Composable Adaptive Secure Protocols without Setup under Polytime Assumptions

Carmit Hazay* Muthuramakrishnan Venkitasubramaniam†

Abstract

All previous constructions of general multiparty computation protocols that are secure against adaptive corruptions in the concurrent setting either require some form of setup or non-standard assumptions. In this paper we provide the first general construction of secure multi-party computation protocol *without* any setup that guarantees composable security in the presence of an *adaptive adversary* based on standard polynomial-time assumptions. We prove security under the notion of "UC with super-polynomial helpers" introduced by Canetti et al. (FOCS 2010), which is closed under universal composition and implies "super-polynomial-time simulation". Moreover, our construction relies on the underlying cryptographic primitives in a black-box manner.

Next, we revisit the zero-one law for two-party secure functions evaluation initiated by the work of Maji, Prabhakaran and Rosulek (CRYPTO 2010). According to this law, every two-party functionality is either trivial (meaning, such functionalities can be reduced to any other functionality) or complete (meaning, any other functionality can be reduced to these functionalities) in the Universal Composability (UC) framework. As our second contribution, assuming the existence of a simulatable public-key encryption scheme, we establish a zero-one law in the adaptive setting. Our result implies that every two-party non-reactive functionality is either trivial or complete in the UC framework in the presence of adaptive, malicious adversaries.

Keywords: UC Security, Adaptive Secure Computation, Coin-Tossing, Black-box construction, Extractable Commitments, Zero-One Law

*Bar-Ilan University, Israel. Email: `carmit.hazay@cs.biu.ac.il`. Research supported by the Israel Ministry of Science and Technology (grant No. 3-10883) and by the BIU Center for Research in Applied Cryptography and Cyber Security in conjunction with the Israel National Cyber Bureau in the Prime Minister's Office.

†University of Rochester, NY. Email: `muthuv@cs.rochester.edu`. Research supported by Google Faculty Research Grant and NSF Award CNS-1526377.

Contents

1 Introduction

Secure computation enables a set parties to mutually run a protocol that computes some function f on their private inputs, while preserving a number of security properties. Two of the most important properties are privacy and correctness. The former implies data confidentiality, namely, nothing leaks by the protocol execution but the computed output. The later requirement implies that no corrupted party or parties can cause the output to deviate from the specified function. It is by now well known how to securely compute any efficient functionality [Yao86, GMW87, MR91, Bea91, Can01] in various models and under the stringent simulation-based definitions (following the ideal/real paradigm). Security is typically proven with respect to two adversarial models: the semi-honest model (where the adversary follows the instructions of the protocol but tries to learn more than it should from the protocol transcript), and the malicious model (where the adversary follows an arbitrary polynomial-time strategy), and feasibility results are known in the presence of both types of attacks. The initial model considered for secure computation was of a static adversary where the adversary controls a subset of the parties (who are called corrupted) before the protocol begins, and this subset cannot change. In a stronger corruption model the adversary is allowed to choose which parties to corrupt throughout the protocol execution, and as a function of its view; such an adversary is called adaptive.

These feasibility results rely in most cases on stand-alone security, where a *single* set of parties run a *single* execution of the protocol. Moreover, the security of most cryptographic protocols proven in the stand-alone setting does not remain intact if many instances of the protocol are executed concurrently [Lin03]. The strongest (but also the most realistic) setting for concurrent security is known by *Universally Composable* (UC) security [Can01]. This setting considers the execution of an unbounded number of concurrent protocols in an arbitrary and adversarially controlled network environment. Unfortunately, stand-alone secure protocols typically fail to remain secure in the UC setting. In fact, without assuming some *trusted help*, UC security is impossible to achieve for most tasks [CF01, CKL06, Lin03]. Consequently, UC secure protocols have been constructed under various *trusted setup* assumptions in a long series of works; see [BCNP04, CDPW06, KLP07, CPS07, LPV09, DMRV13] for few examples.

Concurrent security *without any* setup. In many situations, having a trusted set-up might be hard or expensive. Designing protocols in the plain model that provide meaningful security in a concurrent setting is thus an important challenge. In this regard, a relaxation of UC security allows the adversary in an ideal execution to run in *super-polynomial time*; this notion is referred to as super-polynomial security (or SPS) [Pas03]. On a high-level, this security notion guarantees that any attack carried out by an adversary running in polynomial time can be mounted in the ideal execution with super-polynomial resources. In many scenarios, such a guarantee is meaningful and indeed several past works have designed protocols guaranteeing this relaxed UC security against static adversaries [Pas03, BS05, LPV09] and adaptive adversaries [BS05, DMRV13, Ven14]. While initial works relied on sub-exponential hardness assumptions, more recent works in the static setting have been constructed based on standard polynomial-time hardness assumptions.

The work of [CLP10], put forth some basic desiderata regarding security notions in a concurrent setting. One of them requires supporting *modular analysis*: Namely, there should be a way to deduce security properties of the overall protocol from the security properties of its components. Quite surprisingly, it was shown in [CDPW06] that most protocols in the UC framework that consider both trusted setups and relaxed models of security, in fact, do not support this.

Towards remedying the drawbacks of SPS security, Prabhakaran and Sahai [PS04] put forth the notion of Angel-based UC security that provides guarantees analogous to SPS security while at the same time supporting modular analysis. In this model, both the adversary and the simulator have access to an oracle, referred to as an "angel" that provides judicious use of super-polynomial resources. In the same work and subsequent effort [MMY06] the authors provided constructions under this security notion relying on non-standard hardness assumptions. Recently, Canetti, Lin and Pass [CLP10] provided the first constructions in this model relying on standard polynomial time assumptions. Moreover, to emphasize the modular analysis requirement, they recast the notion of Angel-based security in the extended UC (EUC) framework of [CDPW06] calling it UC with super-polynomial helpers. While prior approaches relied on non-interactive helpers that were stateless, this work designed a helper that was highly interactive and stateful. Since this work, several follow up works [LP12a, GLP$^+$15, Kiy14] have improved both the round complexity and the computational assumptions. The most recent work due to Kiyoshima [Kiy14] provides a $\widetilde{O}(\log^2 n)$-round protocol to securely realize any functionality in this framework based on semi-honest oblivious transfer protocols where the underlying primitives are used in a black-box manner. In this line of research, the work of Canetti, Lin and Pass [CLP13] distinguishes itself by designing protocols that guarantee a stronger notion of security. More precisely, they extend the angel-based security so that protocols developed in this extended framework additionally preserve security of other protocols running the system (i.e. cause minimal "side-effect"). They refer to such protocols "environment friendly" protocols. However, as observed in the same work, this strong notion inherently requires non-black-box simulation techniques. Moreover, the constructions presented in [CLP13] are non-black-box as well.

While considerable progress has been made in constructing protocols secure against static adversaries, very little is known regarding adaptive adversaries. Specifically, the work of Barak and Sahai [BS05] and subsequent works [DMRV13, Ven14] show how to achieve SPS security under non-standard assumptions. Besides these works, every other protocol that guarantees any meaningful security against adaptive adversaries in a concurrent setting has required setup. The main question left open by previous work regarding adaptive security is:

> *Can we realize general functionalities with SPS security in the plain model under standard polynomial time assumptions? and,*
> *Can we show adaptively secure angel-based (or EUC-security) under standard hardness assumptions where the underlying primitives are used in a black-box manner?*

We stress that even the works that provide SPS security require non-standard or sub-exponential hardness assumptions and are *non-black-box*, that is, the constructions rely on the underlying assumptions in non-black-box way. A more ambitious goal would be to construct "environment-friendly" protocols [CLP13] and we leave it as future work.

1.1 Our Results

In this work we resolve both these questions completely and provide the first realizations of general functionalities under EUC security against malicious, adaptive adversaries (See [CDPW06, CLP10] for a formal definition). More formally, we prove the following theorem:

Theorem 1.1 *Assume the existence of a simulatable public-key encryption scheme. Then there exists a sub-exponential time computable (interactive) helper machine $\mathcal{H}$ such that for any "well formed" polynomial-time functionality $\mathcal{F}$, there exists a protocol that realizes $\mathcal{F}$ with $\mathcal{H}$-EUC security, in the plain model secure against malicious, adaptive adversaries. Furthermore, the protocol makes only black-box use of the underlying encryption scheme.*

We recall here that simulatable public-key encryption (PKE), introduced by Damgard and Nielsen [DN00], allows to obliviously sample the public key/ciphertext without the knowledge of the corresponding secret key/plaintext.

As far as we know, this is the first construction based on polynomial-time hardness assumptions of a secure multi-party computation that achieves any non-trivial notion of concurrent security against adaptive adversaries without any trusted-set up (in the plain model) and without assuming an honest majority. Furthermore, the construction supports *modular analysis*and relies on the underlying scheme in a black-box way. In essence, our protocol provides the *strongest* possible security guarantees in the plain model.

A zero-one law for adaptive security In [PR08], Prabhakaran and Rosulek initiated the study of the "cryptographic complexity" of two-party secure computation tasks in the UC framework. Loosely speaking, in their framework a functionality $\mathcal{F}$ *UC-reduces* to another functionality $\mathcal{G}$ if there is a UC secure protocol for $\mathcal{F}$ in the $\mathcal{G}$-hybrid, i.e., using ideal access to $\mathcal{G}$. Under this notion of a reduction in the presence of static adversaries, Maji et al in [MPR10] established a *zero-one* law for two-party (non-reactive) functionalities which states that every functionality is either trivial or complete. In this work, we extend their result to the adaptive setting to obtain the following theorem.

Theorem 1.2 (Informal) *All non-reactive functionalities are either trivial or complete under UC-reductions in the presence of adaptive adversaries.*

1.2 Previous Techniques

All previous approaches for Angel-based UC secure protocols relied on a particular "adaptive hardness" assumption which amounts to guaranteeing security in the presence of an adversary that has adaptive access to a helper function. Indeed, as pursued in the orginal approaches by [PS04, MMY06], complexity leveraging allows for designing such primitives. A major breakthrough was made by Canetti, Lin and Pass [CLP10] that showed that a helper function could be based on standard assumptions. The main technical tool introduced in this work is a new notion of a commitment scheme that is secure against an adaptive chosen commitment attack (CCA security). On a high-level, a tag-based commitment scheme, which are schemes that have additionally a tag as a common input, is said to be CCA-secure if a commitment made with tag id is hiding even if the receiver has access to a (possibly, super-polynomial time) oracle that is capable of "breaking" commitments made using any tag $\mathsf{id}' \neq \mathsf{id}$. In the original work, they constructed a $O(n^\epsilon)$-round CCA-secure commitment scheme based on one-way functions (OWFs) [CLP10]. Since then, several followup works have improved this result, culminating in the work of Kiyoshima [Kiy14] who gave a $\tilde{O}(\log^2 n)$-round construction of a CCA-secure commitment scheme based on OWFs while relying on the underlying OWF in a black-box way.[1] We remark that Angel-based security based on standard polynomial-time assumptions have been constructed only in the static setting. Moreover, all constructions in this line of work, first construct a CCA-secure commitment scheme and then realize a complete UC functionality, such as the commitment or oblivious-transfer functionality using a "decommitment" oracle as the helper functionality.

When considering the adaptive setting, we begin with the observation that any cryptographic primitive in use must be secure in the presence of adaptive corruptions. Namely, we require a simulation that can produce random coins consistent with any honest party during the execution as soon as it is adaptively corrupted. A first attempt would be to enhance a CCA-secure commitment scheme to the adaptive setting. This means there must be a mechanism to equivocate the commitment scheme. It is in fact crucial in all works using CCA-secure commitments that the helper functionality be able to break the commitment and obtain the unique value (if any) that the commitment can be decommitted to. However, equivocal commitments by

[1]We further note that Goyal et al. [GLP+15] gave a $\tilde{O}(\log n)$-round CCA-secure commitment scheme but makes use of the OWF in a non-black-box way.

definition can have commitments that do not have unique decommitments. In essence, standard CCA-secure commitment schemes are necessarily statistically binding (and all previous constructions indeed are statistically binding). Hence, it would be impossible to use any of those schemes in the adaptive setting.

The previous works [DMRV13, Ven14] get around this issue by relying on some sort of setup, namely, a mechanism by which the commitments will be statistically binding in the real world for adversaries, yet can be equivocated in the ideal world by the simulator. The notion of an adaptive instance-dependent scheme [LZ11] provides exactly such a primitive. Informally, such commitment schemes additionally take as input an NP-statement and provide the following guarantee: If the statement is true, the commitment can be equivocated using the witness, whereas if the statement is false then the commitment is statistically binding. Moreover, it admits adaptive corruptions where a simulator can produce random coins for an honest committer, revealing a simulated commitment to any value. The work of [BS05] relies on complexity leveraging in order to generate statements that a simulator, in super-polynomial time can break but an adversary, in polynomial time, cannot break. On the other hand, the works of [DMRV13, Ven14] rely on the so called *UC puzzle*, that provides a similar advantage for the simulator while relying on milder assumptions.

A second issue arises in the adaptive setting where any commitment scheme that tolerates concurrent executions (even with fixed roles) and is equivocal, implies some sort of selective opening security. Indeed, the result of Ostrovsky et al. [ORSV13] proves that it is impossible, in general, to construct concurrent commitments secure w.r.t. selective opening attacks. Getting around this lower bound is harder. Previous results [DMRV13, Ven14] get around this lower bound by first constructing a "weaker" commitment scheme in a limited concurrent environment. Namely, they construct an equivocal non-malleable commitment scheme that can simulate any man-in-the-middle adversary receiving "left" commitments made to independent and identically distributed values (via some *a priori* fixed distribution), and is acting as a committer in many "right" interactions. This allows to get around the [ORSV13] lower bound, as Ostrovsky et al. lower bound holds only if the simulator *does not* know the distribution of the commitments received by the adversary. In any case, all previous works fail to achieve the stronger Angel-based UC security, where the helper function is provided to the adversary and the simulator in the real and ideal world respectively are the same.

Given these bottlenecks, it seems unlikely to use a commitment scheme with such a property. In this work, we introduce a new primitive that will allow to both provide the adaptive hardness property as well as admit adaptive corruptions. This primitive is coin-tossing and will additionally require to satisfy an *adaptive hardness guarantee* that we define in the next section. We chose coin-tossing as a primitive as it does not require any inputs from the parties and the output is independent of any "global" inputs of the parties participating in the coin-tossing. Roughly speaking, if a party is adaptively corrupted it is possible to sample a random string as the output and equivocate the interaction to output this string. On the other hand, a commitment scheme will not allow such a mechanism as corrupting a sender requires equivocating the interaction to a particular value (that could potentially depend on a global input).

2 Our Main Tool: CCA-Secure Coin-Tossing

The main technical tool used in our construction is a new notion of a coin-tossing protocol that is secure against adaptive chosen coins attack (CCA security). Cryptographic primitives with an adaptive hardness property has been studied extensively in the case of the encryption schemes (chosen ciphertext attack security), and more recently in the case of commitments [CLP10, KMO14, Kiy14, GLP$^+$15]. We define here an analogous notion for coin-tossing protocols for the stronger case of adaptive corruptions.

A natural approach is to say that a coin-tossing protocol is CCA-secure if the coin-tossing scheme retains its simulatability even if a "Receiver" has access to a "biasing" oracle $\mathcal{O}$ that has the power to bias the protocol outcome of the coin-tossing to any chosen value. Unfortunately, we do not know how to realize such a notion and will instead, consider a weaker "indistinguishability"-based notion (as opposed to simulation based notion) that will be sufficient for our application.

A motivating example. We motivate our definition by discussing what security properties are desirable for coin-tossing protocols (in general). Consider a public-key cryptosystem that additionally has a property that a public-key can be obliviously sampled using random coins without knowledge of the secret-key (eg, dense cryptosystems, simulatable public-key encryption schemes). Furthermore, semantic security holds for a key sampled using the oblivious strategy. Consider a protocol where the parties after engaging in a coin-tossing protocol sample a public-key using the outcome of the coin-tossing. In such a scenario we would like the coin-tossing scheme to ensure that the semantic-security continues to hold if parties encrypt messages using the public-key.

The natural "simulatable" definition requires the coin-tossing to be "simulatable". If we instantiate a simulatable coin-tossing protocol in our motivating application, semantic security of ciphertexts constructed using the public-key sampled from the coin-tossing outcome indeed holds via a simple security reduction. Suppose there exists an adversary that distinguishes an encryption of 0 from 1 when encrypted under a public-key sampled using the coin-tossing. We can use the simulator to construct an adversary that violates the security of the underlying encryption scheme. Consider a simulator that receives as a challenge a uniformly sampled string and a ciphertext generated with the associated public-key. The simulator can internally simulate the coin-tossing to be this sampled string and thereby use the adversary to break the security of the encryption scheme.

A weaker alternative to simulatability is an information-theoretic based definition where the requirement would be that the entropy of the outcome is sufficiently high. However, such a definition will not suffice in our motivating example.[2] This is because we will not be able to "efficiently" reduce a cheating adversary to the violating the security game of the underlying cryptosystem.

