of-N OT that minimizes number of exponentiations besides achieving adaptive security is also established in this work. We also point out a bug in two of the concurrent works on UC-secure adaptive OT protocols. Our constructions are stated below.

Attack in Concurrent Works on UC-secure Adaptive OT. Concurrent to ours, the works of [BDD+17, HL17] on OT, claim adaptive UC security in the PRO model. However, we observe that both these protocols are prone to a bug when we consider UC-security against a corrupt receiver R^*. We give a justification that these protocols are not secure in the UC model even tolerating a static adversary.

Adaptively Secure 1-out-of-2 Oblivious Transfer. We construct a fundamental building block of MPC, the first Oblivious Transfer protocol that is round-optimal, universally composable and adaptively-secure assuming no erasures. Our construction is motivated by the vital observation of [CO15] that the CRS in OT can be replaced with the PRO. Further, PRO enables equivocation of S's view if utilized appropriately. Thus, we replace CRS in round-optimal PVW protocol with PRO to achieve adaptive security. Additionally, we rely on hardness of the DDH assumption. Despite achieving a stronger notion of security, our protocol incurs a computation overhead of 5 random oracle queries over the static protocol of PVW. In terms of communication, our protocol has merely an overhead of one κ-bit string compared to PVW.

Our construction of adaptively secure 1-out-of-2 OT uses the protocol of [PVW08] as building block. However, the proof of adaptive security poses several challenges: First, it is necessary to set an appropriate CRS (DDH or non-DDH) in accordance to the party being corrupt to facilitate input extraction. Second, we need a mechanism to equivocate the view of receiver/sender upon corruption during or at the end of execution. We counter these challenges with the use of few neat ideas along with a PRO as follows: To address the former case, we generate the CRS appropriately (DDH or non-DDH) as demanded by the simulation proof using PRO. The simulator aptly programs the RO to generate the CRS as a non-DDH tuple in case receiver corruption occurs at the outset of protocol. This enables the simulator to extract receiver's input. In case receiver corruption does not occur at the outset, the simulator programs the RO to generate the CRS as a DDH tuple. This enables extraction of sender's input. However, the real execution would have a non-DDH CRS, except with negligible probability. The case for equivocation is addressed at the receiver side, when the receiver gets corrupted after the outset of the protocol, using the equivocation property that comes in handy as CRS is set to a DDH tuple. At the sender side, we utilize the programmability of random oracle and the hardness assumption of indistinguishability between a DDH and random tuple to enable equivocation. An elaborate discussion on these issues is made in the proof.

Adaptively Secure 1-out-of-N Oblivious Transfer. The work of [NP05] established that $\log \mathsf{N}$ copies of 1-out-of-2 OTs can be transformed to obtain one 1-out-of-N OT,

which is statically-secure against active adversaries. This transformation implies existence of statically-secure 1-out-of-N OT at the expanse of $\mathcal{O}(\log \mathsf{N})$ exponentiations. We extend their result to provide a formal proof that the transformation satisfies adaptive security under PRO assumption. At present, one adaptive 1-out-of-N OT protocol [ABB$^+$13, BC15, BC16, BCG17] incurs atleast $\mathcal{O}(\mathsf{N})$ exponentiations. Our adaptive transformation brings down the number of exponentiations to $\mathcal{O}(\log \mathsf{N})$; thereby matching the efficiency of statically-secure 1-out-of-N OT. Interestingly, we can show that if the 1-out-of-2 OTs support equivocation of receiver's view irrespective of equivocation of sender's view then it is possible to generate adaptively-secure 1-out-of-N OT from our transformation. This implies that we can plug-in statically-secure 1-out-of-2 OTs which satisfy the property of receiver equivocability.

Commitment Schemes. We construct a NICOM that is UC-secure against an adaptive adversary without erasures. Our commitment scheme is based on the observable random oracle (ORO) assumption in the CRS model. Under the hood, the NICOM relies on the Pedersen Commitment [Ped91] for equivocation and the ORO for extraction of a corrupted committer's input. Moreover, the ORO permits committing to a message of length ℓ while incurring the overhead of committing to a single κ bit string, where $\ell = \mathsf{poly}(\kappa)$. Compressing the message from ℓ bits to κ bits does not break binding since we are in the RO model, where it is hard to find two different messages of arbitrary length s.t. the RO returns the same result upon being queried on those two messages. Our protocol involves communication of one κ bit string and three group elements and computation of 4 exponentiations and 4 random oracle queries to commit to ℓ bits.

For completeness, we present a 4 round protocol (a.k.a setup phase) for generation of the CRS for our NICOM, relying on garbled circuits. Once generated, the CRS can be reused for several instances of the commitment scheme. The COMMIT and DECOMMIT phases still remain non-interactive. The reduces our setup assumption to ORO only. However, our setup phase is statically-secure, rendering the resulting commitment protocol to be only statically-secure. If the setup phase is adaptively-secure, then our commitment scheme would be adaptively-secure in the ORO model with no CRS. Despite the above fact, we remark that our statically-secure scheme is an interesting result that can find useful applications in offline-online secure two-party computation (2PC) protocols where the setup phase can be executed in the offline phase while the NICOM can be conveniently used in the online phase. Similar setting has already been considered in the works of [CDD$^+$15, DDGN14] where they have a preprocessing phase and non-interactive COMMIT and DECOMMIT phases. Their protocol requires communication of $\mathcal{O}(\ell + \kappa)$ bits for committing to a message of size ℓ whereas we always need 4κ bits due to the RO.

Our commitment schemes inherently promote commitment length extension. Commitment length extension [GIKW14] is a UC commitment protocol for a long message using a single ideal commitment for a short message. Our commitment length extension does not incur any extra overhead over the commitment to the κ bit string. This also implies that our protocol satisfies a rate which is better than rate-1 [GIKW14] in the COMMIT phase, where rate-r means that the commitment size is at most $r + o(1)$ times of message size. The state-of-the-art UC secure commitment schemes [GIKW14, CDD$^+$16] in the CRS model achieve rate-1 and require communication of bits pro-

portional to the message size, whereas we require 4κ bits only. Table 2 compares our commitment schemes with recent literature.

1.3 Roadmap

The high-level overview of the primitives and notation used is presented in Section 2. In Section 3 we explain the bugs present in the security proofs of [CO15, BDD+17, HL17]. Then we present our adaptively-secure 1-out-of-2 OT protocol appear in Sec. 4. The adaptively-secure 1-out-of-N OT is elaborated in Section 5. The adaptively-secure, non-interactive UC commitment scheme is presented in Sec. 6. The statically-secure scheme for one-time generation of CRS that can be reused is presented in the same section. Finally, we conclude with prospective future work in Section 7. We provide a summary of UC framework for static and adaptive security in Appendix A and B respectively for the sake of completeness. We refer to it for better comprehension of notations used in our security proofs.

2 Preliminaries

In this section, we describe the notations used in our protocols and give a high-level overview of the primitives that are used in the paper.

Notations. For the OT and NICOM protocol, we denote the sender by S and receiver by R. We denote by $a \leftarrow_R D$ the random sampling of a from a distribution D and the set of elements $\{1, \ldots, n\}$ is represented by $[n]$. Let PRF denote a secure pseudorandom function. A function $\mathsf{neg}(\cdot)$ is said to be negligible, if for every polynomial $p(\cdot)$, there exists a constant c, such that for all $n > c$, it holds that $\mathsf{neg}(n) < \frac{1}{p(n)}$. We denote a probabilistic polynomial time algorithm as PPT. We denote the statistical security parameter by μ and the computational security parameter by κ. We use $\overset{c}{\approx}$ and $\overset{s}{\approx}$ to denote computational and statistical indistinguishability, respectively. Let $\mathbb{G}$ be a multiplicative group of prime order p with generator g and $\mathbb{Z}_p$ denote the prime field of order p. For a bit $b \in \{0, 1\}$, we denote $1 - b$ by $\overline{b}$.

Garbled Circuits. The term 'garbled circuit' (GC) was coined by Beaver [BMR90] and used extensively only as a technique in secure protocols until they were formalized as a primitive by Bellare et al. [BHR12]. We use notations consistent with [BHR12] (described in Appendix C) for the garbling primitive used to generate CRS for our commitment scheme. Let GC_k denote the k^{th} garbled circuit instantiating circuit $\mathcal{C}$. We assume that the randomness used for generating circuit k is derived from a κ-bit random string seed_k using a PRF. We assume that the fan-in of each gate is 2. We can assume that each AND gate in the circuit has 2 ciphertexts and XOR gates have 0 ciphertexts, using the Half-Gate construction [ZRE15] as the garbling scheme.

Oblivious Transfer. Oblivious transfer (OT) is a protocol between a sender (S) and a receiver (R). In a 1-out-of-2 OT, the sender holds two inputs $a_0, a_1 \in \{0, 1\}^n$ and the receiver holds a choice bit σ. At the end of the protocol, the receiver obtains a_σ. The

sender learns nothing about the choice bit, and the receiver learns nothing about the sender's other input $a_{\overline{\sigma}}$. The ideal OT functionality $\mathcal{F}_{\mathsf{OT}}$ is recalled below in Figure 1. Similarly a 1-out-of-N OT can be defined as $\mathcal{F}_{\mathsf{N\text{-}OT}}$ functionality in Figure 2.

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

Choose: On input $(\mathsf{rec}, \mathsf{sid}, \sigma)$ from R where $\sigma \in \{0,1\}$; if no message of the form $(\mathsf{rec}, \mathsf{sid}, \sigma)$ has been recorded in the memory, store $(\mathsf{rec}, \mathsf{sid}, \sigma)$ and send $(\mathsf{rec}, \mathsf{sid})$ to S.

Transfer: On input $(\mathsf{sen}, \mathsf{sid}, (a_0, a_1))$ from S with $a_0, a_1 \in \{0,1\}^n$, if no message of the form $(\mathsf{sen}, \mathsf{sid}, (a_0, a_1))$ is recorded and a message of the form $(\mathsf{rec}, \mathsf{sid}, \sigma)$ is stored, send $(\mathsf{sent}, \mathsf{sid}, a_\sigma)$ to R and $(\mathsf{sent}, \mathsf{sid})$ to S.

Fig. 1. The ideal functionality $\mathcal{F}_{\mathsf{OT}}$ for oblivious transfer

$\mathcal{F}_{\mathsf{N\text{-}OT}}$

Choose: On input $(\mathsf{rec}, \mathsf{sid}, \sigma)$ from R where $\sigma \in \{0,1\}^{\log \mathsf{N}}$; if no message of the form $(\mathsf{rec}, \mathsf{sid}, \sigma)$ is present in memory, store $(\mathsf{rec}, \mathsf{sid}, \sigma)$ and send $(\mathsf{rec}, \mathsf{sid})$ to S.

Transfer: On input $(\mathsf{sen}, \mathsf{sid}, \{a_j\}_{j=1}^{\mathsf{N}})$ from S with $a_j \in \{0,1\}^n$, if no message of the form $(\mathsf{sen}, \mathsf{sid}, \{a_j\}_{j=1}^{\mathsf{N}})$ is present in memory and a message of the form $(\mathsf{rec}, \mathsf{sid}, \sigma)$ is stored, send $(\mathsf{sent}, \mathsf{sid}, a_\sigma)$ to R and $(\mathsf{sent}, \mathsf{sid})$ to S.

Fig. 2. The ideal functionality $\mathcal{F}_{\mathsf{N\text{-}OT}}$ for oblivious transfer

Commitment Schemes. Commitment schemes allow a party to commit to a chosen message while keeping it hidden, with the ability to reveal the committed message later. We denote an UC secure commitment to message m with randomness r as $\textsc{Com}(m; r)$. The ideal commitment functionality $\mathcal{F}_{\mathsf{COM}}$ has been depicted in Fig. 3.

Fig. 3. Functionality $\mathcal{F}_{\mathsf{COM}}$

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

$\mathcal{F}_{\mathsf{COM}}$ interacts with sender S and receiver R:

1. On receiving input $(\textsc{Commit}, sid, m)$ from S, if sid has been recorded, ignore the input. Else record the tuple $(sid, \mathsf{S}, \mathsf{R}, m)$ and send $(\textsc{Receipt}, sid, \mathsf{S}, \mathsf{R})$ to R.
2. On receiving input $(\textsc{Decommit}, sid)$ from S, if there is a record of the form $(sid, \mathsf{S}, \mathsf{R}, m')$ return $(\textsc{Decommit}, sid, m')$ to R. Otherwise, ignore the input.

Random Oracle Model. We rely on random oracles (RO) for our schemes. A random oracle query on message m is denoted by $\mathrm{H}(m)$. Our OT protocol requires programmability from the random oracle which allows the simulator to program $\mathrm{H}(m)$ to return

any string upon being queried on m for the first time. Our commitment scheme requires observability from the random oracle which allows the simulator to observe the queries made by the adversary to the oracle.

3 Attack in Concurrent Works on UC-secure Adaptive OT.

Concurrent to our work, the works of [BDD+17, HL17] on OT, claim adaptive UC security in the PRO model. [BDD+17] proposes a general framework for 2-round adaptive OT and provides instantiations under various assumptions such as Learning from Parity with Noise, McEliece cryptosystem, QC-MDPC, Learning With Errors and Computational Diffie Hellman (CDH). [HL17] proposes a construction of 1-out-of-N OT under the CDH assumption. However, both protocols as well as the 'simplest OT' construction of [CO15] are prone to a bug when we consider UC-security even against a statically corrupt receiver R^*. The attack stems from the late input extraction of a statically-corrupt R as detailed below. The simulator for the case of static corruption of R, playing the role of honest S, can only extract R^*'s input by observing R^*'s query to RO, made in an attempt to decrypt its chosen message on receiving the last OT message from the sender. This implies that a corrupt R^* can indefinitely delay the input extraction causing composition-related issues.

The delayed input extraction allows us to demonstrate that their constructions do not realise the OT functionality $\mathcal{F}_{\mathsf{OT}}$ presented in Fig. 1 where S obtains a notification from the functionality, denoting the end of ideal world execution. Rather, they realise only a weaker version of OT functionality $\mathcal{F}_{\mathsf{OT}}^{\mathsf{w}}$ as depicted in Fig. 4. In the weaker variant $\mathcal{F}_{\mathsf{OT}}^{\mathsf{w}}$, S does not obtain any notification from the functionality. Instead its role is limited to sending (sid, a_0, a_1) to $\mathcal{F}_{\mathsf{OT}}^{\mathsf{w}}$, after which S halts. However, $\mathcal{F}_{\mathsf{OT}}^{\mathsf{w}}$ is not composable and cannot be used in a bigger protocol to implement the oblivious transfer functionality. Our observation aligns with the work of [LM16], which states (in page 2, last para of Section 1) that the naive OT functionality (Fig. 1 in their paper), same as $\mathcal{F}_{\mathsf{OT}}^{\mathsf{w}}$, is not composable whereas the modified/revised OT functionality (Fig. 3 of their paper), same as our $\mathcal{F}_{\mathsf{OT}}$, can be proven to be composable. The late input extraction problem is referred to as "timing bug" in their paper (page 3, first paragraph). They explain the issue of composability due to timing bug in the naive OT functionality with an example of OT Extension protocol in Section 3. In Section 4, they address the issue by plugging in the revised OT functionality ($\mathcal{F}_{\mathsf{OT}}$ in our case). Interestingly, all the currently known UC-secure OT protocols, barring the three protocols of [CO15, BDD+17, HL17], implement both $\mathcal{F}_{\mathsf{OT}}$ and $\mathcal{F}_{\mathsf{OT}}^{\mathsf{w}}$ functionalities and hence they are composable. In what follows, we first show that the protocols of [CO15, BDD+17, HL17] do not realise $\mathcal{F}_{\mathsf{OT}}$ functionality, but realise only $\mathcal{F}_{\mathsf{OT}}^{\mathsf{w}}$. Next, we demonstrate the compositional issue of using $\mathcal{F}_{\mathsf{OT}}^{\mathsf{w}}$ in (yet another example) of 2PC protocol based on garbled circuit (GC) approach.

