Fast and 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", efficient 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 realizations of these primitives have been found to be challenging, especially in the no erasure model. We make progress in this direction and provide efficient constructions that are Universally-Composable in the random oracle model.

Oblivious Transfer. We present the first *round optimal* framework for building adaptively-secure OT in the programmable random oracle (PRO) model, relying upon the framework of *Peikert* et al. (Crypto 2008). When instantiated with Decisional Diffie Hellman assumption, 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. This computation overhead translates to 0.02% and 1% in the LAN and WAN setting. Additionally, we obtain 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 in the PRO model. We complete the picture of efficient OT constructions by presenting the first adaptively secure OT Extension, extending the protocol of Asharov et al. (Eurocrypt 2015) for the adaptive setting using PRO. Our OT extension enables us to obtain adaptive OTs at an amortized cost of 3 symmetric key operations and communication of 3κ bit strings. It incurs a runtime overhead of 2% and 11.95%, in the LAN and WAN setting and almost no communication overhead over the static OT extension protocol. In concrete terms, the cost is 2 μs and 115 μs for each OT in LAN and WAN.

Commitment Scheme. We present an adaptively secure commitment scheme in the Global Random Oracle model solely relying on observable random oracle (ORO). Our commitment scheme has a one-time offline setup phase, where a common reference string (crs) is generated between the parties using an ORO. In the online phase, the parties use the crs and ORO to generate commitments in a non-interactive fashion. Our construction incurs communication of 4κ bit strings and computation of 4 exponentiations and 4 random oracle queries for committing to an arbitrary length message. Empirically, it takes around 0.18ms and 0.2 ms for committing to 128 bits and 2048 bits respectively. It finds applications in secure two-party computation (2PC) protocols that adopt offline-online paradigm, where the crs can be generated in the offline phase and the scheme can be used in the online 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, 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 precisely 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]. We aim at obtaining efficient, adaptively-secure primitives which are instrumental in construction of practical 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. Various constructions of OT (mainly 1-out-of-2) have been proposed providing both static and adaptive security. The protocols of [CO15, PVW08, NP05] deal with malicious adversaries in the static model whereas those of [BCG17, BC16, CKWZ13, GWZ09] address the same in adaptive model. However, the OT protocols that achieve adaptive security lack optimality in terms of efficiency- rounds, computation and communication. We have round optimal and efficient protocols in the literature for the static case. However, designing adaptively-secure protocols has been a challenging task. Infact, there is no known OT protocol in the literature which is round optimal. In this direction, we present the first round optimal OT protocol which is secure against adaptive adversaries without erasures.

Another interesting direction to consider for OTs is to reduce the number of public key operations. The impossibility result by [IR89] states that its highly unlikely that OTs can be constructed without public key operations. In order to circumvent this limitation, the concept of OT Extension [Bea96, IKNP03, ALSZ13, ALSZ15, KOS15] was introduced and explored. It allows the parties to execute a small number of OTs, called seed OTs, and then extend them to obtain large number of OTs using cheap symmetric key operations. The amortized cost of generating one single OT reduces to a constant number of symmetric key operations. However, there is no known OT extension protocol in the adaptive setting. An effort towards obtaining adaptively secure OT Extension would be of quite interest as it would open the gates towards constructing large number of efficient adaptively secure OTs using small number of seed OTs. Our paper presents one such result for adaptively-secure OT extension.

Commitment Scheme is another fundamental primitive in the MPC literature that draws attention. 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. Besides its involvement in MPC protocols, commitment schemes in the UC model also have implications in key exchange [DG03] and are non-malleable [CF01] in nature. In this work, we explore UC secure commitment protocols which are secure against adaptive adversaries.

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 of theoretical interest, [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 Learning With Errors (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, the literature can be divided based on the erasure model. The works of [ABB$^+$13, CKWZ13, BCG17, BC16, BC15] rely on secure erasures of the memory, whereas [GWZ09, CDMW09a, CDMW09b, BDD$^+$17] consider the stronger model of no erasures. [CKWZ13] presented a framework for adaptively-secure OT in the global crs model. They provide instantiations under various assumptions- DLIN, Symmetric External Diffie Hellman (SXDH), DDH and DCR. However, their protocols are not round optimal and achieve adaptive security at the cost of significant overhead in communication and computation compared to the PVW protocol. [CKWZ13] also provided two constructions (Appendix A of [CKWZ13]) of [GWZ09] framework under Decisional Linear (DLIN) assumptions. These instantiations are adaptively-secure with erasures, with computational overhead reduced to constant number of exponentiations. There has been a separate line of work [BCG17, BC16, BC15, ABB$^+$13] based on password-authenticated key exchange (PAKE) and smooth projective hash functions. They require atleast 3 rounds of communication, assuming erasures for adaptivity. On the other hand, achieving adaptivity with no erasures is a challenging task. [GWZ09] followed the compiler approach to transform the [PVW08] framework into an adaptively-secure OT using adaptively secure commitments. It was further improved by a recent work of [ABP17]. Both protocols incur an overhead of $\mathcal{O}(n)$ exponentiations, where n is the size of sender's input messages. The works of [CDMW09a, CDMW09b] proposed constructions which are of theoretical interest. [CDMW09a] presented an OT construction using non-committing encryption (NCE). On the other hand, [CDMW09b] compiled protocols that are secure against a semi-honest adaptive adversary into one that is secure against a malicious adaptive adversary.

The work of [CO15], the "simplest OT" protocol explored 3-round OT constructions in the PRO model. Although the paper claimed adaptive security, several bugs have been identified [GIR17, BDD$^+$17, HL17] recently in their static security proof thereby rendering the protocol of [CO15] insecure in the UC model. Recently two more works ([HL17],[BDD$^+$17][1]) have claimed to achieve adaptive security in the same model. However, in Section 3, we give a justification that these protocols are not UC-secure. We note that the authors of [BDD$^+$17] have updated their protocol and their new version presents a 3 round OT framework which can be instantiated under Learning from Parity with Noise, McEliece cryptosystem, QC-MDPC, LWE and Computational Diffie Hellman (CDH) in the PRO model. Their most efficient instantiation (under CDH) incurs twice the amount of communication, while maintaining the same computation cost as ours. Consequently, the problem of attaining an *round-optimal, efficient and adaptively-secure OT* continued to remain open, which we try to address through our work. Table 1 summarizes the literature on adaptively-secure OT protocols and our result. We do not compare with [GWZ09, CDMW09a, CDMW09b, ABP17] in the table since they require atleast $\mathcal{O}(n)$ exponentiations, where sender's input message is of n bits, whereas the other protocols in the table require constant number of exponentiations. Among the PAKE-based schemes we compare with the most efficient works of [ABB$^+$13, BCG17].

Another interesting factor to consider is the model of security that these protocols satisfy. All of the above protocols (except [BDD$^+$17]) are secure in the plain UC-

[1] previous version of their paper

model [Can01] which allows private crs. There is a stronger model of security, i.e. the Generalized UC model (GUC) [CDPW07], which guarantees stronger compositional guarantees than the plain UC model. In the GUC model, the parties are not allowed to possess private crs and the random oracle is a public object, shared between arbitrary protocols. This prevents the simulator from manipulating the crs during simulation, rendering the simulation procedure more difficult. Interestingly, the recent work of [CDG$^+$18] has shown that the random oracle in the GUC model can satisfy the property of programmability. Such a random oracle is formally known as Global Random Oracle (GRO). Utilizing the programmability feature of the GRO, our protocol and the protocol of [BDD$^+$17] achieve adaptive security. Whereas, the other protocols require a local crs, which is private to one instance of the protocol, thereby hampering composition in the GUC model.

Oblivious Transfer Extension. Next, we consider the problem of OT Extension, which was introduced by the work of [Bea96], followed by the seminal work of [IKNP03]. The paper of [IKNP03] presented an efficient 1-out-of-2 semi-honest OT Extension protocol which was secure against static adversaries. An optimized version of this protocol appeared in [ALSZ13]. The paper of [ALSZ15, KOS15] presented the actively secure versions. The paper of [KK13] gave constructions for 1-out-of-N case, which were made actively secure by [PSS17, OOS17]. However all of these protocols are in the static setting and it was not known *whether adaptively-secure OT Extension protocol is possible.* Our work answers it in an affirmative way by proving that existing static OT extension [ALSZ13, ALSZ15] schemes satisfy adaptive security under the PRO assumption.

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] + [FLM11][1]	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$15	3	SXDH	crs	Adaptive with erasures
[CKWZ13]	35	4	19	3	37	4	DDH	crs	Adaptive with erasures
[CKWZ13]	28	4	13	3	26	4	DCR	crs	Adaptive with erasures
[ABB$^+$13]	15	2	13	1	11	3	SXDH	crs	Adaptive with erasures
[BCG17]	10	4	18	4	9	3	SXDH	crs	Adaptive with erasures
PVW	6	-	8	-	3	2	DDH	crs	Static
[BDD$^+$17]	15	5	6	2	5	3	CDH	PRO	Adaptive (GUC model)
Our scheme	7	3	8	2	3	2	DDH	PRO	Adaptive (GUC model)
Our scheme (by OT Extension)	3	2	-	2	-	3	Static Receiver Equivocal OT	PRO	Adaptive (GUC model)

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

Commitment Scheme. 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, ABP17] and many more. We highlight some of the notable works in the relevant literature classified based on their round complexity, the security they achieve and the security model they are proven to be secure in.

The contributions of [DG03, Lin11, GIKW14, Fuj16, FLM11, CJS14, ABP17] 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, ABP17] 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. These schemes [CDD+15, DDGN14] consist of an input independent setup phase, which is run in the offline phase, while the NICOM is efficiently used in the online phase. The cost of the setup phase can be substantial but it gets amortized over multiple commitments. Our scheme also follows this particular model.

Similar to OT literature, 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. [NFT09, FLM11, BCPV13, Fuj16] proposed adaptively-secure protocols which rely on secure erasure of the party's internal state whereas the constructions of [CF01, CLOS02, HM04, ABB+13, HV15, ABP17, HPV17] achieved the same level of security without erasures. We skim through the protocols in the non-erasure model and compare all the above mentioned works with our protocol in Tab. 2.

[HV15] presented a theoretical construction of an interactive adaptive commitment scheme based on the minimal assumption of trapdoor simulatable public-key encryption. Since our focus is on adaptively-secure NICOM schemes, we elaborate the relevant NICOM results. [CF01, CLOS02, ABB+13, ABP17] 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].

The literature can also be classified based on the security model they are proven secure in. All of the above protocols (except [HM04, CJS14, HPV17]) are proven secure in the plain UC-model [Can01]. Recall, that the work of [CDG+18] showed that the GUC model allows observability and programmability from the random oracle. Utilizing the programmability feature of the RO, the protocol of [HM04] achieves adaptive security in the GUC model. Also, the work of [CDG+18] showed that the folklore commitment scheme $\mathcal{F}_{\mathsf{RO}}(m; r)$ is GUC secure assuming programmability from $\mathcal{F}_{\mathsf{RO}}$, where $\mathcal{F}_{\mathsf{RO}}$ is the RO functionality, m is the message and r is the randomness. And it can be trivially shown that it achieves adaptive security too. The work of [CJS14] presents a GUC secure static commitment scheme, relying on the observability property of the GRO. [HPV17] presents a theoretical construction of an adaptively secure commitment scheme relying on one way function. They achieve GUC security in the tamper-proof hardware model. From the literature we can observe that, adaptive commitments in the GUC model either asks for programmability from the RO or it incurs a blowup in terms of efficiency. Hence, we can ask the following question:

"Can we obtain an efficient, non-interactive commitment scheme based on the non-programmable random oracle which is adaptively secure in the GUC model?"

Our paper answers it in affirmative by presenting an adaptively-secure NICOM in the offline-online model, relying solely on the observable property of the RO. 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	≥ 9 + $\mathcal{O}(\kappa^2)$ (one-time)	1/1 + 5 (one-time)	OT	crs	Static
[CJS14]	poly(κ)	7	2/3	DLP	ORO	Static (GUC model)
[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
[HM04]	poly(κ)	5	1/1	DLP	PRO	Adaptive (GUC model)
[ABB$^+$13]	κ	$\mathcal{O}(\kappa)$	1/1	SXDH	crs	Adaptive
[ABP17]	κ	$\mathcal{O}(\kappa)$	1/1	DDH	crs	Adaptive
Our Scheme	poly(κ)	4 + $\mathcal{O}(\nu\lvert\mathcal{C}\rvert)$ (one-time)	1/1 + 4 (one-time)	DLP	ORO	Adaptive (GUC model)

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, GUC - Generalized UC model ORO - observable random oracle, circuit $\mathcal{C}$ computes g^x
Note : The protocol of [CJS14] and [HM04] requires a non-interactive trapdoor commitment scheme. It has been instantiated with Pedersen commitment since to the best of our knowledge such a commitment scheme does not exist based on OWF.

1.2 Our Results

In this work, we focus on optimizing the round complexity while attaining adaptive security in an efficient manner, for two widely used primitives: Oblivious Transfer and Commitment Scheme. We also present an adaptively secure well-defined transformation from $\log \mathsf{N}$ 1-out-of-2 OTs to 1-out-of-N OT, restricting the number of exponentiations to $\mathcal{O}(\log \mathsf{N})$. Finally, we conclude with detailed implementation results of our primitives. Our contributions are stated below.

Adaptively Secure 1-out-of-2 Oblivious Transfer. We construct the first OT framework that is round-optimal, GUC 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. We apply the same observation on the static OT framework of [PVW08]. At the heart of their framework lies the Dual Mode Encryption Scheme (DME) which requires a crs for its functioning. We generate the crs of the Dual Mode Encryption (DME) scheme using the PRO. During simulation the crs can be suitably modified, to equivocate R's view, by programming the PRO. However, for our scheme it should be possible to generate the crs of the DME using an RO. Hence, we customize the definition of DME, based on our requirements, to obtain a stronger version, called Samplable DME. Once the crs has been generated it can be suitably modified, to extract/equivocate R's view, by programming the PRO. On the other hand, S's messages are encrypted using another PRO, such that it enables equivocation of S's view when required. Thus, we replace the crs in the round-optimal [PVW08] framework with PRO to achieve adaptive security. A similar observation was made by the work of [CJS14] where they tried to generate the crs using the observability property of the GRO. However, their goal was to obtain one-sided simulatable static OT in the GUC model. Whereas, we aim for adaptive security in the GUC model relying on the programmability feature. In our framework, the DME scheme can be instantiated under the DDH and LWE assumptions. Additionally, when instantiated with DDH assumption, our protocol incurs a computation overhead of 5 random oracle queries and a minimal communication overhead of one κ-bit string over the static protocol of PVW (the DDH-based instantiation of [PVW08]).Tab. 1 and Tab. 3 compare our scheme with various other schemes. Empirically our protocol requires 1.676s and 4.3s in LAN and WAN setting, incurring a nominal overhead of 0.02% and 1%, in the same settings, over PVW. The protocol of [BDD$^+$17] requires at least the same runtime as ours, since the number of symmetric key operations in [BDD$^+$17] is more than us, although the number of public key operations in both the protocols are same.

Table 3: Table indicating implementation results for Oblivious Transfer protocol in LAN setting, for $n = 128$ and $\kappa = 128$.

Scheme	Communication (bits)	Total Runtime (s)	Rounds	Assumptions	Security
PVW	768	2.645	2	DDH	Static
[BDD$^+$17]	1920	> 2.647	3	CDH	Adaptive
Our Scheme	896	2.647	2	DDH	Adaptive
Our Scheme (from OT extension)	716.7	20.954×10^{-7}	3	DDH	Adaptive
Our Scheme (from OT extension)	702	1.043×10^{-7}	5	DDH	Adaptive

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, it can be shown that for the semi-honest setting the seed OTs can be statically secure, if we consider the simulation of the 1-out-of-2 OTs in a non-blackbox manner. For the active setting, 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, if the simulation of the 1-out-of-2 OTs is performed in a non-blackbox manner. This implies that we can plug-in statically-secure 1-out-of-2 OTs which satisfy the property of receiver equivocability [GKPS18].

Oblivious Transfer Extension. We provide the first adaptively-secure protocols for OT Extension solely relying on the PRO assumption. In this regard we present two results, one corresponding to semi-honest setting and the other for the active setting. Our first result proves that the semi-honest protocol of [ALSZ13] can be made adaptively-secure. Interestingly, we show that the seed OTs can be statically secure, if we invoke them in a non-blackbox way. We know that for adaptive security, blackbox-usage of the seed OTs is not possible in our construction since it would violate the results of [LZ13, IR89]. [LZ13] proves that the existence of OT extension protocol, secure against semi-honest adaptive adversaries, imply OT protocol secure against static semi-honest adversaries. In that case, blackbox usage of seed OTs establishes that PRO would imply static semi-honest OT, contradicting the result of [IR89] which states that public key operations are necessary for static OT. Our second result proves that the actively secure protocol of [ALSZ15] can be made adaptively-secure against active adversaries. The paper of [ALSZ15] gave two constructions - a 3 round and a 5 round OT extension in their paper. The 3 round OT extension consists of pair-wise consistency check for all the columns. Whereas, in the 5 round protocol, the parties performed a coin-tossing subprotocol and based on the outcome of the subprotocol, they performed a consistency check on the columns. The 5 round protocol is more efficient than the 3 round one due to reduced number of checks. Similarly, we consider a 3 round and a 5 round variant of our OT extension protocol. Interestingly, we demonstrate that the seed OTs in our case can be replaced with receiver equivocal static OTs which are secure against active adversaries. Our OT Extension protocols preserve the efficiency of the original static protocols, yielding adaptive 1-out-of-2 OTs at an amortized cost of 3 symmetric key operations and 3κ bits communication per OT. Moreover, if we combine the OT Extension protocol with our 1-out-of-N Transformation, then we obtain 1-out-of-N adaptive OTs at an amortized cost of $\mathsf{N} + 3 \log \mathsf{N} + 1$ symmetric key operations per OT. The other adaptive protocols [ABB$^+$13, BC15, BC16, BCG17, BDD$^+$17] require $\mathcal{O}(\mathsf{N})$ public key operations instead. Next, we briefly highlight the implementation results regarding the OT extension protocol. The 5 round protocol is more efficient than the 3 round one due to reduced number of checks. Empirically, the 3 round protocol has a rough multiplicative overhead of 20 over the 5 round protocol for LAN runtime, while generating 2^{23} OTs together. The overhead will decrease for larger number of extended OTs, as the number of checks remain constant, while the number of extended OTs increase. Throughout the paper our implementation results for the OT extension protocol, correspond to the 5 round protocol unless specified otherwise. Empirically our OT extension protocol has an overhead of 2% and 11.95% over [ALSZ15] in the LAN and WAN setting respectively. The amortized (over $1.25 * 10^5$) cost of generating one adaptive 1-out-of-2 drastically reduces to 2.02 μs and 115.5 μs in LAN and WAN setting for OT output length n=16 bits. We also generate 1-out-of-16 OTs from 4 copies of 1-out-of-2 OTs using our transformation. The amortized cost for each 1-out-of-16 OT is 21.3 μs and 145.5 μs in LAN and WAN setting respectively. In Tab. 3 we compare our OT protocols with PVW and [BDD$^+$17]. Our extended adaptive OTs are useful in practical MPC applications where the MPC protocols require a large number of OTs. In such a situation it can be observed that if we are given 3 rounds, then we can run our 3 round OT extension protocol to obtain adaptive OTs at a minimal cost of 2.1μs (amortized over 10^6 OTs), whereas other 3 round protocols (like [BDD$^+$17]) would require 2.647s, which is 10^6 times slower than our extended OTs. The cost of each extended OT reduces drastically with amortization, i.e. the total number of extended OTs. Given 5 rounds, our OT extension protocol generates adaptive OTs at a cost of 0.1043 μs (amortized over 10^6 OTs). The protocol of [BDD$^+$17] would be 12×10^5 times slower than our extended OTs. If we consider our 5 round OT extension protocol then the overhead factor increases to 25×10^6.