Instead, we take a more direct approach where the security for the coin-tossing is defined so that it will be useful in our motivating example. First, we generalize the security game of the underlying encryption scheme in our motivating example to any indistinguishability based primitive. We model such a primitive via a (possibly) interactive challenger $\mathcal{C}$ that receives as input a random string o and a private bit b. We say that an adversary interacting with $\mathcal{C}$ succeeds if when interacting on a randomly chosen o and bit b, the adversary can guess b with probability better than a $\frac{1}{2}$. Let π be a (two-party) coin-tossing protocol. Our motivating example can be formulated using the following experiment EXP_b with an adversary $\mathcal{A}$:

- $\mathcal{A}$ interacts with an honest party using π to generate o.
- Next, it interacts with a challenger $\mathcal{C}$ on input o and bit b.

We compare this experiment with a stand-alone experiment STA_b where an adversary $\mathcal{B}$ simply interacts with $\mathcal{C}$ on input b and o where o is uniformly sampled. Our security definition of the coin-tossing protocol must preserve the following security property against a challenger $\mathcal{C}$: *if the stand-alone game is hard to distinguish, i.e.* STA_0 *from* STA_1*, then the experiments* EXP_0 *from* EXP_1 *must also be hard to distinguish.* More formally, our definition will (explicitly) give a reduction from any adversary that $\mathcal{A}$ distinguishes EXP_b to a stand-alone adversary $\mathcal{B}$ that can distinguish STA_b. Finally, in a CCA-setting, we generalize this definition by requiring that if there exists any oracle adversary $\mathcal{A}^{\mathcal{O}}$ with access to a biasing oracle $\mathcal{O}$ that

[2] Unless the cryptosystems have additional properties. For instance, consider dual-mode encryption schemes where there are keys sampled via a high-entropy string and could potentially be statistically hiding.

can distinguish EXP_0 from EXP_1, then there exists a stand-alone adversary $\mathcal{B}$ (without access to any oracle) that can distinguish STA_b from STA_1.

Towards formalizing this notion and incorporating adaptive corruptions, we first consider a tag-based coin-tossing protocol between two parties, an *Initiator* I and a *Receiver* R with $l(n)$-bit identities and $m(n)$-bit outcomes. A biasing oracle $\mathcal{O}$ interacts with an adversary $\mathcal{A}$ as follows: $\mathcal{O}$ participates with $\mathcal{A}$ in many sessions using the protocol where the oracle controls the initiator, using identities of length $l(n)$ that are chosen adaptively by $\mathcal{A}$. At the beginning of each session, the adversary produces a coin outcome $c \in \{0,1\}^{m(n)}$ to the oracle where at the end of this session, if the initiator that is initially controlled by the oracle is not (adaptively) corrupted by the adversary, then the outcome of the interaction must result in the *chosen coin* c. If at any point during the interaction the initiator is corrupted, then the oracle simply provides the random-tape of I that is consistent with the partial transcript of the interaction.

We compare an experiment EXP_b with oracle PPT adversary $\mathcal{A}^{\mathcal{O}}$ and a stand-alone experiment STA_b with adversary $\mathcal{B}$. In the man-in-the-middle experiment, an adversary with oracle access to $\mathcal{O}$ interacts with a honest receiver R on identity id to generate an output $o \in \{0,1\}^n$ where n is the security parameter. Then it interacts with a challenger $\mathcal{C}$ on common input (n, o, id) and private input b for $\mathcal{C}$. The adversary is allowed to corrupt the receiver R, challenger $\mathcal{C}$ and any of the interactions with $\mathcal{O}$. If the adversary $\mathcal{A}$ corrupts either $\mathcal{C}$ or I then the output of the experiment is set to $\perp$. If for some identity id' on which $\mathcal{A}$ queries $\mathcal{O}$, it holds that $\mathsf{id}' = \mathsf{id}$, then the output of the experiment is set to $\perp$. Otherwise, the output of the experiment is set to be the output of the adversary.

In the stand-alone experiment STA_b, we consider a PPT adversary $\mathcal{B}$ that interacts with $\mathcal{C}$ on common input (n, o) and private input b for $\mathcal{C}$ where o is uniformly sampled from $\{0,1\}^n$. The output of the experiment is set to be the output of $\mathcal{B}$. Observe that in the stand-alone experiment $\mathcal{B}$ does not get to corrupt $\mathcal{C}$.

Informally, a tag-based coin-tossing scheme $\langle I, R \rangle$ is said to be CCA-secure against a challenger $\mathcal{C}$, if there exists a biasing oracle $\mathcal{O}$ for $\langle I, R \rangle$ such that for every oracle PPT adversary $\mathcal{A}$ and distinguisher $\mathcal{D}$ such that $\mathcal{D}$ distinguishes EXP_0 and EXP_1 with $\mathcal{A}$, then there exist a (stand-alone) PPT $\mathcal{B}$ and distinguisher $\mathcal{D}'$ such that $\mathcal{D}'$ distinguishes STA_0 and STA_1 with $\mathcal{B}$.

In addition to this security requirement we will additionally consider the following definition of CCA-security which simply requires that any adversary with oracle access to a biasing oracle $\mathcal{O}$ can be simulated by a stand-alone PPT machine. In this case, we simply say $\langle I, R \rangle$ is CCA-secure w.r.t $\mathcal{O}$.

Quite surprisingly, we show how to realize such a primitive by relying on a CCA-secure commitment that is secure only against static adversaries. The idea here is that while CCA-secure commitments cannot admit adaptive corruptions, the basic security game ensures that an *unopened* commitment remains hiding in the presence of an adversary having access to a decommitment oracle. We combine such a commitment scheme with the technique of Hazay and Venkitasubramaniam from [HV15] who showed how to construct an adaptive UC-commitment scheme, starting from a public-key encryption scheme (with an oblivious ciphertext generation property) in the CRS model. On a high-level, the protocol can be abstracted as providing a transformation from a extractable (only) commitment scheme (that has a oblivious generation property) to a full adaptively secure UC-commitment. At first, it would be tempting to simply replace the invocations of extractable commitments with a CCA-secure commitment scheme as we only require extraction from these commitments and not equivocation in the simulation. However, this intuition fails in an adaptive setting when considering the fact that we additionally require that the commitment scheme has a oblivious generation property and it is unclear how to construct such a extractable scheme (based on rewinding) to have this property. Nevertheless, we show how to carefully use CCA-secure commitments in the same protocol to obtain a CCA-secure coin-tossing scheme. Next, we show that given a CCA-secure coin-tossing protocol with a biasing oracle $\mathcal{O}$ it is possible to realize the ideal commitment functionality using a helper functionality.

Again, we use another variant of the same protocol from [HV15] to accomplish this transformation. Our constructions and proofs of security are highly modular and quite simple. Moreover, all our transformations rely on the underlying primitives in a black-box manner.

Finally, we show that the black-box construction of an $O(n^{\epsilon})$-round CCA-secure commitment scheme from Lin and Pass [LP12a] will satisfy the required property to be instantiated in our protocol for the CCA-secure coin-tossing scheme.

We remark here that while the focus of the present work is to achieve *plain* angel-based security, we could achieve the stronger "environment-friendly" property if we instead rely on a *strongly unprovable* CCA-secure commitment scheme [CLP13] to construct our CCA-secure coin-tossing scheme. We leave this as future work.

2.1 A Formal Definition of CCA-Secure Coin-Tossing

We begin with the simpler security requirement of CCA-security w.r.t biasing oracles.

Definition 2.1 (CCA-secure coin-tossing) *Let $\langle I, R\rangle$ be a tag-based coin-tossing scheme with $l(n)$-bit identities, $m(n)$-bit outcomes and $\mathcal{O}$ a biasing oracle for it. We say that $\langle I, R\rangle$ is robust CCA-secure w.r.t. $\mathcal{O}$, if for every* PPT *adversary $\mathcal{A}$ there exists a simulator $\mathcal{S}$ such that the following distributions are indistinguishable.*

- $\{\mathcal{A}^{\mathcal{O}}(n, z)\}_{n\in\mathbb{N}, z\in\{0,1\}^*}$
- $\{\mathcal{S}(n, z)\}_{n\in\mathbb{N}, z\in\{0,1\}^*}$

2.2 CCA-Security w.r.t Challengers

Let the random variable $\mathsf{EXP}_b(\langle I, R\rangle, \mathcal{O}, \mathcal{A}, \mathcal{C}, n, z)$ denote the output of the following experiment:

1. On common input 1^n and auxiliary input z, $\mathcal{A}^{\mathcal{O}}$ chooses an identity $\mathsf{id} \in \{0,1\}^{l(n)}$ and first interacts with a honest receiver R using $\langle I, R\rangle$. Let o be the outcome of the execution.
2. Next, it interacts with $\mathcal{C}$ with common input (n, o) and private input b for $\mathcal{C}$.

Finally, the experiment outputs the view of the adversary $\mathcal{A}$ in the experiment and the output is set to $\perp$ unless $\mathcal{A}$ corrupts either $\mathcal{C}$ or I or any of the identities chosen for the interactions of $\mathcal{A}$ with $\mathcal{O}$ is equal to id. Let the random variable $\mathsf{STA}_b(\mathcal{B}, \mathcal{C}, n, z)$ denote the output of $\mathcal{B}$ in an interaction between $\mathcal{B}$ and $\mathcal{C}$ with common input (n, o) where o is uniformly sampled from $\{0,1\}^n$, private input b for $\mathcal{C}$ and auxiliary input c with $\mathcal{B}$.

Definition 2.2 (CCA-secure coin-tossing) *Let $\langle I, R\rangle$ be a tag-based coin-tossing scheme with $l(n)$-bit identities, $m(n)$-bit outcomes and $\mathcal{O}$ a biasing oracle for it. We say that $\langle I, R\rangle$ is CCA-secure w.r.t. $\mathcal{O}$ against a challenger $\mathcal{C}$, if for every* PPT *adversary $\mathcal{A}$ and distinguisher $\mathcal{D}$, if $\mathcal{D}$ distinguishes the following ensembles with non-negligible probability:*

- $\{\mathsf{EXP}_0(\langle I, R\rangle, \mathcal{O}, \mathcal{A}, \mathcal{C}, n, z)\}_{n\in\mathbb{N}, z\in\{0,1\}^*}$
- $\{\mathsf{EXP}_1(\langle I, R\rangle, \mathcal{O}, \mathcal{A}, \mathcal{C}, n, z)\}_{n\in\mathbb{N}, z\in\{0,1\}^*}$

then there exists a stand-alone adversary (that does not have access to $\mathcal{O}$) $\mathcal{B}$ and distinguisher $\mathcal{D}'$ such that $\mathcal{D}'$ distinguishes the following ensembles with non-negligible probability:

- $\{\mathsf{STA}_0(\mathcal{B}, \mathcal{C}, n, z)\}_{n\in\mathbb{N}, z\in\{0,1\}^*}$
- $\{\mathsf{STA}_1(\mathcal{B}, \mathcal{C}, n, z)\}_{n\in\mathbb{N}, z\in\{0,1\}^*}$

We highlight that in a real experiment, o is the result of the outcome of a coin-tossing between the adversary acting as the receiver and an honest initiator. However, the game between $\mathcal{B}$ and $\mathcal{C}$ is instantiated with a randomly chosen o. In essence, the definition says that if a challenge presented by $\mathcal{C}$ is hard to distinguish for a randomly sampled o, then it will be hard to distinguish even if o was sampled according to $\langle I, R\rangle$ with an adversarial receiver R who has access to oracle $\mathcal{O}$.

3 Preliminaries

Basic notations. We denote the security parameter by n. We say that a function $\mu : \mathbb{N} \to \mathbb{N}$ is *negligible* if for every positive polynomial $p(\cdot)$ and all sufficiently large n it holds that $\mu(n) < \frac{1}{p(n)}$. We use the abbreviation PPT to denote probabilistic polynomial-time. We further denote by $a \leftarrow A$ the random sampling of a from a distribution A, and by $[n]$ the set of elements $\{1, \ldots, n\}$. We specify next the definition of computationally indistinguishable.

Definition 3.1 *Let $X = \{X(a,n)\}_{a\in\{0,1\}^*, n\in\mathbb{N}}$ and $Y = \{Y(a,n)\}_{a\in\{0,1\}^*, n\in\mathbb{N}}$ be two distribution ensembles. We say that X and Y are* computationally indistinguishable*, denoted $X \overset{c}{\approx} Y$, if for every* PPT *machine D, every $a \in \{0,1\}^*$, every positive polynomial $p(\cdot)$ and all sufficiently large n:*

$$\Big|\Pr\left[D(X(a,n), 1^n) = 1\right] - \Pr\left[D(Y(a,n), 1^n) = 1\right]\Big| < \frac{1}{p(n)}.$$

3.1 Public Key Encryption Schemes (PKE)

We specify the definitions of public key encryption and IND-CPA.

Definition 3.2 (PKE) *We say that $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ is a* public key encryption scheme *if $\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}$ are polynomial-time algorithms specified as follows:*

- Gen*, given a security parameter n (in unary), outputs keys $(\mathrm{PK}, \mathrm{SK})$, where PK is a public key and SK is a secret key. We denote this by $(\mathrm{PK}, \mathrm{SK}) \leftarrow \mathsf{Gen}(1^n)$.*
- Enc*, given the public key PK and a plaintext message m, outputs a ciphertext c encrypting m. We denote this by $c \leftarrow \mathsf{Enc}_{\mathrm{PK}}(m)$; and when emphasizing the randomness r used for encryption, we denote this by $c \leftarrow \mathsf{Enc}_{\mathrm{PK}}(m; r)$.*
- Dec*, given the public key PK, secret key SK and a ciphertext c, outputs a plaintext message m s.t. there exists randomness r for which $c = \mathsf{Enc}_{\mathrm{PK}}(m; r)$ (or $\perp$ if no such message exists). We denote this by $m \leftarrow \mathsf{Dec}_{\mathrm{PK},\mathrm{SK}}(c)$.*

For a public key encryption scheme $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ and a non-uniform adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, we consider the following *IND-CPA game* denoted by $\text{ADV}_{\Pi,\mathcal{A}}(n)$:

$(\text{PK}, \text{SK}) \leftarrow \mathsf{Gen}(1^n)$.
$(m_0, m_1, history) \leftarrow \mathcal{A}_1(\text{PK})$, s.t. $|m_0| = |m_1|$.
$c \leftarrow \mathsf{Enc}_{\text{PK}}(m_b)$, where $b \leftarrow_R \{0, 1\}$.
$b' \leftarrow \mathcal{A}_2(c, history)$.
Return 1 if $b' = b$, and 0 otherwise.

Definition 3.3 (IND-CPA) *A public key encryption scheme* $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *has* indistinguishable encryptions under chosen plaintext attacks *(IND-CPA), if for every non-uniform adversary* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *there exists a negligible function* negl *such that*

$$\Pr[\text{ADV}_{\Pi,\mathcal{A}}(n) = 1] \leq \frac{1}{2} + \mathsf{negl}(n)$$

where the probability is taken over the random coins used by $\mathcal{A}$, as well as the random coins used in the experiment.

3.2 Commitment Schemes

Commitment schemes are used to enable a party, known as the *sender*, to commit itself to a value while keeping it secret from the *receiver* (this property is called *hiding*). Furthermore, in a later stage when the commitment is opened, it is guaranteed that the "opening" can yield only a single value determined in the committing phase (this property is called *binding*). In this work, we consider commitment schemes that are *statistically-binding*, namely while the hiding property only holds against computationally bounded (non-uniform) adversaries, the binding property is required to hold against unbounded adversaries. More precisely, a pair of PPT machines $\mathsf{Com} = (\text{Rec}, \text{Sen})$ is said to be a commitment scheme if the following two properties hold.

Computational hiding: For every (expected) PPT machine Rec^*, it holds that, the following ensembles are computationally indistinguishable.

- $\{\mathbf{View}^{\text{Rec}^*}_{\mathsf{Com}}(m_1, z)\}_{n\in\mathbb{N},m_1,m_2\in\{0,1\}^n,z\in\{0,1\}^*}$
- $\{\mathbf{View}^{\text{Rec}^*}_{\mathsf{Com}}(m_2, z)\}_{n\in\mathbb{N},m_1,m_2\in\{0,1\}^n,z\in\{0,1\}^*}$

where $\mathbf{View}^{\text{Rec}^*}_{\mathsf{Com}}(m, z)$ denotes the random variable describing the output of Rec^* upon interacting with the sender Sen which commits to m.

Statistical binding: Informally, the statistical-binding property asserts that, with overwhelming probability over the coin-tosses of the receiver Rec, the transcript of the interaction fully determines the value committed to by the sender.

Formally, a receiver's view of an interaction with the sender, denoted $(r, \bar{m})$, consists of the random coins used by the receiver (namely, r) and the sequence of messages received from the receiver (namely, $\bar{m}$). Let $m_1, m_2 \in \mathcal{M}_n$. We say that the receiver's view (of such interaction), $(r, \bar{m})$, is *a possible m-commitment* if there exists a string s such that $\bar{m}$ describes the messages received by Rec when Rec uses local coins r and interacts with Sen which uses local coins s and has input $(1^n, m)$.