To show that the constructions that feature delayed input extraction (a.k.a timing bug) do not realise $\mathcal{F}_{\mathsf{OT}}$ functionality, we consider an adversarial strategy where R^* does not decrypt the last OT message. In the ideal world, Sim will not be able to extract R^*'s input as R^* does not proceed to decrypting its chosen message from the last OT message. Consequently, Sim fails to invoke $\mathcal{F}_{\mathsf{OT}}$ functionality with R^*'s input and

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

Choose: On input $(\mathsf{rec}, \mathsf{sid}, \sigma)$ from R where $\sigma \in \{0,1\}$; if no message of the form $(\mathsf{rec}, \mathsf{sid}, \sigma)$ has been recorded in the memory, store $(\mathsf{rec}, \mathsf{sid}, \sigma)$ and send $(\mathsf{rec}, \mathsf{sid})$ to S.

Transfer: On input $(\mathsf{sen}, \mathsf{sid}, (a_0, a_1))$ from S with $a_0, a_1 \in \{0,1\}^n$, if no message of the form $(\mathsf{sen}, \mathsf{sid}, (a_0, a_1))$ is recorded and a message of the form $(\mathsf{rec}, \mathsf{sid}, \sigma)$ is stored, send $(\mathsf{sent}, \mathsf{sid}, a_\sigma)$ to R.

Fig. 4. Weaker oblivious transfer functionality $\mathcal{F}_{\mathsf{OT}}^{\mathsf{w}}$

as a result, the $\mathcal{F}_{\mathsf{OT}}$ functionality keeps running. This causes a honest S in the ideal world keep waiting for notification from the $\mathcal{F}_{\mathsf{OT}}$ functionality in order to terminate. Therefore, while honest S *does not halt* in the ideal world, it halts in the real world immediately after sending its last message. This difference in the behavior of honest S can be used to distinguish between the two worlds by an environment. With $\mathcal{F}_{\mathsf{OT}}^{\mathsf{w}}$ functionality, in the scenario mentioned above, an honest S would halt in both the worlds, preserving indistinguishability.

We now illustrate the compositional issue resulted from using $\mathcal{F}_{\mathsf{OT}}^{\mathsf{w}}$ by means of a GC based 2PC protocol in the OT-hybrid model, where the evaluator E first involves with the constructor in a set of OT functionalities to choose the circuits that will be used for evaluation and respectively for checking and on completion of OTs, the constructor sends the GCs to the evaluator. When $\mathcal{F}_{\mathsf{OT}}$ was used, a simulator against a corrupt evaluator E^*, extracts the inputs of E^* from the OT and either constructs a fake/simulated GC or a real GC based on the extracted input of E^*. However in $\mathcal{F}_{\mathsf{OT}}^{\mathsf{w}}$-hybrid model, the simulator for a corrupt evaluator E^* (playing the role of R^*) cannot exact the E^*'s input bits to OT and hence cannot substitute certain GCs with simulated ones (without getting caught with high probability). This difference would enable the environment $\mathcal{Z}$ to distinguish between both worlds based on E^*'s view.

4 Adaptively-secure 1-out-of-2 Oblivious Transfer

The 2 round DDH-based OT protocol of PVW is proven to be statically-secure against a malicious adversary and requires CRS as a setup assumption. In this section, we present a DDH-based OT protocol along the same lines that is adaptively-secure. All the existing OT protocols for adaptive security need a minimum of three rounds besides setup assumptions. Our protocol relies on PRO assumption alone and does not require any additional setup. The communication overhead of our protocol compared to PVW is merely one κ bit string. With efficiency comparable to the [PVW08], ours is the first efficient adaptively-secure OT protocol that is round optimal. In the next section, we present our OT protocol π_{OT} (Fig. 5), implementing the ideal functionality $\mathcal{F}_{\mathsf{OT}}$ (Fig. 1). The detailed proof appears subsequently. For an easy sail, we first present static security of our protocol followed by more involved adaptive security proof.

4.1 The Protocol

We first present a brief overview of our protocol and its comparison with respect to the 2-round PVW protocol from DDH assumption and CRS. R commits to his choice bit σ by means of the first OT message. The second OT message sent by S (with input a_0, a_1) enables R to obtain the message a_σ. Our protocol π_{OT} differs from PVW in the following two aspects: First, [PVW08] assumes a trusted CRS setup whereas in our protocol, the receiver R obtains the CRS by querying a PRO (denoted by H_1) at the outset of protocol, which is then verified by the sender S. Second, our protocol needs an additional PRO (denoted by H_2) to formulate sender S's messages. This is essential for adaptive security and is elaborated in detail in the security proof.

Fig. 5. Adaptively-Secure Oblivious Transfer Protocol

π_{OT}

- **Public Inputs:** Generator g of group $\mathbb{G}$. Random Oracles $\mathrm{H}_1 : \{0,1\}^{2\kappa} \to 3\mathbb{G}$ and $\mathrm{H}_2 : \mathbb{G} \to \{0,1\}^{\ell}$.
- **Private Inputs:** S has input messages (a_0, a_1) and R has an input bit σ.

Choose:

- R samples $c \leftarrow_R \{0,1\}^{\kappa}$.
- R generates a tuple (g_0, g_1, h_0, h_1), s.t. $g_0 = \mathsf{g}$, $(g_1||h_0||h_1) = \mathrm{H}_1(\mathsf{sid}||c)$.
- R samples $\alpha \leftarrow_R \mathbb{Z}_p$ and computes $(g, h) = (g_\sigma^\alpha, h_\sigma^\alpha)$.
- R sends $\{c, (g, h)\}$ to S.

Transfer:

- S generates the tuple (g_0, g_1, h_0, h_1) with $g_0 = \mathsf{g}$, $(g_1||h_0||h_1) = \mathrm{H}_1(\mathsf{sid}||c)$.
- S samples $r_0, r_1, s_0, s_1 \leftarrow_R \mathbb{Z}_p$.
- S computes $u_0 = g_0^{r_0} h_0^{s_0}$ and $u_1 = g_1^{r_1} h_1^{s_1}$.
- S sets $w_0 = \mathrm{H}_2(g^{r_0} h^{s_0}) \oplus a_0$ and $w_1 = \mathrm{H}_2(g^{r_1} h^{s_1}) \oplus a_1$.
- S sends $\{(u_0, w_0), (u_1, w_1)\}$ to R.

Local Computation by R:

- Computes a_σ as $a_\sigma = w_\sigma \oplus \mathrm{H}_2(u_\sigma^\alpha)$.

4.2 Static Security

To make the proof of adaptive security more comprehensible, we first prove that π_{OT} realizes the ideal functionality $\mathcal{F}_{\mathsf{OT}}$ (Fig. 1) in presence of static adversaries. This is extended to adaptive security in Section 4.3.

In order to prove static security, we describe a simulator Sim who behaves as the ideal world adversary and generates a view of $\mathcal{Z}$ which is indistinguishable from the view generated by Adv in the real world. It does so by invoking $\mathcal{F}_{\mathsf{OT}}$, on behalf of the adversary in ideal world, and running a copy of Adv internally, in the head. We denote this internal adversary as $\mathsf{Adv}_{\mathsf{Int}}$. Sim simulates the role of the honest parties and the environment to $\mathsf{Adv}_{\mathsf{Int}}$ in the internal execution. Whenever Adv corrupts a party in the real world, $\mathsf{Adv}_{\mathsf{Int}}$ also corrupts that party in the internal execution and Sim corrupts that

party in the ideal world. At the end of the protocol $\mathsf{Adv}_{\mathsf{Int}}$ forwards its view to Sim who forwards it to $\mathcal{Z}$ as its ideal world view. We refer to Appendix A for clarity of $\mathsf{Adv}_{\mathsf{Int}}$ notation and details of static security in the UC model [Can01]. For static security, we prove Theorem 3 by considering the four exhaustive corruption cases namely : (1) Both S and R are honest (2) S^* is corrupt while R is honest (3) S is honest while R^* is corrupt (4) Both S^* and R^* are corrupt. In each of the above cases we describe a simulator Sim and show that the real world view of $\mathcal{Z}$ is indistinguishable from its ideal world view.

We first give a brief intuition of the security proof. Revisiting the proof of security for a static adversary [PVW08], note that Sim sets the CRS in accordance with which party is corrupt to enable input extraction of $\mathsf{Adv}_{\mathsf{Int}}$ in the internal execution. More specifically, while CRS of the internal execution is set to a non-DDH tuple (i.e. messy mode) when R *is corrupted*, it is set to a DDH tuple (i.e. decryption mode) when S *is corrupted*. In the former case, the relation between (g, h) can be suitably matched with either (g_0, h_0) or (g_1, h_1), extracting σ. In the latter case, knowing the relation (in the exponent) between g_0 and g_1 allows Sim to unlock both (a_0, a_1) from (w_0, w_1). The simulation of our protocol for static security is similar to that of PVW.

We are ready to present the formal proof of Theorem. 3. We design an ideal world adversary Sim who creates an ideal world view $\text{IDEAL}_{\mathcal{F},\mathsf{Sim},\mathcal{Z}}(1^\kappa, z)$ of $\mathcal{Z}$ which is indistinguishable from the real world view $\text{REAL}_{\mathcal{F},\mathsf{Adv},\mathcal{Z}}(1^\kappa, z)$ of $\mathcal{Z}$.

Theorem 1. *If* H_1 *and* H_2 *are programmable random oracles and solving Decisional Diffie Hellman is hard in multiplicative group* $\mathbb{G}$*, then protocol* π_{OT} *securely realizes the* $\mathcal{F}_{OT}$ *functionality in the presence of static active adversaries.*

The Simulator: We describe the simulator Sim for each possible case of corruption.

Case 1. S and R are honest: In this case Sim acts on behalf of both parties in the internal execution. At the end, $\mathsf{Adv}_{\mathsf{Int}}$ generates his view without corrupting any party and sends it to Sim who forwards it to $\mathcal{Z}$.

(i) *Simulating the CRS and* R*'s message:* Sim samples $c \leftarrow_R \{0,1\}^\kappa$ and programs H_1 to return a DDH tuple as the CRS upon querying c i.e $(g_1||h_0||h_1) \leftarrow \mathrm{H}_1(\mathsf{sid}||c)$ where $(g_0, g_1, h_0, h_1) = (\mathsf{g}, \mathsf{g}^x, \mathsf{g}^y, \mathsf{g}^{xy})$. On behalf of honest R, Sim samples $\alpha \leftarrow_R \mathbb{Z}$ and sets $(g, h) = (g_0^\alpha, h_0^\alpha) = (\mathsf{g}^\alpha, \mathsf{g}^{\alpha y})$. Sim sends the CRS, c and (g, h) as the first OT message to S on behalf of R in the internal execution.

(ii) *Simulating* S*'s message:* Sim sets (u_0, u_1) as per the protocol and (w_0, w_1) randomly. Sim sends the simulated message on behalf of S.

(iii) *Simulating* R*'s computation:* Sim completes the simulation on behalf of R in the internal world.

Case 2. S^* is corrupted and R is honest: In this case, Sim acts on behalf of R in the internal execution. At the end, $\mathsf{Adv}_{\mathsf{Int}}$ generates the view of S^* and sends it to Sim who forwards it to $\mathcal{Z}$.

(i) *Simulating the CRS and* R*'s message:* Same as Case 1.

(ii) $\mathsf{Adv}_{\mathsf{Int}}$ plays the role of S^* and computes the sender message in the internal world. $\mathsf{Adv}_{\mathsf{Int}}$ sends the second OT message to R in the internal world.

(iii) *Simulating R's computation:* Sim decrypts a_0 and a_1 using the trapdoor of the CRS, which was set as a DDH tuple, as follows:

$$a_0 = w_0 \oplus \mathrm{H}_2(u_0^{\alpha}), a_1 = w_1 \oplus \mathrm{H}_2(u_1^{\alpha x^{-1}})$$

where x is the trapdoor of the CRS. Sim sends (a_0, a_1) to $\mathcal{F}_{\mathsf{OT}}$ and completes the simulation on behalf of R in the internal world.

Case 3. S is honest and R^* is corrupted: In this case, Sim acts on behalf of S in the internal execution. At the end, $\mathsf{Adv}_{\mathsf{Int}}$ generates the view of R^* and sends it to Sim who forwards it to $\mathcal{Z}$.

(i) *Simulating the CRS:* $\mathsf{Adv}_{\mathsf{Int}}$ plays the role of R^* and computes the receiver message in the internal world. When $\mathsf{Adv}_{\mathsf{Int}}$ queries H_1 to generate the CRS, Sim programs it to return a non-DDH tuple. On receiving queries of the form $\mathrm{H}_1(\mathsf{sid}||c)$, Sim samples $x, y, z \leftarrow_R \mathbb{Z}_p$ s.t. the CRS is $(g_0, g_1, h_0, h_1) = (\mathsf{g}, \mathsf{g}^x, \mathsf{g}^y, \mathsf{g}^{xz})$ for $y \neq z$. Sim stores the tuple in a list Q as $(\mathsf{sid}, c, (x, y, z))$. H_1 is programmed to return $(g_1||h_0||h_1) \leftarrow \mathrm{H}_1(\mathsf{sid}||c)$. If a query is repeated then Sim returns the entry in Q indexed by the sid and c values. $\mathsf{Adv}_{\mathsf{Int}}$ sends the first OT message to S in the internal world.

(ii) *Simulating S's message:* Sim extracts choice bit of R^*. If $h = g^y$ then $\sigma = 0$, else if $h = g^z$ then $\sigma = 1$. If neither of the above checks satisfy, Sim sets $\sigma = \perp$. In case $\sigma = \perp$, then Sim sets (w_0, w_1) randomly. Otherwise, Sim sends σ to $\mathcal{F}_{\mathsf{OT}}$ on behalf of R in the ideal world, obtains a_σ sets w_σ appropriately and $w_{\overline{\sigma}}$ at random, where $\overline{\sigma} = 1 - \sigma$. Finally, Sim computes the sender's message and sends it to R^* in the internal world.