Commitment Scheme. We construct a NICOM, in the offline-online model, that is secure against an adaptive adversary without erasures. First, we generate a crs for Pedersen commitment using an ORO in the setup phase. The crs is of the form (g, h), where $h = g^x$ for $g, h \in \mathbb{G}$ and $x \in \mathbb{Z}_p$. $\mathbb{G}$ and $\mathbb{Z}_p$ denote a multiplicative group of prime order p and a prime field of order p respectively. The setup phase is a 4 round protocol where the parties perform a coin tossing protocol and a Zero Knowledge Proof of Knowledge (ZKPoK) protocol, relying on garbled circuits. Once generated, the crs alongwith the ORO can be reused to construct several instances of the NICOM. 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, ORO permits committing to a message of length ℓ while incurring the overhead of committing to a κ bit string, where $\ell = \mathsf{poly}(\kappa)$. Compressing the message from ℓ 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 messages. Our protocol involves communication of one κ bit string, 3 group elements, computation of 4 exponentiations and 4 random oracle queries to commit to ℓ bits. This yields an efficient adaptively secure commitment scheme which is practically motivated for 2PC/MPC protocols based on offline-online paradigm [LR15, RR16]. The setup can be run in the offline phase while the commitment scheme can be conveniently used in the online phase. The ORO in our protocol can be instantiated using a SHA-256 hash function and it can be proven secure in GUC model. This renders our commitment scheme useful in real-life situations where concurrently many protocols are run and share the same hash function. Table 2 compares our commitment schemes with the recent literature. In terms of empirical costs, our scheme takes approx. 0.18 ms and 0.20 ms to obtain commitment of 128 bits and 2048 bits respectively. In comparison, the scheme of [HM04] takes approx 0.42 ms and 0.49 ms for commitment of 128 and 2048 bits respectively. The protocol of [HM04] requires a non-interactive trapdoor commitment scheme which has been instantiated with Pedersen commitment since to the best of our knowledge such a commitment scheme does not exist based on OWF. The comparison can be viewed in Table 4. Based on the results, we conclude that our scheme is approximately 2 times faster than [HM04] even though we rely on a weaker assumption from the RO.

Table 4: Table indicating implementation results for our Commitment Scheme π_{COM} and [HM04] in the LAN setting.

Message Size (bits)	Scheme	Runtime of S (ms)	Runtime of R (ms)	Commitment Size (bits)
128	Our Scheme	0.184	0.185	128 + 512
	[HM04]	0.423	0.418	128 + 640
256	Our Scheme	0.189	0.187	256 + 512
	[HM04]	0.435	0.432	256 + 640
1024	Our Scheme	0.192	0.191	1024 + 512
	[HM04]	0.455	0.469	1024 + 640
2048	Our Scheme	0.198	0.202	2048 + 512
	[HM04]	0.487	0.490	2048 + 640

Note: Commitment Size = message size + (commitment & decommitment size)

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 concurrent adaptive OT papers[CO15, BDD$^+$17, HL17] in the PRO model. We proceed to the definition of Samplable DME (referred as DME only) in Section. 4. Then we present our OT protocol framework in Section 5. The 1-out-of-2 to 1-out-of-N OT transformation is elaborated in Section 6. Section 7 presents the results on OT Extension. Our commitment scheme is covered in Section 8. The implementation details of our protocols have been illustrated in Section 9. Finally, we conclude with prospective future work in Section 10. We provide a summary of UC framework for static and adaptive security in Appendix A for the sake of completeness. We refer to it for better comprehension of notations used in our security proofs. Appendices C and D present instantiations of our DME based on DDH and LWE assumptions.

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 ν and the computational security parameter by κ. We use $\overset{c}{\approx}$ and $\overset{s}{\approx}$ to denote computational and statistical indistinguishability, respectively. Let $\mathbb{Z}_p$ denote the field of order p, where p is a prime. Let $\mathbb{G}$ be the multiplicative group corresponding to $\mathbb{Z}_p^*$ with generator g. 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 B) 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.

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

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

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. 2: The ideal functionality $\mathcal{F}_{\textsf{N-OT}}$ for oblivious transfer

$\mathcal{F}_{\textsf{N-OT}}$

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

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

Commitment Schemes. Commitment schemes allow a party $\textsf{S}$ to commit to a message m using randomness r. It keeps the message hidden, while allowing $\textsf{S}$ 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 $\mathcal{F}_{\textsf{COM}}$ has been depicted in Fig. 3.

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

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

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

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

Random Oracle Model. A random oracle functionality is parametrized by a domain and a range and it is as $\mathcal{F}_{\textsf{RO}}$ in Fig. 4. A random oracle query on message m is denoted by $\mathcal{F}_{\textsf{RO}}(m)$. The random oracle functionality can be broadly classified [CDG+18] into three categories based on its features- plain RO, observable RO and programmable RO. A plain RO returns a random string, from its range, upon being queried on a message m, from its domain. An observable RO inherits the properties of the plain RO but in addition it grants the simulator to observe the queries made, to $\mathcal{F}_{\textsf{RO}}$, by the adversary. A programmable RO allows the simulator to program $\mathcal{F}_{\textsf{RO}}(m)$ to return any string from the range, upon being queried on m for the first time.

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. The previous version of [BDD+17] proposed 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). We note however that the authors of [BDD+17] have fixed their protocol, making it UC secure. [HL17] proposes a construction of 1-out-of-$\textsf{N}$ OT under the CDH assumption. However, both protocols

Fig. 4: Functionality $\mathcal{F}_{\mathsf{RO}}$

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

$\mathcal{F}_{\mathsf{RO}}$ is parameterized by a domain D and range R and it proceeds as follows, running on security parameter k:

– $\mathcal{F}_{\mathsf{RO}}$ maintains a list L (which is initially empty) of pairs of values $(\hat{m}, \hat{h})$, s.t. $\hat{m} \in D$ and $\hat{h} \in R$.
– Upon receiving a value (sid, m) (where $m \in D$) perform the following: If there is a pair $(m, \hat{h})$, for some $\hat{h} \in R$, in the list L, set $h := \hat{h}$. If there is no such pair, sample $h \leftarrow_R R$ and store the pair (m, h) in L. Once h is set, reply to the activating machine with (sid, h).

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 $\mathcal{F}^{\mathsf{w}}_{\mathsf{OT}}$ as depicted in Fig. 5. In the weaker variant $\mathcal{F}^{\mathsf{w}}_{\mathsf{OT}}$, S does not obtain any notification from the functionality. Instead its role is limited to sending (sid, a_0, a_1) to $\mathcal{F}^{\mathsf{w}}_{\mathsf{OT}}$, after which S halts. However, $\mathcal{F}^{\mathsf{w}}_{\mathsf{OT}}$ 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 $\mathcal{F}^{\mathsf{w}}_{\mathsf{OT}}$, 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 protocols of [CO15, HL17], implement both $\mathcal{F}_{\mathsf{OT}}$ and $\mathcal{F}^{\mathsf{w}}_{\mathsf{OT}}$ functionalities and hence they are composable. In what follows, we first show that the protocols of [CO15, HL17] do not realise $\mathcal{F}_{\mathsf{OT}}$ functionality, but realise only $\mathcal{F}^{\mathsf{w}}_{\mathsf{OT}}$. Next, we demonstrate the compositional issue of using $\mathcal{F}^{\mathsf{w}}_{\mathsf{OT}}$ 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 as a result, the $\mathcal{F}_{\mathsf{OT}}$ functionality keeps running. This causes a honest S in the ideal

Fig. 5: The ideal functionality $\mathcal{F}_{\mathsf{OT}}^{\mathsf{w}}$ for weaker OT

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

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.

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 (evaluation to be specific) 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 Samplable Dual-Mode Encryption

The seminal paper of [PVW08] introduced the primitive of *dual mode encryption* (DME). It works like a regular public key encryption scheme, alongside a notion of encryption branches. The key generation algorithm, takes a branch $\sigma \in \{0,1\}$ as an input, alongside the crs, to generate the public and secret key pair $(\mathsf{pk}, \mathsf{sk})$. Messages encrypted using pk, on branch σ, can be decrypted using sk, whereas the messages encrypted on the other branch remains hidden under certain conditions, described next. The encryption scheme can be further initialized in either of the two modes - *messy* or *decryption* mode, based on the setup phase. The setup phase is invoked with the mode and it returns (crs, t) to the invoking party, where t is the trapdoor for the crs. If the mode is initialized to messy, then the message encrypted on branch $1 - \sigma$ remains statistically hidden. Also, it is possible to extract the branch value $1 - \sigma$ (and σ can be computed) given pk and t. Whereas, if the scheme is set to decryption mode, then it is possible to generate secret keys sk_0 and sk_1 which decrypts ciphertexts on branches 0 and 1 respectively. Formally, a dual mode encryption scheme is defined as follows:

- $(\mathsf{crs}, t) \leftarrow \mathsf{SetupMessy}(1^\kappa)$: It is a randomized algorithm that takes as input the security parameter κ and outputs the crs and the trapdoor information t, for the messy mode. It enables the invocation of $\mathsf{FindMessy}$ algorithm.

  $(\mathsf{crs}, t) \leftarrow \mathsf{SetupDec}(1^\kappa)$: It is a randomized algorithm that takes as input the security parameter κ and outputs the crs and the trapdoor information t, for the decryption mode. It enables the invocation of $\mathsf{DecKeyGen}$ algorithm.
- $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(\mathsf{crs}, \sigma)$: It is a randomized algorithm that takes as input the crs and the branch σ and returns the public/encryption and secret/decryption key pair $(\mathsf{pk}, \mathsf{sk})$ for branch σ.
- $(y, r) \leftarrow \mathsf{Enc}(\mathsf{pk}, \sigma, m)$: It is a randomized algorithms that takes as input pk, the branch σ and the message $m \in \{0,1\}^\kappa$. It returns the ciphertext y, encrypted on branch σ and the randomness r, used for the encryption.
- $m \leftarrow \mathsf{Dec}(\mathsf{sk}, y)$: It is a deterministic algorithm that takes in input sk and y and it returns m only if sk and y correspond to the same branch.
- $b \leftarrow \mathsf{FindMessy}(\mathsf{pk}, t)$: It returns the branch value $b \in \{0,1\}$ given pk and the trapdoor t, when the mode is set to messy. The message encrypted on this branch is statistically hidden.
- $(\mathsf{pk}, \mathsf{sk}_0, \mathsf{sk}_1) \leftarrow \mathsf{DecKeyGen}(t)$: It is a randomized algorithm that generates a key pair $(\mathsf{pk}, \mathsf{sk}_0, \mathsf{sk}_1)$ when it is invoked with the trapdoor t as input and the mode is set to decryption. The secret key sk_b, $b \in \{0,1\}$, enables decryption of ciphertexts on branch b.

The above defined scheme satisfies correctness and four security properties, as mentioned in the [PVW08] paper. In addition, the dual mode encryption scheme must satisfy another property for our OT protocol. It must be possible to generate a crs for the messy mode from a random oracle query with overwhelming probability. We outline the correctness and security properties as follows:

- **Correctness**: For every mode, for all $(\mathsf{crs}, t) \leftarrow \mathsf{SetupMessy}(1^\kappa)$ / $\mathsf{SetupDec}(1^\kappa)$, for each $\sigma \in \{0,1\}$, for each $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(\mathsf{crs}, \sigma)$, and for all $m \in \{0,1\}^\kappa$, the following holds $\mathsf{Dec}(\mathsf{sk}, \mathsf{Enc}(\mathsf{pk}, \sigma, m)) = m$.
- **Property 1** (Indistinguishability of modes): The crs generated from either modes are indistinguishable, i.e. for all $(\mathsf{crs}, t) \leftarrow \mathsf{SetupMessy}(1^\kappa)$ and for all $(\mathsf{crs}', t') \leftarrow \mathsf{SetupDec}(1^\kappa)$, the following holds $\mathsf{crs} \overset{c}{\approx} \mathsf{crs}'$.
- **Property 2** (Indistinguishability of Branches in Messy mode): In messy mode, the public key does not leak the branch value to a computational adversary i.e., for all crs generated by $\mathsf{SetupMessy}$, the following condition holds true- $\mathsf{KeyGen}(\mathsf{crs}, 0) \overset{c}{\approx} \mathsf{KeyGen}(\mathsf{crs}, 1)$.
- **Property 3** (Messy Branch Identification): For each crs generated by $\mathsf{SetupMessy}$ and for each pk returned by $\mathsf{KeyGen}(\mathsf{crs}, \sigma)$, $\mathsf{FindMessy}(\mathsf{pk}, t)$ returns the messy branch b, s.t. $b = 1-\sigma$. Moreover, any message encrypted on branch b is statistically hidden, i.e. for every $m_0, m_1 \in \{0,1\}^\kappa$, $\mathsf{Enc}(\mathsf{pk}, b, m_0) \overset{s}{\approx} \mathsf{Enc}(\mathsf{pk}, b, m_1)$.
- **Property 4** (Dual Decryptable Branches in decryption mode): For each $(\mathsf{crs}, t) \leftarrow \mathsf{SetupDec}(1^\kappa)$ and for each $(\mathsf{pk}, \mathsf{sk}_0, \mathsf{sk}_1) \leftarrow \mathsf{DecKeyGen}(t)$, the following three conditions hold for all $m, m_0, m_1 \in \{0,1\}^\kappa$:

i. $(\mathsf{pk}, \mathsf{sk}_0) \overset{s}{\approx} \mathsf{KeyGen}(\mathsf{crs}, 0)$, $(\mathsf{pk}, \mathsf{sk}_1) \overset{s}{\approx} \mathsf{KeyGen}(\mathsf{crs}, 1)$ and $(\mathsf{pk}, \mathsf{sk}_0) \overset{s}{\approx} (\mathsf{pk}, \mathsf{sk}_1)$.
ii. $\mathsf{Dec}(\mathsf{sk}_b, \mathsf{Enc}(\mathsf{pk}, b, m)) = m$ for all $b \in \{0, 1\}$.
iii. $\mathsf{Enc}(\mathsf{pk}, b, m_0) \overset{c}{\approx} \mathsf{Enc}(\mathsf{pk}, b, m_1)$, for all $b \in \{0, 1\}$.

- **Property 5** (Samplable crs in Messy mode): for all $c \in \{0, 1\}^\kappa$, $(\mathsf{crs}, t) \leftarrow \mathsf{SetupMessy}(1^\kappa)$, the random oracle query $\mathcal{F}_{\mathsf{RO}}(c)$ is identically distributed to the crs of messy mode except with negligible probability i.e., $\mathcal{F}_{\mathsf{RO}}(c) \overset{s}{\approx} \mathsf{crs}$.

The above DME scheme can be instantiated under the DDH and the LWE assumption, which has been provided in Appendices C and D respectively. The paper of [PVW08] provides proof for correctness and Properties 1-4 of DME instantiations under the two respective assumptions. We further demonstrate, in Appendices C and D, that they satisfy Property 5 too, thereby yielding a samplable DME.

5 Framework for Adaptive Oblivious Transfer

In this section, we present our round-optimal framework for adaptive OT given a DME scheme, random oracles $\mathcal{F}_{\mathsf{RO1}}$ and $\mathcal{F}_{\mathsf{RO2}}$. We first present a brief overview of our protocol and then we present our proof. R generates the crs by sampling a random string c and invoking the random oracle on c. R obtains a valid crs, in messy mode, except with negligible probability. He invokes the KeyGen with the crs and his choice bit σ to obtain a key pair $(\mathsf{pk}, \mathsf{sk})$ which would allow decryption on branch σ using sk. R sends c and pk to S. Property 2 ensures that pk does not leak about σ. S generates the crs using c and encrypts two random pads on both branches using pk. Finally, S encrypts his messages using RO queries on the pads. R can decrypt the pad corresponding to branch σ, due to correctness of the DME scheme, and thus obtain the message corresponding to choice bit σ. The framework has been presented as protocol π_{OT} in Fig. 6.

5.1 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 5.2.

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 the real world adversary 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 1 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

Fig. 6: Adaptively-Secure Oblivious Transfer Protocol

π_{OT}

- **Functionalities:** Random oracles $\mathcal{F}_{\mathsf{RO1}} : \{0,1\}^{2\kappa} \to \{0,1\}^*$ and $\mathcal{F}_{\mathsf{RO2}} : \{0,1\}^{2\kappa} \to \{0,1\}^{\ell}$ respectively.
- **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 $\mathsf{crs} \leftarrow \mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||c)$.
- R computes $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(\mathsf{crs}, \sigma)$.
- R sends (c, pk) to S.

Transfer:

- S generates $\mathsf{crs} \leftarrow \mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||c)$.
- S samples $r_0, r_1 \leftarrow_R \{0,1\}^{\kappa}$.
- S computes $s_b \leftarrow \mathsf{Enc}(\mathsf{pk}, b, r_b)$, for $b \in \{0,1\}$.
- S sets $y_0 = \mathcal{F}_{\mathsf{RO2}}(\mathsf{sid}||r_0) \oplus a_0$ and $y_1 = \mathcal{F}_{\mathsf{RO2}}(\mathsf{sid}||r_1) \oplus a_1$.
- S sends $\{(s_0, y_0), (s_1, y_1)\}$ to R.

Local Computation by R:

- Computes $r_\sigma = \mathsf{Dec}(\mathsf{sk}, s_\sigma)$ and outputs $a_\sigma = y_\sigma \oplus \mathcal{F}_{\mathsf{RO2}}(\mathsf{sid}||r_\sigma)$.

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 messy mode when *R is corrupted*, it is set to decryption mode when *S is corrupted*. Sim can perform this by invoking $\mathsf{SetupMessy}$ (or $\mathsf{SetupDec}$) to obtain (crs, t) and then program the random oracle to return the same crs. In the former case, $\mathsf{FindMessy}$ can be suitably invoked to extract σ. In the latter case, Sim can invoke $\mathsf{DecKeyGen}$ to obtain secret keys on both branches and unlock both messages using the secret keys. The simulation of our protocol for static security is similar to that of PVW.