We say that the receiver's view $(r, \bar{m})$ is *ambiguous* if is it both a possible m_1-commitment and a possible m_2-commitment. The binding property asserts that, for all but a negligible fraction of the coins toss of the receiver, there exists no sequence of messages (from the sender) which together with these coin toss forms an ambiguous receiver view. Namely, that for all but a negligible function of the $r \in \{0,1\}^{\mathsf{poly} n}$ there is no $\bar{m}$ such that $(r, \bar{m})$ is ambiguous.

We refer the reader to [Gol01] for more details.

3.3 UC Commitment Schemes

The notion of UC commitments was introduced by Canetti and Fischlin in [CF01]. The formal description of functionality $\mathcal{F}_{\rm COM}$ is depicted in Figure 5.

Functionality $\mathcal{F}_{\rm COM}$

Functionality $\mathcal{F}_{\rm COM}$ communicates with sender $\rm Sen$ and receiver $\rm Rec$, and adversary $\mathcal{S}$.

1. Upon receiving input $(\mathsf{commit}, sid, m)$ from $\rm Sen$ where $m \in \{0,1\}^t$, internally record (sid, m) and send message $(sid, \rm Sen, Rec)$ to the adversary. Upon receiving $\mathsf{approve}$ from the adversary send sid, to $\rm Rec$. Ignore subsequent $(\mathsf{commit}, ., ., .)$ messages.
2. Upon receiving (reveal, sid) from $\rm Sen$, where a tuple (sid, m) is recorded, send message m to adversary $\mathcal{S}$ and $\rm Rec$. Otherwise, ignore.

Figure 1: The string commitment functionality.

3.4 Extractable Commitments

In this section we define the notion of (static) extractable commitments which is a weaker notion than UC commitments in the sense that it does not require equivocality. Namely, the simulator cannot commit to one message and then later convince the receiver that it committed to a different value. In what follows, we introduce the definition for the ideal functionality $\mathcal{F}_{\rm EXTCOM}$ from [MPR10]. To the best of our knowledge, this is the only definition that captures straight-line extractability, statistically binding and computationally (stand-alone) hiding. Towards introducing this definition, Maji et al. introduced the following notions first.

Definition 3.4 *A protocol is a* syntactic commitment protocol *if:*

- *It is a two phase protocol between a sender and a receiver (using only plain communication channels).*
- *At the end of the first phase (commitment phase), the sender and the receiver output a transcript* trans. *Furthermore, the sender receives an output (which will be used for opening the commitment).*
- *In the decommitment phase the sender sends a message γ to the receiver, who extracts an output value* $\mathsf{opening}(\mathsf{trans}, \gamma) \in \{0,1\}^n \cup \{\perp\}$.

Definition 3.5 *Two syntactic commitment protocols (Π_L, Π_R) form a pair of complementary statistically binding commitment protocols if the following hold:*

- *Π_R is a statistically binding commitment scheme (with stand-alone security).*

- *In Π_L, at the end of the commitment phase the receiver outputs a string $z \in \{0,1\}^n$. If the receiver is honest, it is only with negligible probability that there exists γ such that* $\mathsf{opening}(\mathsf{trans}, \gamma) \neq \perp$ *and* $\mathsf{opening}(\mathsf{trans}, \gamma) \neq z$.

As noted in [MPR10], Π_L by itself is not an interesting cryptographic goal, as the sender can simply send the committed string in the clear during the commitment phase. Nevertheless, in defining $\mathcal{F}_{\text{EXTCOM}}$ below, there exists a single protocol that satisfies both the security guarantees. We are now ready to introduce the notion of extractable commitments in Figure 2 that is parameterized by (Π_L, Π_R). We additionally include a function pp that will be used as an initialization phase to set up the public parameters for Π_L and Π_R.

Functionality $\mathcal{F}_{\text{EXTCOM}}$ parameterized by $(\mathsf{pp}, \Pi_L, \Pi_R)$

$\mathcal{F}_{\text{EXTCOM}}$ is running with parties $P_1, \ldots, P_n$ and an adversary $\mathcal{S}$: Upon receiving a message $(\mathsf{init-commit}, sid, ssid, P_i, P_j)$ from P_i, it first checks if there is a tuple $(\mathsf{public-params}, sid, P_i, (pp, sp))$. If yes, it sends $(\mathsf{init-commit}, sid, ssid, P_i, P_j)$ to P_j. If not, it runs $(pp, sp) \leftarrow \mathsf{pp}(1^n)$ and sends $(\mathsf{init-commit}, sid, P_i, pp)$ to P_i, P_j and $\mathcal{S}$. It stores $(\mathsf{public-params}, sid, P_i, (pp, sp))$. We denote P_i by the sender and P_j by the receiver in this interaction. Next, the functionality behaves as follows, depending on which party is corrupted.

- P_i IS HONEST AND P_j IS HONEST.

  Commit Phase: Upon receiving $(\mathsf{commit}, sid, ssid, P_i, P_j, m)$ from P_i, it internally simulates a session of Π_R (simulating both the sender and receiver in Π_R), with the sender's input fixed to m. It gives $(\mathsf{transcript}, sid, ssid, \mathsf{trans}, \gamma)$ to P_i and $(\mathsf{receipt}, sid, ssid, P_i, P_j, \mathsf{trans})$ to P_j and $\mathcal{S}$.

  Reveal Phase: Upon receiving $(\mathsf{decommit}, sid, ssid, \cdot)$ from the sender, it sends $(\mathsf{decommit}, sid, ssid, P_i, P_j, z)$ to P_j and $\mathcal{S}$.

- P_i IS CORRUPTED AND P_j IS HONEST.

  Commit Phase: It runs the commitment Π_L with the sender, playing the part of the receiver in Π_L, to obtain $(sid, ssid, \mathsf{trans}, z)$. It sends $(\mathsf{receipt}, sid, ssid, P_i, P_j, \mathsf{trans})$ to P_j and $\mathcal{S}$.

  Reveal Phase: Upon receiving $(\mathsf{decommit}, sid, ssid, \gamma)$ from the sender, if $\mathsf{opening}(\mathsf{trans}, \gamma) = z$, it sends $(\mathsf{decommit}, sid, ssid, P_i, P_j, z)$ to P_j and $\mathcal{S}$. Otherwise ignore.

- P_i IS HONEST AND P_j IS CORRUPT.

  Commit Phase: Upon receiving $(\mathsf{commit}, sid, ssid, P_i, P_j, m)$ from P_i, it runs the commitment phase of Π_R with P_j, playing the sender's role in Π_R with m as input. It obtains the output (trans, γ) at the end of this phase, and sends $(\mathsf{transcript}, sid, ssid, \mathsf{trans}, \gamma)$ to P_i.

  Reveal Phase: Upon receiving $(\mathsf{decommit}, sid, ssid)$ from the sender it sends $(\mathsf{decommit}, sid, ssid, P_i, P_j, (\gamma, z))$ to P_j and $\mathcal{S}$.

The functionality does not do anything when both the sender and the receiver are corrupted.

Figure 2: Extractable commitment functionality.

4 UC Security

We briefly recall the Universal Composability (UC) framework, for more details we refer to [Can01].

Environment. The model of execution includes a special entity called the UC environment (or environment) $\mathcal{Z}$. The environment "manages" the whole execution: it invokes all the parties at the beginning of the execution, generates all inputs and reads all outputs, and finally produces an output for the whole concurrent execution. Intuitively, the environment models the "larger world" in which the concurrent execution takes place (e.g., for a distributed computing task over the Internet, the environment models all the other activities occurring on the Internet at the same time).

Adversarial behavior. The model of execution also includes a special entity called the adversary, that represents adversarial activities that are directly aimed at the protocol execution under consideration. We consider both *static* and *adaptive* adversaries, where static adversarial strategy implies corruptions at the onset of the protocol execution, whereas adaptive strategy implies corruptions at any point during the execution and as a function of what the attacker's view. When a party is corrupted, it shares all its tapes with the adversary and follows its instructions for all its future actions.

While honest parties only communicate with the environment through the input/output of the functions they compute, the adversary is also able to exchange messages with the environment in an arbitrary way throughout the computation.[3] Furthermore, the adversary controls the scheduling of the delivery of all messages exchanged between parties (where messages sent by the environment are directly delivered). Technically, this is modeled by letting the adversary read the outgoing message tapes of all parties and decide whether or not and when (if at all) to deliver the message to the recipient, therefore the communication is asynchronous and lossy. However, the adversary cannot insert messages and claim arbitrary sender identity. In other words, the communication is authenticated.

Protocol execution. The *execution of a protocol π with the environment $\mathcal{Z}$, adversary $\mathcal{A}$ and trusted party* $\mathcal{G}$ proceeds as follows. The environment is the first entity activated in the execution, who then activates the adversary, and invokes other honest parties. At the time an honest party is invoked, the environment assigns it a unique identifier and inquiries the adversary whether it wants to corrupt the party or not. To start an execution of the protocol π, the environment initiates a *protocol execution session*, identified by a session identifier sid, and activates all the participants in that session. An activated honest party starts executing the protocol π thereafter and has access to the trusted party $\mathcal{G}$. We remark that in the UC model the environment only initiates one protocol execution session.

Invoking parties. The environment invokes an honest party by passing input (invoke, P_i) to it. P_i is the globally unique identity for the party and is picked dynamically by the environment at the time it is invoked. Immediately after that, the environment notifies the adversary of the invocation of P_i by sending the message (invoke, P_i) to it, who can then choose to corrupt this party by replying ($\mathsf{corrupt}$, P_i). Note that here as the adversary is static, parties are corrupted only when they are "born" (invoked).

Session initiation. To start an execution of protocol π, the environment selects a subset U of parties that has been invoked so far. For each party $P_i \in U$, the environment activates P_i by sending a start-session message ($\mathsf{start\text{-}session}$, P_i, sid, $c_{i,sid}$, $x_{i,sid}$) to it, where sid is a session id that identifies this execution. We remark that in the UC model, the environment starts only one session, and hence all the activated parties have the same session id.

[3] Through its interaction with the environment, the adversary is also able to influence the inputs to honest parties indirectly.

Honest party execution. An honest party P_i, upon receiving ($\mathsf{start\text{-}session}$, P_i, sid, $c_{i,sid}$, $x_{i,sid}$), starts executing its code $c_{i,sid}$ input $x_{i,sid}$. During the execution,

- The environment can read P_i's output tape and at any time may pass additional inputs to P_i;
- According to its code, P_i can send messages (delivered by the adversary) to the other parties in the session, in the format (P_i, P_j, s, content),[4] where P_j is the identity of the receiver;
- According to its code, P_i can send an input to the trusted party in the format (P_i, $\mathcal{F}$, s, input).

Adversary execution. Upon activation, the adversary may perform one of the following activities at any time during the execution.

- The adversary can read the outgoing communication tapes of all honest parties and decides to deliver some of the messages.
- The adversary can exchange arbitrary messages with the environment.
- The adversary can read the inputs, outputs and incoming messages of a corrupted party, and instruct the corrupted party for any action.
- The (adaptive) adversary can decide to corrupt any party from the set of honest parties at the moment.

Output. The environment outputs a final result for the whole execution in the end.

In the execution of protocol π with security parameter $n \in \mathbb{N}$, environment $\mathcal{Z}$, adversary $\mathcal{A}$ and trusted party $\mathcal{G}$, we define $\mathbf{View}^{\mathcal{G}}_{\pi,\mathcal{A},\mathcal{Z}}(n)$ to be the random variable describing the output of the environment $\mathcal{Z}$, resulting from the execution of the above procedure.

Let $\mathcal{F}$ be an ideal functionality; we denote by π_{IDEAL} the protocol accessing $\mathcal{F}$, called as the ideal protocol. In π_{IDEAL} parties simply interacts with $\mathcal{F}$ with their private inputs, and receive their corresponding outputs from the functionality at the end of the computation. Then the *ideal model execution* of the functionality $\mathcal{F}$ is just the execution of the ideal protocol π_{IDEAL} with environment $\mathcal{Z}$, adversary $\mathcal{S}$ and trusted party $\mathcal{F}$. The output of the execution is thus $\mathbf{View}^{\mathcal{F}}_{\pi_{\text{IDEAL}},\mathcal{S},\mathcal{Z}}(n)$. On the other hand, the real model execution does not require the aid of any trusted party. Let π be a multi-party protocol implementing $\mathcal{F}$. Then, the *real model execution* of π is the execution of π with security parameter n, environment $\mathcal{Z}$ and adversary $\mathcal{A}$, whose output is the random variable $\mathbf{View}_{\pi,\mathcal{A},\mathcal{Z}}(n)$. Additionally, the *$\mathcal{G}$-Hybrid model execution* of a protocol π is the execution of π with security parameter n, environment $\mathcal{Z}$ and adversary $\mathcal{A}$ and ideal functionality $\mathcal{G}$.

Security as emulation of a real model execution in the ideal model. Loosely speaking, a protocol *securely realizes* an ideal functionality if it *securely emulates* the ideal protocol π_{IDEAL}. This is formulated by saying that for every adversary $\mathcal{A}$ in the real model, there exists an adversary $\mathcal{S}$ (a.k.a. *simulator*) in the ideal model, such that no environment $\mathcal{Z}$ can tell apart if it is interacting with $\mathcal{A}$ and parties running the protocol, or $\mathcal{S}$ and parties running the ideal protocol π_{IDEAL}.

Definition 4.1 (UC security) *Let $\mathcal{F}$ and π_{IDEAL} be defined as above, and π be a multi-party protocol in the $\mathcal{G}$-hybrid model. Then protocol π UC realizes $\mathcal{F}$ with static (resp. adaptive) security in $\mathcal{G}$-hybrid model, if*

[4]The session id in the messages enables the receiver to correctly de-multiplexing a message to its corresponding session, even though the receiver may involve in multiple sessions simultaneously.

for every PPT *static (resp. adaptive) adversary $\mathcal{A}$, there exists a* PPT *simulator $\mathcal{S}$, such that for every* PPT *environment $\mathcal{Z}$, the following two ensembles are computationally indistinguishable,*

$$\left\{\textbf{View}^{\mathcal{G}}_{\pi,\mathcal{A},\mathcal{Z}}(n)\right\}_{n\in\mathbb{N}} \overset{c}{\approx} \left\{\textbf{View}^{\mathcal{F}}_{\pi_{\text{IDEAL}},\mathcal{S},\mathcal{Z}}(n)\right\}_{n\in\mathbb{N}}.$$

Extended UC security (EUC). We begin with briefly recalling the definition of EUC security. The extended UC framework [CDPW07] is identical to the standard UC security with the exception that all parties have access to an additional global entity. The global entity used in the work of [CLP10] is a helper oracle that helped breaking commitments. Prior to the work of [CLP10], the global entity has been used to model a trusted setup and as such all parties made use of the global entity. Here, analogous to [CLP10], the global entity will only be used by the corrupted parties. The global entity will be referred to as the helper functionality $\mathcal{H}$ and we will refer to it as $\mathcal{H}$-EUC security. For more details, we refer the reader to [CLP10].

The universal composition theorem. Let $\mathcal{F}$ be an ideal functionality. In its general form, the composition theorem basically says that if π is a protocol that UC realizes $\mathcal{F}$ (resp., EUC realizes $\mathcal{F}$) then, for any $\mathcal{F}$-hybrid protocol ρ, we have that an execution of the composed protocol ρ^{π} "emulates" an execution of protocol ρ. That is, for any adversary $\mathcal{A}$ there exists a simulator $\mathcal{S}$ such that no environment machine $\mathcal{Z}$ can tell with non-negligible probability whether it is interacting with $\mathcal{A}$ and protocol ρ^{π} or with $\mathcal{S}$ and protocol ρ, in a UC (resp., EUC) interaction. As a corollary, we get that if protocol ρ UC realizes $\mathcal{F}$ (resp., EUC realizes $\mathcal{F}$), then so does protocol ρ^{π}.

Theorem 4.2 (Universal composition) *Let $\mathcal{F}$ be an ideal functionality. Let ρ be a $\mathcal{F}$-hybrid protocol, and let π be a protocol that UC realizes $\mathcal{F}$ (resp., EUC realizes $\mathcal{F}$). Then protocol ρ^{π} UC realizes ρ (resp., EUC-realizes ρ).*

4.1 UC Security with Super-Polynomial Helpers

We modify the definitions of UC security by giving the corrupted parties access to an external "helper" entity, in a conceptually similar way to [PS04]. This entity, denoted $\mathcal{H}$, is computationally unbounded, and can be thought of as providing the corrupted parties with some judicious help. (As we shell see, this help will be used to assist the simulator to "reverse engineering" the adversary in order to extract relevant information hidden in its communication.)