(iii) $\mathsf{Adv}_{\mathsf{Int}}$ computes on behalf of R^* and completes the protocol in the internal world.

Case 4. Both S^* and R^* are corrupted: This is a trivial case of corruption. Sim invokes $\mathsf{Adv}_{\mathsf{Int}}$, who simulates messages of both parties and generates the view internally. At the end of execution, $\mathsf{Adv}_{\mathsf{Int}}$ sends the generated view to Sim who forwards it to $\mathcal{Z}$.

Indistinguishability: Here we show that the ideal world view $\text{IDEAL}_{\mathcal{F},\mathsf{Sim},\mathcal{Z}}(1^\kappa, z)$ of $\mathcal{Z}$ is indistinguishable from the real world view $\text{REAL}_{\mathcal{F},\mathsf{Adv},\mathcal{Z}}(1^\kappa, z)$ of $\mathcal{Z}$. We denote $\text{REAL}_{\mathcal{F},\mathsf{Adv},\mathcal{Z}}(1^\kappa, z)$ as hybrid $\mathsf{HYB}_{\mathrm{R}}$. The ideal world view of $\mathcal{Z}$ varies based on the case of corruption. For case κ ($\kappa \in [4]$), we denote the ideal world view as $\mathsf{HYB}_{\mathrm{I}.\kappa}$. We prove that $\mathsf{HYB}_{\mathrm{R}}$ is indistinguishable from the corresponding ideal world view for each of the four exhaustive cases of corruption.

Case 1. S and R are honest: We prove that $\mathsf{HYB}_{\mathrm{R}}$ and the ideal world view i.e $\mathsf{HYB}_{\mathrm{I}.1}$ is indistinguishable through a series of intermediate hybrids.

– **HYB_1 :** We consider a hybrid HYB_1 which is same as $\mathsf{HYB}_{\mathrm{R}}$ except that here, the CRS is set to be a DDH-tuple. Indistinguishability follows from the hardness of the DDH assumption and random sampling of c. A distinguisher for the hybrids can be used to break the DDH assumption or guess the exact value of c, both of which happen with negligible probability.

- **HYB_2 :** We consider a hybrid HYB_2 which is same as HYB_1 except that Sim always encrypts $(g, h) = (g_0^\alpha, h_0^\alpha)$, for some random $\alpha \in \{0, 1\}$. Indistinguishability follows statistically since (g, h) could have been obtained from either (g_0, h_0) and (g_1, h_1) using α or αx^{-1} as exponent.
- **HYB_3 :** We consider a hybrid HYB_3 similar to HYB_2 except that in HYB_3 Sim sets $w_0 = \mathrm{H}_1(v_0') \oplus a_0$ and $w_1 = \mathrm{H}_1(v_1') \oplus a_1$ where $v_0', v_1' \leftarrow_R \mathbb{G}$. The tuples in HYB_3 - (g_0, g, u_0, v_0') and (g_1, g, u_1, v_1') - form Non-DDH tuples, except with negligible probability, as v_0' and v_1' are randomly sampled. Whereas in HYB_2 we have $w_0 = \mathrm{H}_1(v_0) \oplus a_0$ and $w_1 = \mathrm{H}_1(v_1) \oplus a_1$, where $v_0 = g^{r_0} h^{s_0}$ and $v_1 = g^{r_1} h^{s_1}$. The tuples in HYB_2- (g_0, g, u_0, v_0) and (g_1, g, u_1, v_1) - are DDH tuples. If a distinguisher can distinguish between the two hybrids then he must have distinguished based on the distribution of the tuples in the two hybrids. Hence, indistinguishability between the hybrids follows from the DDH assumption.
- **$\mathsf{HYB}_{\mathrm{I.1}}$:** We consider the ideal world hybrid $\mathsf{HYB}_{\mathrm{I.1}}$ similar to HYB_3 except that in $\mathsf{HYB}_{\mathrm{I.1}}$, Sim sets (w_0, w_1) randomly. The values (v_0', v_1') are chosen randomly in HYB_3 and $\mathsf{Adv}_{\mathsf{Int}}$ can distinguish between the hybrids only if the values are guessed precisely and queried to the random oracle. This event occurs with negligible probability in the random oracle model and as a result indistinguishabilty between the hybrids follows.

Case 2. S^* is corrupted and R is honest: We prove that $\mathsf{HYB}_{\mathrm{R}}$ and the ideal world view i.e $\mathsf{HYB}_{\mathrm{I.2}}$ is indistinguishable through an intermediate hybrid.

- **HYB_1 :** We consider hybrid HYB_1 which is same as HYB_1 in previous case. Indistinguishability between $\mathsf{HYB}_{\mathrm{R}}$ and HYB_1 follows (similar to the previous case) from the hardness of DDH assumption.
- **$\mathsf{HYB}_{\mathrm{I.2}}$:** We consider a hybrid $\mathsf{HYB}_{\mathrm{I.2}}$ which is same as HYB_1 except Sim always encrypts $(g, h) = (g_0^\alpha, h_0^\alpha)$, for some random α. Sim extracts (a_0, a_1) from (w_0, w_1) in the internal world, sends (a_0, a_1) to $\mathcal{F}_{\mathsf{OT}}$ and outputs a_σ in the internal world. Indistinguishability follows statistically since (g, h) can be obtained from both (g_0, h_0) and (g_1, h_1) by using α or αx^{-1} as exponent. The extraction of (a_0, a_1) follows from the correctness of protocol.

Case 3. S is honest and R^* is corrupted: We prove that $\mathsf{HYB}_{\mathrm{R}}$ and the ideal world view i.e $\mathsf{HYB}_{\mathrm{I.3}}$ is indistinguishable through a series of intermediate hybrids.

- **HYB_1 :** We consider a hybrid HYB_1 which is same as $\mathsf{HYB}_{\mathrm{R}}$ except here the CRS is set to be a non-DDH-tuple. Indistinguishability follows since the CRS is generated as an output of the random oracle in the real world, which is a non-DDH tuple, except with negligible probability. Another possible way of distinguishing is if the distinguisher can guess the value of the CRS without querying c to the random oracle, but that happens with negligible probability, due to the random oracle assumption.
- **HYB_2 :** If the receiver constructs (g, h) tuple s.t. $g = g_0^{\alpha_0}, h = h_0^{\alpha_0}$ or $g = g_1^{\alpha_1}, h = h_1^{\alpha_1}$ for some $\alpha_0, \alpha_1 \in \mathbb{Z}_p$, then we consider it as well-formed. The correctness of the protocol ensures that R would receive the output if the tuple is well-formed, else correctness is violated and the receiver fails to decrypt w_0 and w_1. Let us

analyze the structure of (g, h), in case it is not well formed. In such a case it can be observed that $g \neq h^y$ and $g \neq h^z$. As a result, Sim fails to extract a valid σ from R^*'s message and he sets $\sigma = \perp$. We consider a hybrid HYB_2, where (w_0, w_1) are set randomly, if $\sigma = \perp$, else it is identical to HYB_1. Indistinguishability follows from the fact that if (g, h) is not well-formed then R^* would fail to decrypt w_0 and w_1 in both HYB_1 and HYB_2, and thus HYB_2 is indistinguishable from HYB_1.

– **HYB_3 :** We consider a hybrid HYB_3 which is same as HYB_2 except that, here, Sim extracts the value of σ, constructs w_σ as per the protocol and sets $w_{\overline{\sigma}} = \mathrm{H}_2(v') \oplus a_{\overline{\sigma}}$, for $v' \leftarrow_R \mathbb{Z}_p$. Indistinguishability results due to the randomness associated with the unknown v and v' values. We demonstrate this by proving that $(u_{\overline{\sigma}}, v)$ is statistically indistinguishable from $(u_{\overline{\sigma}}, v')$ since the CRS is a Non-DDH tuple. Consider $\sigma = 0$, then $u_1 = g_1^{r_1} h_1^{s_1} = g_1^{r_1 + z s_1} = \mathsf{g}^{x r_1 + x z s_1}$ and $v = g^{r_1} h^{s_1} = g_0^{\alpha r_1} h_0^{\alpha s_1} = \mathsf{g}^{\alpha r_1 + \alpha y s_1}$. We analyze the two expressions in exponents: $x r_1 + x z s_1 = x(r_1 + z s_1)$ and $\alpha r_1 + \alpha y s_1 = \alpha(r_1 + y s_1)$. Since $x\alpha(z - y) \neq 0 \in \mathbb{Z}_p$, the two expressions are linearly independent combinations of r_1 and s_1. Therefore, u_1 and v are uniform and independent over the choices of r_1 and s_1 proving $(u_1, v) \overset{s}{\approx} (u_1, v')$. Similarly, if $\sigma = 1$, then $(u_0, v) \overset{s}{\approx} (u_0, v')$ follows where $v = (g^{r_0} h^{s_0})$.

– **$\mathsf{HYB}_{\mathrm{I.3}}$:** Finally we consider our ideal world hybrid $\mathsf{HYB}_{\mathrm{I.3}}$ where $w_{\overline{\sigma}}$ is set randomly. Indistinguishability follows from the random oracle assumption since v' is unknown to the distinguisher and it is possible to distinguish between the two hybrids only if the value of v' is guessed.

Case 4. Both S^* and R^* are corrupted: In this case $\mathsf{HYB}_{\mathrm{R}}$ and $\mathsf{HYB}_{\mathrm{I.4}}$ are generated by Adv and $\mathsf{Adv}_{\mathsf{Int}}$ after being in control of both honest parties in the real world and internal world respectively. As a result, the two views are identical.

4.3 Adaptive Security

Building upon the proof of static security in the previous section, we now prove that π_{OT} securely implements $\mathcal{F}_{\mathsf{OT}}$ in the presence of adaptive adversaries. We refer to Appendix B for details about the security model. We give a brief overview of the proof and then we present it formally. Following the lines of the static proof, Sim programs the CRS of the internal execution to be a non-DDH tuple when the receiver is corrupt in the first round or a DDH tuple otherwise to enable extraction of R's input or S's input. In addition, Sim has to equivocate the view of R (resp. S), in the internal execution, when R (resp. S) gets corrupted adaptively by $\mathsf{Adv}_{\mathsf{Int}}$. The proof demands equivocation only when R gets corrupted after sending the first OT message and/or S gets corrupted after sending the second OT message. This is done to ensure that the ideal world views (messages and internal state) of the simulated honest parties (in the internal execution) are consistent with the real world views (messages and internal state) of the actual honest parties else $\mathcal{Z}$ can distinguish between the two views. In the first scenario, when R is corrupted after the first OT message is sent, Sim has to equivocate (g, h) such that it opens to (g_σ, h_σ). This is facilitated by the DDH property of the CRS since (g, h) can be opened to (g_0, h_0) or (g_1, h_1) based on σ. In the second case when S is corrupted after the second OT message is sent, the random values sent corresponding to (w_0, w_1) by Sim

have to be made consistent with sender's actual input (a_0, a_1). For this, Sim exploits the programmability of the PRO to enforce that (w_0, w_1) decrypts to (a_0, a_1).

We can observe that the CRS plays an instrumental role in simulating the internal execution. We require the CRS to be generated appropriately (DDH or non-DDH) as demanded by the simulation proof. Sim enables this by programming $\mathrm{H}_1(\mathsf{sid}||c)$ to return either a DDH or non-DDH tuple accordingly. Note that if CRS had been generated in a deterministic manner, it would be possible for $\mathsf{Adv}_{\mathsf{Int}}$ to generate it as well, thereby fixing the CRS prior to the corruption of any parties. Later when $\mathsf{Adv}_{\mathsf{Int}}$ corrupts the parties, the simulation proof demands that Sim programs the CRS to be DDH or non-DDH based on the corruption case. However, since the CRS has been already fixed by $\mathsf{Adv}_{\mathsf{Int}}$, it may result in inconsistency. To handle this, we generate the CRS using an identifier c which is chosen randomly by R.

We are ready to present the formal proof of Theorem 2. We design an ideal world adversary Sim who creates an ideal world view $\text{IDEAL}_{\mathcal{F},\mathsf{Sim},\mathcal{Z}}(1^\kappa, z)$ of $\mathcal{Z}$ which is indistinguishable from the real world view $\text{REAL}_{\mathcal{F},\mathsf{Adv},\mathcal{Z}}(1^\kappa, z)$ of $\mathcal{Z}$. Whenever Adv corrupts a party in the real world, Sim corrupts that party in the ideal world and $\mathsf{Adv}_{\mathsf{Int}}$ corrupts that party in the internal world. Upon corruption, Sim has to provide consistent views for the corrupted party in the internal world to $\mathsf{Adv}_{\mathsf{Int}}$ since Sim was behaving on behalf of the party, in the internal world, until it got corrupted.

Theorem 2. *If* H_1 *and* H_2 *are programmable random oracles and solving Decisional Diffie Hellman is hard in multiplicative group* $\mathbb{G}$*, then protocol* π_{OT} *securely realizes the* $\mathcal{F}_{OT}$ *functionality in the presence of adaptive (without erasures) active adversaries.*

Proof. We describe the simulator corresponding to the protocol π_{OT}, for each possible case of adaptive corruption.

The Simulator: The simulator Sim that generates the ideal world view, is initialized with input values from $\mathcal{Z}$ based on which party is corrupted to facilitate simulation.

Outset of the Protocol: $\mathsf{Adv}_{\mathsf{Int}}$ can query H_1 with different values of c to obtain some candidate CRS values which are programmed to be non-DDH tuples. On receiving queries of the form $\mathrm{H}_1(\mathsf{sid}||c)$, Sim samples $x, y, z \leftarrow_R \mathbb{Z}_p$ s.t. $(g_0, g_1, h_0, h_1) = (\mathsf{g}, \mathsf{g}^x, \mathsf{g}^y, \mathsf{g}^{xz})$ for $y \neq z$ and stores the tuple in a list Q as $(\mathsf{sid}, c, (x, y, z))$. H_1 is programmed to return $(g_1||h_0||h_1) \leftarrow \mathrm{H}_1(\mathsf{sid}||c)$. If the same query is made more than once, Sim returns the entry in Q indexed by the sid and c values.

R is honest in the first round: Sim computes R's message similar to case 1(i) (R's message for (S, R) case) of the static proof. Sim sends the CRS (which is a DDH-tuple in this case), c and (g, h) as the first OT message to $\mathsf{Adv}_{\mathsf{Int}}$ on behalf of R in the internal execution.

- **S is honest in the second round:** Sim acts on behalf of S in the internal execution. Sim simulates according to Case 1(ii) (S's message for (S, R) case) of static proof.
  - **Case 1(A). R is honest in the first round, S is honest in the second round, R^* is corrupted after second OT message:** Sim obtains σ and a_σ in the ideal world. Sim equivocates (g, h) by setting β s.t. $(g, h) = (g_\sigma^\beta, h_\sigma^\beta)$ for $\beta = \alpha x^{-\sigma}$. Sim returns β and c as internal randomness of R^* to $\mathcal{Z}$. Additionally,

Sim equivocates w_σ s.t. a_σ can be obtained from w_σ. For this, Sim programs the random oracle as follows: $\mathrm{H}_2(g^{r_\sigma}h^{s_\sigma}) = w_\sigma \oplus a_\sigma$. At the end of the protocol, $\mathsf{Adv}_{\mathsf{Int}}$ outputs $\perp$ and sends its internal state to Sim who forwards it to $\mathcal{Z}$.