We are ready to present the formal proof of Theorem. 1. 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 DME is a samplable dual mode encryption scheme then protocol π_{OT} securely realizes the $\mathcal{F}_{\mathsf{OT}}$ functionality against static active adversaries in $(\mathcal{F}_{\mathsf{RO1}}, \mathcal{F}_{\mathsf{RO2}})$-hybrid model.*

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 obtains $(\mathsf{crs}, t) \leftarrow \mathsf{SetupDec}$. Sim samples $c \leftarrow_R \{0,1\}^\kappa$ and programs $\mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||c)$ to return crs. Sim invokes $\mathsf{DecKeyGen}(t)$ to obtain $(\mathsf{pk}, \mathsf{sk}_0, \mathsf{sk}_1)$ and sends (c, pk) as the first OT message to S on behalf of R in the internal execution.

(ii) *Simulating S's message:* Sim sets (s_0, s_1) as per the protocol and (y_0, y_1) are set 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 r_0 and r_1 using secret keys sk_0 and sk_1 and obtains a_0 and a_1 from y_0 and y_1. Sim invokes $\mathcal{F}_{\mathsf{OT}}$ with (a_0, a_1) 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. Whenever $\mathsf{Adv}_{\mathsf{Int}}$ queries $\mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||c)$, Sim invokes $\mathsf{SetupMessy}$ to obtain (crs, t) and programs $\mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||c)$ to return crs. Sim stores the tuple in a list Q as $(\mathsf{sid}, c, \mathsf{crs}, t)$. If a query is repeated then Sim returns the crs corresponding to 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 the the choice bit, i.e. σ, of R^* by invoking $\mathsf{FindMessy}(\mathsf{pk}, t)$. Sim sends σ to $\mathcal{F}_{\mathsf{OT}}$ on behalf of R in the ideal world, obtains a_σ and constructs (s_σ, y_σ) honestly. On the other hand, $s_{\overline{\sigma}}$ is set honestly and $y_{\overline{\sigma}}$ is set randomly. 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 $\textsc{ideal}_{\mathcal{F},\mathsf{Sim},\mathcal{Z}}(1^\kappa, z)$ of $\mathcal{Z}$ is indistinguishable from the real world view $\textsc{real}_{\mathcal{F},\mathsf{Adv},\mathcal{Z}}(1^\kappa, z)$ of $\mathcal{Z}$. We denote $\textsc{real}_{\mathcal{F},\mathsf{Adv},\mathcal{Z}}(1^\kappa, z)$ as hybrid HYB_R. The ideal world view of $\mathcal{Z}$ varies based on the case of corruption. For case $\textsc{d}$ ($\textsc{d} \in [4]$), we denote the ideal world view as $\mathsf{HYB}_{I.d}$. We prove that HYB_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 HYB_R and the ideal world view i.e $\mathsf{HYB}_{I.1}$ is indistinguishable through a series of intermediate hybrids.

- $\mathbf{HYB}_1$ **:** We consider a hybrid HYB_1 which is same as HYB_R except that here, the crs is generated using $\mathsf{SetupDec}$. Indistinguishability follows from Property 1 of DME, i.e. indistinguishability of the two modes, and random sampling of c. A distinguisher for the hybrids can be used to break Property 1 or guess the exact value of c, both of which happen with negligible probability.
- $\mathbf{HYB}_2$ **:** We consider a hybrid HYB_2 which is same as HYB_1 except that Sim generates pk using $\mathsf{DecKeyGen}$. Indistinguishability follows statistically from Property 4i of DME, i.e. the branch remains statistically hidden when the DME is set to decryption mode.
- $\mathbf{HYB}_3$ **:** We consider a hybrid HYB_3 similar to HYB_2 except that in HYB_3 Sim constructs s_0 and s_1 using r_0 and r_1 whereas y_0 and y_1 are formed using different random pads, r'_0 and r'_1. Indistinguishability between the hybrids follows from Property 4iii of DME, i.e. indistinguishability of ciphertexts in decryption mode.
- $\mathbf{HYB}_{I.1}$ **:** We consider the ideal world hybrid $\mathsf{HYB}_{I.1}$ similar to HYB_3 except that in $\mathsf{HYB}_{I.1}$, Sim sets y_0 and y_1 randomly. $\mathsf{Adv}_{\mathsf{Int}}$ can distinguish between the hybrids only if the values r'_0 or r'_1 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 HYB_R and the ideal world view i.e $\mathsf{HYB}_{I.2}$ is indistinguishable through an intermediate hybrid.

- $\mathbf{HYB}_1$ **:** We consider hybrid HYB_1 which is same as HYB_1 in previous case. Indistinguishability between HYB_R and HYB_1 follows (similar to the previous case) from Property 1 of DME.
- $\mathbf{HYB}_{I.2}$ **:** We consider a hybrid $\mathsf{HYB}_{I.2}$ similar to HYB_1 except that in $\mathsf{HYB}_{I.2}$ Sim decrypts r_0 and r_1 and obtains a_0 and a_1 respectively using secret keys on both branches. Indistinguishability follows from Property 4ii of DME, which ensures that Sim can decrypt messages on both branches using the secret keys in the decryption mode.

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

- $\mathbf{HYB}_1$ **:** We consider a hybrid HYB_1 which is same as HYB_R except here the crs is generated using $\mathsf{SetupMessy}$. Indistinguishability follows from Property 5 and 1, i.e. the crs in the messy mode can be sampled using the random oracle and the crs in the messy mode is indistinguishable from the crs in decryption mode. 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.
- $\mathbf{HYB}_2$ **:** We consider a hybrid HYB_2 which is same as HYB_1 except that, here, Sim extracts the value of σ, by invoking $\mathsf{FindMessy}(\mathsf{crs}, t)$, and sends it to $\mathcal{F}_{\mathsf{OT}}$ and obtains a_σ. Indistinguishability follows from Property 3 of DME.

- HYB_3 **:** We consider a hybrid HYB_3 similar to HYB_2 except that in HYB_3, Sim constructs $s_{\overline{\sigma}}$ using $r_{\overline{\sigma}}$, whereas $y_{\overline{\sigma}}$ is formed using different random pad, $r'_{\overline{\sigma}}$. Indistinguishability between the hybrids follows from the fact that $r'_{\overline{\sigma}}$ remains statistically hidden, due to Property 3, in the messy mode.
- $\mathsf{HYB}_{I.3}$ **:** Finally we consider our ideal world hybrid $\mathsf{HYB}_{I.3}$ where $y_{\overline{\sigma}}$ is set randomly. Indistinguishability follows from the random oracle assumption since the distinguisher has to guess $r'_{\overline{\sigma}}$ to distinguish between the two hybrids and this happens with negligible probability.

Case 4. Both S^* and R^* are corrupted: In this case HYB_R and $\mathsf{HYB}_{I.4}$ 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.

5.2 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 A 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 in messy mode, when the receiver is corrupt in the first round, or in the decryption mode otherwise, to enable extraction of R's input or S's input respectively.

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, the mode is set to decryption, and Sim can extract secrets keys for both branches, corresponding to pk, by invoking $\mathsf{DecKeyGen}$. When R gets corrupted and Sim obtains σ as R's input, Sim can provide sk_σ as its secret key on branch σ. Equivocation is successful due to Property 4ii of DME. In the second case when S is corrupted after the second OT message is sent, the random values sent corresponding to (y_0, y_1) by Sim have to be made consistent with sender's actual input (a_0, a_1). For this, Sim exploits the programmability of $\mathcal{F}_{\mathsf{RO2}}$ to enforce that (y_0, y_1) decrypts to (a_0, a_1). We are ready to present the proof of Theorem 2.

Theorem 2. *If DME is a samplable dual mode encryption scheme then protocol π_{OT} securely realizes $\mathcal{F}_{\mathsf{OT}}$ functionality against adaptive active adversaries in $(\mathcal{F}_{\mathsf{RO1}}, \mathcal{F}_{\mathsf{RO2}})$-hybrid model.*

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: Whenever $\mathsf{Adv_{Int}}$ queries $\mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||c)$, Sim invokes the $\mathsf{SetupMessy}$ algorithm to obtain (crs, t) and programs $\mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||c)$ to return crs. Sim stores the tuple in a list Q as $(\mathsf{sid}, c, \mathsf{crs}, t)$. If a query is repeated then Sim returns the crs corresponding to 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. The crs is set in the decryption mode by programming $\mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||c)$. Sim invokes $\mathsf{DecKeyGen}$ to obtain $(\mathsf{pk}, \mathsf{sk}_0, \mathsf{sk}_1)$ and he sends c and pk as the first OT message to $\mathsf{Adv_{Int}}$ 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 pk by setting sk_σ as the secret key on branch σ. Additionally, Sim equivocates y_σ s.t. a_σ can be obtained from it. For this, Sim programs $\mathcal{F}_{\mathsf{RO2}}(\mathsf{sid}||r_\sigma) = a_\sigma \oplus y_\sigma$. At the end of the protocol, $\mathsf{Adv_{Int}}$ 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. y_0 and y_1 open to a_0 and a_1. We note that y_σ was previously equivocated. Equivocation of $y_{\overline{\sigma}}$ is performed by programming $\mathcal{F}_{\mathsf{RO2}}(\mathsf{sid}||r_{\overline{\sigma}}) = a_{\overline{\sigma}} \oplus y_{\overline{\sigma}}$. 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, $\mathsf{Adv_{Int}}$ 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 $\mathcal{Z}$. 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. $\mathsf{Adv_{Int}}$ 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: Whenever $\mathsf{Adv_{Int}}$ queries $\mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||c)$, Sim invokes the $\mathsf{SetupMessy}$ algorithm to obtain crs and programs $\mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||c)$ to return

crs. Sim stores the tuple in a list Q as $(\mathsf{sid}, c, \mathsf{crs}, t)$. If a query is repeated then Sim returns the the crs corresponding to entry in Q indexed by the sid and c values. $\mathsf{Adv}_{\mathsf{Int}}$ generates the first OT message which is sent to S in the internal execution. This is similar to Case 3(i) (R's message for (S, R^*) case) in static proof.

- **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 $y_{\overline{\sigma}}$ such that it opens to $a_{\overline{\sigma}}$. Sim performs this by programming $\mathcal{F}_{\mathsf{RO2}}(\mathsf{sid}||r_{\overline{\sigma}}) = y_{\overline{\sigma}} \oplus a_{\overline{\sigma}}$. The internal state of S^* is revealed to $\mathcal{Z}$ who halts with an output.
- **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 HYB_R and show that in each simulation case HYB_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 provide sk_σ as the secret key for branch σ. Equivocation is successful due to Property 4ii of DME, i.e. σ is statistically hidden in pk when the crs is generated in decryption mode. In case of post execution corruption, Sim successfully equivocates $y_{\overline{\sigma}}$, s.t. it unlocks $a_{\overline{\sigma}}$ by programming $\mathcal{F}_{\mathsf{RO2}}(\mathsf{sid}||r_{\overline{\sigma}})$. Adv can query $r_{\overline{\sigma}}$ to $\mathcal{F}_{\mathsf{RO2}}$ only with negligible probability and hence equivocation is successful. This follows from Property 4iii of DME, i.e. indistinguishability of ciphertexts in the decryption mode.

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 $\mathsf{HYB}_{I.2}$ from HYB_R in the static proof. In case of post execution corruption, equivocation of R's view is possible since the crs is in decryption mode and Sim has secret keys for both branches. sk_σ can be provided as the secret key for branch σ. Indistinguishability follows from Property 4ii of DME.

Case 3. R^* is corrupted in the first round, S is honest in the second round: Indistinguishability follows from the indistinguishability of $\mathsf{HYB}_{I.3}$ from HYB_R in the static proof. In case of post execution corruption equivocation can fail if during the simulation $\mathsf{Adv}_{\mathsf{Int}}$ unlocks $r_{\overline{\sigma}}$ from $s_{\overline{\sigma}}$ and queries it to $\mathcal{F}_{\mathsf{RO2}}$. However, this occurs with negligible probability due to Property 3 of DME, i.e. ciphertext indistinguishability in messy mode, and the random oracle assumption.

Case 4. R^* is corrupted in the first round, S^* is corrupted in the second round: This is identical to case 4 (both $(\mathsf{S}^*, \mathsf{R}^*)$ are corrupted) in static proof. Hence, indistinguishability of simulation for this case follows from indistinguishability of $\mathsf{HYB}_{I.4}$ from HYB_R in the static proof.

5.3 Concrete Instantiations and Optimized Protocol based on DDH

We provide concrete instantiations of our protocol by instantiating the DME scheme under the DDH and LWE assumptions in Appendix C and D respectively. Furthermore, we can optimize our protocol, while considering the DDH-based instantiation of the DME. In Figure. 7 we present our optimized adaptive OT protocol $\pi_{\mathsf{OT}}^{\mathrm{DDH}}$ based on the DDH assumption. It differs from the original framework since protocol $\pi_{\mathsf{OT}}^{\mathrm{DDH}}$ does not send separate messages (s_0, s_1) for encrypting the random pads. Rather, the random pads and the messages are encrypted in the same ciphertext (w_0, w_1). This protocol incurs a computation cost of 11 exponentiations and 5 random oracle queries and it has an optimal round complexity of 2. It requires sending a κ bit string - c, two ℓ bit strings - (w_0, w_1); and 4 group elements– (u_0, u_1) and (g, h). Interestingly, it is the first round-optimal adaptively-secure OT protocol and it has an overhead of 5 random oracle queries and κ-bit string communication overhead over the static protocol of [PVW08]. The security of our protocol follows from the security of the OT framework and the DME instantiation presented in Appendix C. Hence we avoid presenting it and summarize it in Theorem. 3.

Theorem 3. *If the Decisional Diffie Hellman problem is hard in group $\mathbb{G}$ then protocol π_{OT}^{DDH} securely realizes the $\mathcal{F}_{OT}$ functionality against adaptive active adversaries in $(\mathcal{F}_{RO1}, \mathcal{F}_{RO2})$- hybrid model.*

5.4 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 [PVW08] framework, except here we generate the crs

Fig. 7: Optimized Adaptively-Secure Oblivious Transfer Protocol from DDH

$\pi_{\mathsf{OT}}^{\mathrm{DDH}}$

- **Functionalities:** Random oracles $\mathcal{F}_{\mathsf{RO1}} : \{0,1\}^{2\kappa} \to 4\mathbb{G}$ and $\mathcal{F}_{\mathsf{RO2}} : \{0,1\}^{\kappa} \times \mathbb{G} \to \{0,1\}^{\ell}$.
- **Private Inputs:** S has ℓ-bit input messages (a_0, a_1) and R has an input bit σ.

Choose:

- R samples $c \leftarrow_R \{0,1\}^{\kappa}$.
- R generates $\mathsf{crs} = (g_0, g_1, h_0, h_1)$, s.t. $(g_0, g_1, h_0, h_1) \leftarrow \mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||c)$.
- 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 $\mathsf{crs} = (g_0, g_1, h_0, h_1)$, s.t. $(g_0, g_1, h_0, h_1) \leftarrow \mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||c)$.
- 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 $w_0 = \mathcal{F}_{\mathsf{RO2}}(\mathsf{sid}||g^{r_0} h^{s_0}) \oplus a_0$ and $w_1 = \mathcal{F}_{\mathsf{RO2}}(\mathsf{sid}||g^{r_1} h^{s_1}) \oplus a_1$.
- S sends $\{(u_0, w_0), (u_1, w_1)\}$ to R.

Local Computation by R:

- Computes a_σ as $a_\sigma = w_\sigma \oplus \mathcal{F}_{\mathsf{RO2}}(\mathsf{sid}||u_\sigma^\alpha)$.

using a PRO. We can obtain π'_{OT} from π_{OT} by removing the random oracle invocation $\mathcal{F}_{\mathsf{RO2}}$. $\mathcal{F}_{\mathsf{RO2}}$ is not required since static corruption does not demand equivocation, of sender's message, by the Sim. The security of our protocol is summarized in Theorem 4 and the proof is similar to the static security proof of π_{OT}.

Theorem 4. *If DME is a samplable dual mode encryption scheme then protocol π_{OT} securely realizes the $\mathcal{F}_{\mathsf{OT}}$functionality against static active adversaries in $\mathcal{F}_{\mathsf{RO1}}$- hybrid model.*

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. In this case the crs is set to messy mode. For the second case, the simulator sets the crs in the decryption mode which enables him to equivocate R's view on obtaining σ, after R gets corrupted.

6 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 $p_0^i, p_1^i \leftarrow \{0,1\}^{\kappa}$ 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 :

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

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. 8 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 $\mathcal{F}_{\mathsf{RO}}$ and modify the formation of w_j as follows, where j_i denotes the ith bit of j.

$$w_j = a_j \oplus \mathcal{F}_{\mathsf{RO}}(\mathsf{sid}||j, p_{j_1}^1||p_{j_2}^2|| \ldots ||p_{j_{\log \mathsf{N}}}^{\log \mathsf{N}}) = a_j \oplus \mathcal{F}_{\mathsf{RO}}(\mathsf{sid}||j, v_j),$$

where $v_j = (p_{j_1}^1||p_{j_2}^2|| \ldots ||p_{j_{\log \mathsf{N}}}^{\log \mathsf{N}})$. This reduces the $\mathsf{N} \log \mathsf{N}$ PRF evaluations to N random oracle queries. Later, in this section we further demonstrate that the underlying $\mathcal{F}_{\mathsf{OT}}$ need not be full adaptively secure. Static OTs with security against adaptive receivers, i.e. receiver equivocal OTs, suffice for the transformation. Hence, we obtain adaptively-secure 1-out-of-N from $\log \mathsf{N}$ 1-out-of-2 receiver equivocal OTs.

6.1 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 $\mathcal{F}_{\mathsf{RO}}(j, v_j)$ 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 $\mathcal{F}_{\textsf{OT}}$ messages can be trivially equivocated by invoking the adaptive simulator for $\mathcal{F}_{\textsf{OT}}$. The simulated ciphertexts can be equivocated by programming $\mathcal{F}_{\textsf{RO}}(\textsf{sid}||j, v_j) = w_j \oplus a_j$, for $j \in [\textsf{N}]/\sigma$. Adversary can query $\mathcal{F}_{\textsf{RO}}(\textsf{sid}||j, v_j)$ with negligible probability since he lacks atleast one pad in v_j and hence simulator can program $\mathcal{F}_{\textsf{RO}}(\textsf{sid}||j, v_j)$ successfully to equivocate correctly. The proof of indistinguishability is similar to the one for statically corrupted $\textsf{R}^*$.