The definition uses the formalism of EUC security [CDPW07]. Specifically, we model the helper entity as an ITM that is invoked directly by the environment, and that interacts with the environment and the corrupted parties. More formally, let $\mathcal{H}$ be an ITM. An environment $\mathcal{Z}$ is called aided by $\mathcal{H}$ if: (a) $\mathcal{Z}$ invokes a single instance $\mathcal{H}$ immediately after invoking the adversary; (b) As soon as a party (i.e., an ITI) P is corrupted (i.e., P receives a corrupted message), $\mathcal{Z}$ lets $\mathcal{H}$ know of this fact; (c) $\mathcal{H}$ interacts only with the corrupted parties. Then:

Definition 4.3 *Let π and ϕ be protocols, and let $\mathcal{H}$ be a helper functionality (i.e., an ITM). We say that π H-EUC realizes ϕ with static (resp. adaptive) security if for any static (resp. adaptive) adversary $\mathcal{A}$ there exists an adversary $\mathcal{S}$ such that for any environment $\mathcal{Z}$ that's aided by $\mathcal{H}$ the following two ensembles are computationally indistinguishable,*

$$\left\{\textbf{View}_{\pi,\mathcal{A},\mathcal{Z}}(n)\right\}_{n\in\mathbb{N}} \overset{c}{\approx} \left\{\textbf{View}_{\phi,\mathcal{S},\mathcal{Z}}(n)\right\}_{n\in\mathbb{N}}.$$

The meaningfulness of relativized UC security of course depends on the particular helper ITM in use. Still, it is easy to see that if protocol π H-EUC realizes protocol ϕ where $\mathcal{H}$ obeys the above rules and runs in time $T(n)$, then π UC realizes ϕ according to a relaxed notion where the adversary $\mathcal{S}$ can run in time $\mathsf{poly}(T(n))$. As noted in the past, for many protocols and ideal functionalities, this relaxed notion of security suffices even when $T(n) = \mathsf{exp}(n)$.

Universal composition with super-polynomial helpers. The universal composition theorem generalizes naturally to the case of EUC, even with super-polynomial helper functionalities:

Theorem 4.4 (Universal composition for relativized UC) *Let $\mathcal{F}$ be an ideal functionality, let $\mathcal{H}$ be a helper functionality, let π be an $\mathcal{F}$-hybrid protocol, and let ρ be a protocol that H-EUC realizes $\mathcal{F}$. Then protocol π^ρ H-EUC realizes π.*

See [LP12a] for the proof.

5 Black-Box Adaptive UC Secure Protocols with Super-Polynomial Helpers

We consider the model of UC with super-polynomial helpers introduced in [PS04, CLP10]. Informally speaking, in this UC model, both the adversary and the environment in the real and ideal worlds have access to a super-polynomial time functionality that assists the parties. For more details, we refer the reader to [CLP10]. In the original work of [CLP10] as well as subsequent works, only static adversaries were considered. In this work, we consider the stronger adaptive adversary and obtain the following theorem in this model.

Theorem 5.1 *Assume the existence of a simulatable public-key encryption scheme. Then, for every $\epsilon > 0$ there exists a super-polynomial time helper functionality $\mathcal{H}$, such that for every well-formed functionality $\mathcal{F}$, there exists a $\tilde{O}(d_{\mathcal{F}} n^\epsilon)$-round protocol Π that $\mathcal{H}$-EUC emulates $\mathcal{F}$ where $d_{\mathcal{F}}$ is the depth of the circuit implementing the functionality $\mathcal{F}$. Furthermore, the protocol uses the underlying encryption scheme in a black-box way.*

We will rely in our proof the following two lemmas.

Lemma 5.1 *Assume the existence of a simulatable public-key encryption scheme and a T_{COIN}-round CCA-secure coin-tossing protocol. Then, there exists a super-polynomial time helper functionality $\mathcal{H}$, such that there exists a $O(T_{\text{COIN}})$-round protocol Π that $\mathcal{H}$-EUC emulates $\mathcal{F}_{\text{COM}}$ against malicious adaptive adversaries. Furthermore, the protocol uses the underlying encryption scheme in a black-box way.*

Lemma 5.2 *Assume the existence of one-way functions, the for every $\epsilon > 0$ there exists a $O(n^\epsilon)$-round CCA-secure coin-tossing scheme against malicious adaptive adversaries. Furthermore, the protocol uses the underlying primitives in a black-box way.*

First, we prove the theorem assuming the lemmas hold and then prove the lemmas in the following sections. Towards this, we first describe our helper functionality $\mathcal{H}$. The biasing oracle for the CCA-secure coin-tossing scheme provided in Lemma 5.2 will serve as $\mathcal{H}$. This in turn relies on Lin and Pass construction from [LP12a] of a $\tilde{O}(n^\epsilon)$-round black-box construction of a CCA-secure commitment scheme based on one-way functions. Since one-way functions can be constructed from a simulatable public-key encryption scheme in a black-box way, combining [LP12a] with Lemmas 5.1 and 5.2 we have a $O(n^\epsilon)$-round protocol that $\mathcal{H}$-EUC that emulates $\mathcal{F}_{\text{COM}}$. We conclude the proof of the theorem by combining the following three results:

1. The work of Choi et al. [CDMW09] provides a $O(T_{\mathrm{OT}})$-round construction that realizes $\mathcal{F}_{\mathrm{OT}}$ in the $\mathcal{F}_{\mathrm{COM}}$-hybrid assuming the existence of a T_{OT}-round stand-alone adaptively-secure semi-honest oblivious-transfer protocol where the underlying protocol is used in a black-box way.
2. The work of Damgard and Nielsen [DN00] provides a black-box construction of a $O(1)$-round stand-alone adaptively-secure semi-honest oblivious-transfer protocol assuming the existence of simulatable public-key encryption schemes.
3. The work of Ishai et al. [IPS08] provides a $O(d_{\mathcal{F}})$-round protocol that realizes any well-formed functionality $\mathcal{F}$ in the $\mathcal{F}_{\mathrm{OT}}$-hybrid, where $d_{\mathcal{F}}$ is the depth of the circuit implementing functionality $\mathcal{F}$.

We rely on the $O(n^{\epsilon})$ construction of CCA-secure commitment of Lin and Pass [LP12a] instead of the more round efficient construction of Kiyoshima [Kiy14] because we additionally need to prove that the commitment is secure in the presence of adaptive adversaries and we are able to achieve this only for the [LP12a] construction (See Theorem A.3 in Appendix A). We leave it as future work to improve it with respect to the [Kiy14] construction.

6 CCA-Secure Coin-Tossing from CCA-Secure Commitments

In this section, we provide our construction of CCA-secure coin-tossing protocol $\langle I, R\rangle$. The two primitives we will require are CCA-secure commitments and one-way functions. Recall that, standard CCA-secure commitments require that a value committed to, using a tag id, remains hidden even to an adversary who has access to a "decommitment oracle". We will additionally require that if we consider an adversary that can adaptively corrupt receivers in its interactions with the decommitment oracle, the value committed to the adversary is hidden as long as the committer in this interaction is not corrupted. We show that the CCA-secure commitment scheme of [LP12a] satisfies this guarantee in Appendix A. More formally, let $\langle C, R\rangle$ be a CCA-secure commitment scheme and Com be a statistically-binding commitment scheme with pseudorandom commitments. For instance, the 2-round commitment scheme of Naor [Nao91] based on one-way function satisfies this notion. Next, we prove that the scheme from Figure 3 is CCA-secure and CCA-secure against challengers.

Proof overview. Our starting point is the protocol of [HV15] who constructs an adaptive UC-commitment protocol in the CRS model relying on a public-key encryption scheme that has pseudorandom ciphertexts (i.e. the ciphertexts are pseudorandom). We briefly describe this protocol next.

Roughly speaking, in their protocol, the CRS contains two public-keys of the encryption scheme, one used by the sender and the other by the receiver and proceeds in two phases: An *input encoding phase*, where the sender encodes its input via a n-out-of-$(2n+1)$ Shamir secret sharing scheme and commits to them in a specific way followed by a *cut-and-choose phase* where the receiver asks the sender to reveal n of the shares. In slight more detail, in the input encoding phase, the sender encodes its message m via n-degree polynomial $p(\cdot)$ such that $p(0) = m$ and commits to $p(1), \ldots, p(2n+1)$ as follows: For each i, it sends two strings, one is a ciphertext containing an encryption of $p(i)$ under the public-key in the CRS meant for the sender and the other is a random string. Furthermore, the sender randomly decides which one of the two strings is the encryption of $p(i)$. In the cut-and-choose phase, the parties engage in a coin-toss where the receiver first encrypts its share for the coin-tossing using the receiver public-key from the CRS, followed by the sender providing its share of the coin-tossing. Then the receiver opens its share and the result of the coin-tossing decided by the XOR of the shares determines a subset Γ of $[2n+1]$ of size n. The sender reveals the $p(i)$ for every $i \in \Gamma$ and the randomness used for generating the ciphertext. In the decommitment phase, the sender reveals the entire randomness used for the encoding phase.

Straight-line equivocation is achieved by considering encodings of both 0 and 1 and for each i, via polynomials $p(\cdot)$ and $q(\cdot)$, such that they agree on n points randomly chosen from $\{1, \ldots, 2n+1\}$, call this set Γ^*. Then the simulator encodes by having one ciphertext with the value $p(i)$ and the other $q(i)$. Finally, the simulator biases the coin-tossing outcome so that the outcome results is the set Γ^*. This can be achieved in a straight-line manner as the simulator will possess the secret key corresponding to the receiver public-key in the CRS. Since the receiver encrypts its share first in the coin-tossing, the simulator can extract the value and send the sender's share accordingly to the biased outcome. Finally, since p and q agree on this set Γ^*, in the decommitment phase, the sender will be able to open either $p(\cdot)$ or $q(\cdot)$, depending on what the message is. Furthermore, we require one of the two strings in each coordinate to be random and this can be faked as the encryption scheme has pseudorandom ciphertexts.

Straight-line extraction on the other hand requires an information theoretic lemma which states that there exists a unique set Γ^* after the encoding phase that the sender needs as the outcome of the coin-tossing in the cut-and-choose phases for it to equivocate. First, using the semantic-security of the receiver public-key we show that the probability that the sender can bias the coin-tossing is negligible. Then we show that the simulator can extract the message of the sender by using the secret key corresponding to the sender public-key used by the sender in the encoding phase. This it will accomplish by using the n values that have been revealed and finding a consistent polynomial for the remaining shares (See [HV15] for more details).

Finally, we remark that this protocol also tolerates adaptive corruptions and this essentially follows from the pseudorandomness of the ciphertexts.

Now, we move on to our protocol. On a high-level, our coin-tossing protocol will be the 4-round Blum coin-tossing protocol [Blu86] where the initiator makes the first move committing to its share using a variant of the [HV15] protocol followed by the receiver committing to its share using a CCA-secure commitment scheme. Then both parties decommit their shares and compute the XOR to obtain the outcome. To argue security it suffices for the initiator's commitment to be straight-line equivocable and the receiver's commitment straight-line extractable. CCA-secure commitment schemes with their associated oracle exactly provide this, where we can extract the receiver's commitment using the oracle. We modify the [HV15] commitment protocol to depend only on one-way functions and is only straight-line equivocable (i.e., our variant looses straight-line extraction). In [HV15], recall that the sender encodes its message using the encryption scheme and the receiver encodes its share for the coin-tossing subprotocol. In our variant, first we will replace the sender's encoding algorithm with the Naor commitment that has pseudorandom ciphertexts. Observe that this will satisfy all the property required except for extraction. Next, we will make the receiver send its share by committing to it via a CCA-secure commitment. This variant is straight-line equivocable because the simulator can bias the coin-toss outcome (to be the specific set Γ^*) by using the committed-value oracle to extract the receiver's share. The scheme is not straight-line extractable but is binding using the same information-theoretic lemma from [HV15].

Finally, to argue security against adaptive corruptions we will rely on the adaptive security guarantee of the [HV15] protocol and extend the CCA-security of the commitment schemes to allow adaptive corruptions. However, we will require the actual hiding property of the CCA-secure commitment scheme in the indistinguishability game to hold only when the sender in the game is not corrupted. In essence, we will set the outcome of the games to $\perp$ if the adversary corrupts the sender in between the games. See Appendix A for more details.

Theorem 6.1 *Suppose, $\langle C, R\rangle$ is a 0-robust CCA secure commitment scheme in the presence of adaptive adversaries. Then there exists an oracle helper $\mathcal{O}$ such that $\langle I, R\rangle$ is a CCA-secure coin tossing protocol*

Protocol $\pi_{\text{COIN}} = \langle I, R \rangle$**.**

Let 1^n be the common input to the initiator I and receiver R and the identity of the interaction $\mathsf{id} \in \{0,1\}^{l(n)}$.

Stage 1: Commit Phase: The receiver sends the first message σ of the Naor's commitment scheme. The initiator first picks a random bit m and chooses a random n-degree polynomial $p(\cdot)$ over a field $\mathbb{F}[x]$ such that $p(0) = m$. Namely, it randomly chooses $a_i \leftarrow \mathbb{F}$ for all $i \in [n]$ and sets $a_0 = m$, and defines the polynomial $p(x) = a_0 + a_1 x + \cdots + a_n x^n$. The initiator then creates a commitment to m as follows. For every $i = [3n+1]$, it first picks $b_i \leftarrow \{0,1\}$ at random and then computes:

$$c_i^{b_i} = \mathsf{Com}_\sigma(p(i); t_i) \text{ and } c_i^{1-b_i} = r_i$$

where $r_i, t_i \leftarrow \{0,1\}^n$. The initiator sends $(c_0^0, c_0^1), \ldots, (c_{3n+1}^0, c_{3n+1}^1)$ to the receiver.

Stage 2: Cut-and-Choose Phase: The initiator and receiver interact in a coin-tossing protocol to obtain the cut-and-choose set that is carried out as follows.

1. The receiver chooses a random σ_0 and commits to the initiator using $\langle C, R \rangle$ using identity id.
2. The initiator picks $\sigma_1 \leftarrow \{0,1\}^N$ at random and sends it in the clear to the receiver.
3. The receiver decommits r_{σ_0}.

Both the initiator and the receiver compute $\sigma = \sigma_0 \oplus \sigma_1$ and use σ as the random string to sample a random subset $\Gamma \subset [3n+1]$ of size n.The initiator provides the decommitments for $\{c_i^{b_i}\}_{i \in \Gamma}$ by sending the sequence $\{b_i, p(i), t_i\}_{i \in \Gamma}$. The receiver verifies that all the decommitments are correct and aborts otherwise.

Stage 3: Coin-Toss Phase: In the first two stages, the initiator essentially commits to the string m. Next they continue with the coin-tossing protocol.

1. The receiver commits to m' using $\langle C, R \rangle$.
2. This is followed by the initiator revealing its input m as follows: Let $\Gamma' = [3n+1] - \Gamma$. The initiator decommits $\{c_i^{b_i}\}_{i \in \Gamma'}$ to their respective messages. The receiver checks if the decommitments are correct and aborts otherwise. Using the n polynomial evaluations revealed relative to $i \in \Gamma$ and any additional polynomial evaluation that was revealed relative to Γ', the receiver reconstructs the polynomial $p(\cdot)$ (via polynomial interpolation of $n+1$ points). Next, the receiver verifies whether $p(0) = m$, and that for every $i \in [3n+1]$ the point $p(i)$ is the decrypted value within $c_i^{m_i}$.
3. The receiver decommits m'

Finally, the outcome of the coin-tossing is $m \oplus m'$. More formally, $\mathsf{out}(\tau)$ where τ is the transcript of this protocol is set to $m \oplus m'$.

Figure 3: Our CCA-secure coin-tossing protocol $\langle I, R \rangle$.

w.r.t $\mathcal{O}$.

Proof: To demonstrate that our scheme is CCA-secure, we construct a biasing oracle $\mathcal{O}$ and show that given any PPT adversary $\mathcal{A}$, there exists a PPT simulator $\mathcal{S}$ such that:

$$\{\mathcal{A}^{\mathcal{O}}(n,z)\}_{n \in \mathbb{N}, z \in \{0,1\}^*} \approx \{\mathcal{S}(n,z)\}_{n \in \mathbb{N}, z \in \{0,1\}^*}.$$

Description of biasing oracle $\mathcal{O}$: For the protocol described in Figure 1, our biasing oracle $\mathcal{O}$ on input c proceeds as follows:

- In Stage 1, picks a random subset $\widetilde{\Gamma} \subset [3n+1]$ of size n and two random n-degree polynomials $p_0(\cdot)$ and $p_1(\cdot)$ such that p_0 and p_1 agree on all points $i \in \widetilde{\Gamma}$ and $p_0(0) = 0$ and $p_1(0) = 1$.
  - For every $i \in \widetilde{\Gamma}$ the simulator proceeds as the honest sender would with polynomial $p_0(\cdot)$. Namely, it first picks $b_i \leftarrow \{0,1\}$ at random and then sets
$$c_i^{b_i} = \mathsf{Com}(p_0(i); t_i) \text{ and } c_i^{1-b_i} = r_i$$
where $t_i \leftarrow \{0,1\}^n$ (we recall that $p_0(i) = p_1(i)$ for all $i \in \widetilde{\Gamma}$).
  - For every $i \in \widetilde{\Gamma}' = [3n+1] - \widetilde{\Gamma}$, the simulator picks $b_i \leftarrow \{0,1\}$ at random and then sets
$$c_i^{b_i} = \mathsf{Com}(p_0(i); t_i^0) \text{ and } c_i^{1-b_i} = \mathsf{Com}(p_1(i); t_i^1)$$
where $t_i^0, t_i^1 \leftarrow \{0,1\}^n$ are chosen uniformly at random.