Post Execution. In case of post execution corruption of S^*, Sim obtains (a_0, a_1) and needs to provide the internal randomness of S^* s.t. w_0 and w_1 open to a_0 and a_1. We note that w_σ was equivocated already. Equivocation of $w_{\overline{\sigma}}$ is performed by programming $\mathrm{H}_2(g^{r_{\overline{\sigma}}}h^{s_{\overline{\sigma}}}) = w_{\overline{\sigma}} \oplus a_{\overline{\sigma}}$. Sim sends the internal state of S^* to $\mathcal{Z}$ who halts with an output.

- **Case 1(B). R is honest in the first round, S is honest in the second round, R is honest after second OT message:** In this case, Sim acts on behalf of both parties throughout the protocol in the internal execution. At the end of the protocol, $\mathsf{Adv}_{\mathsf{Int}}$ outputs his random tape as its internal state to Sim who forwards it to $\mathcal{Z}$.

  Post Execution. In case of post execution corruption of R^* and S^*, Sim obtains (σ, a_σ) and (a_0, a_1), and has to provide the internal randomness of R^* and S^* to $\mathcal{Z}$. The equivocation of both views is similar to the previous case (Case 1(A) of adaptive simulation) where the view of R^* is equivocated first and then the view of S^* is equivocated. Sim sends the equivocated views of R^* and S^* to $\mathcal{Z}$, who halts with an output.

– **Case 2. R is honest in the first round, S^* is corrupted in the second round:** Sim receives the second message on behalf of R from S^* in the internal execution. Simulation is performed as in Case 2(iii) (R's computation for (S^*, R) case) of static proof. $\mathsf{Adv}_{\mathsf{Int}}$ outputs $\perp$ at the end of the protocol and sends its internal state to Sim who forwards it to $\mathcal{Z}$.

Post Execution. In case of post execution corruption of R^*, Sim obtains σ and a_σ and he proceeds like the simulator for case 1(A) of adaptive simulation.

R^* is corrupted in the first round: When $\mathsf{Adv}_{\mathsf{Int}}$ (or R^*) queries $\mathrm{H}_1(\mathsf{sid}||c)$, Sim programs the RO to return a non-DDH CRS as Case 3(i) (R's message for (S, R^*) case) in static proof. $\mathsf{Adv}_{\mathsf{Int}}$ generates the first OT message which is sent to S in the internal execution.

– **Case 3. R^* is corrupted in the first round, S is honest in the second round:** Sim receives the first OT message, on behalf of S, from $\mathsf{Adv}_{\mathsf{Int}}$ controlling R^* in the internal execution. Sim continues simulation as in Case 3(ii) (S's message for (S, R^*) case) of static proof.

Post Execution. In case of post execution corruption of S^*, Sim obtains (a_0, a_1) and needs to equivocate $w_{\overline{\sigma}}$ such that it is consistent with $a_{\overline{\sigma}}$. For simplicity, let us assume that $\sigma = 0$. Then $w_1 = \mathrm{H}_2(g^{r_1}h^{s_1}) \oplus a_1 = \mathrm{H}_2(g_0^{\alpha r_1}h_0^{\alpha s_1}) \oplus a_1$. Sim can simply compute $g_0^{\alpha r_1}h_0^{\alpha s_1}$ given $g = g_0^\alpha$, $h = h_0^\alpha$ and the internal randomness (r_1, s_1), which was sampled earlier to construct u_1. H_2 is programmed such that $\mathrm{H}_2(g^{r_1}h^{s_1}) = w_1 \oplus a_1$. The internal state of S^* comprising of (r_0, r_1, s_0, s_1) is revealed to $\mathcal{Z}$ who halts with an output. Likewise, equivocation can be performed for $\sigma = 1$.

– **Case 4. R^* is corrupted in the first round, S^* is corrupted in the second round:** This is a trivial case since both parties are corrupted by the adversary. The parties are controlled by $\mathsf{Adv}/\mathsf{Sim}/\mathsf{Adv}_{\mathsf{Int}}$ in the real/ideal/internal world. $\mathsf{Adv}_{\mathsf{Int}}$ generates the second OT message similar to Case 4 ((S^*, R^*) case) of static proof. At the end of execution, $\mathsf{Adv}_{\mathsf{Int}}$ outputs a special symbol $\perp$ on behalf of the corrupted parties and hands over the internal state to Sim who in turn forwards it to $\mathcal{Z}$.
Post Execution. There is no post execution corruption since both parties are corrupted and $\mathcal{Z}$ halts with an output computed from the internal state of $\mathsf{Adv}_{\mathsf{Int}}$.

Indistinguishability : Here we show that the ideal world view $\text{IDEAL}_{\mathcal{F},\mathsf{Sim},\mathcal{Z}}(1^\kappa, z)$ of $\mathcal{Z}$ is indistinguishable from the real world view $\text{REAL}_{\mathcal{F},\mathsf{Adv},\mathcal{Z}}(1^\kappa, z)$ of $\mathcal{Z}$. We denote $\text{REAL}_{\mathcal{F},\mathsf{Adv},\mathcal{Z}}(1^\kappa, z)$ as hybrid $\mathsf{HYB}_{\mathrm{R}}$ and show that in each simulation case $\mathsf{HYB}_{\mathrm{R}}$ is indistinguishable from the ideal world by relying on the static indistinguishability proof.

Case 1(A). R is honest in the first round, S is honest in the second round, R^* is corrupted after second OT message: Simulation of both OT messages follows along the same direction as Case 1 of static proof. When R^* gets corrupted after second OT message Sim obtains σ and he can open (g, h) correctly to (g_σ, h_σ) since the CRS is a DDH tuple. He programs the random oracle H_2 on v_σ to open to w_σ. Equivocation is successful since $\mathsf{Adv}_{\mathsf{Int}}$ can query v_σ, before corrupting R^*, to H_2 only with negligible probability. This follows from the DDH assumption (HYB_3 of Case 1 in static proof) and the random oracle assumption ($\mathsf{HYB}_{\mathrm{I}.1}$ of Case 1 in static proof). In case of post execution corruption, Sim can successfully equivocate $w_{\overline{\sigma}}$ to open $a_{\overline{\sigma}}$ since Adv can query $v_{\overline{\sigma}}$ to H_2 with negligible probability. This is also follows from HYB_3 and $\mathsf{HYB}_{\mathrm{I}.1}$ of Case 1 in static proof. Thus indistinguishability of simulation follows from indistinguishability of $\mathsf{HYB}_{\mathrm{I}.1}$ from $\mathsf{HYB}_{\mathrm{R}}$ in the static proof.

Case 1(B). R is honest in the first round, S is honest in the second round, R is honest after second OT message: In this case we can consider that the post execution corruption occurs in two phases - first R^* gets corrupted and only after that S^* gets corrupted. Then it becomes identical to the previous case (case 1(A) of adaptive proof) and indistinguishability follows from the previous argument.

Case 2. R is honest in the first round, S^* is corrupted in the second round: The first message is computed according to Case 1(i) of static proof, which is identical to Case 2(i) of static proof. The rest of the simulation proceeds as Case 2 of static proof. Indistinguishability follows from the indistinguishability of $\mathsf{HYB}_{\mathrm{I}.2}$ from $\mathsf{HYB}_{\mathrm{R}}$ in the static proof. In case of post execution corruption equivocation of R's view always succeeds since the CRS is a DDH tuple and (g, h) can open to both (g_0, h_0) and (g_1, h_1) with $\beta = \alpha$ and $\beta = \alpha x^{-1}$ respectively.

Case 3. R^* is corrupted in the first round, S is honest in the second round: Indistinguishability follows from the indistinguishability of $\mathsf{HYB}_{\mathrm{I}.3}$ from $\mathsf{HYB}_{\mathrm{R}}$ in the static proof. In case of post execution corruption equivocation can fail if during the simulation $\mathsf{Adv}_{\mathsf{Int}}$ queries $v = g^{r_{\overline{\sigma}}} h^{s_{\overline{\sigma}}}$ to H_2 before Sim programs $\mathrm{H}_2(v)$. This would happen only if the query was made before Sim obtains $a_{\overline{\sigma}}$ in the post execution corruption. However,

it has been proven in HYB_3 and $\mathsf{HYB}_{\mathrm{I}.3}$ of Case 3, of static case, that this happens with negligible probability, proving correctness of equivocation of S's view by Sim.

Case 4. R^* is corrupted in the first round, S^* is corrupted in the second round: In this case the first OT message is simulated according to Case 3(i) of static proof and the rest of the protocol is simulated according to Case 4 of static proof. However, Case 4(i) and Case 3(i) differs only in terms of CRS generation. In Case 4(i) it is randomly sampled whereas in Case 3(i) it is a non-DDH tuple and so they are indistinguishable except with negligible probability. Thus indistinguishability of simulation for this case follows from indistinguishability of $\mathsf{HYB}_{\mathrm{I}.4}$ from $\mathsf{HYB}_{\mathrm{R}}$ in the static proof.

□

4.4 Efficiency

Our protocol requires 11 exponentiations and 5 random oracle queries and it has an optimal round complexity of 2. It requires sending 3 κ bit strings - c and (w_0, w_1); and 4 group elements– (u_0, u_1) and (g, h). S already has $g_0 = \mathsf{g}$ and so R avoids sending it. Furthermore, our protocol is based on DDH and PRO assumptions and requires no CRS setup unlike most other existing adaptively-secure OT protocols. Interestingly, our protocol is the first round-optimal adaptively-secure OT protocol. Our protocol has an overhead of 5 random oracle queries and κ-bit string communication overhead over the static protocol of [PVW08].

4.5 Static Version of our Protocol

We can optimize our adaptive π_{OT} protocol to obtain an efficient static OT protocol, denoted as π'_{OT}. It is similar to the PVW protocol, except here we generate the CRS using a PRO. We can obtain π'_{OT} from π_{OT} by removing the random oracle H_2 . H_2 is not required since Sim is not required to equivocate w_0 and w_1 in the case of static corruption. This yields the first 2 round OT protocol secure under PRO assumption with an overhead of 11 exponentiations and 2 oracle queries. The security of our protocol is summarized in Theorem 3 and the proof is similar to the static security proof of π_{OT}.

Theorem 3. *If H_1 is a programmable random oracle and solving Decisional Diffie Hellman is hard in multiplicative group $\mathbb{G}$, then protocol π'_{OT} securely implements the $\mathcal{F}_{\mathsf{OT}}$ functionality in the presence of static active adversaries.*

Interestingly, π'_{OT} satisfies the property of receiver equivocality, which states that the view of the receiver can be equivocated if the receiver gets adaptively corrupted during/after the protocol execution. We analyze this property as two exhaustive sub-cases as follows : (1) R gets corrupted before the first OT message is sent (2) R gets corrupted after the first OT message is sent. For the first case, the simulator has not sent any message on behalf of R and hence equivocation is not required. For the second case, the simulator sets the CRS as a DDH tuple and he sends the first OT message as (g, h) and c. He can equivocate (g, h) (Similar to Case 1(A) of adaptive simulation for π_{OT}) on obtaining σ, after R gets corrupted, since he knows the trapdoor for the CRS.

5 Adaptively Secure 1-out-of-N Oblivious Transfer.

The work of [NP05] implements $\mathcal{F}_{\mathsf{N\text{-}OT}}$, against static adversaries in the $\mathcal{F}_{\mathsf{OT}}$-hybrid model by relying on pseudorandom functions. S has N strings $(a_1, a_2, \ldots, a_{\mathsf{N}})$ as input and R has a $\log \mathsf{N}$ bit input choice string σ. S samples $2 \log \mathsf{N}$ random pads $p_0^i, p_1^i \leftarrow_R \{0,1\}^\kappa$ for $i \in [\log \mathsf{N}]$. The two parties invoke $\log \mathsf{N}$ copies of $\mathcal{F}_{\mathsf{OT}}$ with S's input as (p_0^i, p_1^i) and R's input as σ_i respectively. The ith invocation of $\mathcal{F}_{\mathsf{OT}}$ outputs $p_{\sigma_i}^i$ to R, i.e the pad corresponding to his σ_i choice bit. Finally, S encrypts a_j as w_j using the pads corresponding to the bit representation of j, $j \in [\mathsf{N}]$. The message a_j is encrypted as follows :

$$w_j = a_j \oplus \bigoplus_{i=1}^{\log N} \mathrm{PRF}_{p_c^i}(j),$$

where $c = j_i$, i.e. the ith bit of string j. R obtains $(w_0, w_1, \ldots, w_{\mathsf{N}})$ and decrypts w_σ for which he possesses all the $\log \mathsf{N}$ pads. For other w values, R lacks at least one random pad and security follows by applying PRF on that secret random pad. The transformation communicates N ciphertexts, and requires $\log \mathsf{N}$ invocations of $\mathcal{F}_{\mathsf{OT}}$ and $\mathsf{N} \log \mathsf{N}$ evaluations of a PRF. It guarantees security against a statically corrupted active adversary.

We show in Fig. 6 that the same transformation can be made adaptively-secure (by protocol $\pi_{\mathsf{N\text{-}OT}}$) by implementing the underlying $\mathcal{F}_{\mathsf{OT}}$ functionality in an adaptive secure manner. In addition, we replace PRF with a programmable random oracle H and modify the formation of w_j as follows, where j_i denotes the ith bit of j. This reduces the $\mathsf{N} \log \mathsf{N}$ PRF evaluations to N random oracle queries.

$$w_j = a_j \oplus \mathrm{H}(j, p_{j_1}^1 || p_{j_2}^2 || \ldots || p_{j_{\log \mathsf{N}}}^{\log \mathsf{N}}) = a_j \oplus \mathrm{H}(j, v_j),$$

where $v_j = (p_{j_1}^1 || p_{j_2}^2 || \ldots || p_{j_{\log \mathsf{N}}}^{\log \mathsf{N}})$.