Fig. 8: Adaptively-Secure 1-out-of-N Oblivious Transfer Protocol

$\pi_{\textsf{N-OT}}$

- **Functionalities:** Random oracle $\mathcal{F}_{\textsf{RO}} : \{0,1\}^{\kappa+\log\textsf{N}(1+\kappa)} \rightarrow \{0,1\}^n$. $\mathcal{F}_{\textsf{OT}}$ denotes a 1-out-of-2 OT functionality.
- **Private Inputs:** $\textsf{S}$ has input messages $\{a_j\}_{j=1}^{\textsf{N}}$ and $\textsf{R}$ has an input choice string σ, where $|\sigma| = \log\textsf{N}$.

Choose:

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

Transfer:

- $\textsf{S}$ samples $2\log\textsf{N}$ random pads (p_0^i, p_1^i), where $p_0^i, p_1^i \in \{0,1\}^\kappa$.
- $\textsf{S}$ invokes the ith copy of $\mathcal{F}_{\textsf{OT}}$ with input $(\textsf{sen}, i, (p_0^i, p_1^i))$.
- $\textsf{S}$ encrypts his input messages as $w_j = a_j \oplus \mathcal{F}_{\textsf{RO}}(\textsf{sid}||j, p_{j_1}^1||p_{j2}^2||\ldots||p_{j_{\log\textsf{N}}}^{\log\textsf{N}})$, for $j \in [\textsf{N}]$.

Local Computation by R:

- Computes a_σ as $a_\sigma = w_\sigma \oplus \mathcal{F}_{\textsf{RO}}(\textsf{sid}||\sigma, p_{\sigma_1}^1||p_{\sigma_2}^2||\ldots||p_{\sigma_{\log\textsf{N}}}^{\log\textsf{N}})$.

6.2 Efficiency

Our protocol invokes $\mathcal{F}_{\textsf{OT}}$ functionality $\log\textsf{N}$ times and queries $\mathcal{F}_{\textsf{RO}}$ for $\textsf{N}$ times. It incurs communication of $\log\textsf{N}$ copies of $\mathcal{F}_{\textsf{OT}}$ and $\textsf{N}$ ciphertexts of size n bits.

6.3 Instantiating $\mathcal{F}_{\textsf{OT}}$ using Static Receiver Equivocal 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 $\textsf{S}^*$ or $\textsf{R}^*$ is corrupted, inputs are extracted by invoking the simulator for the actively-secure static OT protocol. For adaptive security, we need equivocation of $\textsf{S}$'s and $\textsf{R}$'s views if corruption occurs during the course of execution or at the end of protocol. In the transformation, $\textsf{S}$'s view consists of the OT messages and ciphertexts, i.e. $\{w_j\}_{j\in[\textsf{N}]}$. A simulator, playing the role of $\textsf{S}$ can trivially simulate the OT messages by running the honest $\textsf{S}$ algorithm with random pads, which are independent of $\textsf{S}$'s inputs. The ciphertexts are equivocated by programming $\mathcal{F}_{\textsf{RO}}(\textsf{sid}||j, v_j)$ (see Section 6.1). On the other hand, $\textsf{R}$'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 5.4) 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. The security of $\pi_{\mathsf{N\text{-}OT}}$ is summarized in Theorem 5.

Theorem 5. *If $\mathcal{F}_{\mathsf{RO}}$ is a programmable random oracle and $\mathcal{F}_{\mathsf{OT}}$ is a static actively secure OT functionality satisfying receiver equivocation, 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.*

7 Adaptively Secure Oblivious Transfer Extension

In this section we present our semi-honest and actively secure OT Extension protocols which are secure against adaptive adversaries, without assuming erasures.

7.1 Adaptively Secure OT Extension against Semi-Honest Adversaries

We prove that the semi-honest OT Extension protocol of [ALSZ13] (denoted as ALSZ13 hereby) can be made adaptively secure using PRO. We initiate our proof by briefly recalling the protocol of ALSZ13. In ALSZ13, S has m pairs of messages - $\{\mathbf{x}_{j,0}, \mathbf{x}_{j,1}\}$ for $j \in [m]$, and R has m selection bits, denoted as vector $\mathbf{r} = (r_1, \cdots, r_m)$. R samples two random strings $(\mathbf{k}_i^0, \mathbf{k}_i^1)$ and S samples a random string $\mathbf{s}$, $i \in [\kappa]$. R and S performs the seed OT phase by invoking $\mathcal{F}_{\mathsf{OT}}$ on inputs $(\mathbf{k}_i^0, \mathbf{k}_i^1)$ and $\mathbf{s}_i$ as sender and receiver respectively. S obtains $\mathbf{k}_i^{\mathbf{s}_i}$ from $\mathcal{F}_{\mathsf{OT}}$. R computes matrices $\mathbf{B}$ and $\mathbf{E}$ s.t. $\mathbf{B}^i = G(\mathbf{k}_i^0)$ and $\mathbf{E}^i = \mathbf{r}$, where G is a PRG. R sends a correction matrix $\mathbf{D}$ to S, s.t. $\mathbf{D}^i = \mathbf{B}^i \oplus G(k_i^1) \oplus \mathbf{E}^i$. S forms the matrix $\mathbf{Q}$ using $G(\mathbf{k}_i^{s_i})$, $\mathbf{s}$ and $\mathbf{D}$. The jth row of $\mathbf{Q}$ is used to encrypt the jth message pair $\{\mathbf{x}_{j,0}, \mathbf{x}_{j,1}\}_{j\in[m]}$ using pads $H(j, \mathbf{Q}_j)$ and $H(j, \mathbf{Q}_j \oplus \mathbf{s})$ respectively, where H is a correlation robust function [IKNP03]. S sends the ciphertexts to R, who decrypts and obtains $\mathbf{x}_{j,r_j}$ using $H(j, \mathbf{B}_j)$. The ALSZ13 is similar to the protocol presented in Fig. 9, except $\mathcal{F}_{\mathsf{RO1}}$ and $\mathcal{F}_{\mathsf{RO2}}$ will be replaced by G and H. Security against a statically corrupt S^* follows from the sender privacy of $\mathcal{F}_{\mathsf{OT}}$, which ensures that one of R's input (acting as sender in $\mathcal{F}_{\mathsf{OT}}$) to $\mathcal{F}_{\mathsf{OT}}$ remains hidden from S. On the other hand, a statically corrupt R can break honest sender privacy if he obtains both $\mathbf{x}_{j,0}$ and $\mathbf{x}_{j,1}$ for some j. However, this happens with negligible probability since R has to either guess the value of $\mathbf{s}$ or obtain a collision in the random oracle query, s.t. $H(j, \mathbf{B}_j) = H(j, \mathbf{B}_j \oplus \mathbf{s})$. We refer to [ALSZ13] for the full proof.

While considering adaptive security, the messages of S and R require equivocation. These messages can be classified into three exhaustive cases. Below, we give a high level overview for equivocating each case :

- $\mathcal{F}_{\mathsf{OT}}$ messages: It can be observed that S and R invoke $\mathcal{F}_{\mathsf{OT}}$ on random inputs - $\mathbf{s}_i$ and $(\mathbf{k}_i^0, \mathbf{k}_i^1)$ respectively, independent of their actual inputs. This can be leveraged to reduce the assumption on $\mathcal{F}_{\mathsf{OT}}$. More specifically, $\mathcal{F}_{\mathsf{OT}}$ can be secure only against static adversaries, since the simulator can simulate the view of an honest sender (resp. receiver) by running the honest sender (resp. receiver) algorithm on random inputs. When sender (resp. receiver) gets corrupted, the simulator can produce the

Fig. 9: Adaptive OT Extension Protocol secure against passive adversaries

Protocol $\pi^{\mathsf{p}}_{\mathsf{ALSZ}}$ for obtaining $\binom{2}{1}$-OT^{m}_{ℓ} from $\binom{2}{1}$-OT^{κ}_{m}

- **Input of S:** m pairs $\{\mathbf{x}_{j,0}, \mathbf{x}_{j,1}\}_{j\in[m]}$ of ℓ bit strings.
- **Input of R:** m selection bit vector $\mathbf{r} = (r_1, \cdots, r_m)$ such that each $r_j \in \{0,1\}$.
- **Functionalities:** Random Oracles $\mathcal{F}_{\mathsf{RO1}} : \{0,1\}^{2\kappa} \rightarrow \{0,1\}^{m}$ and $\mathcal{F}_{\mathsf{RO2}} : \{0,1\}^{\kappa} \times [m] \times \{0,1\}^{\kappa} \rightarrow \{0,1\}^{\ell}$ respectively. $\mathcal{F}_{\mathsf{OT}}$ denotes a 1-out-of-2 OT functionality.
- **Notations:** In the protocol $j \in [m]$ and $i \in [\kappa]$, unless specified otherwise.

Seed OT Phase:

1. S chooses $\mathbf{s} \leftarrow \{0,1\}^{\kappa}$ at random.
2. S computes the first message of π_{OT} as $\pi^{1}_{\mathsf{OT}}(\mathbf{s})$.
3. R chooses κ pairs of seeds $(\mathbf{k}^0_i, \mathbf{k}^1_i)$ each of length κ.
4. S and R invokes the $\mathcal{F}_{\mathsf{OT}}$ functionality for κ times with inputs $(\mathsf{rec}, \mathsf{sid}, \mathbf{s}_i)$ and $(\mathsf{rec}, \mathsf{sid}, (\mathbf{k}^0_i, \mathbf{k}^0_i))$ respectively. S receives $\{\mathbf{k}^{s_i}_i\}$ as output.

OT Extension Phase :

1. R forms three $m \times \kappa$ matrices $\mathbf{B}$, $\mathbf{E}$ and $\mathbf{C}$ in the following way and sends $\mathbf{D}$ to S:
   - Set $\mathbf{B}^i = \mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||\mathbf{k}^0_i)$.
   - Set $\mathbf{E}_j = (r_j||\ldots||r_j)$. Clearly, $\mathbf{E}^i = \mathbf{r}$.
   - Set $\mathbf{D}^i = \mathbf{B}^i \oplus \mathcal{F}_{\mathsf{RO1}}(\mathbf{k}^1_i) \oplus \mathbf{E}^i$.
2. On receiving $\mathbf{D}$, S forms $m \times \kappa$ matrix $\mathbf{Q}$ with the jth column of $\mathbf{Q}$ set as $\mathbf{Q}^i = \big(s_i \odot \mathbf{D}^i\big) \oplus \mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||\mathbf{k}^{s_i}_i)$. Clearly, (i) $\mathbf{Q}^i = \big(\mathbf{B}^i \oplus (s_i \odot \mathbf{E}^i)\big)$ and (ii) $\mathbf{Q}_j = \big(\mathbf{B}_j \oplus (\mathbf{s} \odot \mathbf{E}_j)\big) = \big(\mathbf{B}_j \oplus (\mathbf{s} \odot r_j)\big)$.
3. For every $j \in [m]$, S computes $\mathbf{y}_{j,0} = \mathbf{x}_{j,0} \oplus \mathcal{F}_{\mathsf{RO2}}(\mathsf{sid}||j, \mathbf{Q}_j)$ and $\mathbf{y}_{j,1} = \mathbf{x}_{j,1} \oplus \mathcal{F}_{\mathsf{RO2}}(\mathsf{sid}||j, \mathbf{Q}_j \oplus \mathbf{s})$. S sends $\{\mathbf{y}_{j,0}, \mathbf{y}_{j,1}\}_{j\in[m]}$ to R.
4. For every $j \in [m]$, R recovers $\mathbf{z}_j = \mathbf{y}_{j,r_j} \oplus \mathcal{F}_{\mathsf{RO2}}(\mathsf{sid}||j, \mathbf{B}_j)$. R outputs $\{\mathbf{z}_j\}_{j\in[m]}$.

view, generated using random inputs, which is indistinguishable from the honest sender's (resp. receiver's) view.

- R's messages : It consists of matrix $\mathbf{D}$ which depends on R's input $\mathbf{r}$. When R gets corrupted, the simulator has to equivocate R's message, i.e. matrix $\mathbf{D}$, based on R's input. This is ensured by replacing G with a PRO $\mathcal{F}_{\mathsf{RO1}}$.
- S's messages: It consists of messages - $(\mathbf{y}_{j,0}, \mathbf{y}_{j,1})$, which are dependent on S's inputs $(\mathbf{x}_{j,0}, \mathbf{x}_{j,1})$. When S gets corrupted, the simulator has to equivocate S's message based on S's inputs. This is ensured by replacing H with a PRO $\mathcal{F}_{\mathsf{RO2}}$.

The explanation for the last two subcases will be clear from the next few paragraphs, where we elaborately discuss the equivocation cases. We consider equivocation of S and R in two separate cases. Our passively secure adaptive OT extension protocol $\pi^{\mathsf{p}}_{\mathsf{ALSZ}}$ is presented in Fig. 9 and the security proof has been summarized in Theorem 6.

Theorem 6. *The protocol $\pi^{\mathsf{p}}_{\mathsf{ALSZ}}$ UC-securely realizes the Oblivious Transfer Extension functionality in the $(\mathcal{F}_{\mathsf{OT}}, \mathcal{F}_{\mathsf{RO1}}, \mathcal{F}_{\mathsf{RO2}})$ hybrid model against adaptive (without erasures) passive adversaries.*

- Equivocation of S's view: When S is honest, Sim plays the role of S in the internal world. The view of S consists of his input $\{\mathbf{x}_{j,0}, \mathbf{x}_{j,1}\}$, random vector $\mathbf{s}$, matrix

$\mathbf{Q}$ and the ciphertexts $(\mathbf{y}_{j,0}, \mathbf{y}_{j,1})$. The $\mathbf{s}$ vector and $\mathbf{Q}$ matrix is independent of S's inputs and can be simulated correctly without equivocation. The ciphertexts are simulated by relying on the programmability property of $\mathcal{F}_{\mathsf{RO2}}$. We first consider the case where both parties are honest. The formal simulation for S is presented as follows. Sim invokes the $\mathcal{F}_{\mathsf{OT}}$ simulator with random input $\mathbf{s}_i$ to simulate the ith seed-OT. Sim generates $\mathbf{Q}$ following honest sender algorithm. In the OT Extension phase, Sim sends random values for $\mathbf{y}_{j,b}$, where $b \in \{0,1\}$. In case of post execution corruption of S, Sim obtains the inputs of S. Sim provides $\mathbf{s}$ and $\mathbf{Q}$ as part of the view, and this is indistinguishable from the real world view. In addition, Sim opens $\mathbf{y}_{j,b}$ to $\mathbf{x}_{j,b}$ by programming $\mathcal{F}_{\mathsf{RO2}}$ as follows:

$$\mathcal{F}_{\mathsf{RO2}}(j, \mathbf{Q}_j) = \mathbf{y}_{j,0} \oplus \mathbf{x}_{j,0},$$

$$\mathcal{F}_{\mathsf{RO2}}(j, \mathbf{Q}_j \oplus \mathbf{s}) = \mathbf{y}_{j,1} \oplus \mathbf{x}_{j,1}.$$

Equivocation is is successful due to the randomness of $\mathbf{s}$ and random oracle assumption. An adversary $\mathsf{Adv}_{\mathsf{Int}}$ can prevent Sim from programming $\mathcal{F}_{\mathsf{RO2}}$ if he queries $\mathbf{Q}_j$ or $\mathbf{Q}_j \oplus \mathbf{s}$ to $\mathcal{F}_{\mathsf{RO2}}$. This happens with negligible probability as $\mathbf{Q}$ and $\mathbf{s}$ are randomly chosen. This proves that real and ideal world are indistinguishable. Now, we consider the case where $\mathsf{Adv}_{\mathsf{Int}}$ corrupts R^*. Sim obtains the choice vector $\mathbf{r}$. Sim invokes the OT Extension functionality with $\mathbf{r}$ and obtains $\{\mathbf{x}_{j,r_j}\}_{j\in[m]}$. Sim computes y_{j,r_j} honestly like an honest sender, s.t. y_{j,r_j} opens to $\mathbf{x}_{j,r_j}$. In case of post execution corruption, Sim obtains $(\mathbf{x}_{j,0}, \mathbf{x}_{j,1})$ and he programs $\mathcal{F}_{\mathsf{RO2}}(j, \mathbf{Q}_j \oplus (\mathbf{s} \odot \overline{r_j}))$, s.t. $y_{j,\overline{r_j}}$ opens to $\mathbf{x}_{j,\overline{r_j}}$. To prevent equivocation $\mathsf{Adv}_{\mathsf{Int}}$ has to guess the value of $\mathbf{s}$ and query $\mathbf{Q}_j \oplus (\mathbf{s} \odot \overline{r_j}) = \mathbf{B}_j \oplus \mathbf{s}$ to $\mathcal{F}_{\mathsf{RO2}}$ which happens with negligible probability, proving indistinguishability of the two worlds.

- Equivocation of R's view: When R is honest, Sim plays the role of R in the internal world. The view of R consists of his inputs, the seeds $(\mathbf{k}_i^0, \mathbf{k}_i^1)$ and the matrices, $\mathbf{B}$, $\mathbf{E}$ and $\mathbf{D}$. The seeds and $\mathbf{B}$ matrix can be simulated without any equivocation since they are independent of R's input. The other two matrices are simulated by programming $\mathcal{F}_{\mathsf{RO1}}$, since they depend on R's input. We first consider the case where both parties are uncorrupted. The formal simulation is presented as follows. Sim randomly generates $(\mathbf{k}_i^0, \mathbf{k}_i^1)$ and simulates ith $\mathcal{F}_{\mathsf{OT}}$ by invoking $\mathcal{F}_{\mathsf{OT}}$ with inputs $(\mathbf{k}_i^0, \mathbf{k}_i^1)$ like an honest R. Sim sets $\mathbf{D}$ matrix randomly and the $\mathbf{B}$ and $\mathbf{E}$ matrices are constructed later as they are not required for the simulation. Sim sends $\mathbf{D}$ to S and ends simulation as both parties are honest. The adversary $\mathsf{Adv}_{\mathsf{Int}}$ (without corrupting S) observes R's message, which consists only of $\mathbf{D}$ matrix. Based on that, he cannot distinguish between real and ideal world since in ideal world $\mathbf{D}$ is a random matrix, whereas in real world, it is padded with $\mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}, \mathbf{k}_i^0)$ and $\mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}, \mathbf{k}_i^1)$, where $\mathbf{k}_i^0$ and $\mathbf{k}_i^1$ remains hidden from $\mathsf{Adv}_{\mathsf{Int}}$. Thus, indistinguishability follows from the random oracle assumption. In case of post execution corruption of R, Sim obtains $\mathbf{r}$ and he programs $\mathcal{F}_{\mathsf{RO}}(\mathsf{sid}||\mathbf{k}_i^0)$ and $\mathcal{F}_{\mathsf{RO}}(\mathsf{sid}||\mathbf{k}_i^1)$ s.t. the following holds true:

$$\mathcal{F}_{\mathsf{RO1}}(\mathbf{k}_i^0) \oplus \mathcal{F}_{\mathsf{RO1}}(\mathbf{k}_i^1) = \mathbf{D}^i \oplus \mathbf{r}. \tag{1}$$

The $\mathbf{B}$ matrix is constructed after $\mathcal{F}_{\mathsf{RO1}}$ has been programmed according to Eq. 1. Equivocation is successful since $\mathsf{Adv}_{\mathsf{Int}}$ does not obtain $\mathbf{k}_i^{\overline{0}}$ and $\mathbf{k}_i^{\overline{1}}$ from $\mathcal{F}_{\mathsf{OT}}$ and he has to guess it, which occurs with negligible probability, in order to prevent equivocation. If $\mathsf{Adv}_{\mathsf{Int}}$ corrupts S^*, then Sim obtains $\mathbf{s}$ from $\mathcal{F}_{\mathsf{OT}}$ simulator (since S^* is the OT receiver) and he sets $\mathcal{F}_{\mathsf{RO1}}(\mathbf{k}_i^{\mathbf{s}_i})$ randomly concluding the simulation. In case of post execution corruption of R^*, Sim obtains $\mathbf{r}$ and constructs $\mathbf{E}$ matrix s.t. $\mathbf{E}^i = \mathbf{r}$. Sim equivocates $\mathbf{D}$ matrix by programming $\mathcal{F}_{\mathsf{RO1}}(\mathbf{k}_i^{\overline{\mathbf{s}_i}})$ as follows:

$$\mathcal{F}_{\mathsf{RO1}}(\mathbf{k}_i^{\overline{\mathbf{s}_i}}) = \mathcal{F}_{\mathsf{RO1}}(\mathbf{k}_i^{\mathbf{s}_i}) \oplus \mathbf{D}^i \oplus \mathbf{r}. \tag{2}$$

The $\mathbf{B}$ matrix is constructed after $\mathcal{F}_{\mathsf{RO1}}$ has been programmed according to Eq. 2. Equivocation is successful since $\mathsf{Adv}_{\mathsf{Int}}$ does not obtain $\mathbf{k}_i^{\overline{\mathbf{s}_i}}$ from $\mathcal{F}_{\mathsf{OT}}$ and he has to guess it, in order to prevent equivocation. However, it occurs with negligible probability and hence the two worlds are indistinguishable.