  Finally, it sends $(c_0^0, c_0^1), \ldots, (c_{3n+1}^0, c_{3n+1}^1)$ as the Stage 1 message of the initiator.

- In Stage 2, it breaks the commitment made using $\langle C, R\rangle$ and obtains the decommitted value σ_0. Next, it sets σ_1 so that $\sigma = \sigma_1 \oplus \sigma_0$ yields the set $\widetilde{\Gamma}$ as the outcome in Stage 2.

- In Stage 3, it breaks the commitment made using $\langle C, R\rangle$ and obtains m'. Then it decommits to $m = c \oplus m'$ using the following strategy. Recall first as the initiator it needs to reveal points on a polynomial $p(\cdot)$ and pairs $\{(b_i, t_i)\}_{i \in [3n+1]}$ such that $p(0) = m$ and $c_i^{b_i} = \mathsf{Com}(p(i); t_i)$. Let $\hat{b}_i = b_i \oplus m$ for all $i \in \widetilde{\Gamma}'$, then $\mathcal{S}$ reveals $p_m(\cdot)$, $\{\hat{b}_i, t_i^{\hat{b}_i}, r_i = c_i^{1-m}\}_{i \in \widetilde{\Gamma}'}$.

Figure 4: Biasing oracle $\mathcal{O}$.

We provide the description of our biasing oracle $\mathcal{O}$ in Figure 2. On a high-level, this oracle follows the equivocation strategy analogous to the simulation in [HV15]. In slight more detail, this protocol that is a variant of the protocol in [HV15] allows for the initiator to equivocate m in Stage 3 if for a chosen set S at the beginning of the execution, the outcome of the coin-tossing in Stage 2 can be biased to yield S. Our oracle $\mathcal{O}$ will be able to accomplish this by breaking the commitment made by the receiver R in Stage 2 using $\langle C, R\rangle$ in exponential time. Next, given an adversary $\mathcal{A}$, we construct a simulator $\mathcal{S}$ in two steps:

Step 1: Suppose $\mathcal{O}'$ is the oracle w.r.t which $\langle C, R\rangle$ is 0-robust. From the description of our oracle $\mathcal{O}$, it follows that every query to $\mathcal{O}$ can be simulated by a PPT algorithm with access to $\mathcal{O}'$. Recall that the only super-polynomial computation made by $\mathcal{O}$ is breaking a commitment made using $\langle C, R\rangle$, which can be done using $\mathcal{O}'$.[5] Therefore, given any adversary $\mathcal{A}$, there exists another oracle adversary $\widehat{\mathcal{A}}$ such that the following distributions are identically distributed:

- $\{\mathcal{A}^{\mathcal{O}}(n, z)\}_{n \in \mathbb{N}, z \in \{0,1\}^*}$

[5]We remark here that typical CCA-secure commitment schemes are statistically binding and such schemes can be easily broken in exponential time. However, the CCA-secure commitment of [LP12a] is not statistically binding. Yet, as shown in [LP12a] it is "strongly" computationally binding which will suffice.

- $\{\widehat{\mathcal{A}}^{\mathcal{O}'}(n,z)\}_{n\in\mathbb{N},z\in\{0,1\}^*}$

Step 2: Relying on the 0-robustness CCA-security of the $\langle C,R\rangle$ commitment scheme, it follows that given $\widehat{\mathcal{A}}$, there exists a simulator $\mathcal{S}$ such that the following distributions are indistinguishable.

- $\{\widehat{\mathcal{A}}^{\mathcal{O}'}(n,z)\}_{n\in\mathbb{N},z\in\{0,1\}^*}$
- $\{\mathcal{S}(n,z)\}_{n\in\mathbb{N},z\in\{0,1\}^*}$

The statement of the theorem now follows using a standard hybrid argument. ■

■

Next, we proceed to show the stronger security-preserving property of our scheme.

Theorem 6.2 *Suppose, $\langle C,R\rangle$ is a k-robust CCA-secure commitment scheme in the presence of adaptive adversaries. Then for every k-message* PPT *$\mathcal{C}$, $\langle I,R\rangle$ is a CCA-secure coin-tossing scheme w.r.t. the biasing oracle $\mathcal{O}$ against $\mathcal{C}$.*

Proof: Assume for contradiction there exist an adversary $\mathcal{A}$, sequence $\{z_n\}_{n\in\mathbb{N}}$ and distinguisher $\mathcal{D}$ such that $\mathcal{D}$ distinguishes the following ensembles

- $\{\mathsf{EXP}_0(\langle I,R\rangle,\mathcal{O},\mathcal{A},\mathcal{C},n,z_n)\}_{n\in\mathbb{N}}$
- $\{\mathsf{EXP}_1(\langle I,R\rangle,\mathcal{O},\mathcal{A},\mathcal{C},n,z_n)\}_{n\in\mathbb{N}}$

with non-negligible probability. Namely, it distinguishes with probability $p(n)$ for some polynomial $p(\cdot)$ and infinitely many n's. We need to construct a machine $\mathcal{B}$ and distinguisher $\mathcal{D}'$ that will distinguish STA_0 from STA_1. Let $\mathcal{O}'$ be the committed-value oracle guaranteed by the k-robust CCA-security of $\langle C,R\rangle$ in the presence of adaptive adversaries. We will accomplish our goal of constructing $\mathcal{B}$ in two steps.

Step 1: First we construct a simulator $\widetilde{\mathcal{S}}$ such that the following distributions are distinguishable with non-negligible probability.

- $\{\mathsf{STA}_0(\widetilde{\mathcal{S}}^{\mathcal{O}'},\mathcal{C},n,z)\}_{n\in\mathbb{N},z\in\{0,1\}^*}$
- $\{\mathsf{STA}_1(\widetilde{\mathcal{S}}^{\mathcal{O}'},\mathcal{C},n,z)\}_{n\in\mathbb{N},z\in\{0,1\}^*}$

Step 2: Since $\mathcal{C}$ interacts in at most k-messages, we obtain the required $\mathcal{B}$ directly by relying on the k-robustness of the CCA-security of $\langle C,R\rangle$ in the presence of an adaptive adversary.

Step 1: Constructing $\widetilde{\mathcal{S}}^{\mathcal{O}'}$. Fix an n for which $\mathcal{D}$ distinguishes the two ensembles with probability $p = p(n)$. Recall that in the EXP experiment, $\mathcal{A}$ first interacts with an external R and then interacts with $\mathcal{C}_b$.

In a random instance of the EXP_b experiment, let T be the random variable representing the partial transcript up until the end of Stage 1 in $\mathcal{A}$'s interaction with external R. Next, we consider the modified experiment $\widetilde{\mathsf{EXP}}_b^T$ which starts from the partial transcript T and proceeds identically to the EXP_b. Namely, we condition on the partial transcript T.[6]

Now, using an averaging argument, we can conclude that with probability at least $p/2$ over partial transcript $\tau_n \leftarrow T$ it holds that $\mathcal{D}$ distinguishes the following two ensembles with probability at least $p/2$.

[6]This can be achieved by instantiating the adversary with the same random coins and feeding the messages from T, and then running the rest of the experiment with fresh randomness.

- $\{\widetilde{\mathsf{EXP}}_0^{\tau_n}(\langle I, R\rangle, \mathcal{O}, \mathcal{A}, \mathcal{C}, n, z_n)\}_{n\in\mathbb{N}}$
- $\{\widetilde{\mathsf{EXP}}_0^{\tau_n}(\langle I, R\rangle, \mathcal{O}, \mathcal{A}, \mathcal{C}, n, z_n)\}_{n\in\mathbb{N}}$

Now, we are ready to construct $\widetilde{\mathcal{S}}$. The high-level approach is as follows:

- First, we show that, except with non-negligible probability over random executions starting from τ_n, there is a fixed value m_n that the adversary will decommit to in the Stage 3 of its interaction with R. We will rely on an information theoretic lemma from [HV15] for this. We state this step in the Claim 6.1 below.

  Claim 6.1 *There exists a string m_n such that, starting from partial transcript τ_n, the probability that $\mathcal{A}$ successfully decommits to a message different from m_n in Stage 3 is negligible.*

  On a high-level the idea is that given the transcript until end of Stage 1, there is a unique set S that needs to be the outcome of Stage 2 in order for the an initiator to equivocate in Stage 3. This means that if an adversarial initiator can equivocate with non-negligible probability, then it has to bias the coin-tossing in Stage 2 to yield this unique set S with non-negligible probability. Such an adversary can then be exploited in order to violate the CCA-security of the commitment made using $\langle C, R\rangle$ in Stage 2. We provide a formal proof of this claim at the end of this section.

- Next, for a fixed transcript τ_n, we will give τ_n, m_n and partial view of $\mathcal{A}$ in the execution as the non-uniform advice. Our simulator $\widetilde{\mathcal{S}}$ will start an execution with $\mathcal{A}$ from the partial view with transcript τ_n and will use m_n to bias the outcome of the coin-tossing to o by setting $m' = m_n \oplus o$ in Stage 3 of the execution. Now, we observe that, if o is uniformly distributed, then m' chosen by $\widetilde{\mathcal{S}}$ will also be (non-negligibly) close to the uniform distribution given m_n and hence the view of $\mathcal{S}$ output with $\mathcal{C}_b$ will be statistically close to the distribution of $\mathcal{A}$ when interacting with $\mathcal{C}_b$ starting from τ_n. This means that if $\mathcal{D}$ distinguishes the view of $\mathcal{A}$ starting from τ_n in both the experiments, then it will also distinguish the output of $\widetilde{\mathcal{S}}^{\mathcal{O}'}$ in the two experiments.

We now construct our simulation $\widetilde{\mathcal{S}}$. On input $(1^n, o, (z, \tau_n, m_n, r_n))$, $\widetilde{\mathcal{S}}^{\mathcal{O}'}$ internally emulates an execution of $\mathcal{A}(1^n, z; r)$ in the real experiment starting from the partial transcript τ_n. On the left, $\widetilde{\mathcal{S}}^{\mathcal{O}'}$ needs to provide messages for the initiator I such that the outcome is o while simultaneously answering all oracle queries to $\mathcal{O}$. This it accomplished by committing to $m' = o \oplus m_n$ in Stage 3. Then if the adversary reveals anything other than the m_n, it simply aborts.

Answering $\mathcal{O}$ queries. In any interaction, the oracle $\mathcal{O}$ first receives a coin c. In the internal emulation $\widetilde{\mathcal{S}}^{\mathcal{O}'}$ obtains c and needs to emulate $\mathcal{O}$. It carries out the actions exactly as $\mathcal{O}$ with the exception that instead of breaking the commitments made using $\langle C, R\rangle$ (as $\mathcal{O}$ does) $\widetilde{\mathcal{S}}^{\mathcal{O}'}$ simply forwards it to $\mathcal{O}'$ which breaks them for $\mathcal{S}$.

It follows from the construction and Claim 6.1 that the following distributions are statistically close:

- $\{\widetilde{\mathsf{EXP}}_b^{\tau_n}(\langle I, R\rangle, \mathcal{O}, \mathcal{A}, \mathcal{C}, n, z_n)\}_{n\in\mathbb{N}}$
- $\{\mathsf{STA}_b(\widetilde{\mathcal{S}}^{\mathcal{O}'}, \mathcal{C}, n, (z_n, \tau_n, m_n, r_n))\}_{n\in\mathbb{N}, z\in\{0,1\}^*}$

and therefore $\mathcal{D}$ distinguishes the distribution $\mathsf{STA}_0(\widetilde{\mathcal{S}}^{\mathcal{O}'}, \mathcal{C}, n, (z_n, \tau_n, m_n, r_n))$ from $\mathsf{STA}_1(\widetilde{\mathcal{S}}^{\mathcal{O}'}, \mathcal{C}, n, (z_n, \tau_n, m_n, r_n))$ with with probability at least $p/2 - \nu(n) > p/4$ for all sufficiently large n's.

Step 2: Constructing a stand-alone $\mathcal{B}$. In Step 1, we constructed a machine $\widetilde{\mathcal{S}}^{\mathcal{O}'}$ that with access to $\mathcal{O}'$ can violate the game. Next, in order to obtain a stand-alone $\mathcal{B}$, we rely on the k-robustness property of $\langle C, R\rangle$ with $\widetilde{\mathcal{S}}^{\mathcal{O}'}$ according to Definition A.2. More precisely, using the robustness we have that the following distributions are computationally indistinguishable:

- $\{\mathsf{STA}_b(\widetilde{\mathcal{S}}^{\mathcal{O}'}, \mathcal{C}, n, (z_n, \tau_n, m_n, r_n))\}_{n\in\mathbb{N}, z\in\{0,1\}^*}$
- $\{\mathsf{STA}_b(\mathcal{B}, \mathcal{C}, n, (z_n, \tau_n, m_n, r_n))\}_{n\in\mathbb{N}, z\in\{0,1\}^*}$

and therefore $\mathcal{D}$ distinguishes the distribution $\mathsf{STA}_0(\mathcal{B}, \mathcal{C}, n, (z_n, \tau_n, m_n, r_n))$ from $\mathsf{STA}_1(\mathcal{B}, \mathcal{C}, n, (z_n, \tau_n, m_n, r_n))$ with probability at least $p/4 - \nu(n) > p/8$ for all sufficiently large n's and this completes the proof of the theorem.

To conclude the proof of Theorem 6.2, it only remains to prove Claim 6.1.

Proof of Claim 6.1. Assume for contradiction, the adversary $\mathcal{A}$ equivocates with non-negligible probability starting from τ_n. We now show that $\mathcal{A}^{\mathcal{O}'}$ violates the CCA-security of $\langle C, R\rangle$ w.r.t $\mathcal{O}'$, namely, it violates the hiding property of the commitment made using $\langle C, R\rangle$ in Stage 2.

As stated above, we use an information theoretic lemma from [HV15]. On a high-level, the lemma states that for the adversary to be able to equivocate in Stage 3, there exists a unique set S that it must bias the outcome of the coin-tossing in Stage 2 so that the resulting set is S. On a high-level, we can rely on this lemma, as a malicious initiator that equivocates must bias the outcome to a particular set S and using the set S. Then, we can construct an adversary $\widehat{\mathcal{A}}^{\mathcal{O}'}$ that violates the CCA-game for $\langle C, R\rangle$ by simply detecting this set S in the outcome of Stage 2.

More formally, given τ_n, and a partial view of $\mathcal{A}$, let us assume that $\mathcal{A}$ equivocates with probability $\frac{1}{q(n)}$ for some polynomial $q(\cdot)$ and infinitely many n.

Before we recall the information theoretic lemma from [HV15], we first explain how our protocol is an instance of the protocol in their work. In [HV15], they construct an adaptively secure UC-commitment in the CRS hybrid where the protocol proceeds as follows:

1. In Stage 1, the committer using the same strategy as the initiator in our Stage 1 commits to a string m, where instead of using Com_σ, it uses an encryption scheme with oblivious ciphertext generation property (where the public-key for this scheme is placed in the CRS).
2. In Stage 2, the committer and receiver execute a coin-tossing protocol where the receiver makes the first move just as in $\langle I, R\rangle$ with the exception that the receiver in the their protocol uses again an encryption scheme (with the public-key in the CRS) instead of a commitment scheme to commit to σ_0.
3. In the decommitment phase of their protocol, the committer reveals its commitment just as the initiator does in Step 2 of Stage 3 in our protocol.

We remark that in essence, the protocol in [HV15] is used as a subprotocol in our work here where the initiator commits to a string m and then reveals it. The only property they need of the encryption scheme is that it is statistically binding[7] and has the oblivious generation property. In our protocol, the Naor commitment scheme has both these properties. (See our next protocol for such a variant)

[7]Any commitment can either be revealed a encryption of a unique string x or as a ciphertext that was obliviously generated. In particular, it cannot be revealed as a an encryption of two distinct messages x and x'.

Claim 6.2 **(Restatement of Claim 5.5 [HV15])** *Let τ be a fixed partial transcript up until end of Stage 1. Then, except with negligible probability, there exists no two transcripts* $\mathsf{trans}_1, \mathsf{trans}_2$ *that satisfy the following conditions:*

1. trans_1 *and* trans_2 *are complete and accepting transcripts of* π_{COM} *with* τ *being their prefix.*
2. *There exists two distinct sets* S_1, S_2 *such that* S_1 *and* S_2 *are the respective outcomes of the coin-tossing phase within* trans_1 *and* trans_2.
3. *There exist valid decommitments to two distinct strings in* trans_1 *and* trans_2.