5.1 Security

The security of the protocol is proven by constructing a simulator Sim for $\pi_{\mathsf{N\text{-}OT}}$. We first present the static proof and then discuss the adaptive proof. For a statically corrupted S^*, Sim can extract the pads by invoking the simulator for $\mathcal{F}_{\mathsf{OT}}$. Sim forms $\{v_j\}_{j \in [\mathsf{N}]}$ and decrypts $\{w_j\}_{j \in [\mathsf{N}]}$ to unlock all the input messages of S^*, i.e. $\{a_j\}_{j \in [\mathsf{N}]}$, and completes the simulation by invoking $\mathcal{F}_{\mathsf{N\text{-}OT}}$ with the input messages. Indistinguishability follows from the $\mathcal{F}_{\mathsf{OT}}$ hybrid model. For a statically corrupted R^*, Sim can extract the choice bit σ_i by invoking the simulator for ith copy of $\mathcal{F}_{\mathsf{OT}}$. Sim reconstructs σ and invokes $\mathcal{F}_{\mathsf{N\text{-}OT}}$ with σ to obtain a_σ. Sim concludes the simulation by setting w_σ correctly while $\{w_j\}_{j \in [\mathsf{N}]/\sigma}$ are set as random strings. R^* obtains all the random pads (corresponding to σ) from $\log \mathsf{N}$ invocations of $\mathcal{F}_{\mathsf{OT}}$ and constructs v_σ to unlock a_σ. The other $\{a_j\}_{j \in [\mathsf{N}]/\sigma}$ values remain hidden since R^* lacks at least one random pad for $\{v_j\}_{j \in [\mathsf{N}]/\sigma}$, and hence $\mathrm{H}(j, v_j)$ will be indistinguishable from a random string, except with negligible probability, due to the random oracle assumption.

In order to prove adaptivity, we need the property of equivocation. The simulated messages have to be equivocated appropriately so that they are indistinguishable from the real world messages of honest parties. Sim can equivocate the simulated message (consisting of only $\mathcal{F}_{\mathsf{OT}}$ messages) of R by invoking the adaptive simulator for $\mathcal{F}_{\mathsf{OT}}$. The simulated message of S consists of the $\mathcal{F}_{\mathsf{OT}}$ messages and the simulated ciphertexts. The $\mathcal{F}_{\mathsf{OT}}$ messages can be trivially equivocated by invoking the adaptive simulator for $\mathcal{F}_{\mathsf{OT}}$. The simulated ciphertexts can be equivocated by programming $\mathrm{H}(j, v_j) = w_j \oplus a_j$, for $j \in [\mathsf{N}]/\sigma$. Adversary can query $\mathrm{H}(j, v_j)$ with negligible probability since he lacks atleast one pad in v_j and hence simulator can program $\mathrm{H}(j, v_j)$ successfully and hence equivocate correctly. The proof of indistinguishability is similar to the one for statically corrupted R^*. The security of $\pi_{\mathsf{N\text{-}OT}}$ is summarized in Theorem 4.

Theorem 4. *If* H *is a programmable random oracle then* $\pi_{\mathsf{N\text{-}OT}}$ *UC-securely realizes the* $\mathcal{F}_{\mathsf{N\text{-}OT}}$ *functionality in the* $\mathcal{F}_{\mathsf{OT}}$ *hybrid model against adaptive (without erasures) active adversaries.*

5.2 Efficiency

Our protocol invokes $\mathcal{F}_{\mathsf{OT}}$ functionality $\log \mathsf{N}$ times and queries the PRO N times. It incurs communication of $\log \mathsf{N}$ copies of $\mathcal{F}_{\mathsf{OT}}$ and N ciphertexts on n bits.

5.3 Instantiating $\mathcal{F}_{\mathsf{OT}}$ using Static Receiver Equivocable OT

We observe that our transformation continues to be adaptively-secure, despite replacing the adaptively-secure 1-out-of-2 OTs in the above result with statically-secure OTs, that support equivocation of receiver's view irrespective of equivocation of sender's view. When S^* or R^* is corrupted, inputs are extracted by invoking the simulator for the actively-secure static OT protocol. For adaptive security, we need equivocation of S's and R's views if corruption occurs during the course of execution or at the end of protocol. In the transformation, S's view consists of the OT messages and ciphertexts, i.e. $\{w_j\}_{j\in[\mathsf{N}]}$. A simulator, playing the role of S can trivially simulate the OT messages by running the honest S algorithm with random pads, which are independent of S's inputs. The ciphertexts are equivocated by programming $\mathrm{H}(j, v_j)$ (see Section 5.1). On the other hand, R's view consists only of the OT messages which can be trivially equivocated by relying on the receiver equivocal property of the OT. Our π'_{OT} protocol (Section 4.5) is a statically-secure protocol satisfying receiver equivocal property; hence we can plug in our protocol to obtain adaptively-secure 1-out-of-N OTs.

6 Non-Interactive UC-Secure Commitment Scheme

In this section we present our non-interactive UC-secure commitment scheme COM that is adaptively-secure. The protocol π_{COM} (described in Fig. 7) is universally composable and securely realizes the functionality $\mathcal{F}_{\mathsf{COM}}$ (described in Fig. 3) in the CRS model under ORO model and Discrete Log (over a group $\mathbb{G}$) assumption. Later in this section, we demonstrate a protocol π_{CRS} to generate CRS of the form (g, h) (denoted by $\mathcal{F}_{\mathsf{CRS}}$ in Fig. 8) in 4 rounds. Our CRS generation algorithm is statically-secure and once the CRS is generated, it can be used for subsequent instances of COM.

Fig. 6. Adaptively-Secure 1-out-of-N Oblivious Transfer Protocol

$\pi_{\mathsf{N\text{-}OT}}$

– **Public Inputs:** $\mathrm{H} : \{0,1\}^{\kappa} \rightarrow \{0,1\}^{n}$ denotes a programmable random oracle. $\mathcal{F}_{\mathsf{OT}}$ denotes a 1-out-of-2 OT functionality.
– **Private Inputs:** S has input messages $\{a_j\}_{j=1}^{\mathsf{N}}$ and R has an input choice string σ.

Choose:

– R invokes $\mathcal{F}_{\mathsf{OT}}$ functionality $\log \mathsf{N}$ times. He invokes the ith copy of $\mathcal{F}_{\mathsf{OT}}$ with input $(\mathsf{rec}, i, \sigma_i)$ for $i \in [\log \mathsf{N}]$.

Transfer:

– S samples $2 \log \mathsf{N}$ random pads as $(p_0^i, p_1^i) \leftarrow_R \{0,1\}^{2\kappa}$.
– S invokes the ith copy of $\mathcal{F}_{\mathsf{OT}}$ with input $(\mathsf{sen}, i, (p_0^i, p_1^i))$.
– S encrypts his input messages as $w_j = a_j \oplus \mathrm{H}(j, p_{j_1}^1 || p_{j_2}^2 || \ldots || p_{j_{\log \mathsf{N}}}^{\log \mathsf{N}})$, for $j \in [\mathsf{N}]$.

Local Computation by R:

– Computes a_σ as $a_\sigma = w_\sigma \oplus \mathrm{H}(\sigma, p_{\sigma_1}^1 || p_{\sigma_2}^2 || \ldots || p_{\sigma_{\log \mathsf{N}}}^{\log \mathsf{N}})$.

6.1 Protocol Overview

We build upon the commitment scheme of Pedersen [Ped91], that relies on hardness of discrete log problem. The Pedersen commitment inherently supports equivocation as the message is statistically hidden in their case. However, for UC security the simulator, acting on behalf of R, has to extract the message, of a corrupted S^*, from the commitment. Our first approach was to apply an observable RO on the message being committed, i.e. $\mathrm{H}_1(m)$, and then commit the response of the RO query in the Pedersen commitment. This would allow the simulator to observe the queries and obtain candidate message values. However, the simulator cannot uniquely identify the message committed. It is necessary to extract the randomness, say r_1, used in the commit phase so that the simulator can match the (message, randomness) with the commitment value. We achieve this by enforcing S to bind to r_1 using a second RO, i.e. H_2. S commits to r_1 by means of the query $\mathrm{H}_2(r_1)$. The hardness of the Discrete Log Problem ensures that a corrupted S^* is unable construct more than one such (message, randomness) pair that matches the Pedersen commitment.

However, the above technique demands binding to r_1 using RO which in turn prevents equivocation by simulator, acting on behalf of S. In order to restore the equivocal property, S is required to pad $\mathrm{H}_2(r_1)$ with fresh randomness r_2. This allows the simulator to equivocate the commitments by equivocating the first part of the commitment c_1 (Pedersen commitment on the message) separately, fixing r_1 to a new value. Now, the second part of the commitment c_2 can be equivocated by using r_1 and setting r_2 accordingly.

6.2 Static Security

We show that our non-interactive commitment scheme COM is secure against static active adversaries and securely realizes the functionality $\mathcal{F}_{\mathsf{COM}}$ in the UC model by proving theorem 5. We refer to Section 4.2 for the security model.

Fig. 7. Non-Interactive UC-Secure Commitment Scheme

π_{COM}

- **Public Inputs:** Generator g of group $\mathbb{G}$, $\mathrm{H}_1 : \{0,1\}^{\mathsf{poly}(\kappa)} \to \mathbb{Z}_p$ and $\mathrm{H}_2 : \mathbb{Z}_p \to \{0,1\}^{\kappa}$ denote two random oracles. CRS: (g, h) s.t. $g = \mathsf{g}$ and $h = g^x$.
- **Private Inputs:** S has input message m and R does not have any input.

Commit Phase:
On receiving input $(\textsc{Commit}, sid, m)$ S performs the following:

- He computes $a = \mathrm{H}_1(m)$.
- He samples $r_1, r_2 \leftarrow_R \mathbb{Z}_p$ and forms $\textsc{Com}(m; r_1, r_2) = (c_1, c_2) = (g^a h^{r_1}, \mathrm{H}_2(r_1) \oplus r_2)$.
- He sends $\textsc{Com}(m; r_1, r_2)$ to R as commitment to m.

Decommitment Phase:
On receiving input $(\textsc{Decommit}, sid)$ S sends (m, r_1, r_2) to R. R performs canonical verification of $\textsc{Com}$ using (m, r_1, r_2) and outputs $\textsc{accept}$ if verification succeeds, else outputs $\textsc{reject}$.

Theorem 5. *If H_1 and H_2 are observable random oracles and solving the Discrete Log Problem is hard in multiplicative group $\mathbb{G}$, then π_{COM} UC-securely realizes the $\mathcal{F}_{\mathsf{COM}}$ functionality in the CRS model against static active adversaries.*

Proof. Our proof is in the CRS model where it is assumed that Sim knows the trapdoor x s.t. $h = g^x$ for the CRS $(g; h)$. The proof proceeds in two cases - first, where Adv corrupts R^* and second, where the Adv corrupts S^*.

R^* is corrupted: Adv corrupts R^* in the real world, Sim corrupts R^* in the ideal world and $\mathsf{Adv}_{\mathsf{Int}}$ corrupts R^* in the internal execution. During the commit phase, Sim commits to a random message m' using (r'_1, r'_2) as randomness, thereby computing $\textsc{Com}$, similar to an honest sender and sends $\textsc{Com}(m'; r'_1, r'_2)$ to $\mathsf{Adv}_{\mathsf{Int}}$. Sim further invokes $\mathcal{F}_{\mathsf{COM}}$ on behalf of R^* to obtain the message $(\textsc{Receipt}, sid, \mathsf{S}, \mathsf{R})$. In the decommit phase, Sim invokes $\mathcal{F}_{\mathsf{COM}}$ to obtain the message $(\textsc{Decommit}, sid, m)$. On obtaining the committed message m, Sim provides randomness (r_1, r_2) s.t. $\textsc{Com}$ decommits to m. The The randomness (r_1, r_2) is computed by Sim as follows:

1. Let $a = \mathrm{H}_1(m)$ and $a' = \mathrm{H}_1(m')$.
2. The trapdoor x is known to Sim. Sim generates (r_1, r_2) s.t the values of (c_1, c_2) remain unchanged while the commitment is being equivocated. Sim does so by solving equations 1 and 2:

$$a + r_1 x = a' + r'_1 x \implies r_1 = (a' - a + r'_1 x) x^{-1} \tag{1}$$

$$\mathrm{H}_2(r_1) \oplus r_2 = \mathrm{H}_2(r'_1) \oplus r'_2 \implies r_2 = \mathrm{H}_2(r'_1) \oplus r'_2 \oplus \mathrm{H}_2(r'_1) \tag{2}$$

   Sim provides (m, r_1, r_2), as opening to the commitment $\textsc{Com}$, to $\mathsf{Adv}_{\mathsf{Int}}$.

At the end of the protocol, $\mathsf{Adv}_{\mathsf{Int}}$ sends its view to Sim. Sim forwards the view to $\mathcal{Z}$ who halts with an output.

We show that the real world view of $\mathcal{Z}$ is indistinguishable from the ideal world view by showing that the following two hybrids are statistically indistinguishable.

– HYB_0 : Real world execution of the protocol.
– HYB_1 : Same as HYB_0, except that, Sim commits to a random message m' using (r'_1, r'_2) as randomness and opens to m in the decommit phase using different randomness (r_1, r_2).

$\mathsf{HYB}_0 \overset{s}{\approx} \mathsf{HYB}_1$: HYB_0 corresponds to the real world view and HYB_1 corresponds to the ideal world view of $\mathcal{Z}$. It follow from Eq. 1 and 2 that the committed message remains statistically hidden in COM. Hence, $\forall m, m', r'_1, r'_2$, Sim can always find a consistent pair of randomness (r_1, r_2) s.t the commitment opens to m, provided Sim knows the trapdoor value x. This proves statistical indistinguishability of the two worlds.

S^* is corrupted: Adv corrupts S^* in the real world, Sim corrupts S^* in the ideal world and $\mathsf{Adv}_{\mathsf{Int}}$ corrupts S^* in the internal execution. Sim emulates the role of an honest R against $\mathsf{Adv}_{\mathsf{Int}}$ in the internal execution. Sim plays the role of S^* in $\mathcal{F}_{\mathsf{COM}}$. During the commit phase, Sim obtains the commitment $\text{COM}(m)$ from $\mathsf{Adv}_{\mathsf{Int}}$ in the internal execution. He observes the random oracle queries (both H_1 and H_2) made by $\mathsf{Adv}_{\mathsf{Int}}$ and tries to extract the committed message m. Sim aborts if it fails to extract the message. Let us assume that $\mathsf{Adv}_{\mathsf{Int}}$ makes s random oracle queries during the commit phase and Sim records them as $(q_1, q_2, \ldots, q_s)$. We denote a (q_i, q_j) pair as valid, if q_i was queried to H_1, q_j was queried to H_2 and the following holds:

$$g^{\mathrm{H}_1(q_i)} h^{q_j} = c_1. \tag{3}$$

where $\text{COM}(m) = (c_1, c_2)$ is received from $\mathsf{Adv}_{\mathsf{Int}}$ by Sim. Sim runs over all possible pairs of (q_i, q_j) to find the valid pair(s). Based on the number of valid pair(s) discovered, Sim performs the following :

– If there does not exist any valid pair then Sim samples $m' \leftarrow_R \mathbb{G}$ and sends (COMMIT, sid, m') to $\mathcal{F}_{\mathsf{COM}}$.
– If there exists a unique valid (q_i, q_j) pair then Sim sends $(\text{COMMIT}, sid, q_i)$ to $\mathcal{F}_{\mathsf{COM}}$.
– If there exists more than one valid pair then Sim samples $m' \leftarrow_R \mathbb{G}$ and sends (COMMIT, sid, m') to $\mathcal{F}_{\mathsf{COM}}$.