7.2 Adaptively Secure OT Extension against Active Adversaries

The ALSZ13 protocol is actively secure against S^* and passively secure against R^*, assuming seed OTs to be actively secure. An active R^* can send a bad $\mathbf{D}$ matrix such that $\mathbf{Q}$ is malformed and it leaks the $\mathbf{s}$ vector. We refer to [ALSZ15] (denoted as ALSZ henceforth) for the exact attack. To address this attack, the ALSZ15 protocol introduces column wise consistency check for every pair of columns. Moreover, for active security we require that the simulator Sim can extract a corrupted party's input in the protocol- $\mathbf{s}$ from S^* and $(\mathbf{k}_i^0, \mathbf{k}_i^1)$ from R^*, even when the parties are adaptively corrupted. For adaptive security, the simulator also has to simulate the consistency checks when Sim plays the role of R in the ideal world, in addition to the usual equivocation of views of honest parties.

In order to extract corrupted parties' inputs and simulate the consistency checks properly, we observe that the $\mathcal{F}_{\mathsf{OT}}$ messages are random in the OT extension protocol of ALSZ13 and ALSZ15. Hence, we can modify the protocol s.t. the parties invoke $\mathcal{F}_{\mathsf{ROT}}$ (Fig. 11) instead of $\mathcal{F}_{\mathsf{OT}}$. $\mathcal{F}_{\mathsf{ROT}}$ is an OT functionality where R has an input whereas S does not have any input. The functionality returns random messages to S and one of the random messages to R based on his input. Formally stating, $\mathcal{F}_{\mathsf{ROT}}$ takes choice bit σ as input from receiver of $\mathcal{F}_{\mathsf{ROT}}$, i.e. R_{OT}, and generates two random pads (a_0, a_1) for sender of $\mathcal{F}_{\mathsf{ROT}}$, i.e. S_{OT}. It sends a_σ to R_{OT} and (a_0, a_1) to S_{OT}. On a high level, when $\mathcal{F}_{\mathsf{ROT}}$ is invoked in our OT Extension protocol the input of S (playing as R_{OT}) to the $\mathcal{F}_{\mathsf{ROT}}$ can be extracted by invoking the static simulator for $\mathcal{F}_{\mathsf{ROT}}$. The seeds for R (playing the role of S_{OT}) are obtained from $\mathcal{F}_{\mathsf{ROT}}$. For equivocation of R's view, Sim runs the honest S_{OT} algorithm to obtain random seeds. The consistency checks are simulated by programming $\mathcal{F}_{\mathsf{RO1}}$ and $\mathcal{F}_{\mathsf{RO2}}$. For equivocation of S's view, Sim invokes the $\mathcal{F}_{\mathsf{ROT}}$ simulator for R_{OT} with a random $\mathbf{s}$ vector. Indistinguishability follows since the value of $\mathbf{s}$, sampled by honest S, is random in real world as well. The ciphertexts $(\mathbf{y}_{j,0}, \mathbf{y}_{j,1})$ are equivocated by programming $\mathcal{F}_{\mathsf{RO3}}$. Invoking $\mathcal{F}_{\mathsf{ROT}}$ has a significant advantage - $\mathcal{F}_{\mathsf{ROT}}$ can be adaptively implemented by a static receiver equivocal OT (Sec. 5.4), since S does not have any message, whereas implementing $\mathcal{F}_{\mathsf{OT}}$ in an adaptive way is expensive.

Now we formally describe our protocol $\pi^{\mathsf{a}}_{\mathsf{ALSZ}}$, which has been presented in Fig. 10. Our protocol is same as ALSZ15, except here the PRG, hash and correlation robust function invocations of ALSZ15 are replaced by PRO invocations. Our protocol has a seed OT phase similar to the $\pi^{\mathsf{p}}_{\mathsf{ALSZ}}$ protocol. The OT extension phase is divided into two parts, based on R's and S's role. R's role consist of phase I of OT extension, followed by the consistency checks and finally phase II of OT extension consists of S's role in the OT extension. Similar to ALSZ15, our protocol also contains column wise consistency check for every pair of columns. It has been recently identified by [ALSZ15] team that the checks leak κ^2 bits of randomness, on R's end. In order to counter that, they add κ^2 bits of randomness by adding κ columns and κ rows, as dummy choice bits τ. As a result the new choice bit string is $\mathbf{r}' = \mathbf{r}||\tau$. The checks ensure that the same $\mathbf{r}'$ has been used in all the columns of $\mathbf{D}$, i.e. $\mathbf{E}^i = \mathbf{r}'$ for all $i \in [k]$, where $k = 2\kappa$. Our static proof of security follows from the security proof of [ALSZ15] and we refer to their paper for the static proof against an active adversary. Adaptive security for our protocol can be proved against active adversaries in the PROM model, provided the underlying seed OTs are adaptive actively secure implementation of $\mathcal{F}_{\mathsf{ROT}}$ (Fig. 11) functionality. Security of $\pi^{\mathsf{a}}_{\mathsf{ALSZ}}$ has been summarized in Theorem 7.

Theorem 7. *If $\mathcal{F}_{\mathsf{RO1}}$, $\mathcal{F}_{\mathsf{RO2}}$ and $\mathcal{F}_{\mathsf{RO3}}$ are programmable random oracles then $\pi^{\mathsf{a}}_{\mathsf{ALSZ}}$ UC-securely realizes the Oblivious Transfer Extension functionality in the $\mathcal{F}_{\mathsf{ROT}}$ hybrid model against adaptive (without erasures) active adversaries.*

For proving adaptive security, the views of S and R require equivocation. Equivocation of S's view follows from the equivocation of S's view in ALSZ13 protocol. Equivocation of R's view is similar to that of ALSZ13 but in addition it requires simulating the consistency checks correctly.

- Equivocation of S's view: Sim simulates the seed-OT by invoking the $\mathcal{F}_{\mathsf{ROT}}$ functionality with a random $\mathbf{s}$ to obtain $\{\mathbf{k}_i^{\mathbf{s}_i}\}$, for $i \in [k]$. If both parties are honest then the simulation proceeds similar to the simulation for "Equivocation of S's view" (passive case) in Sec. 7.1. The consistency checks can be trivially simulated since S's role is limited to verification of the checks. If $\mathsf{Adv}_{\mathsf{Int}}$ corrupts R^* then Sim extracts the choice vector $\mathbf{r}'$. More specifically, Sim extracts R^*'s input, $(\mathbf{k}_i^0, \mathbf{k}_i^1)$, by invoking the $\mathcal{F}_{\mathsf{ROT}}$ simulator and computes $\mathbf{r}' = \mathbf{E}^i = \mathbf{r}||\tau$ from $\mathbf{D}^i$ using the seeds. Sim invokes the OT Extension functionality with $\mathbf{r}$ and obtains $\{\mathbf{x}_{j,r_j}\}_{j\in[m+\kappa]}$. Sim programs $\mathcal{F}_{\mathsf{RO3}}(\mathsf{sid}||j, \mathbf{Q}_j \oplus (\mathbf{s} \odot r_j))$ s.t. y_{j,r_j} opens to $\mathbf{x}_{j,r_j}$. In case of post execution corruption, Sim obtains $(\mathbf{x}_{j,0}, \mathbf{x}_{j,1})$ and he programs $\mathcal{F}_{\mathsf{RO3}}(\mathsf{sid}||j, \mathbf{Q}_j \oplus (\mathbf{s} \odot \overline{r_j}))$, s.t. $y_{j,\overline{r_j}}$ opens to $\mathbf{x}_{j,\overline{r_j}}$. To prevent equivocation $\mathsf{Adv}_{\mathsf{Int}}$ has to guess the value of $\mathbf{s}$ and query $\mathbf{Q}_j \oplus (\mathbf{s} \odot \overline{r_j}) = \mathbf{B}_j \oplus \mathbf{s}$ to $\mathcal{F}_{\mathsf{RO3}}$ which happens with negligible probability.
- Equivocation of R's view: The view of R consists of his inputs, the seeds $(\mathbf{k}_i^0, \mathbf{k}_i^1)$ and the matrices, $\mathbf{B}$, $\mathbf{E}$ and $\mathbf{D}$. When R is honest, Sim plays the role of R in the internal world where Sim obtains $(\mathbf{k}_i^0, \mathbf{k}_i^1)$ from $\mathcal{F}_{\mathsf{ROT}}$. If both parties are honest then simulation proceeds similar to the simulation for "Equivocation of R's view" (passive case) in Sec. 7.1. The consistency checks are simulated by setting $h^{a,b}_{\alpha,\beta}$ values randomly, for all $a, b \in \{0,1\}$ and all pairs $\alpha, \beta \subseteq [k^2]$. In case of post execution corruption, Sim obtains $\mathbf{r}$, and then based on the equivocated $\mathbf{B}$, $\mathbf{E}$ and $\mathbf{D}$ matrices, Sim can equivocate the $h^{a,b}_{\alpha,\beta}$ values correctly by programming $\mathcal{F}_{\mathsf{RO2}}$.

Fig. 10: Adaptive OT Extension Protocol secure against active adversaries

Protocol $\pi^{\mathsf{a}}_{\mathsf{ALSZ}}$ for obtaining $\binom{2}{1}$-OT^m_ℓ from $\binom{2}{1}$-OT^κ_m

- **Input of S:** m pairs $\{\mathbf{x}_{j,0}, \mathbf{x}_{j,1}\}_{j \in [m]}$ of ℓ bit strings.
- **Input of R:** m selection bit vector $\mathbf{r} = (r_1, \cdots, r_m)$ such that each $r_j \in \{0,1\}$.
- **Security Parameter:** $k = 2\kappa$.
- **Functionalities:** $\mathcal{F}_{\mathsf{RO1}} : \{0,1\}^{2\kappa} \to \{0,1\}^{m+\kappa}$, $\mathcal{F}_{\mathsf{RO2}} : \{0,1\}^{m+\kappa} \to \{0,1\}^{\kappa}$ and $\mathcal{F}_{\mathsf{RO3}} : \{0,1\}^{\kappa} \times [m] \times \{0,1\}^{k} \to \{0,1\}^{\ell}$ respectively. $\mathcal{F}_{\mathsf{OT}}$ denotes OT functionality.
- **Notations:** In the protocol $j \in [m+\kappa]$ and $i \in [k]$, unless specified otherwise.

Seed OT Phase:

1. S samples $\mathbf{s} \in \{0,1\}^k$ and invokes $\mathcal{F}_{\mathsf{ROT}}$ with message $(\mathsf{rec}, \mathsf{sid}, (\kappa, \mathbf{s}_i))$ for k times to obtain $\{\mathbf{k}_i^{\mathbf{s}_i}\}$ seeds.
2. R invokes $\mathcal{F}_{\mathsf{ROT}}$ with message $(\mathsf{sen}, \mathsf{sid}, (\mathsf{transfer}, \kappa))$ for k times to obtain $\{(\mathbf{k}_i^0, \mathbf{k}_i^1)\}$ seeds.

OT Extension Phase I:

1. R forms three $(m+\kappa) \times k$ matrices $\mathbf{B}$, $\mathbf{E}$ and $\mathbf{C}$ in the following way and sends $\mathbf{D}$ to S:
   - Sets $\mathbf{B}^i = \mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||\mathbf{k}_i^0)$.
   - Samples $\tau \leftarrow \{0,1\}^\kappa$ and sets $\mathbf{r}' = \mathbf{r}||\tau$.
   - Sets $\mathbf{E}_j = (r'_j|| \ldots ||r'_j)$. Clearly, $\mathbf{E}^i = \mathbf{r}'$.
   - Set $\mathbf{D}^i = \mathbf{B}^i \oplus \mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||\mathbf{k}_i^1) \oplus \mathbf{E}^i$.
2. On receiving $\mathbf{D}$, S forms $(m+\kappa) \times k$ matrix $\mathbf{Q}$ with the jth column of $\mathbf{Q}$ set as $\mathbf{Q}^i = \big(s_i \odot \mathbf{D}^i\big) \oplus \mathcal{F}_{\mathsf{RO1}}(\mathbf{k}_i^{s_i})$. Clearly, (i) $\mathbf{Q}^i = \big(\mathbf{B}^i \oplus (s_i \odot \mathbf{E}^i)\big)$ and (ii) $\mathbf{Q}_j = \big(\mathbf{B}_j \oplus (\mathbf{s} \odot \mathbf{E}_j)\big) = \big(\mathbf{B}_j \oplus (\mathbf{s} \odot r_j)\big)$.

Check Phase:

1. For every pair $\alpha, \beta \subseteq [k^2]$ and $a, b \in \{0,1\}$, R defines the following four values:

$$h^{a,b}_{\alpha,\beta} = \mathcal{F}_{\mathsf{RO2}}(\mathsf{sid}||\mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||\mathbf{k}^a_\alpha) \oplus \mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||\mathbf{k}^b_\beta))$$

   R sends each $h^{a,b}_{\alpha,\beta}$ to S.
2. For every pair $\alpha, \beta \subseteq [k^2]$, S knows $\mathbf{s}_\alpha$, $\mathbf{s}_\beta$, $\mathbf{k}_\alpha^{\mathbf{s}_\alpha}$, $\mathbf{k}_\beta^{\mathbf{s}_\beta}$, $\mathbf{D}^\alpha$ and $\mathbf{D}^\beta$. S checks that:
   (a) $h^{s_\alpha, s_\beta}_{\alpha,\beta} = \mathcal{F}_{\mathsf{RO2}}(\mathsf{sid}||\mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||\mathbf{k}_\alpha^{s_\alpha}) \oplus \mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||\mathbf{k}_\beta^{s_\beta}))$.
   (b) $h^{\overline{s_\alpha}, \overline{s_\beta}}_{\alpha,\beta} = \mathcal{F}_{\mathsf{RO2}}(\mathsf{sid}||\mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||\mathbf{k}_\alpha^{s_\alpha}) \oplus \mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||\mathbf{k}_\beta^{s_\beta}) \oplus \mathbf{D}^\alpha \oplus \mathbf{D}^\beta) = \mathcal{F}_{\mathsf{RO2}}(\mathsf{sid}||\mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||\mathbf{k}_\alpha^{\overline{s_\alpha}}) \oplus \mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||\mathbf{k}_\beta^{\overline{s_\beta}}) \oplus \mathbf{r}^\alpha \oplus \mathbf{r}^\beta)$, where $\mathbf{r}^\alpha$, $\mathbf{r}^\beta$ denotes $\mathbf{r}$ used in $\mathbf{D}^\alpha$, $\mathbf{D}^\beta$ respectively.
   (c) $\mathbf{D}^\alpha \neq \mathbf{D}^\beta$.

   In case one of these checks fails, S aborts and outputs $\perp$.

OT Extension Phase II:

1. For every $j \in [m]$, S computes $\mathbf{y}_{j,0} = \mathbf{x}_{j,0} \oplus \mathcal{F}_{\mathsf{RO3}}(\mathsf{sid}||j, \mathbf{Q}_j)$ and $\mathbf{y}_{j,1} = \mathbf{x}_{j,1} \oplus \mathcal{F}_{\mathsf{RO3}}(\mathsf{sid}||j, \mathbf{Q}_j \oplus \mathbf{s})$. S sends $\{\mathbf{y}_{j,0}, \mathbf{y}_{j,1}\}_{j \in [m]}$ to R.
2. For every $j \in [m]$, R recovers $\mathbf{z}_j = \mathbf{y}_{j,r_j} \oplus \mathcal{F}_{\mathsf{RO3}}(\mathsf{sid}||j, \mathbf{B}_j)$. R outputs $\{\mathbf{z}_j\}_{j \in [m]}$.

Fig. 11: The ideal functionality $\mathcal{F}_{\mathsf{ROT}}$ for Random Oblivious Transfer

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

Initiate: On input $(\mathsf{rec}, \mathsf{sid}, (\ell, \sigma))$ from R; if no message of the form $(\mathsf{sid}, (\ell, \sigma))$ is present in memory, store $(\mathsf{sid}, (\ell, \sigma))$. Send $(\mathsf{rec}, \mathsf{sid})$ to S.

Transfer: On input $(\mathsf{sen}, \mathsf{sid}, (\mathsf{transfer}, \ell))$ from S, if no message of the form $(\mathsf{sid}, (\ell, \sigma))$ is present in memory, then abort. Else sample $a_0, a_1 \leftarrow_R \{0,1\}^\ell$. Send $(\mathsf{sent}, \mathsf{sid}, a_\sigma)$ to R and $(\mathsf{sent}, \mathsf{sid}, (a_0, a_1))$ to S.

Corruption: If Adv corrupts S^* then receive (a_0, a_1) from Adv and continue execution as above using these values. If Adv corrupts R^* the usual execution continues.