Since the commitment made by our Initiator can be viewed as an instance of their protocol, we can conclude that there exists a unique set S that should be the outcome of the coin-tossing in Stage 2 for a malicious initiator to equivocate m. Since $\mathcal{A}$ equivocates with probability $\frac{1}{q(n)}$ it holds, there is a set S such that with the probability negligible close[8] to $\frac{1}{q(n)}$, starting from τ_n, the outcome of Stage 2 is S. To construct an adversary $\widehat{\mathcal{A}}$ that violates the CCA-security of the underling $\langle C, R\rangle$ scheme, we simply incorporate $\mathcal{A}$ and use as auxiliary input τ_n, S and the partial view of $\mathcal{A}$. Next, it forwards the $\langle C, R\rangle$ interaction in Stage 2 to an external committer. All queries to the helper oracle $\mathcal{O}$ by $\mathcal{A}$ can be simulated using $\mathcal{H}$ and $\widehat{\mathcal{A}}$ simply uses $\mathcal{H}$ to emulate $\mathcal{O}$. Then it halts the execution right after the adversary in the internal emulation reveals σ_1. Now, $\widehat{\mathcal{A}}$ simply outputs $\sigma_0 = \sigma \oplus \sigma_1$ where σ is the string that maps to the set S. This violates the CCA game as with probability close to $\frac{1}{q(n)}$, $\widehat{\mathcal{A}}$ identifies the message committed using $\langle C, R\rangle$. □ ■

■

7 Realizing $\mathcal{F}_{\text{COM}}$ Using CCA-Secure Coin-Tossing

In this section, we provide our black-box construction of $\mathcal{H}$-EUC secure protocol Π_{COM}. Our protocol is a variant of the protocol described in [HV15] where it is shown how to realize $\mathcal{F}_{\text{COM}}$ in the CRS model assuming only public-key encryption that admits oblivious-ciphertext generation with adaptive UC-security. While the [HV15] protocol assumes that every pair of parties share an independently generated CRS, in this work we assume no setup, but will require the stronger simulatable public-key encryption scheme. Assume that $\langle I, R\rangle$ is a CCA-secure coin-tossing scheme and that the public-key encryption scheme $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ is augmented with algorithms $(\mathsf{oGen}, \mathsf{oRndEnc}, \mathsf{rGen}, \mathsf{rRndEnc})$ which implies a simulatable public-key encryption scheme. Then we start with a formal description of our protocol.

Consider a helper functionality $\mathcal{H}$ that "biases" the coin-tossing in an interaction using $\langle I, R\rangle$ in the same way as the biasing oracle $\mathcal{O}$ does, subject to the condition that player P_i in a protocol instance sid can only query the functionality on interactions that use identity (P_i, sid). More precisely, every party P_i can simultaneously engage with $\mathcal{H}$ in multiple sessions of $\langle I, R\rangle$ as an initiator using identity P_i where the functionality simply forwards all the messages internally to the biasing oracle $\mathcal{O}$, and ensures that the result of the coin-tossing is biased to a prescribed outcome at the end of each session. See Figure 5 for a formal description of the functionality. We note here that since $\mathcal{O}$ can be implemented in super-polynomial time, this functionality can also be implemented in super-polynomial time.

We proceed with a proof sketch of Lemma 5.1 before we proceed to a formal proof.

[8]The probability is not identically equal to $\frac{1}{q(n)}$ since the commitment scheme is only statistically binding and not perfectly binding.

Functionality $\mathcal{H}$

Initialization: Upon receiving an input $(\mathsf{Init}, P_i, sid, k)$ from party P_i in the protocol instance sid, if there is a previously recorded session (P_i, sid, k), ignore this message; otherwise, initialize a session of $\langle I, R\rangle$ with $\mathcal{O}$ using identity (P_i, sid) and record session (P_i, sid, k).
Accessing O: Upon receiving an input $(\mathsf{Mesg}, P_i, sid, k, m)$ from party P_i in the protocol instance sid, if there is no previously recorded session (P_i, sid, k), ignore the message; otherwise, forward m to $\mathcal{O}$ in the k^{th} session that uses identity (P_i, sid), obtain a reply m', and return $(\mathsf{Mesg}, P_i, sid, k, m')$ to P_i.

Figure 5: The Helper Functionality $\mathcal{H}$ (i.e. angel).

Protocol π_{COM}.

Sender's Input: A message $m \in \{0,1\}$ and a security parameter 1^n.

Commitment Phase:

Stage 1: Key Generations Phase: The sender and receiver engage in a protocol using $\langle I, R\rangle$ where the receiver acts as the initiator I and the sender acts as R. Let $\mathsf{PK} = \mathsf{oGen}(\mathsf{out}(\tau_{S\to R}))$ where $\tau_{S\to R}$ is the transcript of the interaction.

Stage 2: Input Encoding Phase: The sender chooses a random n-degree polynomial $p(\cdot)$ over a field $\mathbb{F}[x]$ such that $p(0) = m$. Namely, it randomly chooses $a_i \leftarrow \mathbb{F}$ for all $i \in [n]$ and sets $a_0 = m$, and defines the polynomial $p(x) = a_0 + a_1 x + \cdots + a_n x^n$. The sender then creates a commitment to m as follows. For every $i = [3n+1]$, it first pick $b_i \leftarrow \{0,1\}$ at random and then computes:

$$c_i^{b_i} = \mathsf{Enc}_{\mathsf{PK}}(p_0(i); t_i) \text{ and } c_i^{1-b_i} = \mathsf{oRndEnc}(\mathsf{PK}, r_i)$$

where $r_i, t_i \leftarrow \{0,1\}^n$. The sender sends $(c_0^0, c_0^1), \ldots, (c_{3n+1}^0, c_{3n+1}^1)$ to the receiver.

Stage 3: Cut-and-choose Phase: The sender and receiver engage in a protocol using $\langle I, R\rangle$ where the sender acts as the initiator I and the receiver acts as R . Define a subset $\Gamma \subset [3n+1]$ of size n using the outcome $\mathsf{out}(\tau_{R\to S})$) where $\tau_{R\to S}$ is the transcript of the interaction. The sender provides the plaintexts encrypted in $\{c_i^{b_i}\}_{i\in\Gamma}$ by sending the sequence $\{b_i, p(i), t_i\}_{i\in\Gamma}$. The receiver verifies that all the decryptions are correct and aborts otherwise.

Decommitment Phase: Let $\Gamma' = [3n+1] - \Gamma$. The sender reveals its input m and all the plaintexts encrypted in $\{c_i^{b_i}\}_{i\in\Gamma'}$. The receiver checks if all the decryptions are correct and aborts otherwise. Using the n polynomial evaluations revealed relative to $i \in \Gamma$ and any additional polynomial evaluation that was revealed relative to Γ', the receiver reconstructs the polynomial $p(\cdot)$ (via polynomial interpolation of $n+1$ points). Next, the receiver verifies whether $p(0) = m$, and that for every $i \in [3n+1]$ the point $p(i)$ is the decrypted value within $c_i^{m_i}$.

Figure 6: Protocol Π_{COM} that realizes $\mathcal{F}_{\text{COM}}$ using a CCA-secure coin-tossing protocol $\langle I, R\rangle$

Proof overview: Recalling that an adversary can adaptively corrupt both parties, for the overview, we present the hardest cases for simulation, which is static corruption of one party and adaptive corruption of the other party.

Simulating static corruption of receiver and post-execution corruption of sender. To simulate the messages for a honest sender, the simulator generates random shares for 0 and 1 that agree on a randomly chosen n subset $\widetilde{\Gamma}$ (chosen in advance). It then encrypts these shares in Stage 2 where for each index it randomly

positions the shares for 0 and 1. Next, in Stage 3, the simulator biases $\tau_{R\to S}$ using the helper $\mathcal{H}$ so that the subset generated using $\mathsf{out}(\tau_{R\to S})$ is exactly $\widetilde{\Gamma}$. As these shares are common for a sharing of 0 and 1, revealing them in the commit phase will go undetected. Later in the decommit phase, it can chose to reveal shares of 0 or 1 depending on the real message m (to show that the unopened shares were obliviously generated will be done by exploiting the invertible sampling algorithm for the simulatable encryption scheme). The core argument in proving indistinguishability of simulation will be to reduce the hiding property of Stage 2 to the semantic-security of the underlying encryption scheme on a public-key generated using Gen, i.e., the CPA-security of the encryption scheme, where we will rely on the CCA-security game w.r.t challengers for our coin-tossing protocol to achieve this. We discuss this reduction on a high-level below. Before that we remark that the adversary will not be able to use the helper oracle $\mathcal{H}$ to bias the outcome of the coin-tossing in Stage 1 because the helper oracle will not provide access to the biasing oracle on sessions where the party querying the helper is not the responder R of that coin-tossing session.
Reduction: The challengers $\mathcal{C}$ for our CCA-game, on input a string o will set $\mathsf{PK} = \mathsf{rGen}(o)$ and for a predetermined message t it proceed as follows:

- If its private input $b = 0$, $\mathcal{C}$ will output a ciphertext that is an honest encryption of t using Enc.
- If its private input $b = 1$, $\mathcal{C}$ will obliviously generate a ciphertext using $\mathsf{oRndEnc}$.

It will follow from the security guarantees of the simulatable public-key encryption that for a randomly chosen o, no (stand-alone) adversary can distinguish the outputs of $\mathcal{C}|_{b=0}$ or $\mathcal{C}|_{b=1}$ even given o (i.e. $\mathsf{STA}_0 \approx \mathsf{STA}_1$).

Now given an adversary $\mathcal{A}$ controlling the receiver in our coin-tossing scheme $\langle I, R\rangle$ we consider a sequence of hybrid experiments where we replace the encryptions in Stage 2 from the honest sender's strategy to the simulated strategy. Namely, obliviously generated ciphertexts $c_j^{1-b_j}$ will be generated using the encryption algorithm. More precisely, we consider a sequence of hybrids $H^0 = \textbf{REAL}, H^1 \ldots, H^{3n+1}$ where in the H^i we generate $c_j^{1-b_j}$ for $j = 1, \ldots, i$ in Stage 2 according to the simulator's strategy (i.e. encryption of valid messages as opposed to being obliviously generated). Next we show that H^{i-1} and H^i are indistinguishable. The only difference between the two hybrids is in how $c_i^{1-b_i}$ is generated. More precisely, in H^{i-1}, $c_i^{1-b_i}$ is generated using $\mathsf{oRndEnc}$ and in H^i it is generated using Enc. We now reduce the indistinguishability of the hybrids to the semantic-security of the encryption scheme via the CCA-game of $\langle I, R\rangle$. Towards this, we consider a challenger $\mathcal{C}$ described above for which the stand-alone game is hard.

Next, consider an oracle adversary $\widetilde{\mathcal{A}}$ that internally incorporates $\mathcal{A}$ and the environment and proceeds as follows: $\widetilde{\mathcal{A}}$ forwards every oracle query made by $\mathcal{A}$ to its oracle and forwards the interaction using $\langle I, R\rangle$ in Stage 1 externally to an honest receiver. $\widetilde{\mathcal{A}}$ then stalls the internal emulation upon having the interaction within $\langle I, R\rangle$ complete, and outputs the view of $\mathcal{A}$ and the outcome of the coin-tossing o from the internal emulation, in the external interaction. Then it interacts with $\mathcal{C}$ that on input o produces a ciphertext. Internally, $\widetilde{\mathcal{A}}$ feeds the ciphertext in place of $c_i^{1-b_i}$ in Stage 2. The rest of the encryptions are honestly generated according to the strategy in H^i.

It now follows that if the message t is chosen according to the strategy in H^i, then we have that $\mathsf{hyb}^{i-1} = \mathsf{EXP}_1(\langle I, R\rangle, \mathcal{O}, \mathcal{A}, \mathcal{C}, n, z)\}_{n\in\mathbb{N}, z\in\{0,1\}^*}$ and $\mathsf{hyb}^i = \mathsf{EXP}_0(\langle I, R\rangle, \mathcal{O}, \mathcal{A}, \mathcal{C}, n, z)\}_{n\in\mathbb{N}, z\in\{0,1\}^*}$ where hyb^{i-1} and hyb^i are the views of the adversary $\mathcal{A}$ in the hybrids H^{i-1} and H^i. Therefore, if hyb^{i-1} and hyb^i are distinguishable by the CCA-security of $\langle I, R\rangle$ we have that there exists a stand-alone PPT algorithm B for which STA_0 and STA_1 are distinguishable. Recalling that $\mathsf{STA}_0 \approx \mathsf{STA}_1$ by the hiding property of obliviously generated ciphertexts in the underlying encryption scheme and thus we arrive at a contradiction. Therefore, hyb^{i-1} and hyb^i must be indistinguishable.

To complete this case, we need to handle post-execution corruption of the sender. This can be achieved exactly as in the decommitment phase which reveals all the randomness used in the commitment phase.

Simulating malicious senders. For a honest receiver, the simulator first biases the outcome of the coin-tossing in Stage 1, so that PK is a public-key for which it knows the corresponding secret-key. This will allow the simulator to decrypt the ciphertexts provided by the adversary in Stage 2. However, this does not ensure extraction as an adversarial sender can equivocate just as the simulator for honest senders. Showing that there is a unique value that can be extracted requires showing that a corrupted sender cannot successfully predict exactly the n indexes Γ from $\{1,\ldots,3n+1\}$ that will be chosen in the coin-tossing protocol. Using an information-theoretic argument from [HV15], we know that after an encoding phase, for any adversary to break binding (i.e. equivocate) it must ensure that the coin-tossing phase results in a particular set Γ. We can reduce the binding property of our scheme to the CCA-security of underlying coin-tossing scheme. First, we observe that the helper functionality cannot be directly used by the adversary to bias the coin-tossing as $\mathcal{H}$ will not help in sessions where the identity of the party controlling $\mathcal{I}$ is the party requesting the help. We will in fact rely on the CCA-security against challengers to guarantee that the coin-tossing outcome has high-entropy. Once we have established that the adversary cannot bias the coin-tossing used to determine the set Γ, we can obtain extraction by relying on a strategy from [HV15], that can determine the message using the decryptions from Stage 1 and the coin-tossing outcome in Stage 3. Finally, to address post-execution corruption of the receiver we observe that it suffices to generate the messages for the receiver honestly and upon corruption simply provide the random coins of this honest receiver.

Formal proof of Correctness of UC-Commitment Protocol:

Let $\mathcal{A}$ be a PPT adversary that attacks Protocol Π_{COM} described in Figure 7 and recall that simulator $\mathcal{S}$ interacts with the ideal functionality $\mathcal{F}_{\mathrm{COM}}$ and with the environment $\mathcal{Z}$. Then $\mathcal{S}$ starts by invoking a copy of $\mathcal{A}$ and running a simulated (internal) interaction of $\mathcal{A}$ with the environment $\mathcal{Z}$ and parties running the protocol. We fix the following notation. First, the session and sub-session identifiers are respectively denoted by sid and $ssid$. Next, the committing party is denoted P_i and the receiving party P_j. $\mathcal{S}$ proceeds as follows:

Simulating the communication with $\mathcal{Z}$: Every message that $\mathcal{S}$ receives from $\mathcal{Z}$ it internally feeds to $\mathcal{A}$ and every output written by $\mathcal{A}$ is relayed back to $\mathcal{Z}$.

Simulating the commitment phase when the receiver is statically corrupted: In this case $\mathcal{S}$ uses the honest sender's algorithm in Stage 1 and in Stage 2 proceeds as follows. Upon receiving message $(sid, \mathrm{Sen}, \mathrm{Rec})$ from $\mathcal{F}_{\mathrm{COM}}$, the simulator picks a random subset $\widetilde{\gamma} \subset [3n+1]$ of size n and two random n-degree polynomials $p_0(\cdot)$ and $p_1(\cdot)$ such that p_0 and p_1 agree on all points $i \in \widetilde{\Gamma}$ and $p_0(0) = 0$ and $p_1(0) = 1$.

- For every $i \in \widetilde{\Gamma}$ the simulator proceeds as the honest sender would with polynomial $p_0(\cdot)$. Namely, it first picks $b_i \leftarrow \{0,1\}$ at random and then sets the following pairs, $c_i^{b_i} = \mathsf{Enc}_{\mathrm{PK}}(p_0(i); t_i)$ and $c_i^{1-b_i} = \mathsf{oRndEnc}(\mathrm{PK}, r_i)$ where $r_i, t_i \leftarrow \{0,1\}^n$ (we recall that $p_0(i) = p_1(i)$ for all $i \in \widetilde{\Gamma}$).
- For every $i \in \widetilde{\Gamma}' = [3n+1] - \widetilde{\Gamma}$ the simulator picks $b_i \leftarrow \{0,1\}$ at random and then uses the points on both polynomials $p_0(\cdot)$ and $p_1(\cdot)$ to calculate the following pairs, namely $c_i^{b_i} = \mathsf{Enc}_{\mathrm{PK}}(p_0(i); t_i^0)$ and $c_i^{1-b_i} = \mathsf{Enc}_{\mathrm{PK}}(p_1(i), t_i^1)$ where $t_i^0, t_i^1 \leftarrow \{0,1\}^n$ are chosen uniformly at random.

Finally, the simulator sends the pairs $(c_0^0, c_0^1), \ldots, (c_{3n+1}^0, c_{3n+1}^1)$ to the receiver.

Next, in Stage 3, the simulator biases the coin-tossing result so that the set Γ that is chosen in this phase is identical to $\widetilde{\Gamma}$. More precisely, produces coins c that will yield $\widetilde{\Gamma}$ in Stage 3 and sends c to $\mathcal{H}$. Next, it forwards the messages the simulator receives from $\mathcal{A}$ controlling R in this interaction using $\langle I, R\rangle$ to $\mathcal{H}$. Recall that the helper function will bias the outcome of this interaction to c (as the identity of this interaction is not equal to any identity made by the $\mathcal{A}$). Finally, the simulator reveals the plaintexts in all the ciphertexts within $\{c_i^{b_i}\}_{i\in\widetilde{\Gamma}}$.