In the decommitment phase, $\mathsf{Adv}_{\mathsf{Int}}$ sends (m, r_1, r_2) to Sim. Sim verifies the commitment and aborts if verification fails in the internal execution. Sim aborts in internal world and $\mathcal{F}_{\mathsf{COM}}$ if the following holds:

– **Case 1:** If there was no valid pair.
– **Case 2:** If there was one valid (q_i, q_j) pair, and $q_i \neq m$.
– **Case 3:** If there exists more than one valid pair.

If none of the above conditions hold, then Sim has an unique valid (q_i, q_j) pair, s.t. $q_i = m$ and $q_j = r_1$. He sends $(\text{DECOMMIT}, sid)$ to $\mathcal{F}_{\mathsf{COM}}$ to complete simulation of $\mathcal{F}_{\mathsf{COM}}$. At the end of the protocol, $\mathsf{Adv}_{\mathsf{Int}}$ sends its view to Sim. Sim forwards the view to $\mathcal{Z}$ who halts with an output

We show that the real world view of $\mathcal{Z}$ is indistinguishable from the ideal world view by showing that the following two hybrids are computationally indistinguishable.

– HYB_0 : Real world execution of the protocol.
– HYB_1 : Ideal World execution of the protocol

$\mathsf{HYB}_0 \overset{c}{\approx} \mathsf{HYB}_1$: $\mathcal{Z}$ can distinguish between the two hybrids (or worlds) if Sim aborts in ideal world and $\mathcal{F}_{\mathsf{COM}}$, while the honest R completes the protocol in real world. This occurs when the decommitment to COM provided by $\mathsf{Adv}_{\mathsf{Int}}$ verifies correctly but Sim fails to extract the underlying committed message in the internal execution. This event has been captured as an union of three exhaustive cases presented in the simulation. We will show that each case occurs with negligible probability:

– **Case 1:** This case indicates that $\mathsf{Adv}_{\mathsf{Int}}$ obtained $\mathrm{H}_1(m)$ and $\mathrm{H}_2(r_1)$ without querying m or r_1 to the random oracle during the commit phase. The random oracle assumption ensures that the query results would be random. Hence $\mathsf{Adv}_{\mathsf{Int}}$ can guess $\mathrm{H}_1(m)$ (or $\mathrm{H}_2(r_1)$) without querying m (or r_1) with negligible probability.
– **Case 2:** This case indicates that either $\mathsf{Adv}_{\mathsf{Int}}$ obtained $\mathrm{H}_1(m)$ and $\mathrm{H}_2(r_1)$ without querying m or r_1 to the random oracle during the commit phase or the $\mathsf{Adv}_{\mathsf{Int}}$ possesses two valid pairs (q_i, q_j) and (m, r). The first event occurs with negligible probability as we are in the RO model. And the occurrence of the second event implies that the DLP problem can be solved by using $\mathsf{Adv}_{\mathsf{Int}}$ as a blackbox. We address this implication in Case 3 and show that it occurs with negligible probability following from the hardness of the DLP problem.
– **Case 3:** This case indicates that $\mathsf{Adv}_{\mathsf{Int}}$ obtains two or more valid pairs. However this implies that the DLP problem can be solved by using $\mathsf{Adv}_{\mathsf{Int}}$ as a blackbox. Let us denote two such valid pairs as (q_i, q_j) and (q'_i, q'_j). We will further split this case into two more subcases for analysis based on the equality of $\mathrm{H}_1(q_i)$ and $\mathrm{H}_1(q'_i)$ values.
  (a) $q_i \neq q'_i, \mathrm{H}_1(q_i) = \mathrm{H}_1(q'_i), q_j = q'_j$: This indicates that $\mathsf{Adv}_{\mathsf{Int}}$ found a collision in the random oracle queries as $q_i \neq q'_i$. However, this event occurs with negligible probability as we are in the random oracle model.
  (b) $q_i \neq q'_i, \mathrm{H}_1(q_i) \neq \mathrm{H}_1(q'_i), q_j \neq q'_j$: In this case, $\mathsf{Adv}_{\mathsf{Int}}$ can be used as blackbox by another adversary Adv_D to solve the DLP problem by finding the trapdoor x as follows. Adv_D sets the CRS for commitment scheme as the DLP challenge and participates in the commitment game with $\mathsf{Adv}_{\mathsf{Int}}$. On observing two valid pairs (q_i, q_j) and (q'_i, q'_j), s.t. $q_i \neq q'_i$ and $\mathrm{H}_1(q_i) \neq \mathrm{H}_1(q'_i)$ - Adv_D can find x as follows:

$$\mathrm{H}_1(q_i) + q_j x = \mathrm{H}_1(q'_i) + q'_j x \implies x = (\mathrm{H}_1(q_i) - \mathrm{H}_1(q'_i))(q'_j - q_j)^{-1}$$

  This occurs with negligible probability as solving DLP is hard in multiplicative group $\mathbb{G}$.

The other two cases involve $\mathrm{H}_1(q_i) = \mathrm{H}_1(q'_i), q_j \neq q'_j$ and $\mathrm{H}_1(q_i) \neq \mathrm{H}_1(q'_i), q_j = q'_j$ for $q_i \neq q'_i$. However, in these cases, it is not possible for both (q_i, q_j), (q'_i, q'_j) to be valid pairs (refer condition for valid pair in Eq 3) as $g^{\mathrm{H}_1(q_i)} h^{q_j} \neq g^{\mathrm{H}_1(q'_i)} h^{q'_j}$. There maybe atmost one valid pair, for which the analysis follows from Case 1 and 2.

□

6.3 Adaptive Security

Interestingly, our commitment scheme π_{COM} (Fig 7) satisfies the stronger security notion of adaptivity under the observable random oracle assumption in the CRS setup. We briefly discuss the proof in this section.

Theorem 6. *If* H_1 *and* H_2 *are observable random oracles and solving the Discrete Log Problem is hard in multiplicative group* $\mathbb{G}$*, then protocol* π_{COM} *UC-securely realizes the* $\mathcal{F}_{\mathsf{COM}}$ *functionality in the CRS model against adaptive active adversaries (without erasures).*

Proof. To prove adaptive security, we require Sim to equivocate the views of R and S appropriately on adaptive corruption in addition to static security. We divide our simulation into cases based on the party being corrupted:

R^* is corrupted: R does not have any input or input randomness, and so the role of R in the protocol is restricted to verifying the commitments upon obtaining the message (m) and randomness $(r_1$ and $r_2)$ during the decommitment phase. When $\mathsf{Adv}_{\mathsf{Int}}$ corrupts R^* at any stage, i.e. commit phase, decommitment phase or post execution of the protocol, Sim returns a random tape as the internal randomness of R^*.

S^* is corrupted: Sim closely imitates the role of the simulator for static corruption when S^* gets corrupted adaptively. If $\mathsf{Adv}_{\mathsf{Int}}$ corrupts S^* in the beginning of the protocol or before the commitment is sent, Sim returns a random tape as the internal randomness of S^*. If $\mathsf{Adv}_{\mathsf{Int}}$ corrupts S^* after the commitment is sent, i.e. in decommitment phase or post execution, then Sim needs to equivocate. Sim initially commits to a dummy message m' and upon corruption of S^*, Sim obtains message m and successfully equivocates the commitment to open to m using randomness (r_1, r_2) (computed as described in the proof of Theorem 5). $\mathcal{Z}$ cannot distinguish between the commitment to a dummy message and commitment to the actual value due to the hiding property of the scheme. Equivocation follows from the equivocal property as proven for the static case.

□

6.4 Generation of CRS

For our protocol π_{COM}, the involved parties obtain CRS by invoking the functionality $\mathcal{F}_{\mathsf{CRS}}$ (Fig. 8). The CRS required in the commitment scheme is (g, h), where $h = g^x$ and it is necessary for the simulator to know the trapdoor x in order to perform simulation. The CRS can be trivially generated by invoking a PRO. The parties generate the CRS as $\mathrm{H}(\mathsf{sid}||\text{“}com\text{”})$. Sim samples x and programs the RO to return (g, h) s.t. $h = g^x$. This preserves adaptive security of π_{COM} when the CRS generation algorithm is included as part of π_{COM}. However we are interested in generating the CRS without relying on the programmability of the RO. This can be achieved by executing a 2PC protocol π_{CRS} (Fig. 9) relying solely on the observability property of the RO. Once the CRS is generated it can be reused for subsequent commitments between the parties.

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

On input $(\text{CRSGEN}, \mathsf{sid})$ from party P_I, if $(\mathsf{sid}, ((s_1, s_2), s_3))$ is present in memory then send $(\mathsf{sid}, (s_1, s_2))$ to P_I. Else sample $x \leftarrow_R \mathbb{Z}_p$, compute $h = g^x$, store $(\mathsf{sid}, ((g, h), x))$ in the memory and return $(\mathsf{sid}, (g, h))$ to P_I.

Fig. 8. The ideal functionality $\mathcal{F}_{\mathsf{CRS}}$ for generating CRS

Fig. 9. Implementing $\mathcal{F}_{\mathsf{CRS}}$ functionality for π_{COM}

π_{CRS}

– **Public Inputs:** Generator gen of group $\mathbb{G}$ and $\text{H} : \{0,1\}^{\mathsf{poly}(\kappa)} \rightarrow \{0,1\}^\kappa$ be a random oracle.
– **Private Inputs:** The parties do not have any input.

Coin Tossing:

– Round 1:
  - S samples $x_\mathsf{S} \leftarrow_R \mathbb{Z}_p$, computes $h_\mathsf{S} = g^{x_\mathsf{S}}$ and sends $\text{H}(h_\mathsf{S})$ to R.
  - R samples $x_\mathsf{R} \leftarrow_R \mathbb{Z}_p$, computes $h_\mathsf{R} = g^{x_\mathsf{R}}$ and sends $\text{H}(h_\mathsf{R})$ to S.
– Round 2:
  - S sends h_S to R.
  - R sends h_R to S.
– Computation:
  - S verifies $\text{H}(h_\mathsf{R})$ and computes $h = h_\mathsf{S}.h_\mathsf{R}$, else aborts.
  - R verifies $\text{H}(h_\mathsf{S})$ and computes $h = h_\mathsf{S}.h_\mathsf{R}$, else aborts.

Zero Knowledge Proof of Knowledge:
S and R perform the following steps in parallel with their roles interchanged.

– Round 1:
  - R samples a challenge string $c \leftarrow_R \{0,1\}^\mu$ and sends $\text{H}(c)$ to S.
– Round 2:
  - S computes μ garbled circuits, by sampling $\mathsf{seed}_i \leftarrow_R \{0,1\}^\kappa$ and $(\mathsf{GC}_i, \mathsf{e}_i, \mathsf{d}_i) \leftarrow \mathsf{Gb}(\text{PRF}(\mathsf{seed}_i), \mathcal{C})$ where $\mathcal{C}$ computes g^x, $i \in [\mu]$.
  - S sends $\text{H}(\mathsf{seed}_i)$, $\text{H}(\mathsf{GC}_i)$ and d_i to R.
– Round 3:
  - R reveals c to S. S verifies $\text{H}(c)$ and aborts if verification fails.
– Round 4: (Let c_i denote i^{th} bit of c)
  - If $c_i = 0$, then GC_i is a check circuit and S sends seed_i to R.
  - If $c_i = 1$, then GC_i is an evaluation circuit and S sends $\mathsf{X} = \mathsf{En}(x_\mathsf{S}, \mathsf{e}_i)$ to R.
– Computation:
  - R verifies the check circuit GC_i as $\mathsf{Ve}(\mathcal{C}, \mathsf{GC}_i, \mathsf{e}_i)$, if $c_i = 0$, else he aborts.
  - R computes $y = \mathsf{De}(\mathsf{Ev}(\mathsf{GC}_i, \mathsf{X}))$, and aborts if $y \neq h_\mathsf{S}$ and $c_i = 1$.
  - R stores (g, h) as the CRS.

We proceed to describe the π_{CRS} protocol. Our protocol has two parts - coin tossing and zero knowledge proof of knowledge (ZKPoK). The parties perform coin tossing to generate random shares h_S and h_R. These shares are then used to obtain $h = h_\mathsf{S}.h_\mathsf{R}$. Once coin tossing is performed, they engage in ZKPoK in order to prove the knowledge

of trapdoors to their respective shares. The ZKPoK enables the simulator to extract the corrupted party's share in order to obtain the trapdoor to (g, h). Our coin tossing protocol requires 2 rounds and ZKPoK consumes 4 rounds. However, the first 2 rounds of the ZKPoK can overlap with the coin tossing protocol, thus yielding a 4 round protocol for CRS generation. The coin tossing is performed using the random oracle and the ZKPoK is performed by plugging a simplified ZK version of [HV16] which uses garbled circuits.

The security of our protocol relies on the hardness of the DLP problem and the underlying properties of the garbling scheme. Informally, the security relies on the following three observations:

- h *should be a random group element:* The randomness of h is ensured as one of the parties will be honest and hence either h_{S} or h_{R} will be random. Moreover, the adversary cannot manipulate the value of h in the second round of coin tossing, after seeing the honest party's share. This is because the adversary's share has already been committed in the first round using the RO.
- *Sim should extract the trapdoor:* Sim will extract the adversary's share from the ZKPoK by relying on the observability property of $\mathrm{H}(\mathsf{seed}_i)$. Sim plays the role of an evaluator whereas the $\mathsf{Adv}_{\mathsf{Int}}$ plays the role of the constructor in the ZKPoK. Sim obtains seeds for all circuits by observing the queries made to $\mathrm{H}()$ and matching them with $\mathrm{H}(\mathsf{seed}_i)$. Sim generates e_i from $\mathrm{PRF}(\mathsf{seed}_i)$. Then Sim extracts the input of the corrupted constructor, i.e. x, by matching X_i with e_i. Hence Sim can extract the share of the adversarial party and compute the trapdoor to the CRS.
- *Sim should simulate ZK correctly:* Sim can simulate the ZKPoK, on behalf of constructor, by extracting the challenge string sent by $\mathsf{Adv}_{\mathsf{Int}}$, on behalf of evaluator, in Round 1 of ZKPoK. Sim constructs the check circuits correctly and for the evaluation circuits he invokes the privacy simulator of the garbling scheme.
  The security of our π_{CRS} protocol has been summarized in Theorem 7.

The π_{CRS} protocol is secure against static adversaries and not adaptive ones. Hence if our commitment scheme has a CRS generation phase which is implemented using π_{CRS} protocol then the whole scheme relies only on the observable random oracle. However, the drawback is that the commitment scheme becomes statically-secure.

Theorem 7. *If* H *is an observable random oracle,* Garble *is a private, verifiable garbling scheme and solving the Discrete Log Problem is hard in multiplicative group* $\mathbb{G}$*, then* π_{CRS} *UC-securely realizes* $\mathcal{F}_{CRS}$ *functionality (in Fig. 8) in the presence of static active adversaries.*

6.5 Efficiency

The length of our commitment is one group element and one κ bit string, independent of the message length. Decommitment incurs communication of two group elements. The computation is also minimal, incurring one random oracle query on $|m|$ bits, one oracle query on a κ bits string and two exponentiations on sender's side for committing. Decommitment incurs similar computation overhead on the receiver's end. Our protocol in the CRS model is non-interactive - in both commitment and decommitment phases.