$\mathsf{Adv}_{\mathsf{Int}}$ lacks $\mathbf{k}_i^0$ and $\mathbf{k}_i^1$, hence he cannot query the required argument to $\mathcal{F}_{\mathsf{RO2}}$ s.t. it would prevent equivocation of $h_{\alpha,\beta}^{a,b}$. Thus, the consistency checks can be correctly simulated. If $\mathsf{Adv}_{\mathsf{Int}}$ corrupts S^* then Sim extracts s vector from $\mathcal{F}_{\mathsf{ROT}}$ simulator. S computes $\mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||\mathbf{k}_i^{\mathbf{s}_i})$ and simulates the $\mathbf{B}$, $\mathbf{E}$ and $\mathbf{D}$ matrices according to the simulation strategy in Section 7.1. For simulating the consistency checks the $h_{\alpha,\beta}^{a,b}$ values are randomly set, for all $a, b \in \{0,1\}$. For every pair $\alpha, \beta \subseteq [k^2]$, Sim programs $\mathcal{F}_{\mathsf{RO2}}$ as follows :

$$\mathcal{F}_{\mathsf{RO2}}(\mathsf{sid}||\mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||\mathbf{k}_\alpha^{\mathbf{s}_\alpha}) \oplus \mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||\mathbf{k}_\beta^{\mathbf{s}_\beta})) = h_{\alpha,\beta}^{\mathbf{s}_\alpha,\mathbf{s}_\beta},$$
$$\mathcal{F}_{\mathsf{RO2}}(\mathsf{sid}||\mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||\mathbf{k}_\alpha^{s_\alpha}) \oplus \mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||\mathbf{k}_\beta^{s_\beta}) \oplus \mathbf{D}^\alpha \oplus \mathbf{D}^\beta) = h_{\alpha,\beta}^{\overline{\mathbf{s}_\alpha},\overline{\mathbf{s}_\beta}}. \qquad (3)$$

If $\mathcal{F}_{\mathsf{RO2}}$ is queried on other values by the adversary $\mathsf{Adv}_{\mathsf{Int}}$ then Sim answers it with a random value. When R^* gets corrupted, Sim programs $\mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||\mathbf{k}_i^{\overline{\mathbf{s}_i}})$ according to Eq. 1. Furthermore, he programs $\mathcal{F}_{\mathsf{RO2}}$ as follows:

$$\mathcal{F}_{\mathsf{RO2}}(\mathsf{sid}||\mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||\mathbf{k}_\alpha^{\mathbf{s}_\alpha}) \oplus \mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||\mathbf{k}_\beta^{\overline{\mathbf{s}_\beta}})) = h_{\alpha,\beta}^{\mathbf{s}_\alpha,\overline{\mathbf{s}_\beta}}$$
$$\mathcal{F}_{\mathsf{RO2}}(\mathsf{sid}||\mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||\mathbf{k}_\alpha^{\overline{\mathbf{s}_\alpha}}) \oplus \mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||\mathbf{k}_\beta^{\mathbf{s}_\beta})) = h_{\alpha,\beta}^{\overline{\mathbf{s}_\alpha},\mathbf{s}_\beta} \qquad (4)$$

$\mathsf{Adv}_{\mathsf{Int}}$ corrupting S^* does not obtain $\mathbf{k}_\alpha^{\overline{s_\alpha}}$ and $\mathbf{k}_\beta^{\overline{s_\beta}}$ except with negligible probability as we are in the $\mathcal{F}_{\mathsf{ROT}}$ hybrid model. It allows Sim to correctly program $\mathcal{F}_{\mathsf{RO2}}$ involving $\mathbf{k}_\alpha^{\overline{s_\alpha}}$ and $\mathbf{k}_\beta^{\overline{s_\beta}}$, according to Eq. 4, permitting equivocation of R's view and completing the simulation for the consistency checks.

Optimized OT Extension: Our 3 round OT extension protocol requires consistency check for each pair of columns, incurring an overhead of $\mathcal{O}(\kappa^2)$ checks. It can be optimized to $\mathcal{O}(\kappa)$ checks by following the 5 round protocol of [ALSZ15]. In the optimized protocol, the parties perform a 3 round coin-tossing subprotocol, after "OT Extension Phase I", where the first round of the subprotocol can be run in parallel with the last round of "OT Extension Phase I". The subprotocol returns a set of $\mathcal{O}(\kappa)$ pairs of column indices. R performs consistency check on those $\mathcal{O}(\kappa)$ pairs of columns and sends the $h_{\alpha,\beta}^{a,b}$ to S in parallel with the last round of the coin-tossing protocol. S verifies the outcome of the coin-tossing protocol and the consistency checks. After successful verification, S executes “OT Extension Phase II”. The work of [ALSZ15] proved static security for this protocol. Adaptive security follows immediately if we perform the consistency checks using PRO ($\mathcal{F}_{\mathsf{RO1}}$ and $\mathcal{F}_{\mathsf{RO2}}$), similar to the 3 round protocol. S will lack $\mathbf{k}_{\alpha}^{\overline{\mathbf{s}_{\alpha}}}$ and $\mathbf{k}_{\beta}^{\overline{\mathbf{s}_{\beta}}}$ and Sim can equivocate R's view according to Eq. 3. For empirical results, we implement the 5 round protocol and compare it with the state-of-the-art OT extension protocols.

7.3 Efficiency and Implications

Our protocols $\pi^{\mathsf{p}}_{\mathsf{ALSZ}}$ and $\pi^{\mathsf{a}}_{\mathsf{ALSZ}}$ requires one extra round, compared to the seed OTs, and it matches with the round complexity of the state-of-the-art static OT extension protocols [ALSZ13, KOS15, PSS17]. They also preserve the efficiency of the semi-honest and actively secure protocols of [ALSZ13] and [ALSZ15] respectively. In concrete terms, the amortized cost for generating m copies of both semi-honest and active adaptively secure 1-out-of-2 OTs is 3κ bits communication and 3 symmetric key operations per OT. The other costs - seed OTs, PRG expansion and consistency checks, are independent of m. We present a few implications resulting from our OT Extension protocols:

- Our semi-honest OT Extension protocol requires static semi-honest OT protocol for the seed-OTs in a non-blackbox fashion, where we explicitly modified the simulation strategy for the static semi-honest OT. Thus, static semi-honest OT implies adaptive semi-honest OT Extension under the PROM assumption in a non-blackbox manner. Our theorem is summarized in Thm. 8. [LZ11] states that adaptive semi-honest oblivious transfer cannot be constructed from static semi-honest oblivious transfer in a black-box manner and we demonstrate that its possible in a non-blackbox way.
- If the random OT is instantiated using our receiver equivocal static OT protocol then our actively secure OT Extension protocols requires 3 and 5 rounds, thereby efficiently generating large number of OTs at an amortized cost of 3κ bits communication and 3 symmetric key operations per OT. However, the state-of-the-art adaptively secure 1-out-of-2 OT protocols [BCG17] that have a round complexity of 3 rounds, assume erasures, incur 27 exponentiations and communication of 10 κ bits. Thus, we drastically reduce the cost of generating adaptive OTs.
- We can efficiently generate $m \log \mathsf{N}$ copies of 1-out-of-2 OTs using our OT Extension protocol (following the previous approach) and then apply our adaptive transformation (Sec. 6) to obtain m copies of 1-out-of-N OTs. The amortized cost for each 1-out-of-N OT is $\mathsf{N} + 3\log\mathsf{N} + 1$ symmetric key operations. Whereas the state-of-the-art adaptively secure 1-out-of-N OT protocols [BCG17, BDD$^+$17] require atleast $\mathcal{O}(\mathsf{N})$ exponentiations.

We have implemented our adaptively-secure variant of [ALSZ15] and presented the results in Sec. 9. For implementation we have considered the optimized variant where $\mathcal{O}(\kappa)$ are performed.

Theorem 8. *If there exists an Oblivious Transfer protocol that is secure in the presence of static semi-honest adversaries, then there exists an Oblivious Transfer Extension protocol from κ to $\mathsf{poly}(\kappa)$ that is secure in the presence of adaptive semi-honest adversaries, in a non-blackbox way, under the programmable random oracle assumption.*

8 Non-Interactive UC-Secure Commitment Scheme

In this section we present our adaptively-secure NICOM scheme Com, implemented by protocol π_{COM}. The protocol π_{COM} (described in Fig. 12) is universally composable and securely realizes the functionality $\mathcal{F}_{\mathsf{COM}}$ (described in Fig. 3) in the crs model under ORO assumption and Discrete Log (over a group $\mathbb{G}$) assumption. Later in this section, we demonstrate a protocol π_{CRS} to generate an instance of the Discrete Log problem, as the crs, in 4 rounds. Once the crs is generated, it can be used for subsequent instances of Com. Our crs generation algorithm is independent of the parties' inputs and hence it is adaptively-secure, rendering the whole protocol adaptively-secure under the ORO assumption. Our protocol is also secure in the Global Random Oracle model [CJS14] since we generate the local crs within the protocol, using π_{CRS}.

8.1 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, i.e. $\mathcal{F}_{\mathsf{RO1}} : \{0,1\}^{\mathsf{poly}(\kappa)} \to \mathbb{Z}_p$, on the message being committed, i.e. $\mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||m)$, and then commit the response of the ORO 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 ORO, i.e. $\mathcal{F}_{\mathsf{RO2}} : \{0,1\}^{\kappa} \times \mathbb{Z}_p \to \{0,1\}^{\kappa}$. S commits to r_1 by means of the query $\mathcal{F}_{\mathsf{RO2}}(\mathsf{sid}||r_1)$. 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 $\mathcal{F}_{\mathsf{RO2}}(\mathsf{sid}||r_1)$ 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.

Fig. 12: Non-Interactive UC-Secure Commitment Scheme COM

π_{COM}

- **Public Inputs:** The generator of group $\mathbb{G}$ is g. crs: (g, h) s.t. $g = \mathsf{g}$ and $h = g^x$.
- **Functionalities:** Random Oracles $\mathcal{F}_{\mathsf{RO1}} : \{0,1\}^{\mathsf{poly}(\kappa)} \to \mathbb{Z}_p$ and $\mathcal{F}_{\mathsf{RO2}} : \mathbb{Z}_p \to \{0,1\}^{\kappa}$ denote two random oracles.
- **Private Inputs:** S has input message m and R does not have any input.

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

- Computes $a = \mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||m)$.
- Samples $r_1, r_2 \leftarrow_R \mathbb{Z}_p$ and forms $\textsc{Com}(m; r_1, r_2) = (g^a h^{r_1} \bmod \mathrm{p}, \mathcal{F}_{\mathsf{RO2}}(\mathsf{sid}||r_1) \oplus r_2) = (c_1, c_2)$.
- Sends $\textsc{Com}(m; r_1, r_2)$ to R as commitment to m.

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

8.2 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 9.

Theorem 9. *If $\mathcal{F}_{\mathsf{RO1}}$ and $\mathcal{F}_{\mathsf{RO2}}$ 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 $\mathsf{crs} - (g; h)$. 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 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 $(\textsc{Receipt}, \mathsf{sid}, \mathsf{S}, \mathsf{R})$. In the decommit phase, Sim invokes $\mathcal{F}_{\mathsf{COM}}$ to obtain the message $(\textsc{Decommit}, \mathsf{sid}, m)$. On obtaining the committed message m, Sim provides randomness (r_1, r_2) s.t. COM decommits to m. The randomness (r_1, r_2) is computed by Sim as follows:

- Let $a = \mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||m)$ and $a' = \mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||m')$.
- 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 5 and 6:

$$\begin{aligned} & g^{a} h^{r_1} \bmod \mathrm{p} = g^{a'} h^{r'_1} \bmod \mathrm{p} \\ \Longrightarrow\ & g^{a+r_1 x} \bmod \mathrm{p} = g^{a'+r'_1 x} \bmod \mathrm{p} \\ \Longrightarrow\ & a + r_1 x = a' + r'_1 x \\ \Longrightarrow\ & r_1 = (a' - a + r'_1 x) x^{-1} \end{aligned} \tag{5}$$

$$\begin{aligned} & \mathcal{F}_{\mathsf{RO2}}(\mathsf{sid}||r_1) \oplus r_2 = \mathcal{F}_{\mathsf{RO2}}(\mathsf{sid}||r'_1) \oplus r'_2 \\ \Longrightarrow\ & r_2 = \mathcal{F}_{\mathsf{RO2}}(\mathsf{sid}||r'_1) \oplus r'_2 \oplus \mathcal{F}_{\mathsf{RO2}}(\mathsf{sid}||r_1) \end{aligned} \tag{6}$$

Sim provides (m, r_1, r_2), as opening to the commitment 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.

Indistinguishbaility : 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_R : Real world execution of the protocol.
- HYB_I : Same as HYB_R, 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). HYB_I corresponds to the ideal world view of $\mathcal{Z}$. It follow from Eq. 5 and 6 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 $\textsc{Com}(m)$ from $\mathsf{Adv}_{\mathsf{Int}}$ in the internal execution. He observes the random oracle queries (both $\mathcal{F}_{\mathsf{RO1}}$ and $\mathcal{F}_{\mathsf{RO2}}$) 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 $\mathcal{F}_{\mathsf{RO1}}$, q_j was queried to $\mathcal{F}_{\mathsf{RO2}}$ and the following holds:

$$g^{\mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||q_i)} h^{q_j} \bmod \mathrm{p} = c_1. \tag{7}$$

where $\textsc{Com}(m) = (c_1, c_2)$ 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 $(\textsc{Commit}, \mathsf{sid}, m')$ to $\mathcal{F}_{\mathsf{COM}}$.

– If there exists a unique valid (q_i, q_j) pair then Sim sends $(\textsc{Commit}, \mathsf{sid}, q_i)$ to $\mathcal{F}_{\mathsf{COM}}$.
– If there exists more than one valid pair then Sim samples $m' \leftarrow_R \mathbb{G}$ and sends $(\textsc{Commit}, \mathsf{sid}, m')$ 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. Whereas, Sim aborts in the ideal world as well as in $\mathcal{F}_{\mathsf{COM}}$ 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 $(\textsc{Decommit}, \mathsf{sid})$ 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.

Indistinguishability : 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_R : Real world execution of the protocol.
– HYB_I : It represents the ideal world execution of the protocol $\mathcal{Z}$ 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 $\textsc{Com}$ 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 shows that $\mathsf{Adv}_{\mathsf{Int}}$ obtained $\mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||m)$ and $\mathcal{F}_{\mathsf{RO2}}(\mathsf{sid}||r_1)$ 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 $\mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||m)$ (or $\mathcal{F}_{\mathsf{RO2}}(\mathsf{sid}||r_1)$) without querying m (or r_1) with negligible probability.
  - **Case 2:** This case demonstrates that either $\mathsf{Adv}_{\mathsf{Int}}$ obtained $\mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||m)$ and $\mathcal{F}_{\mathsf{RO2}}(\mathsf{sid}||r_1)$ 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. The adversary $\mathsf{Adv}_{\mathsf{DLP}}$ will obtain a challenge instance $(g, h) = (g, g^x)$ from the challenger $\mathsf{Chall}_{\mathsf{DLP}}$ of the DLP game. $\mathsf{Adv}_{\mathsf{DLP}}$ will invoke the $\mathsf{Adv}_{\mathsf{Int}}$ while simulating the role of an honest R in the commitment scheme with $\mathsf{crs} = (g, h)$. $\mathsf{Adv}_{\mathsf{DLP}}$ will observe the queries made by $\mathsf{Adv}_{\mathsf{Int}}$ to obtain a valid pair (q_i, q_j). He will receive the commitment $\textsc{Com}$

from $\mathsf{Adv_{Int}}$. Upon obtaining the decommitment, (m, r), to COM, $\mathsf{Adv_{DLP}}$ will find x as follows:

$$\begin{aligned} & g^{\mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||q_i)} h^{q_j} = g^{\mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||m)} h^r \bmod \mathrm{p} \\ \implies \quad & g^{\mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||q_i)+q_j x} = g^{\mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||m)+rx} \bmod \mathrm{p} \\ \implies \quad & \mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||q_i) + q_j x = \mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||m) + rx \\ \implies \; & x = (\mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||q_i) - \mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||m))(r - q_j)^{-1} \end{aligned} \tag{8}$$

However, we assume that the DLP problem is hard in group $\mathbb{G}$ and hence this case occurs with negligible probability.

- **Case 3:** This case indicates that $\mathsf{Adv_{Int}}$ obtains two or more valid pairs. This again implies that either the DLP problem has been solved by $\mathsf{Adv_{Int}}$ or $\mathsf{Adv_{Int}}$ found a collision in the random oracle queries. 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 sub-cases for analysis based on the equality of $\mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||q_i)$ and $\mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||q'_i)$ values.

  i. $q_i \neq q'_i, \mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||q_i) = \mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||q'_i), q_j = q'_j$: This indicates that $\mathsf{Adv_{Int}}$ 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.

  ii. $q_i \neq q'_i, \mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||q_i) \neq \mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||q'_i), q_j \neq q'_j$: In this case, $\mathsf{Adv_{Int}}$ can be used as blackbox by $\mathsf{Adv_{DLP}}$ to solve the DLP problem. $\mathsf{Adv_{DLP}}$ will invoke the $\mathsf{Adv_{Int}}$ while simulating the role of an honest R in the commitment scheme with $\mathsf{crs} = (g, h)$. $\mathsf{Adv_{DLP}}$ will observe the queries made by $\mathsf{Adv_{Int}}$ to obtain two valid pairs (q_i, q_j) and (q'_i, q'_j) s.t. $q_i \neq q'_i$ and $q_j \neq q'_j$. $\mathsf{Adv_{DLP}}$ solves the DLP problem by finding x as follows:

$$\begin{aligned} & g^{\mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||q_i)} h^{q_j} = g^{\mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||q'_i)} h^{q'_j} \bmod \mathrm{p} \\ \implies \quad & g^{\mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||q_i)+q_j x} = g^{\mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||q'_i)+q'_j x} \bmod \mathrm{p} \\ \implies \quad & \mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||q_i) + q_j x = \mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||q'_i) + q'_j x \\ \implies \; & x = (\mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||q_i) - \mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||q'_i))(q'_j - q_j)^{-1} \end{aligned} \tag{9}$$

  However, we assume that the DLP problem is hard in group $\mathbb{G}$ and hence this case occurs with negligible probability.

  The other two cases involve $\mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||q_i) = \mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||q'_i), q_j \neq q'_j$ and $\mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||q_i) \neq \mathcal{F}_{\mathsf{RO1}}(\mathsf{sid}||q'_i), q_j = q'_j$ 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 7) as $g^{\mathcal{F}_{\mathsf{RO1}}(q_i)} h^{q_j} \neq g^{\mathcal{F}_{\mathsf{RO1}}(q'_i)} h^{q'_j}$. There maybe atmost one valid pair, for which the analysis follows from Case 1 and 2.

8.3 Adaptive Security

Interestingly, our commitment scheme π_{COM} (Fig 12) also 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 10. *If $\mathcal{F}_{\mathsf{RO1}}$ and $\mathcal{F}_{\mathsf{RO2}}$ 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 private 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 9). $\mathcal{Z}$ cannot distinguish between the commitment to m' and commitment to the actual value due to the statistical hiding property of the scheme. Equivocation follows from the equivocal property as proven for the static case.