Simulating the decommitment phase where the receiver is statically corrupted: Upon receiving a message $(\mathsf{reveal}, sid, m)$ from $\mathcal{F}_{\rm COM}$, $\mathcal{S}$ generates a simulated decommitment message as follows. Recall first that the simulator needs to reveal points on a polynomial $p(\cdot)$ and pairs $\{(b_i, t_i)\}_{i\in[3n+1]}$ such that $p(0) = m$ and $c_i^{b_i} = \mathsf{Enc}_{\mathsf{PK}}(p(i); t_i)$. Let $\hat{b}_i = b_i \oplus m$ for all $i \in \widetilde{\Gamma}'$, then $\mathcal{S}$ reveals $p_m(\cdot)$, $\{\hat{b}_i, t_i^{\hat{b}_i}, r_i = \mathsf{rRndEnc}(\mathsf{PK}, t_i^{1-m}, p_{1-m}(i))\}_{i\in\widetilde{\Gamma}'}$.

Simulating the commit phase when the sender is statically corrupted: Simulating the sender involves extracting the committed value as follows. In Stage 1, $\mathcal{S}$ first samples $(\mathsf{PK}, \mathsf{SK})$ using the Gen algorithm with randomness r_G. Then it runs rGen on r_G to obtain c which it forwards to the helper $\mathcal{H}$. Then, it forwards the messages the simulator receives from $\mathcal{A}$ controlling R in this interaction using $\langle I, R\rangle$ to $\mathcal{H}$. Recall that the helper function will bias the outcome of this interaction to c. This means that the public-key obtained from the coin-tossing is PK.

The simulation next uses the honest receiver's algorithm in Stages 2 and 3. Let Γ be the set obtained from the outcome of the coin-tossing phase. To extract the input, $\mathcal{S}$ chooses an arbitrary index $j \in [3n+1] - \Gamma$ and reconstructs two polynomials $q(\cdot)$ and $\widetilde{q}(\cdot)$ such that

$$q(i) = \widetilde{q}(i) = \beta_i^{b_i} \qquad \forall i \in \Gamma$$
$$q(j) = \beta_j^0 \text{ and } \widetilde{q}(j) = \beta_j^1 \text{ and } q(0), \widetilde{q}(0) \in \{0,1\}.$$

It then verifies whether for all $i \in [3n+1]$, $q(i) \in \{\beta_i^0, \beta_i^1\}$ and $\widetilde{q}(i) \in \{\beta_i^0, \beta_i^1\}$. The following cases arise:

Case 1: *Both $q(\cdot)$ and $\widetilde{q}(\cdot)$ satisfy the condition and $\widetilde{q}(0) \neq q(0)$.* Then $\mathcal{S}$ halts returning fail. Below we prove that the simulator outputs fail with negligible probability.

Case 2: *At most one of $q(\cdot)$ and $\widetilde{q}(\cdot)$ satisfy the condition or $\widetilde{q}(0) = q(0)$.* $\mathcal{S}$ sends $(\mathsf{commit}, sid, q(0))$ to the $\mathcal{F}_{\rm COM}$ functionality and stores the committed bit $q(0)$. Otherwise, $\mathcal{S}$ sends a default value.

Case 3: *Neither $q(\cdot)$ or $\widetilde{q}(\cdot)$ satisfy the condition.* $\mathcal{S}$ sends a default value to the ideal functionality and need not store the committed bit since it will never be decommitted correctly.

Simulating adaptive corruptions: We remark that we only provide the description of the simulator for static corruption. If any honest party is adaptively corrupted during the simulation, since the simulation is straight-line and admits post-execution corruption, it can directly generate coins even in the middle of the execution.

Below we analyze each of the above scenarios and show that no environment $\mathcal{Z}$ interacting with $\mathcal{S}$ in the ideal-world is distinguishable from that with $\mathcal{A}$ in the real-world in each of the cases.

Analysis of receiver corruptions: Our proof follows a sequence of hybrids from the real world execution to the ideal world execution.

Hybrid H_0: H_0 is identical to the real world execution.

Hybrid H_1: The hybrid experiment H_1 proceeds identically to H_0 with the exception that a set $\widetilde{\Gamma}$ of size n is chosen at random and the coin-tossing interaction using $\langle I, R\rangle$ in Stage 3 is biased so that the outcome yields $\widetilde{\Gamma}$. Hybrids H_0 and H_1 are identically distributed except when the oracle $\mathcal{O}$ fails. Since this happens only with negligible probability, the outputs of the two experiments are statistically close.

Hybrid H_2**:** We gradually change the ciphertexts generated in Stage 2 from the real committer to the simulation. Indistinguishability of experiment H_1 and H_2 will rely on the security of the encryption scheme. However, to reduce the indistinguishability to the security game of the simulatable public-key encryption scheme, we will require to bias the PK chosen in Stage 1 to a challenge public-key obtained from the challenger for the encryption security game. We will be able to do this by relying on the security game of our CCA-secure coin-tossing protocol.

More formally, consider a sequence of hybrids $H_1^0, \ldots, H_1^{3n+1}$ where in the H_1^i we generate $c_j^{1-b_j}$ for $j = 1, \ldots, i$ according to the simulator's strategy (i.e. encryption of valid messages as opposed to being obliviously generated). Now we show that H_1^{i-1} and H_1^i are indistinguishable. The only difference between the two hybrids is in how $c_i^{1-b_i}$ is generated. More precisely, in H_1^{i-1}, $c_i^{1-b_i}$ is generated using $\mathsf{oRndEnc}$ and in H_1^i it is generated using Enc. We now reduce the indistinguishability of the hybrids to the semantic-security of the encryption scheme via the CCA-game of $\langle I, R\rangle$.

Towards this, we give a challenger $\mathcal{C}$ for which the stand-alone game is hard. On a high-level this game will be the semantic-security of the underlying simulatable public-key encryption scheme where the public-key is sampled using rGen on the coin-tossing o.

Reduction: More formally, given a message t, define $\mathcal{C}(o, b)$ as the strategy that sets PK $= \mathsf{rGen}(o)$ and outputs a ciphertext that was honest encryption of t using Enc when $b = 0$ and obliviously generated using $\mathsf{oRndEnc}$ when $b = 1$.

Next consider an oracle adversary $\widetilde{\mathcal{A}}$ that internally incorporates $\mathcal{A}$ and the environment and proceeds as follows: $\widetilde{\mathcal{A}}$ forwards every oracle query made by $\mathcal{A}$ to its oracle and forwards the interaction using $\langle I, R\rangle$ in Stage 1 externally to an honest receiver. Let o be the outcome of the interaction in the internal emulation. an encryption of a message using Enc or generates one obliviously.

Then it interacts with $\mathcal{C}$ that on input o produces a ciphertext. Internally, $\widetilde{\mathcal{A}}$ feeds the ciphertext in place of $c_i^{1-b_i}$ in Stage 2.

It now follows that if the message t is chosen according to the strategy in H_1^i, then

$$\mathsf{hyb}_1^{i-1} = \mathsf{EXP}_1(\langle I, R\rangle, \mathcal{O}, \mathcal{A}, \mathcal{C}, n, z)\}_{n\in\mathbb{N}, z\in\{0,1\}^*}$$
$$\mathsf{hyb}_1^{i} = \mathsf{EXP}_0(\langle I, R\rangle, \mathcal{O}, \mathcal{A}, \mathcal{C}, n, z)\}_{n\in\mathbb{N}, z\in\{0,1\}^*}$$

where hyb_1^{i-1} and hyb_1^i are the views of the adversary $\mathcal{A}$ in the hybrids H_1^{i-1} and H_1^i. Therefore, if hyb_1^{i-1} and hyb_1^i are distinguishable by the CCA-security of $\langle I, R\rangle$ we have that there exists a stand-alone PPT algorithm B that distinguish the interaction with $\mathcal{C}_0$ and $\mathcal{C}_1$ for a randomly sampled coin-tossing outcome o. This violates the semantic-security of the encryption scheme and thus we arrive at a contradiction. Therefore, hyb_1^{i-1} and hyb_1^i must be indistinguishable.

Hybrid H_3**:** In this hybrid, we follow H_2 except that we use the simulation strategy to decommit to the message m received from the $\mathcal{F}_{\mathrm{COM}}$-functionality. Since in H_2 the commitment phase has been setup to be equivocated, this follows directly. Again using the CCA-security of $\langle I, R\rangle$ just as we used to argue indistinguishability for hybrids H_1 and H_2, we can reduce the indistinguishability of H_2 and H_3 to the security of the underlying simulatable public-key encryption scheme.

Finally, we conclude by observing the H_3 is identical to the ideal world experiment.

Analysis of sender corruptions: Our proof follows a sequence of hybrids from the real world execution to the ideal world execution.

Hybrid H_0: H_0 is identical to the real world execution.

Hybrid H_1: This experiment proceeds identical to H_0 with the exception that we forward the interaction using $\langle I, R\rangle$ in Stage 1 to the oracle $\mathcal{H}$. More precisely, we pick $(\mathsf{PK}, \mathsf{SK})$ using the Gen algorithm with randomness r_G. Then rGen is invoked on r_G to obtain c which it forwards to the helper $\mathcal{H}$. Recall that $\mathcal{H}$ will bias the coin-tossing outcome to c and the resulting public-key agreed upon will be PK. Indistinguishability of H_1 and H_0 can be reduced directly to the indistinguishability of real and obliviously generated public-keys of the simulatable public-key encryption scheme using the CCA-security of $\langle I, R\rangle$.

Hybrid H_2: H_2 is the same as H_1 with the exception that the value committed to by the adversary is extracted using the simulator's strategy and forwarded to $\mathcal{F}_{\mathrm{COM}}$. The only difference between the hybrids H_1 and H_2 is that in H_2 we extract a value for the commitment from the adversarial sender. This means that to argue indistinguishability it suffices to show that the value extracted is correct (i.e. the scheme is binding). We argue this by relying on the information-theoretic lemma proved in [HV15]. In more detail, this lemma shows that at the end of Stage 2, it is possible to define a set Γ such that for any adversarial sender to equivocate it needs to bias the outcome of the coin toss in Stage 3 to result in this set Γ. This coin-tossing is decided using our protocol $\langle I, R\rangle$ where the adversarial sender controls the initiator and by relying on CCA-security we argue next that there exists no adversary that can bias the outcome to result in a particular set with non-negligible probability.

Suppose for contradiction there exists an adversary $\mathcal{A}$ that can bias the outcome to Γ in H_1 with non-negligible probability. We now construct an adversary $\mathcal{A}'$ that incorporates $\mathcal{A}$ and internally emulates the hybrid experiment H_2 with the exception that it forwards the interaction of $\mathcal{A}$ in Stage 3 to an external honest receiver. Now, consider the challengers $\mathcal{C}$ for the CCA-security game where $\mathcal{C}|_{b=0}$ outputs 1 if the outcome o results in Γ and 0 otherwise. $\mathcal{C}|_{b=1}$ outputs 0 irrespective of the outcome. By our assumption on $\mathcal{A}$, this means that EXP_0 and EXP_1 with the adversary $\mathcal{A}'$ are distinguishable because the adversary biases the coin-toss to result in Γ with non-negligible probability. However, since a uniformly sampled coin will result in Γ with at most negligible probability we have that STA_0 and STA_1 are indistinguishable which is a contradiction. Therefore, we have that the value extracted by our simulator is correct except with non-negligible probability and this concludes the proof. Finally, we conclude by observing the H_2 is identical to the ideal world experiment.

8 Application: Zero-One Law for Adaptive UC Security

We extend the result of [MPR10] and establish a zero-one law under adaptive UC-reduction. More formally, we show that all (non-reactive)[9] functionalities fall into two categories: *trivial* functionalities, those which can be UC-reduced to any other functionality; and *complete* functionalities, to which any other functionality can be UC-reduced.

Theorem 8.1 *Assume the existence of simulatable public-key encryption scheme. Then every two-party non-reactive functionality is either trivial or complete in the UC framework in the presence of adaptive, malicious adversaries.*

Proof: Our starting point, is the characterization in [MPR10] that shows that any non-trivial (non-reactive) functionality UC realizes one of the following three two-party functionalities.

[9]Such functionalities are computed in a single round of communication with the functionality.

$\mathcal{F}_{\text{OT}}$**:** On input $x_0, x_1 \in \{0,1\}$ from P_1 and $y \in \{0,1\}$ from P_2, $\mathcal{F}_{\text{OT}}$ outputs x_y to P_2.

$\mathcal{F}_{\text{XOR}}$**:** On input $x \in \{0,1\}$ from P_1 and $y \in \{0,1\}$ from P_2, $\mathcal{F}_{\text{XOR}}$ outputs $x \oplus y$ to both parties.

$\mathcal{F}_{\text{CC}}$**:** On input $x \in \{0,1\}$ from P_1 and $y \in \{0,1\}$ from P_2, $\mathcal{F}_{\text{CC}}$ outputs x if $y = 0$ and outputs a special symbol, say $\perp$, otherwise.

More formally, they showed that every non-reactive functionality unconditionally realizes one of the three functionalities in the UC framework with information-theoretic security. Specifically, since the reductions are information-theoretically secure, they are further secure in the presence of adaptive corruptions. Therefore, to prove the zero-one law when considering malicious adaptive adversaries, it suffices to demonstrate the completeness of the three functionalities $\mathcal{F}_{\text{OT}}$, $\mathcal{F}_{\text{XOR}}$, and $\mathcal{F}_{\text{CC}}$. In more details,

1. As mentioned before, the work of [IPS08] proves that $\mathcal{F}_{\text{OT}}$ is complete unconditionally in the UC framework in the presence of adaptive malicious adversaries.

2. Next, we argue that $\mathcal{F}_{\text{COIN}}$ is complete under the assumption of existence of simulatable PKE scheme, and since $\mathcal{F}_{\text{XOR}}$ statistically implements $\mathcal{F}_{\text{COIN}}$, it establishes the completeness of $\mathcal{F}_{\text{XOR}}$. More formally, it follows directly that $\mathcal{F}_{\text{COIN}}$ implements $\mathcal{F}_{\text{COM}}$ in the presence of adaptive malicious adversaries by simply replacing all instantiations of our CCA-secure coin-tossing protocol $\langle I, R\rangle$ in Protocol Π_{COM} with a call to the $\mathcal{F}_{\text{COIN}}$ functionality. Then, following the same argument as in Section 7, combining the works of [DN00, CDMW09, IPS08], we have that assuming the existence of simulatable public-key encryption, $\mathcal{F}_{\text{COIN}}$ implements $\mathcal{F}_{\text{OT}}$ and is therefore complete. More formally, we have the following lemma.

   Lemma 8.1 *Assume the existence of simulatable public-key encryption scheme. Then $\mathcal{F}_{\text{COM}}$ can be realized in the $\mathcal{F}_{\text{COIN}}$-hybrid model in the presence of adaptive, malicious adversaries, using black-box access to the encryption scheme.*

3. In Section 8.1, we show that $\mathcal{F}_{\text{CC}}$ realizes weaker commitment functionality $\mathcal{F}_{\text{EXTCOM}}$ that implements extractable commitments. Next, we observe that $\mathcal{F}_{\text{EXTCOM}}$ is stronger than a CCA-secure commitments (as it is non-interactive and provides straight-line extraction) and replacing all instantiations to the protocol $\langle C, R\rangle$ in the protocol Π_{COIN} we obtain a CCA-secure coin-tossing. Now, completeness follows from the completeness of the CCA-secure coin-tossing via Π_{COM} that realizes the $\mathcal{F}_{\text{COM}}$-functionality.

■

8.1 Implementing $\mathcal{F}_{\text{EXTCOM}}$ Using $\mathcal{F}_{\text{CC}}$ in the Adaptive Setting

We briefly recall the construction from [MPR10] for the static case and argue that the same construction works even for adaptive corruptions.