The static protocol in the observable RO model incurs same overhead for commitment and decommitment. In addition to that, the π_{CRS} protocol requires 4 rounds. The computation cost is 2 exponentiations, $8\mu + 8$ oracle queries and construction and evaluation of 2μ circuits. The communication cost is 2μ circuits + $(8 + 4\mu + 2\kappa)$ strings of κ bits. However, it is a one-time cost which would get amortized when multiple commitments are performed using the same CRS. Our protocol is practically motivated especially for offline-online 2PC/MPC protocols [HKK+14, LR14, LR15, RR16]. The π_{CRS} protocol can be run in the offline phase while the commitment scheme can be conveniently used in the online phase. In comparison, the random oracle based interactive scheme of [CJS14] generates a trapdoor in-protocol which they use for equivocation. However unlike our protocol, their protocol cannot be optimized to include a preprocessing stage to generate a one-time trapdoor (or a CRS). Hence, in case of multiple commitments, their cost of trapdoor generation is incurred for each commitment. The other random oracle based commitment scheme of [HM04] requires a PRO.

7 Conclusion

In this paper, we presented UC-secure, round optimal protocols for cryptographic primitives, secure against adaptive adversaries. Our OT scheme relies solely on the programmability of random oracle as its setup assumption, while our commitment scheme relies on the observability of random oracle and CRS. The transformation from $\mathcal{F}_{\mathsf{OT}}$ to $\mathcal{F}_{\mathsf{N\text{-}OT}}$ against adaptive adversaries described in Section 4 is also based on PRO assumption. We further defined a statically-secure protocol for one-time CRS generation. It removes the reliance of our commitment scheme on a trusted setup for CRS, thus reducing the setup assumption to ORO only. However, the resulting scheme is secure against static adversary. We leave it as an open problem to obtain an adaptively-secure protocol for generating the CRS without relying on additional trusted setup assumption, such as public key infrastructure or PRO. Obtaining such a reusable CRS would uplift the security guarantee of our protocol to the adaptive notion, with assumptions based solely on ORO. Another interesting open problem is to obtain a NICOM for κ bit string which would require $\mathcal{O}(1)$ communication and computation, while relying on assumptions weaker than the random oracle.

Acknowledgements

We would like to thank Yashvanth Kondi for pointing out the delayed input extraction issue (a.k.a. the timing bug) in the security proofs of adaptively-secure OT protocols of [BDD+17, HL17] that claim UC security, in the PRO model. It helped us formalize a concrete attack to demonstrate the protocols are not composable.

References

ABB+13. Michel Abdalla, Fabrice Benhamouda, Olivier Blazy, Céline Chevalier, and David Pointcheval. Sphf-friendly non-interactive commitments. In *Advances in Cryptol-*

ogy - ASIACRYPT 2013 - 19th International Conference on the Theory and Application of Cryptology and Information Security, Bengaluru, India, December 1-5, 2013, Proceedings, Part I, pages 214–234, 2013.

AIR01. William Aiello, Yuval Ishai, and Omer Reingold. Priced oblivious transfer: How to sell digital goods. In *Advances in Cryptology - EUROCRYPT 2001, International Conference on the Theory and Application of Cryptographic Techniques, Innsbruck, Austria, May 6-10, 2001, Proceeding*, pages 119–135, 2001.

BC15. Olivier Blazy and Céline Chevalier. Generic construction of uc-secure oblivious transfer. In *Applied Cryptography and Network Security - 13th International Conference, ACNS 2015, New York, NY, USA, June 2-5, 2015, Revised Selected Papers*, pages 65–86, 2015.

BC16. Olivier Blazy and Céline Chevalier. Structure-preserving smooth projective hashing. In *Advances in Cryptology - ASIACRYPT 2016 - 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part II*, pages 339–369, 2016.

BCG17. Olivier Blazy, Céline Chevalier, and Paul Germouty. Almost optimal oblivious transfer from QA-NIZK. In *Applied Cryptography and Network Security - 15th International Conference, ACNS 2017, Kanazawa, Japan, July 10-12, 2017, Proceedings*, pages 579–598, 2017.

BCPV13. Olivier Blazy, Céline Chevalier, David Pointcheval, and Damien Vergnaud. Analysis and improvement of lindell's uc-secure commitment schemes. In *Applied Cryptography and Network Security - 11th International Conference, ACNS 2013, Banff, AB, Canada, June 25-28, 2013. Proceedings*, pages 534–551, 2013.

BCR86. Gilles Brassard, Claude Crépeau, and Jean-Marc Robert. All-or-nothing disclosure of secrets. In *Advances in Cryptology - CRYPTO '86, Santa Barbara, California, USA, 1986, Proceedings*, pages 234–238, 1986.

BDD+17. Paulo S. L. M. Barreto, Bernardo David, Rafael Dowsley, Kirill Morozov, and Anderson C. A. Nascimento. A framework for efficient adaptively secure composable oblivious transfer in the ROM. *IACR Cryptology ePrint Archive*, abs/1710.08256, 2017.

BHR12. Mihir Bellare, Viet Tung Hoang, and Phillip Rogaway. Foundations of garbled circuits. In *the ACM Conference on Computer and Communications Security, CCS'12, Raleigh, NC, USA, October 16-18, 2012*, pages 784–796, 2012.

BMR90. Donald Beaver, Silvio Micali, and Phillip Rogaway. The round complexity of secure protocols (extended abstract). In *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing, May 13-17, 1990, Baltimore, Maryland, USA*, pages 503–513, 1990.

Can01. Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *42nd Annual Symposium on Foundations of Computer Science, FOCS 2001, 14-17 October 2001, Las Vegas, Nevada, USA*, pages 136–145, 2001.

CDD+15. Ignacio Cascudo, Ivan Damgård, Bernardo Machado David, Irene Giacomelli, Jesper Buus Nielsen, and Roberto Trifiletti. Additively homomorphic UC commitments with optimal amortized overhead. In *Public-Key Cryptography - PKC 2015 - 18th IACR International Conference on Practice and Theory in Public-Key Cryptography, Gaithersburg, MD, USA, March 30 - April 1, 2015, Proceedings*, pages 495–515, 2015.

CDD+16. Ignacio Cascudo, Ivan Damgård, Bernardo David, Nico Döttling, and Jesper Buus Nielsen. Rate-1, linear time and additively homomorphic UC commitments. In *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part III*, pages 179–207, 2016.

CDMW09a. Seung Geol Choi, Dana Dachman-Soled, Tal Malkin, and Hoeteck Wee. Improved non-committing encryption with applications to adaptively secure protocols. In *Advances in Cryptology - ASIACRYPT 2009, 15th International Conference on the Theory and Application of Cryptology and Information Security, Tokyo, Japan, December 6-10, 2009. Proceedings*, pages 287–302, 2009.

CDMW09b. Seung Geol Choi, Dana Dachman-Soled, Tal Malkin, and Hoeteck Wee. Simple, black-box constructions of adaptively secure protocols. In *Theory of Cryptography, 6th Theory of Cryptography Conference, TCC 2009, San Francisco, CA, USA, March 15-17, 2009. Proceedings*, pages 387–402, 2009.

CF01. Ran Canetti and Marc Fischlin. Universally composable commitments. In *Advances in Cryptology - CRYPTO 2001, 21st Annual International Cryptology Conference, Santa Barbara, California, USA, August 19-23, 2001, Proceedings*, pages 19–40, 2001.

CJS14. Ran Canetti, Abhishek Jain, and Alessandra Scafuro. Practical UC security with a global random oracle. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, Scottsdale, AZ, USA, November 3-7, 2014*, pages 597–608, 2014.

CKWZ13. Seung Geol Choi, Jonathan Katz, Hoeteck Wee, and Hong-Sheng Zhou. Efficient, adaptively secure, and composable oblivious transfer with a single, global CRS. In *Public-Key Cryptography - PKC 2013 - 16th International Conference on Practice and Theory in Public-Key Cryptography, Nara, Japan, February 26 - March 1, 2013. Proceedings*, pages 73–88, 2013.

CLOS02. Ran Canetti, Yehuda Lindell, Rafail Ostrovsky, and Amit Sahai. Universally composable two-party and multi-party secure computation. In *Proceedings on 34th Annual ACM Symposium on Theory of Computing, May 19-21, 2002, Montréal, Québec, Canada*, pages 494–503, 2002.

CO15. Tung Chou and Claudio Orlandi. The simplest protocol for oblivious transfer. In *Progress in Cryptology - LATINCRYPT 2015 - 4th International Conference on Cryptology and Information Security in Latin America, Guadalajara, Mexico, August 23-26, 2015, Proceedings*, pages 40–58, 2015.

DDGN14. Ivan Damgård, Bernardo Machado David, Irene Giacomelli, and Jesper Buus Nielsen. Compact VSS and efficient homomorphic UC commitments. In *Advances in Cryptology - ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, R.O.C., December 7-11, 2014, Proceedings, Part II*, pages 213–232, 2014.

DG03. Ivan Damgård and Jens Groth. Non-interactive and reusable non-malleable commitment schemes. In *Proceedings of the 35th Annual ACM Symposium on Theory of Computing, June 9-11, 2003, San Diego, CA, USA*, pages 426–437, 2003.

FLM11. Marc Fischlin, Benoît Libert, and Mark Manulis. Non-interactive and re-usable universally composable string commitments with adaptive security. In *Advances in Cryptology - ASIACRYPT 2011 - 17th International Conference on the Theory and Application of Cryptology and Information Security, Seoul, South Korea, December 4-8, 2011. Proceedings*, pages 468–485, 2011.

Fuj16. Eiichiro Fujisaki. Improving practical uc-secure commitments based on the DDH assumption. In *Security and Cryptography for Networks - 10th International Conference, SCN 2016, Amalfi, Italy, August 31 - September 2, 2016, Proceedings*, pages 257–272, 2016.

GH08. Matthew Green and Susan Hohenberger. Universally composable adaptive oblivious transfer. In *Advances in Cryptology - ASIACRYPT 2008, 14th International Conference on the Theory and Application of Cryptology and Information Security, Melbourne, Australia, December 7-11, 2008. Proceedings*, pages 179–197, 2008.

GIKW14. Juan A. Garay, Yuval Ishai, Ranjit Kumaresan, and Hoeteck Wee. On the complexity of UC commitments. In *Advances in Cryptology - EUROCRYPT 2014 - 33rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Copenhagen, Denmark, May 11-15, 2014. Proceedings*, pages 677–694, 2014.

GIR17. Ziya Alper Genç, Vincenzo Iovino, and Alfredo Rial. "the simplest protocol for oblivious transfer" revisited. *IACR Cryptology ePrint Archive*, 2017:370, 2017.

GMW87. Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or A completeness theorem for protocols with honest majority. In *Proceedings of the 19th Annual ACM Symposium on Theory of Computing, 1987, New York, New York, USA*, pages 218–229, 1987.

GMY04. Juan A. Garay, Philip MacKenzie, and Ke Yang. *Efficient and Universally Composable Committed Oblivious Transfer and Applications*, pages 297–316. Springer Berlin Heidelberg, Berlin, Heidelberg, 2004.

GWZ09. Juan A. Garay, Daniel Wichs, and Hong-Sheng Zhou. Somewhat non-committing encryption and efficient adaptively secure oblivious transfer. In *Advances in Cryptology - CRYPTO 2009, 29th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2009. Proceedings*, pages 505–523, 2009.

HK07. Omer Horvitz and Jonathan Katz. Universally-composable two-party computation in two rounds. In *Advances in Cryptology - CRYPTO 2007, 27th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2007, Proceedings*, pages 111–129, 2007.

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

HKK+14. Yan Huang, Jonathan Katz, Vladimir Kolesnikov, Ranjit Kumaresan, and Alex J. Malozemoff. Amortizing garbled circuits. In *Advances in Cryptology - CRYPTO 2014 - 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part II*, pages 458–475, 2014.

HL17. Eduard Hauck and Julian Loss. Efficient and universally composable protocols for oblivious transfer from the cdh assumption. *IACR Cryptology ePrint Archive*, 2017:1011, 2017.

HM04. Dennis Hofheinz and Jörn Müller-Quade. Universally composable commitments using random oracles. In *Theory of Cryptography, First Theory of Cryptography Conference, TCC 2004, Cambridge, MA, USA, February 19-21, 2004, Proceedings*, pages 58–76, 2004.

HV16. Carmit Hazay and Muthuramakrishnan Venkitasubramaniam. On the power of secure two-party computation. In *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part II*, pages 397–429, 2016.

IPS08. Yuval Ishai, Manoj Prabhakaran, and Amit Sahai. Founding cryptography on oblivious transfer - efficiently. In *Advances in Cryptology - CRYPTO 2008, 28th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2008. Proceedings*, pages 572–591, 2008.

JKO13. Marek Jawurek, Florian Kerschbaum, and Claudio Orlandi. Zero-knowledge using garbled circuits: how to prove non-algebraic statements efficiently. In *2013 ACM SIGSAC Conference on Computer and Communications Security, CCS'13, Berlin, Germany, November 4-8, 2013*, pages 955–966, 2013.

JS07. Stanislaw Jarecki and Vitaly Shmatikov. Efficient two-party secure computation on committed inputs. In *Advances in Cryptology - EUROCRYPT 2007, 26th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Barcelona, Spain, May 20-24, 2007, Proceedings*, pages 97–114, 2007.

Kil88. Joe Kilian. Founding cryptography on oblivious transfer. In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing, May 2-4, 1988, Chicago, Illinois, USA*, pages 20–31, 1988.

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

Lin11. Yehuda Lindell. Highly-efficient universally-composable commitments based on the DDH assumption. In *Advances in Cryptology - EUROCRYPT 2011 - 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tallinn, Estonia, May 15-19, 2011. Proceedings*, pages 446–466, 2011.

Lin13. Yehuda Lindell. Fast cut-and-choose based protocols for malicious and covert adversaries. In *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part II*, pages 1–17, 2013.

LM16. Baiyu Li and Daniele Micciancio. Equational security proofs of oblivious transfer protocols. Cryptology ePrint Archive, Report 2016/624, 2016.

LR14. Yehuda Lindell and Ben Riva. Cut-and-choose yao-based secure computation in the online/offline and batch settings. In *Advances in Cryptology - CRYPTO 2014 - 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part II*, pages 476–494, 2014.

LR15. Yehuda Lindell and Ben Riva. Blazing fast 2pc in the offline/online setting with security for malicious adversaries. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, CO, USA, October 12-6, 2015*, pages 579–590, 2015.

MR17. Payman Mohassel and Mike Rosulek. Non-interactive secure 2pc in the offline/online and batch settings. In *Advances in Cryptology - EUROCRYPT 2017 - 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Paris, France, April 30 - May 4, 2017, Proceedings, Part III*, pages 425–455, 2017.