8.4 Generation of crs using Observable Random Oracle

For our protocol π_{COM}, the involved parties require a crs of the form (g, h), where $h = g^x$. Also, the simulator should have the knowledge of the trapdoor x in order to perform simulation. The crs can be trivially generated if we assume a PRO. The parties can generate the crs as $\mathcal{F}_{\mathsf{RO}}(\mathsf{sid}||\text{“}\mathsf{com}\text{”})$. 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. 13) relying solely on the observability property of the RO. Once the crs is generated it can be reused for subsequent commitments between the parties. By generating the crs as part of the protocol it can be concluded that our commitment scheme is adaptively secure in the Global Random Oracle model.

Intuition: Our π_{CRS} protocol proceeds in two phases - 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 the coin tossing is performed, they engage in ZKPoK in order to prove the knowledge of trapdoors to their respective shares, i.e. h_{S} and h_{R} respectively. The ZKPoK enables the simulator of π_{CRS} to extract the corrupted party's share in order to obtain the trapdoor

Fig. 13: Generating $\mathsf{crs} = (g, h)$ for π_{COM} using $\mathcal{F}_{\mathsf{RO}}$

π_{CRS}

- **Public Inputs:** The generator of group $\mathbb{G}$ is g.
- **Functionality:** Random oracle $\mathcal{F}_{\mathsf{RO}} : \{0,1\}^{\mathsf{poly}(\kappa)} \rightarrow \{0,1\}^{\kappa}$.
- **Private Inputs:** The parties do not have any input.

Coin Tossing:

- Round 1:
  - S samples $x_{\mathsf{S}} \leftarrow_R \mathbb{Z}_p$, computes $h_{\mathsf{S}} = g^{x_{\mathsf{S}}}$ mod p and sends $\mathcal{F}_{\mathsf{RO}}(\mathsf{sid}||h_{\mathsf{S}})$ to R.
  - R samples $x_{\mathsf{R}} \leftarrow_R \mathbb{Z}_p$, computes $h_{\mathsf{R}} = g^{x_{\mathsf{R}}}$ mod p and sends $\mathcal{F}_{\mathsf{RO}}(\mathsf{sid}||h_{\mathsf{R}})$ to S.
- Round 2:
  - S sends h_{S} to R.
  - R sends h_{R} to S.
- Computation:
  - S verifies $\mathcal{F}_{\mathsf{RO}}(\mathsf{sid}||h_{\mathsf{R}})$ and computes $h = h_{\mathsf{S}}.h_{\mathsf{R}}$ mod p, else aborts.
  - R verifies $\mathcal{F}_{\mathsf{RO}}(\mathsf{sid}||h_{\mathsf{S}})$ and computes $h = h_{\mathsf{S}}.h_{\mathsf{R}}$ mod p, 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 $c \leftarrow_R \{0,1\}^{\nu}$ and sends $\mathcal{F}_{\mathsf{RO}}(\mathsf{sid}||c)$ to S.
- Round 2:
  - S computes ν garbled circuits, by sampling $\mathsf{seed}_i \leftarrow_R \{0,1\}^{\kappa}$ and $(\mathsf{GC}_i, \mathsf{e}_i, \mathsf{d}_i) \leftarrow \mathsf{Gb}(\mathrm{PRG}(\mathsf{seed}_i), \mathcal{C})$ where $\mathcal{C}$ computes g^x, for $i \in [\nu]$.
  - S sends $\mathcal{F}_{\mathsf{RO}}(\mathsf{sid}||\mathsf{seed}_i)$, $\mathcal{F}_{\mathsf{RO}}(\mathsf{sid}||\mathsf{GC}_i)$ and d_i to R.
- Round 3:
  - R reveals c to S. S verifies $\mathcal{F}_{\mathsf{RO}}(\mathsf{sid}||c)$ 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 $y = \mathsf{De}(\mathsf{Ev}(\mathsf{GC}_i, \mathsf{X}))$, if $c_i = 1$, and aborts if $y \neq h_{\mathsf{S}}$.
  - R stores (g, h) as the crs.

to (g, h). Our coin tossing protocol requires 2 rounds and ZKPoK consumes 4 rounds. However, the first 2 rounds of the ZKPoK can be parallelized 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 and random oracle only. Note that we need a ZKPoK protocol which can be implemented using an ORO and without relying on any other setup assumption. This rules out the possibilities of using efficient ZKPoK protocol of [JKO13] since it uses an OT, which would further require other setup assumptions. However, the interactive ZKPoK variants of “MPC-in-the-head” protocols [GMO16, AHIV17] also suffices for our purpose.

Static Security: We prove that π_{CRS} generates crs in the presence of static active adversaries. Since our protocol is symmetric, we consider the case where S^* is corrupted and simulator Sim plays the role of honest R. Sim behaves like an honest R throughout the protocol and extracts the trapdoor x_{S} from the ZKPoK sent by S^*. Sim plays the role of an evaluator in the ZKPoK whereas S^* plays the role of a constructor. Sim extracts the seeds, for all circuits sent by S^*, by observing the queries made to $\mathcal{F}_{\mathsf{RO}}$ and matching them with $\mathcal{F}_{\mathsf{RO}}(\mathsf{sid}||\mathsf{seed}_i)$. Sim generates the encoding information of all evaluation circuits from the seeds. Finally, Sim extracts the input witness x_{S} of S^*, by matching the input wire labels, of the evaluation circuit, with the encoding information, of the evaluation circuit. On the other hand, Sim can simulate the ZKPoK, on behalf of constructor, by behaving like an honest R since he knows the trapdoor x_{R} sampled by him. A corrupt S^* cannot extract any information about x_{R} due to the privacy of the garbling scheme.

Adaptive Security: We prove that protocol π_{CRS} is adaptively-secure by observing that the parties do not have any private inputs at the outset of the protocol. The simulator can simulate by running honest sender/receiver algorithms as required. Upon corruption of a party, the simulator can reveal its random tape for the corresponding party. The simulated honest party's view will be indistinguishable from the real honest party's view since in both cases the protocol transcripts are honestly generated using a random tape. This proves that our protocol π_{CRS} UC-securely generates the crs required for our commitment scheme.

8.5 Final Commitment Scheme $\pi = \pi_{\mathsf{CRS}} + \pi_{\mathsf{COM}}$

We can combine π_{CRS} and π_{COM} to obtain an UC-secure protocol π which implements $\mathcal{F}_{\mathsf{COM}}$ functionality in the presence of adaptive adversaries solely relying on ORO. The security has been summarized in Theorem. 11. It can be proven secure in the GRO of [CJS14] since it does not require a local crs.

Theorem 11. *If* $\mathsf{Garble} = (\mathsf{Gb}, \mathsf{En}, \mathsf{Ev}, \mathsf{De}, \mathsf{Ve})$ *is a private, verifiable garbling scheme and solving the Discrete Log Problem is hard in multiplicative group $\mathbb{G}$, if π_{COM} implements $\mathcal{F}_{\mathsf{COM}}$ functionality against active adaptive adversaries in the crs, ORO-model and π_{CRS} generates the crs against active adaptive adversaries in the ORO model, then $\pi = \pi_{\mathsf{CRS}} + \pi_{\mathsf{COM}}$ implements $\mathcal{F}_{\mathsf{COM}}$ functionality against active adaptive adversaries (without erasures) in the ORO-model.*

8.6 Efficiency of π

We analyze the efficiency of π by analyzing the efficiency of π_{CRS} and π_{COM} separately. The π_{CRS} protocol requires 4 rounds and has a computation cost of 2 exponentiations, $8\nu + 8$ oracle queries, construction and evaluation of 2ν circuits. The communication cost is 2ν circuits + $(8 + 4\nu + 2\kappa)$ strings of κ bits. However, it is a one-time cost which would get amortized when multiple commitments are performed using the same crs. Next, we analyze the efficiency of π_{COM}. 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. It is non-interactive in both commitment and decommitment phase. This yields an efficient adaptively secure commitment scheme which is practically motivated 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.

9 Implementation Details

In this section, we support our theoretical claims with empirical details. First, we talk about the hardware configuration of our system and then we compare our results with other relevant protocols in the literature. We specifically compare against the state-of-the-art statically secure protocols to establish the fact that we achieve upgrade in security with minimal overhead.

Hardware Details: We have tested the code in a standard LAN network and a simulated WAN setting. Our machine has 8GB RAM and an Intel Core i5-4690 CPU with 3.5 GHz processor speed. For WAN simulation, we used the tc tool of Linux and introduced a round trip delay of 100 milliseconds into the network, with a bandwidth of 20Mbps.

Software Details: To establish OT results, first we compare the performance of our adaptively secure OT with the statically secure PVW OT, both under DDH assumption. Table 5 shows the results for LAN and WAN setting for a message size of 128 bits. Second, for OT Extension, we build upon the OT extension code made available by Encrypto group on github [OTC]. It contains the OT extension implementations of KK13, NNOB and ALSZ in C++, using the Miracl library for elliptic curve arithmetic. Additionally, we have received the code from the authors of [PSS17] and used it in our comparison for 1-out-of-N OT extension. The random oracle in our protocol has been implemented by the use of standard AES-128 and SHA-256 hash function. We require 190, 342, 256 and 190 seed OTs for ALSZ, NNOB, PSS and our protocol respectively to obtain the necessary security. Tables 6 and 7 give a detailed view of cost involved in attaining 1-out-of-2 OT Extension and 1-out-of-N (1-out-of-16) OT Extension respectively in both LAN and WAN setting over various values of m which is the number of OTs generated. For our adaptively secure commitment scheme, we use SHA-256 as a standard replacement to Observable Random Oracle and tabulate efficiency values over various message lengths as depicted in 4. For all implementation purposes, we set the computational security parameter to be $\kappa = 128$ and the statistical security parameter to be $\nu = 40$.

Comparison: Tables 5, 6 and 7 compares the performance of our adaptively secure protocols against the existing state-of-the-art static protocols. Our adaptively secure OT protocol has a mere overhead of 0.02% and 1.0% over the PVW protocol in the LAN and WAN setting respectively. This overhead is nominal compared to the security

Table 5: Table indicating implementation results for Oblivious Transfer protocol in the LAN setting for κ=128.

Scheme	Msg Size n (bits)	Setting	Runtime of S (s)	Runtime of R (s)	Comm. (bits) of S (s)	Comm. (bits) of R (s)
PVW	128	LAN	1.675	0.970	512	256
(Static, DDH based)		WAN	4.321	1.681	512	256
Our Scheme	128	LAN	1.676	0.971	512	384
(Adaptive, DDH-based)		WAN	4.329	1.690	512	384

achieved. We can in fact consider this to be an upgrade in security and ideally suited for real world applications, at the cost of static OT. From Fig. 14 we can see that our protocol has minimal overhead over the static 1-out-of-2 OT extension protocols. We incur an overhead of 2% and 11.95% over ALSZ, 8% and almost no overhead over NNOB in the LAN and WAN setting respectively in terms of computation. Communication wise, we incur 2.4% overhead over ALSZ and none over NNOB. For generating 1-out-of-16 OT, we incur an overhead of 0.6% and 0.6% over ALSZ, 2% and 7% NNOB, 16% and 24% over PSS in LAN and WAN runtime. In concrete terms, each extended 1-out-of-2 adaptive OT costs around .104-2 μs based on the number of OTs generated together. The implementation of KOS [KOS15] is not available in the public domain. ALSZ has 24% overhead over [IKNP03] and KOS has 5% over [IKNP03] in LAN setting. This translates to 18% overhead of ALSZ over KOS. Based on that we can compare accordingly with KOS, by comparing with ALSZ. Similarly each extended 1-out-of-16 adaptive OT costs around 21μs. Hence, our OT extension protocols generates practically efficient OTs which can find useful applications in real-life protocols that demand adaptive security. We compare our commitment scheme with the state-of-the-art adaptive protocol of [HM04]. Our scheme requires around 0.20 ms for committing to a string of length 2048 bits, whereas they require around 0.48ms. This implies our scheme is approx. 2 times faster compared to [HM04]. Tab. 4 compares our protocol with [HM04] for various message sizes and demonstrates our efficiency gain.

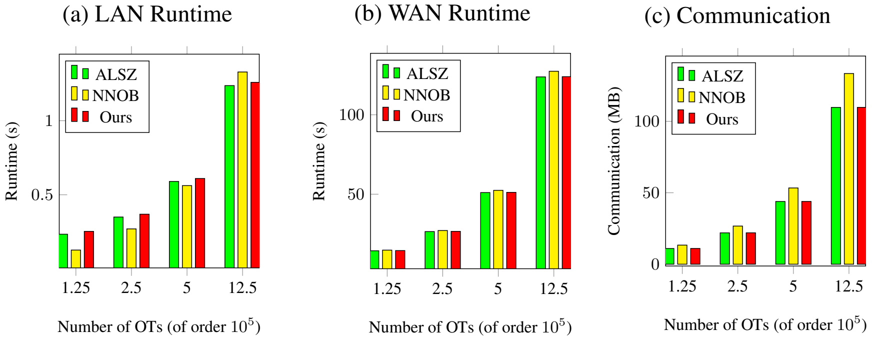
Table 6: Performance Comparison of our adaptive OT extension with the state-of-the-art static OT extensions for generating 1-out-of-2 OTs. The value of m denotes the number of OTs generated and $\kappa = 128$.

m	Runtime in LAN (in sec)			Runtime in WAN (in sec)			Communication (in MB)		
(10^5)	ALSZ	NNOB	**Ours**	ALSZ	NNOB	**Ours**	ALSZ	NNOB	**Ours**
1.25	0.234	0.127	0.253	14.35	14.75	14.469	11.00	13.36	11.024
2.5	0.350	0.270	0.369	26.44	27.11	26.569	21.96	26.70	21.984
5	0.590	0.562	0.610	51.02	52.43	51.14	43.90	53.38	43.924
12.5	1.237	1.329	1.259	123.9	127.43	124.024	109.74	133.45	109.764

Table 7: Performance Comparison of our adaptive OT extension with the state-of-the-art static OT extensions for generating 1-out-of-16 OTs. The value of m denotes the number of OTs generated and the value of the security parameter is $\kappa = 128$.

m	Runtime in LAN (in sec)				Runtime in WAN (in sec)				Communication (in MB)			
	ALSZ	NNOB	PSS	**Ours**	ALSZ	NNOB	PSS	**Ours**	ALSZ	NNOB	PSS	**Ours**
1.25×10^5	02.65	02.61	2.30	2.668	18.10	16.90	14.62	18.218	16.67	21.60	4.77	16.693
2.5×10^5	05.26	05.05	4.50	5.287	33.80	31.40	26.45	34.019	33.33	43.21	9.54	33.353
5×10^5	10.04	10.10	9.0	10.062	65.00	60.40	50.75	65.119	66.62	86.39	19.08	66.643
1.25×10^6	24.81	24.84	22.50	24.836	158.60	143.20	121.94	159.76	166.54	215.95	47.70	166.563

Fig. 14: Performance Comparison of OT extensions for producing 1-out-of-2 OTs on 16 bits. The x-axis denotes number of OTs (order 10^5) and y-axis denotes runtime for graphs (a); (b) and communication (MB) for graph (c).


(a) LAN Runtime

(b) WAN Runtime

(c) Communication

10 Conclusion

In this paper, we presented UC-secure, efficient round-optimal protocols for cryptographic primitives, secure against adaptive adversaries aided with asymptotic and empirical costs. Our OT and OT extension schemes rely solely on programmability of random oracle as a setup assumption, while our commitment scheme relies on the observability of random oracle. Our OT and commitment scheme are secure in the GRO model, making it the first adaptive OT and commitment scheme in the strong GUC model and thereby realistic for applications. We have validated our claims with robust implementation results supported by tables; graphs depicting the cost incurred for computation and communication in various settings. These results highlight the prospective practicality of our protocols in the adaptive setting. Finally, we leave it as an open problem to obtain a fully simulatable adaptive OT in the GRO model and an OT extension based on assumptions weaker than random oracle.

References

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

ABP17. Michel Abdalla, Fabrice Benhamouda, and David Pointcheval. Removing erasures with explainable hash proof systems. In *Public-Key Cryptography - PKC 2017 - 20th IACR International Conference on Practice and Theory in Public-Key Cryptography, Amsterdam, The Netherlands, March 28-31, 2017, Proceedings, Part I*, pages 151–174, 2017.

AHIV17. Scott Ames, Carmit Hazay, Yuval Ishai, and Muthuramakrishnan Venkitasubramaniam. Ligero: Lightweight sublinear arguments without a trusted setup. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017*, pages 2087–2104, 2017.

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.

ALSZ13. Gilad Asharov, Yehuda Lindell, Thomas Schneider, and Michael Zohner. More efficient oblivious transfer and extensions for faster secure computation. In *2013 ACM SIGSAC Conference on Computer and Communications Security, CCS'13, Berlin, Germany, November 4-8, 2013*, pages 535–548, 2013.

ALSZ15. Gilad Asharov, Yehuda Lindell, Thomas Schneider, and Michael Zohner. More efficient oblivious transfer extensions with security for malicious adversaries. 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 I*, pages 673–701, 2015.

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.

Bea96. Donald Beaver. Correlated pseudorandomness and the complexity of private computations. In *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing, Philadelphia, Pennsylvania, USA, May 22-24, 1996*, pages 479–488, 1996.

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.

CDG$^+$18. Jan Camenisch, Manu Drijvers, Tommaso Gagliardoni, Anja Lehmann, and Gregory Neven. The wonderful world of global random oracles. Cryptology ePrint Archive, Report 2018/165, 2018. https://eprint.iacr.org/2018/165.

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.

CDPW07. Ran Canetti, Yevgeniy Dodis, Rafael Pass, and Shabsi Walfish. Universally composable security with global setup. In *Theory of Cryptography, 4th Theory of Cryptography Conference, TCC 2007, Amsterdam, The Netherlands, February 21-24, 2007, Proceedings*, pages 61–85, 2007.

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.

GKPS18. Chaya Ganesh, Yashvanth Kondi, Arpita Patra, and Pratik Sarkar. Efficient adaptively secure zero-knowledge from garbled circuits. Cryptology ePrint Archive, Report 2018/043, 2018. https://eprint.iacr.org/2018/043.

GMO16. Irene Giacomelli, Jesper Madsen, and Claudio Orlandi. Zkboo: Faster zero-knowledge for boolean circuits. In *25th USENIX Security Symposium, USENIX Security 16, Austin, TX, USA, August 10-12, 2016.*, pages 1069–1083, 2016.

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.

GPV08. Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*, pages 197–206, 2008.

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.

HPV17. Carmit Hazay, Antigoni Polychroniadou, and Muthuramakrishnan Venkitasubramaniam. Constant round adaptively secure protocols in the tamper-proof hardware model. In *Public-Key Cryptography - PKC 2017 - 20th IACR International Conference on Practice and Theory in Public-Key Cryptography, Amsterdam, The Netherlands, March 28-31, 2017, Proceedings, Part II*, pages 428sss–460, 2017.