Recall first that the $\mathcal{F}_{\text{CC}}$ functionality takes as input $x \in \{0,1\}$ from party P_1 and $y \in \{0,1\}$ from P_2 and outputs x to both parties if $y = 0$, and outputs nothing otherwise.[10] To realize $\mathcal{F}_{\text{EXTCOM}}$ using $\mathcal{F}_{\text{CC}}$ the high-level idea of the protocol in [MPR10] is as follows: To commit to a bit m, P_1 first chooses a random bit string s and encodes it to a codeword $t \in \{0,1\}^n$ using a linear error-correcting code. Then P_1 commits

[10] We can consider outputs nothing as a special output $\perp$.

to the bits of t using a statistically-binding commitment scheme. Next, for each $i \in [n]$, P_1 and P_2 invoke the $\mathcal{F}_{\text{CC}}$ functionality where P_1's input is t_i, the ith bit of t and P_2 picks a bit according to some distribution that allows it learn the bits of t with some specific probability. According to the functionality $\mathcal{F}_{\text{CC}}$, P_1 knows for which indexes i, P_2 learns t_i. P_1 makes sure that number of bits learned by P_2 is insufficient to reconstruct t by counting the number of bits learned by P_2 and aborts if it learns too much. Next, using the remaining entropy in s, P_1 masks the bit m and sends it to P_2. In the reveal phase, P_1 simply decommits t. The protocols Π_L and Π_R corresponding to this protocol are simply the same protocol with the exception of which party emulates the $\mathcal{F}_{\text{CC}}$ functionality. Namely, in Π_L the receiver emulates the $\mathcal{F}_{\text{CC}}$ functionality, whereas in Π_R the sender does so. Then it follows from the hiding property of the statistically-binding commitment scheme and the error-correcting code (when only a few bits are learned), that s has sufficient (computational) entropy and m is hidden. To achieve extraction from a malicious sender, the receiver of Π_R (i.e., the simulator) sees all the queries of P_1 to $\mathcal{F}_{\text{CC}}$, namely the bits of t, and can reconstruct s using the error-correcting property of the code and unmask m.

To argue that the same protocol is secure in the adaptive setting, we need to argue that adaptive corruption of either the sender or the receiver of the protocol implementing $\mathcal{F}_{\text{EXTCOM}}$ can be achieved. Regarding sender corruption, we first observe that the message m to be committed is fixed and known to the simulation (even for honest senders), namely, there is no equivocation required. Furthermore, if in the protocol, simulating an honest sender for Π_L and Π_R simply involves running the code of the honest sender then providing randomness for the sender upon corruption is achieved directly. Arguing receiver corruption is not immediate as in protocol Π_R achieving sender's input extraction might involve a strategy different from the honest receiver. However, in the preceding protocol, extraction is achieved by simply looking at the sender's input to the $\mathcal{F}_{\text{CC}}$ functionality and all messages of the receiver are generated honestly in Π_L and Π_R. Therefore, providing randomness for the receiver upon corruption can be directly achieved.

References

[BCNP04] Boaz Barak, Ran Canetti, Jesper Buus Nielsen, and Rafael Pass. Focs. pages 186–195, 2004.

[Bea91] Donald Beaver. Foundations of secure interactive computing. In *CRYPTO*, pages 377–391, 1991.

[Blu86] Manuel Blum. Independent unbiased coin flips from a correlated biased source-a finite stae markov chain. *Combinatorica*, 6(2):97–108, 1986.

[BS05] Boaz Barak and Amit Sahai. How to play almost any mental game over the net - concurrent composition via super-polynomial simulation. In *FOCS*, pages 543–552, 2005.

[Can01] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *FOCS*, pages 136–145, 2001.

[CDMW09] Seung Geol Choi, Dana Dachman-Soled, Tal Malkin, and Hoeteck Wee. Simple, black-box constructions of adaptively secure protocols. In *TCC*, pages 387–402, 2009.

[CDPW06] Ran Canetti, Yevgeniy Dodis, Rafael Pass, and Shabsi Walfish. Universally composable security with global setup. *IACR Cryptology ePrint Archive*, 2006:432, 2006.

[CDPW07] Ran Canetti, Yevgeniy Dodis, Rafael Pass, and Shabsi Walfish. Universally composable security with global setup. In *TCC*, pages 61–85, 2007.

[CF01] Ran Canetti and Marc Fischlin. Universally composable commitments. In *CRYPTO*, pages 19–40, 2001.

[CKL06] Ran Canetti, Eyal Kushilevitz, and Yehuda Lindell. On the limitations of universally composable two-party computation without set-up assumptions. *J. Cryptology*, 19(2):135–167, 2006.

[CLP10] Ran Canetti, Huijia Lin, and Rafael Pass. Adaptive hardness and composable security in the plain model from standard assumptions. In *FOCS*, pages 541–550, 2010.

[CLP13] Ran Canetti, Huijia Lin, and Rafael Pass. From unprovability to environmentally friendly protocols. In *FOCS*, pages 70–79, 2013.

[CPS07] Ran Canetti, Rafael Pass, and Abhi Shelat. Cryptography from sunspots: How to use an imperfect reference string. In *FOCS*, pages 249–259, 2007.

[DDN03] Danny Dolev, Cynthia Dwork, and Moni Naor. Nonmalleable cryptography. *SIAM Review*, 45(4):727–784, 2003.

[DMRV13] Dana Dachman-Soled, Tal Malkin, Mariana Raykova, and Muthuramakrishnan Venkitasubramaniam. Adaptive and concurrent secure computation from new adaptive, non-malleable commitments. In *ASIACRYPT*, pages 316–336, 2013.

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

[GLP$^+$15] Vipul Goyal, Huijia Lin, Omkant Pandey, Rafael Pass, and Amit Sahai. Round-efficient concurrently composable secure computation via a robust extraction lemma. In *TCC*, pages 260–289, 2015.

[GMW87] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or A completeness theorem for protocols with honest majority. In *STOC*, pages 218–229, 1987.

[Gol01] Oded Goldreich. *Foundations of Cryptography: Basic Techniques*. Cambridge University Press, 2001.

[HV15] Carmit Hazay and Muthuramakrishnan Venkitasubramaniam. On black-box complexity of universally composable security in the CRS model. In *ASIACRYPT*, pages 183–209, 2015.

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

[Kiy14] Susumu Kiyoshima. Round-efficient black-box construction of composable multi-party computation. In *CRYPTO*, pages 351–368, 2014.

[KLP07] Yael Tauman Kalai, Yehuda Lindell, and Manoj Prabhakaran. Concurrent composition of secure protocols in the timing model. *J. Cryptology*, 20(4):431–492, 2007.

[KMO14] Susumu Kiyoshima, Yoshifumi Manabe, and Tatsuaki Okamoto. Constant-round black-box construction of composable multi-party computation protocol. In *TCC*, pages 343–367, 2014.

[Lin03] Yehuda Lindell. General composition and universal composability in secure multi-party computation. In *FOCS*, pages 394–403, 2003.

[LP12a] Huijia Lin and Rafael Pass. Black-box constructions of composable protocols without set-up. In *CRYPTO*, pages 461–478, 2012.

[LP12b] Huijia Lin and Rafael Pass. Black-box constructions of composable protocols without set-up (full version). Available at `https://www.cs.ucsb.edu/~rachel.lin`, 2012.

[LPV08] Huijia Lin, Rafael Pass, and Muthuramakrishnan Venkitasubramaniam. Concurrent non-malleable commitments from any one-way function. In *TCC*, pages 571–588, 2008.

[LPV09] Huijia Lin, Rafael Pass, and Muthuramakrishnan Venkitasubramaniam. A unified framework for concurrent security: universal composability from stand-alone non-malleability. In *STOC*, pages 179–188, 2009.

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

[MMY06] Tal Malkin, Ryan Moriarty, and Nikolai Yakovenko. Generalized environmental security from number theoretic assumptions. In *TCC*, pages 343–359, 2006.

[MPR10] Hemanta K. Maji, Manoj Prabhakaran, and Mike Rosulek. A zero-one law for cryptographic complexity with respect to computational UC security. In *CRYPTO*, pages 595–612, 2010.

[MR91] Silvio Micali and Phillip Rogaway. Secure computation (abstract). In *CRYPTO*, pages 392–404, 1991.

[Nao91] Moni Naor. Bit commitment using pseudorandomness. *J. Cryptology*, 4(2):151–158, 1991.

[ORSV13] Rafail Ostrovsky, Vanishree Rao, Alessandra Scafuro, and Ivan Visconti. Revisiting lower and upper bounds for selective decommitments. In *TCC*, pages 559–578, 2013.

[Pas03] Rafael Pass. Simulation in quasi-polynomial time, and its application to protocol composition. In *EUROCRYPT*, pages 160–176, 2003.

[PR08] Manoj Prabhakaran and Mike Rosulek. Cryptographic complexity of multi-party computation problems: Classifications and separations. In *CRYPTO*, pages 262–279, 2008.

[PS04] Manoj Prabhakaran and Amit Sahai. New notions of security: achieving universal composability without trusted setup. In *STOC*, pages 242–251, 2004.

[RK99] Ransom Richardson and Joe Kilian. On the concurrent composition of zero-knowledge proofs. In *EUROCRYPT*, pages 415–431, 1999.

[Ven14] Muthuramakrishnan Venkitasubramaniam. On adaptively secure protocols. In *SCN*, pages 455–475, 2014.

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

A Adaptive Extension to CCA-Secure Commitments

In our work, we need to consider the CCA-Security game in the presence of an adaptive adversary $\mathcal{A}$. This definition as we will see does not require full-fledged adaptive security and in particular our definition will not imply equivocability of the commitments.

First, we recall the CCA-security game for the commitments as introduced in [CLP10]. Roughly speaking, a commitment scheme is CCA-secure if the commitment scheme retains its hiding property even if the receiver has access to a "decommiment oracle". The experiment considers an oracle adversary $\mathcal{A}$ with oracle access to a helper function $\mathcal{H}$ and interacts as the receiver with an honest committer C. In our adaptive setting, we will require two additional properties:

- The adversary will be allowed to corrupt the external committer C. However, security is required to hold, i.e. hiding property of the left commitment, only if the committer is not corrupted.
- In the interaction between the adversary and the helper oracle, where it interacts as the committer, the adversary will be allowed to corrupt the receiver. In this case, the helper oracle is required to provide random coins for the receiver consistent with the transcript.

The second property does not require any explicit change in the definition of the security as it only alters the semantics of the interaction between $\mathcal{A}$ and $\mathcal{H}$. The first property however needs to be incorporated in the definition which we do next.

Modifying the IND_b random variable in the definition. In the standard definition $\mathsf{IND}_b(\langle C, R\rangle, \mathcal{O}, \mathcal{A}, n, z)$ represents the output of the $\mathcal{A}^{\mathcal{O}}$ in a experiment where it interacts with an honest committer with input $b \in \{0,1\}^n$. This output is set to $\perp$, if the identity of the execution with C is the same as the identity of any interaction of $\mathcal{A}$ with $\mathcal{O}$. We define a new random variable $\overline{\mathsf{IND}}_b(\langle C, R\rangle, \mathcal{O}, \mathcal{A}, n, z)$ which is equal to $\mathsf{IND}_b(\langle C, R\rangle, \mathcal{O}, \mathcal{A}, n, z)$ only if $\mathcal{A}^{\mathcal{O}}$ does not corrupt the honest committer C in the execution. Otherwise it is set to $\perp$.

Definition A.1 (CCA-secure commitments with an adaptive adversary) *Let $\langle C, R\rangle$ be a tag-based commitment scheme with $l(n)$-bit identities, and $\mathcal{O}$ a decommitment oracle for it. We say that $\langle C, R\rangle$ is CCA-secure w.r.t. $\mathcal{O}$* in the presence of an adaptive adversary, *if for every* PPT $\mathcal{A}$, *the following ensembles are computationally indistinguishable:*

- $\{\overline{\mathsf{IND}}_0(\langle C, R\rangle, \mathcal{O}, \mathcal{A}, n, z)\}_{n\in\mathbb{N}}$
- $\{\overline{\mathsf{IND}}_1(\langle C, R\rangle, \mathcal{O}, \mathcal{A}, n, z)\}_{n\in\mathbb{N}}$

We say that $\langle C, R\rangle$ is CCA-secure if there exists a decommitment oracle $\mathcal{O}'$, such that, $\langle C, R\rangle$ is CCA-secure w.r.t. $\mathcal{O}'$.

We additionally require an analogous adaptive-extension to the k-robustness property of CCA-secure commitments as defined in [CLP10].

Definition A.2 *Let $\langle C, R\rangle$ be a tag-based commitment scheme, and $\mathcal{O}$ a committed-value oracle for it. We say that $\langle C, R\rangle$ is k-robust CCA-secure w.r.t. $\mathcal{O}$* in the presence of an adaptive adversary, *if $\langle C, R\rangle$ is CCA-secure w.r.t. $\mathcal{O}$, and for every* PPT *adversary $\mathcal{A}$, there exists a* PPT *simulator $\mathcal{S}$, such that, for every* PPT *k-round interactive Turing Machine B, the following two ensembles are computationally indistinguishable.*

- $\{\mathbf{View}_{\mathcal{A}}(\langle B, \mathcal{A}^{\mathcal{O}}(z)\rangle(1^n))\}_{n\in\mathbb{N}, z\in\{0,1\}^n}$

- $\{\textbf{View}_{\mathcal{S}}(\langle B, \mathcal{S}(z)\rangle(1^n))\}_{n\in\mathbb{N}, z\in\{0,1\}^n}$

where $\textbf{View}_M(\langle B, M\rangle(1^n))$ *refers to the view of party* M *at the end of an interaction with* B *with common input* 1^n.

Theorem A.3 *Assume the existence of one-way functions. Then, for every* $\epsilon > 0$, *there exists a* $O(n^\epsilon)$, *there exists a* $O(n^\epsilon)$*-round commitment scheme that is CCA-secure w.r.t. the committed-value oracle* in the presence of an adaptive adversary *and only relies on black-box access to one-way functions (where n is the security parameter).*

Proof Sketch: Lin and Pass [LP12a] gave a black-box construction of a $O(n^\epsilon)$-round CCA-secure commitment scheme $\langle C, R\rangle$. We rely on the same construction for our stronger definition of security in the presence of an adaptive adversary. We provide a high-level proof sketch of its correctness. We begin with a short overview of the proof in [LP12a].

In the proof of standard security of the scheme provided in [LP12a], the idea is to reduce the indistinguishability of the IND_b experiments to the stand-alone hiding property of a different commitment scheme $\langle \tilde{C}, \tilde{R}\rangle$ (that is a slight variant of $\langle C, R\rangle$). The main part of the proof is to show that given an oracle and an adversary for $\langle C, R\rangle$ there exists a stand-alone malicious receiver R^* (that does not have access to the oracle) for $\langle \tilde{C}, \tilde{R}\rangle$. On a high-level, R^* will internally incorporate $\mathcal{A}$ and emulate the committed-value oracle for $\mathcal{A}$ while forwarding the left interaction externally to $\tilde{C}$ (which it can do as it is a variant that has a "similar" structure). To emulate the oracle, R^* needs to extract the value committed value which it will accomplish by rewinding the right interactions. Two issues arise:

- Since the left interaction is forwarded to an external committer, R^* needs to be able to rewind the right interactions without rewinding the left. The main idea here that is reminiscent of previous work [DDN03, LPV08] is to identify the so-called *safe-points* where this can be done. In slight more detail, when rewinding from a safe-point the only thing the adversary can do in the left interaction is to request "complete" (3-round witness-indistinguishable) proofs and such a request will be accommodated by the variant $\langle \tilde{C}, \tilde{R}\rangle$.

- There are unbounded-many right interactions that will result in R^* recursively rewinding interactions to extract the committed value in the interactions. In [LP12a], they achieve this by providing several points to rewind from and rely on the [RK99] to ensure that expected running time of the rewindings in each level is polynomial and the recursive depth is at most a constant.

Next, we argue why the same protocol satisfies our stronger definition of security. We begin with the observation that if the adversary $\mathcal{A}$ does not corrupt the left or right interactions, then our definition reduces to the standard CCA-security. We will prove security identically to [LP12a] by reducing it to the stand-alone hiding property of $\langle \tilde{C}, \tilde{R}\rangle$. We will employ the exact rewinding strategy as in [LP12a] for R^* with the following exception: Our definition of *safe-point* will have one additional requirement: A *safe-point* for our scheme is any *safe-point* according to [LP12a] with the added requirement that the adversary corrupts neither the committer in the left-interaction or the receiver of the right interaction (associated with the safe-point) before the 3-round witness indistinguishable (WI) proof associated with the safe-point completes.[11]

We remark that our definition of *safe-point* can modularly replace the definition in [LP12a] and the entire proof goes through. This is because the definition affects only the run-time analysis of the reduction. For the

[11]In [LP12b] the definition of a safe-point is parameterized with the depth of the recursion and our additional requirement naturally extends to the definition.

run-time analysis to go through the only requirements are that there are sufficiently many *safe-point*'s and when rewound from a safe-point, it continues to be a *safe-point* with at least the same probability (See Step 1 in Sub-Claim 2 of [LP12b]).[12] The first property holds because, a right receiver needs to be rewound only if $\mathcal{A}$ completes the entire right session without corrupting the right receiver or the left committer. In this event there will be as many safe points according to the definition of [LP12a] as there according to ours. The second property holds because a rewinding will be cancelled only if the point is not safe. This concludes the proof of the theorem.

Adaptive extension to k-Robust CCA-secure commitments. We pursue the same extension as above for the definition of k-robust CCA-secure commitments. We merely point out that we require our simulation to be indistinguishable conditioned on $\mathcal{A}$ not corrupting the left interaction. We only require a k-robust CCA-Secure Commitment for constant k in this work and the protocol from above is k-robust for constant k and the proof follows analogously.

[12]The second property is to ensure that the expected running time of the rewinding is polynomial in each recursive depth. In more detail, if a point in the transcript is safe with probability p, then if the property holds the expected number of times before which a rewinding is successful is at most $O(\frac{1}{p})$.