NFT09. Ryo Nishimaki, Eiichiro Fujisaki, and Keisuke Tanaka. Efficient non-interactive universally composable string-commitment schemes. In *Provable Security, Third International Conference, ProvSec 2009, Guangzhou, China, November 11-13, 2009. Proceedings*, pages 3–18, 2009.

NP01. Moni Naor and Benny Pinkas. Efficient oblivious transfer protocols. In *Proceedings of the Twelfth Annual Symposium on Discrete Algorithms, January 7-9, 2001, Washington, DC, USA.*, pages 448–457, 2001.

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

Ped91. Torben P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In *Advances in Cryptology - CRYPTO '91, 11th Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15, 1991, Proceedings*, pages 129–140, 1991.

PVW08. Chris Peikert, Vinod Vaikuntanathan, and Brent Waters. A framework for efficient and composable oblivious transfer. In *Advances in Cryptology - CRYPTO 2008, 28th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2008. Proceedings*, pages 554–571, 2008.

Rab81. Michael O. Rabin. How to exchange secrets with oblivious transfer, 1981. Harvard University Technical Report 81 talr@watson.ibm.com 12955 received 21 Jun 2005.

RR16. Peter Rindal and Mike Rosulek. Faster malicious 2-party secure computation with online/offline dual execution. In *25th USENIX Security Symposium, USENIX Security 16, Austin, TX, USA, August 10-12, 2016.*, pages 297–314, 2016.

Yao82. Andrew Chi-Chih Yao. Protocols for secure computations (extended abstract). In *23rd Annual Symposium on Foundations of Computer Science, Chicago, Illinois, USA, 3-5 November 1982*, pages 160–164, 1982.

ZRE15. Samee Zahur, Mike Rosulek, and David Evans. Two halves make a whole - reducing data transfer in garbled circuits using half gates. In *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part II*, pages 220–250, 2015.

A Static Security in the UC Model

In this subsection we provide an informal description of static security of a protocol π, implementing the two party protocol $\mathcal{F}$, in the UC security model of [Can01]. We refer to their paper for better comprehension of the model.

In this model, the real world execution of protocol π is carried out between the honest parties P_1 and P_2 and an adversary Adv, in the presence of an external entity called the environment $\mathcal{Z}$. All the parties are PPT Turing machines and $\mathcal{Z}$ has an auxiliary information z. At the outset of the protocol the environment initiates the parties with inputs and provides some initial information to Adv. $\mathcal{Z}$ initiates the protocol by activating the parties with inputs. At the outset of the protocol, Adv may or may not corrupt a party. Upon corruption of a party, Adv gets access to the internal state and input of that party. From now on the party will behave according to Adv's instructions (since we are in the malicious model). At the end of the protocol, the honest parties send their output to $\mathcal{Z}$ while Adv outputs $\perp$ on behalf of the corrupted parties and its internal state to $\mathcal{Z}$. We denote the view of $\mathcal{Z}$ as $\text{REAL}_{\mathcal{F},\mathsf{Adv},\mathcal{Z}}(1^\kappa, z)$.

In the ideal world we consider the honest parties P_1 and P_2, a PPT adversary Sim, $\mathcal{Z}$ and the functionality $\mathcal{F}$. Sim has a random tape r and security parameter κ. He simulates the role of Adv in the ideal world and whenever Adv corrupts a party in the real world Sim corrupts that party in the ideal world and gets access to its internal state. Sim invokes the algorithm of Adv, in his head, in another internal protocol execution where Sim simulates the view of the honest parties to Adv. We will denote this internal copy of Adv as $\mathsf{Adv}_{\mathsf{Int}}$. Based on the reply of $\mathsf{Adv}_{\mathsf{Int}}$ in the internal execution, Sim behaves accordingly in the ideal world execution. He extracts the inputs of the corrupted parties in the internal execution and invokes $\mathcal{F}$ in the ideal world with those inputs to obtain the output. In the internal execution he simulates the protocol in such a way that $\mathsf{Adv}_{\mathsf{Int}}$ obtains that output. At the end of the protocol, $\mathsf{Adv}_{\mathsf{Int}}$ forwards his view to Sim who forwards it to $\mathcal{Z}$. We denote the view of $\mathcal{Z}$ as $\text{IDEAL}_{\mathcal{F},\mathsf{Sim},\mathcal{Z}}(1^\kappa, z)$.

Definition 1. *Let π be a protocol for computing a functionality $\mathcal{F}$. We say that π UC-securely computes the two party protocol functionality $\mathcal{F}$ in the presence of static adversaries if for every PPT adaptive real-world adversary Adv and every environment $\mathcal{Z}$, there exists a PPT ideal-world adversary Sim, such that:*

$$\text{REAL}_{\mathcal{F},\mathsf{Adv},\mathcal{Z}}(1^\kappa, z) \overset{c}{\approx} \text{IDEAL}_{\mathcal{F},\mathsf{Sim},\mathcal{Z}}(1^\kappa, z)$$

B Adaptive Security in the UC Model

In this subsection we provide an informal description of adaptive security of a protocol π, implementing the two party protocol $\mathcal{F}$, in the UC security model of [Can01]. We refer to their paper for better comprehension of the model.

In this model, the real world execution of protocol π is carried out between the honest parties P_1 and P_2 and an adversary Adv, in the presence of an external entity called the environment $\mathcal{Z}$. All the parties are PPT Turing machines and $\mathcal{Z}$ has an auxiliary information z. At the outset of the protocol the environment initiates the parties with inputs and provides some initial information to Adv. $\mathcal{Z}$ and Adv can interact with each other throughout the protocol . The execution of the protocol proceeds in rounds where in each round Adv might corrupt a party who is supposed to be active in that round. Next, Adv activates the party which is supposed to be active in this round. He sees the protocol transcript of that round and based on that he might corrupt an honest party in the future. Upon corruption of a party, Adv gets access to the internal state and input of that party. From now on the party will behave according to Adv's instructions (since we are in the malicious model). At the end of the protocol, the honest parties send their output to $\mathcal{Z}$ while Adv outputs $\perp$ on behalf of the corrupted parties and its internal state to $\mathcal{Z}$. A post execution corruption occurs, where $\mathcal{Z}$ can corrupt an honest party and obtain his internal state. We denote the view of $\mathcal{Z}$ as $\text{REAL}_{\mathcal{F},\mathsf{Adv},\mathcal{Z}}(1^\kappa, z)$.

In the ideal world we consider the honest parties P_1 and P_2, a PPT adversary Sim, $\mathcal{Z}$ and the functionality $\mathcal{F}$. Sim has a random tape r and security parameter κ. He simulates the role of Adv in the ideal world and whenever Adv corrupts a party in the real world Sim corrupts that party in the ideal world and gets access to its internal state. Sim invokes the algorithm of Adv, in his head, in another internal protocol execution where Sim simulates the view of the honest parties to Adv. We will denote this internal copy of Adv as $\mathsf{Adv}_{\mathsf{Int}}$. Based on the reply of $\mathsf{Adv}_{\mathsf{Int}}$ in the internal execution, Sim behaves accordingly in the ideal world execution. He extracts the inputs of the corrupted parties in the internal execution and invokes $\mathcal{F}$ in the ideal world with those inputs to obtain the output. In the internal execution he simulates the protocol in such a way that $\mathsf{Adv}_{\mathsf{Int}}$ obtains that output. At the end of the protocol, $\mathsf{Adv}_{\mathsf{Int}}$ forwards his view to Sim who forwards it to $\mathcal{Z}$. A post execution phase occurs in the ideal world similar to the real world. We denote the view of $\mathcal{Z}$ as $\text{IDEAL}_{\mathcal{F},\mathsf{Sim},\mathcal{Z}}(1^\kappa, z)$.

Definition 2. *Let π be a protocol for computing a functionality $\mathcal{F}$. We say that π UC-securely computes the two party protocol functionality $\mathcal{F}$ in the presence of adaptive adversaries if for every PPT adaptive real-world adversary Adv and every environment $\mathcal{Z}$, there exists a PPT ideal-world adversary Sim, such that:*

$$\text{REAL}_{\mathcal{F},\mathsf{Adv},\mathcal{Z}}(1^\kappa, z) \overset{c}{\approx} \text{IDEAL}_{\mathcal{F},\mathsf{Sim},\mathcal{Z}}(1^\kappa, z)$$

C Garbled Circuits

Bellare et al [BHR12] gave an abstraction of garbling schemes for circuits and formalized several notions of security. Using the language of [BHR12] for circuits; the

circuit itself is a directed acyclic graph, where each gate g is indexed by its outgoing wire, and its left and right incoming wires $A(g)$ and $B(g)$ are numbered such that $g > B(g) > A(g)$. Also, a circuit output wire can not be an input wire to any gate. We denote the number of input wires, gates and output wires using n, q and m respectively in a circuit $\mathcal{C}$.

At a high-level, a garbling scheme consists of the following algorithms: Gb takes a circuit as input and outputs a garbled circuit, encoding information, and decoding information. En takes an input x and encoding information and outputs a garbled input X. Ev takes a garbled circuit and garbled input X and outputs a garbled output Y. Finally, De takes a garbled output Y and decoding information and outputs a plain circuit-output (or an error, $\perp$).

In [JKO13], there is an additional verification algorithm in the garbling scheme which when accepts a given $(\mathsf{GC}, \mathsf{e})$ signifies that the GC is correct, and that the garbled output corresponding to any clear output can be extracted. Formally, a *garbling scheme* is defined by a tuple of functions $\mathsf{Garble} = (\mathsf{Gb}, \mathsf{En}, \mathsf{Ev}, \mathsf{De}, \mathsf{Ve})$, described as follows:

- *Garble* algorithm $\mathsf{Gb}\,(1^\kappa, \mathcal{C})$: A randomized algorithm which takes as input the security parameter and a circuit $\mathcal{C} : \{0,1\}^n \to \{0,1\}^m$ and outputs a tuple of strings $(\mathsf{GC}, \mathsf{e}, \mathsf{d})$, where GC is the garbled circuit, e denotes the input-wire labels, and d denotes the decoding information.
- *Encode* algorithm $\mathsf{En}\,(x, \mathsf{e})$: a deterministic algorithm that outputs the garbled input X corresponding to input x.
- *Evaluation* algorithm $\mathsf{Ev}\,(\mathsf{GC}, \mathsf{X})$: A deterministic algorithm which evaluates garbled circuit GC on garbled input X, and outputs a garbled output Y.
- *Decode* algorithm $\mathsf{De}\,(\mathsf{Y}, \mathsf{d})$: A deterministic algorithm that outputs the plaintext output corresponding to Y or $\perp$ signifying an error if the garbled output Y is invalid.
- *Verify* algorithm $\mathsf{Ve}\,(\mathcal{C}, \mathsf{GC}, \mathsf{e})$: A deterministic algorithm which takes as input a circuit $\mathcal{C} : \{0,1\}^n \mapsto \{0,1\}^m$, a garbled circuit (possibly malicious) GC, encoding information e, and outputs d when GC is a valid garbling of $\mathcal{C}$, and $\perp$ otherwise.

A garbling scheme may satisfy several properties such as *correctness, privacy, obliviousness, authenticity and verifiability*. We review some of these notions below. The definitions for correctness and authenticity are standard: correctness enforces that a correctly garbled circuit, when evaluated, outputs the correct output of the underlying circuit; authenticity enforces that the evaluator can only learn the output label that corresponds to the value of the function. *Verifiability* [JKO13] allows one to check that the garbling of a circuit indeed implements the specified plaintext circuit $\mathcal{C}$. Given that verification succeeds for a candidate $(\mathcal{C}, \mathsf{GC}, \mathsf{e})$, the garbled output corresponding to a given clear output can be extracted. We provide definitions of correctness, privacy and verifiability as we need it for our π_{CRS} protocol.

Definition 3. *(Correctness) A garbling scheme* Garble *is* ***correct*** *if for all input lengths* $n \leq \mathsf{poly}(\kappa)$*, circuits* $\mathcal{C} : \{0,1\}^n \to \{0,1\}^m$ *and inputs* $x \in \{0,1\}^n$*, the following probability is negligible in* κ*:*

$$\Pr\left(\mathsf{De}(\mathsf{Ev}(\mathsf{GC}, \mathsf{En}(\mathsf{e}, x)), \mathsf{d}) \neq \mathcal{C}(x) : (\mathsf{GC}, \mathsf{e}, \mathsf{d}) \leftarrow \mathsf{Gb}(1^\kappa, \mathcal{C})\right).$$

Definition 4. *(Privacy) A garbling scheme* Garble *is* $\mathsf{private}$ *if for all input lengths* $n \leq \mathsf{poly}(\kappa)$*, circuits* $\mathcal{C} : \{0,1\}^n \to \{0,1\}^m$*, there exists a* PPT *simulator* Sim *such that for all inputs* $x \in \{0,1\}^n$*, for all probabilistic polynomial-time adversaries* Adv*, the following two distributions are computationally indistinguishable:*

- $\text{REAL}(\mathcal{C}, x)$: *run* $(\mathsf{GC}, \mathsf{e}, \mathsf{d}) \leftarrow \mathsf{Gb}(1^\kappa, \mathcal{C})$*, and output* $(\mathsf{GC}, \mathsf{En}(x, \mathsf{e}), \mathsf{d})$.
- $\text{IDEAL}_{\mathsf{Sim}}(\mathcal{C}, \mathcal{C}(x))$*: output* $(\mathsf{GC}', \mathsf{X}, \mathsf{d}') \leftarrow \mathsf{Sim}(1^\kappa, \mathcal{C}, \mathcal{C}(x))$

Definition 5. *(Verifiability) A garbling scheme* Garble*is* $\mathsf{verifiable}$ *if for all input lengths* $n \leq \mathsf{poly}(\kappa)$*, circuits* $\mathcal{C} : \{0,1\}^n \to \{0,1\}^m$*, inputs* $x \in \{0,1\}^n$*, and PPT adversaries* Adv*, the following probability is negligible in* κ*:*

$$\Pr\left(\mathsf{De}(\mathsf{Ev}(\mathsf{GC}, \mathsf{En}(x, \mathsf{e})), \mathsf{d}) \neq \mathcal{C}(x) \ : \ \begin{array}{c}(\mathsf{GC}, \mathsf{e}, \mathsf{d}) \leftarrow \mathsf{Adv}(1^\kappa, \mathcal{C}) \\ \mathsf{Ve}\,(\mathcal{C}, \mathsf{GC}, \mathsf{e}) = \mathsf{d} \neq \bot\end{array}\right)$$

We are interested in a class of garbling schemes referred to as *projective* in [BHR12]. When garbling a circuit $\mathcal{C} : \{0,1\}^n \mapsto \{0,1\}^m$, a projective garbling scheme produces encoding information of the form $\mathsf{e} = \left(\mathbb{K}_i^0, \mathbb{K}_i^1\right)_{i \in [n]}$, and the encoded input X corresponding to $x = (x_i)_{i \in [n]}$ can be interpreted as $\mathsf{X} = \mathsf{En}(x, \mathsf{e}) = (\mathbb{K}_i^{x_i})_{i \in [n]}$.