HV15. Carmit Hazay and Muthuramakrishnan Venkitasubramaniam. On black-box complexity of universally composable security in the CRS model. In *Advances in Cryptology - ASIACRYPT 2015 - 21st International Conference on the Theory and Application of Cryptology and Information Security, Auckland, New Zealand, November 29 - December 3, 2015, Proceedings, Part II*, pages 183–209, 2015.

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.

IKNP03. Yuval Ishai, Joe Kilian, Kobbi Nissim, and Erez Petrank. Extending oblivious transfers efficiently. In *Advances in Cryptology - CRYPTO 2003, 23rd Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 2003, Proceedings*, pages 145–161, 2003.

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.

IR89. Russell Impagliazzo and Steven Rudich. Limits on the provable consequences of one-way permutations. In *Proceedings of the 21st Annual ACM Symposium on Theory of Computing, May 14-17, 1989, Seattle, Washigton, USA*, pages 44–61, 1989.

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.

KK13. Vladimir Kolesnikov and Ranjit Kumaresan. Improved OT extension for transferring short secrets. In *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part II*, pages 54–70, 2013.

KOS15. Marcel Keller, Emmanuela Orsini, and Peter Scholl. Actively secure OT extension with optimal overhead. In *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part I*, pages 724–741, 2015.

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.

LZ11. Yehuda Lindell and Hila Zarosim. Adaptive zero-knowledge proofs and adaptively secure oblivious transfer. *J. Cryptology*, 24(4):761–799, 2011.

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

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.

OOS17. Michele Orrù, Emmanuela Orsini, and Peter Scholl. Actively secure 1-out-of-n OT extension with application to private set intersection. In *Topics in Cryptology - CT-RSA 2017 - The Cryptographers' Track at the RSA Conference 2017, San Francisco, CA, USA, February 14-17, 2017, Proceedings*, pages 381–396, 2017.

OTC. Encrypto group otextension code. https://github.com/encryptogroup/OTExtension.

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.

PSS17. Arpita Patra, Pratik Sarkar, and Ajith Suresh. Fast actively secure OT extension for short secrets. In *24th Annual Network and Distributed System Security Symposium, NDSS*, 2017.

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 Universal Composability Model

We prove security of our protocol in the standard Universal Composability (UC) framework of Canetti [Can01], with static and adaptive corruptions. And we conclude this section with the definition of $\mathcal{F}$-hybrid model, which is instrumental for security proofs in the UC model.

A.1 Static Security in the UC 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. $\mathcal{Z}$ is allowed to interact with Adv throughout the protocol. 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 $\text{IDEAL}_{\mathcal{F},\mathsf{Sim},\mathcal{Z}}(1^\kappa, z)$. We say that a protocol π UC-securely implements a functionality $\mathcal{F}$ in the presence of static adversaries if the real world and ideal world views are indistinguishable.

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:*

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

A.2 Adaptive Security in the UC Model

In the adaptive setting, $\mathcal{Z}$ can ask the real world adversary Adv to corrupt an honest party during the real world execution of the protocol or after the execution completes. During the execution, Adv can observe the public transcript of the protocol and based on that he can adaptively corrupt an honest party. Once a party gets corrupted, Adv gets access to the input and private randomness of the party, thus controlling the party from thereon. In case of post execution corruption, Adv observes the output and the transcript of the protocol, and then he corrupts the honest party to get access to the input and private randomness of the party. After post execution corruption occurs, Adv forwards its view to $\mathcal{Z}$. Based on that, $\mathcal{Z}$ constructs its real world view, which we denote as $\text{REAL}_{\mathcal{F},\mathsf{Adv},\mathcal{Z}}(1^\kappa, z)$.

Similarly, in the ideal world $\mathcal{Z}$ can ask the ideal world adversary Sim to corrupt an honest party during the ideal world execution of the protocol or after the execution completes. When $\mathcal{Z}$ instructs Sim to corrupt an honest party in the ideal world, Sim obtains the input of the honest party, in the ideal world, and he instructs the internal world adversary $\mathsf{Adv}_{\mathsf{Int}}$ to corrupt the corresponding honest party in the internal world. Recall that Sim simulates the honest parties in the internal execution. When $\mathsf{Adv}_{\mathsf{Int}}$ corrupts an honest party in the internal world, Sim has to produce a private randomness for the simulated honest party such that it matches with the input of the honest party and the simulated transcript produced by Sim, in the internal world, on behalf of the honest party. Sim provides this matching randomness and the input of the simulated honest party to $\mathsf{Adv}_{\mathsf{Int}}$ in the internal world. In case of post execution corruption of an honest party, Sim obtains the honest party's input in the ideal world and produces the matching randomness (corresponding to the simulated transcript) in a similar fashion to $\mathsf{Adv}_{\mathsf{Int}}$ in the internal world. After post execution corruption occurs, Adv forwards its view to Sim, who forwards it to $\mathcal{Z}$. Based on that, $\mathcal{Z}$ constructs its ideal world view, which we denote as $\text{IDEAL}_{\mathcal{F},\mathsf{Sim},\mathcal{Z}}(1^\kappa, z)$. We say that a protocol π UC-securely implements a functionality $\mathcal{F}$ in the presence of adaptive adversaries if the real world and ideal world views are indistinguishable.

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:*

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

A.3 The $\mathcal{F}$-hybrid model.

In order to construct our protocols, we utilize other secure two-party protocols as subprotocols. The standard way of doing this is to work in a "*hybrid model*" where both the parties interact with each other (as in the real model) in the outer protocol and use ideal functionality calls (as in the ideal world) for the subprotocols. The UC composition theorem states that if a protocol ρ UC-securely implements a functionality $\mathcal{F}$, then any execution of ρ in a bigger protocol can be replaced with ideal calls to the functionality $\mathcal{F}$. Specifically, while constructing a protocol π that uses ρ as subprotocol, for securely computing some functionality $\mathcal{F}$, the parties can run π and invoke $\mathcal{F}$. The execution of π that invokes $\mathcal{F}$, for each execution of ρ, is called the *$\mathcal{F}$-hybrid execution of* π and is denoted as $\pi^{\mathcal{F}}$. The hybrid ensemble $\mathsf{HYB}_{\pi^{\mathcal{F}},\,\mathsf{ADV},\,\mathcal{Z}}(1^\kappa, z)$ describes $\mathcal{Z}$'s output after interacting with Adv and the parties running protocol $\pi^{\mathcal{F}}$. Whereas, the execution of π that considers execution of ρ is denoted as π^{ρ}. The hybrid ensemble $\mathsf{HYB}_{\pi^{\rho},\,\mathsf{ADV},\,\mathcal{Z}}(1^\kappa, z)$ describes $\mathcal{Z}$'s output after interacting with Adv and the parties running protocol π^{ρ}. By UC security, the two hybrids $\mathsf{HYB}_{\pi^{\mathcal{F}},\,\mathsf{ADV},\,\mathcal{Z}}(1^\kappa, z)$ and

$\mathsf{HYB}_{\pi^\rho,\,\mathsf{ADV},\,\mathcal{Z}}(1^\kappa, z)$ are indistinguishable. This permits replacing executions of ρ, in π, with ideal calls to $\mathcal{F}$ functionality; thereby allowing π to execute in the $\mathcal{F}$-hybrid model. It simplifies the security proof of $\pi^{\mathcal{F}}$ as it can be performed in the $\mathcal{F}$-hybrid model, instead of proving security of ρ within the proof of π^ρ.

B 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:*

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

Definition 4. *(Privacy) A garbling scheme* Garble *is* ***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:*

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

Definition 5. *(Verifiability) A garbling scheme* Garble*is* ***verifiable*** *if for all input lengths $n \leq \mathit{poly}(\kappa)$, 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:*

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

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 $\mathsf{X} = \mathsf{En}(x, \mathsf{e}) = \left(\mathbb{K}_i^{x_i}\right)_{i \in [n]}$.

C Realization of OT Framework Based on DDH

In this section we present an instantiation of the DME (from [PVW08]) based on the DDH assumption:

- $(\mathsf{crs}, t) \leftarrow \mathsf{SetupMessy}(1^\kappa)$: In messy mode, the crs is a non-DDH tuple and it is generated as follows: Sample $g_0, g_1 \leftarrow_R \mathbb{G}$, $x, y \leftarrow_R \mathbb{Z}_p$ and initialize $h_0 = g_0^x$, $h_1 = g_1^y$. Set $\mathsf{crs} = (g_0, g_1, h_0, h_1)$ and the trapdoor t is set to (x, y).
- $(\mathsf{crs}, t) \leftarrow \mathsf{SetupDec}(1^\kappa)$: In decryption mode, the crs is a DDH tuple and it is generated as follows: Sample $g_0 \leftarrow_R \mathbb{G}$, $x, y \leftarrow_R \mathbb{Z}_p$ and initialize $g_1 = g_0^y$, $h_0 = g_0^x$, $h_1 = g_1^x$. Set $\mathsf{crs} = (g_0, g_1, h_0, h_1)$ and the trapdoor t is set to (x, y).
- $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(\mathsf{crs}, \sigma)$ Given $\sigma \in \{0,1\}$ and $\mathsf{crs} = (g_0, g_1, h_0, h_1)$, set $\mathsf{pk} = (g, h) = (g_\sigma^\alpha, h_\sigma^\alpha)$, where $\alpha \leftarrow_R \mathbb{Z}_p$. The secret key sk is set to α.
- $y \leftarrow \mathsf{Enc}(\mathsf{pk}, b, m)$: Given $\mathsf{pk} = (g, h)$, m and $b \in \{0,1\}$, m is encrypted as follows: Sample $s, r \leftarrow_R \mathbb{Z}_p$ and set $u = g_b^s h_b^r, v = g^s h^r$. The ciphertext is $y = (u, v.m)$ and the corresponding randomness is (s, r).

- $m \leftarrow \mathsf{Dec}(\mathsf{sk}, y)$: Given $\mathsf{sk} = \alpha$ and ciphertext $y = (c_0, c_1)$, the corresponding plaintext s obtained as $m = c_1/c_0^{\alpha}$.
- $b \leftarrow \mathsf{FindMessy}(\mathsf{pk}, t)$: Given $\mathsf{pk} = (g, h)$ and $t = (x, y)$, if $h = g^x$ then output that branch $b = 0$ is messy else output that branch $b = 1$ is messy.
- $(\mathsf{pk}, \mathsf{sk}_0, \mathsf{sk}_1) \leftarrow \mathsf{DecKeyGen}(t)$: Given $\mathsf{crs} = (g_0, g_1, h_0, h_1)$ in decryption mode and $t = (x, y)$ generate public and secret key pairs as follows: Sample $r_0 \leftarrow_R \mathbb{Z}_p$ and compute $r_1 = r_0/y$. Set $\mathsf{pk} = (g, h) = (g_0^{r_0}, h_0^{r_0})$, $\mathsf{sk}_0 = r_0$ and $\mathsf{sk}_1 = r_1$.

The above DDH-based instantiation correctly implements a DME scheme and it satisfies Properties 1-5 (Section 4). Proof of Property 1-4 follows from the paper of [PVW08]. Next, we show that it also satisfies Property 5, i.e. the crs in the messy mode can be sampled using a random oracle. The crs for the messy mode is required to be a non-DDH tuple. A random oracle query result of size $4|\mathbb{G}|$ bits returns a DDH tuple, with probability :

$$\frac{|\mathbb{Z}_p|^3}{|\mathbb{Z}_p|^4} = \frac{1}{|\mathbb{Z}_p|}$$

With $1 - \frac{1}{|\mathbb{Z}_p|}$ probability the tuple will be non-DDH and hence a valid crs for messy mode will be generated, satisfying Property 5.

D Instantiation of DME based on LWE

In this section we provide an instantiation of our DME based on LWE. Before describing the DME instantiation we recall the definition of an LWE encryption scheme from [GPV08], which will be instrumental in the instantiation. The encryption scheme is a collection of following three algorithms:

- $\mathsf{LWESetup}$: Choose a matrix $\mathbf{A} \leftarrow \mathbb{Z}_q^{m \times \kappa}$ uniformly at random.
- $\mathsf{LWEKeyGen}$: Choose a secret decryption key $\mathbf{s} \leftarrow \mathbb{Z}_q^{\kappa}$ uniformly at random. The public key is the vector $\mathbf{p} = \mathbf{A}^{\mathsf{T}}\mathbf{s} + \mathbf{x} \in \mathbb{Z}_q^m$, where $\mathbf{x} = (x_1, \ldots, x_m)$ and each x_i is chosen independently from the error distribution χ for $i \in [m]$.
- $\mathsf{LWEEnc}(\mathbf{p}, b)$: To encrypt a bit $b \in \{0, 1\}$, choose a vector $\mathbf{e} \in \mathbb{Z}^m$ uniformly at random and set the ciphertext as $(\mathbf{u}, c) = (\mathbf{Ae}, \mathbf{p}^{\mathsf{T}}\mathbf{e} + b.\lceil q/2 \rceil) \in \mathbb{Z}_q^{\kappa+1}$.
- $\mathsf{LWEDec}(\mathbf{s}, (\mathbf{u}, c))$: Compute $b' = c - \mathbf{s}^{\mathsf{T}}\mathbf{u} \in \mathbb{Z}_q$. Output 0 if b' is closer to 0 than to $\lceil q/2 \rceil \bmod q$, otherwise output 1.

The above encryption scheme satisfies the notion of IND-CPA security if we assume that the LWE problem is hard for parameters $q = \mathcal{O}(\kappa^3), m = \mathcal{O}(\kappa \log \kappa)$ (Lemma. 1). The proof appears in the work of [GPV08] and we refer to their paper for more details.

Lemma 1. *The cryptosystem above is CPA-secure, assuming that LWE is hard for parameters q, m.*

Next, we will use the encryption scheme in our DME instantiation. In our instantiation we need an additional algorithm, $\mathsf{IsMessy}$, besides the usual algorithms of DME. $\mathsf{IsMessy}(t, \mathsf{pk})$ answers whether pk is messy or not, when it is invoked with trapdoor t on public key pk. Now we are ready to instantiate the algorithms for the DME scheme based on the LWE assumption. The description of the algorithms has been borrowed from the paper of [PVW08].

- $(\mathsf{crs}, t) \leftarrow \mathsf{SetupMessy}(1^\kappa)$: In the messy mode the crs is generated as follows: Sample a matrix $\mathbf{A} \leftarrow \mathbb{Z}_q^{\kappa \times m}$ uniformly at random, along with a trapdoor $t = (\mathbf{S}, \mathbf{A})$ (as described in Section 5.3.2 of [GPV08]). Sample an independent row vector $\mathbf{v}_b \leftarrow \mathbb{Z}_q^{l \times m}$ uniformly at random for every $b \in \{0, 1\}$. Set $\mathsf{crs} = (\mathbf{A}, \mathbf{v}_0, \mathbf{v}_1)$ and $t = (\mathbf{S}, \mathbf{A})$.
- $(\mathsf{crs}, t) \leftarrow \mathsf{SetupDec}(1^\kappa)$: In the decryption mode the crs is generated as follows: Sample a matrix $\mathbf{A} \leftarrow \mathbb{Z}_q^{\kappa \times m}$ uniformly at random. Choose a row vector $\mathbf{w} \leftarrow \mathbb{Z}_q^{1 \times m}$ uniformly at random. For every $b \in \{0, 1\}$, sample a secret $\mathbf{s}_b \leftarrow \mathbb{Z}_q^n$ uniformly at random and an error row vector $\mathbf{x}_b \leftarrow \chi^{1 \times m}$ (i.e., the m entries are chosen independently from error distribution χ). Let $\mathbf{v}_b = \mathbf{s}_b^\mathsf{T} \mathbf{A} + \mathbf{x}_b \mathbf{w}$. Set $\mathsf{crs} = (\mathbf{A}, \mathbf{v}_0, \mathbf{v}_1)$ and $t = (\mathbf{w}, \mathbf{s}_0, \mathbf{s}_1)$.
- $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(\mathsf{crs}, \sigma)$: Given crs and σ, sample a secret $\mathbf{r} \leftarrow \mathbb{Z}_q^n$ and a row vector $\mathbf{x} \leftarrow \chi^{l \times m}$. Set $\mathsf{pk} = \mathbf{r}^\mathsf{T} \mathbf{A} + \mathbf{x}\mathbf{v}$ and $\mathsf{sk} = \mathbf{r}$.
- $y \leftarrow \mathsf{Enc}(\mathsf{pk}, b, m)$: Given $\mathsf{pk} = \mathbf{A}$, m and $b \in \{0, 1\}$, m is encrypted as $y = \mathsf{LWEEnc}((\mathbf{A}, \mathsf{pk} + \mathbf{v}_b), m)$.
- $m \leftarrow \mathsf{Dec}(\mathsf{sk}, y)$: Given $\mathsf{sk} = \mathbf{r}$ and ciphertext v, the underlying plaintext message is decrypted as $m = \mathsf{LWEDec}(\mathsf{sk}, y)$.
- $b \leftarrow \mathsf{FindMessy}(\mathsf{pk}, t)$: Given pk and $t = (\mathbf{S}, \mathbf{A})$ in messy mode, invoking $\mathsf{IsMessy}((\mathbf{S}, \mathbf{A}), \mathsf{pk} + \mathbf{v}_b)$ for each $b \in \{0, 1\}$, outputs the messy branch value for b, and it is correct with overwhelming probability.
- $(\mathsf{pk}, \mathsf{sk}_0, \mathsf{sk}_1) \leftarrow \mathsf{DecKeyGen}(t)$: Given the $\mathsf{crs} = (\mathbf{A}, \mathbf{v}_0, \mathbf{v}_1)$ in decryption mode and the trapdoor $t = (\mathbf{w}, \mathbf{s}_0, \mathbf{s}_1)$, the public and secret keys pair is formed as follows: $(\mathsf{pk}, \mathsf{sk}_0, \mathsf{sk}_1) = (\mathbf{w}, \mathbf{s}_0, \mathbf{s}_1)$.

The paper of [PVW08] has proven that the above instantiation satisfies correctness and Properties 1-4 of DME. It has also shown in Lemma 7.4 that most of the keys are messy, proving Property 5. In particular, if the crs in the messy mode is generated as $\mathsf{crs} = (\mathbf{A}, \mathbf{v}_0, \mathbf{v}_1) \leftarrow \mathcal{F}_{\mathsf{RO}}(\mathsf{sid}||c)$, then $\mathcal{F}_{\mathsf{RO}}(\mathsf{sid}||c)$ returns a messy key except with negligible probability for a random value c, where $\mathcal{F}_{\mathsf{RO}} : \{0, 1\}^{2\kappa} \rightarrow \mathbb{Z}_q^{\kappa \times m} \times \mathbb{Z}_q^{2l \times m}$.