CONCURRENT SECURITY

A Dissertation

Presented to the Faculty of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

by

Huijia Rachel Lin

January 2012

CONCURRENT SECURITY

Huijia Rachel Lin, Ph.D.

Cornell University 2012

Traditionally, cryptographic protocols are analyzed in a “stand-alone” setting, where a single protocol execution takes place *in isolation*. In the age of the Internet, however, a great number of executions of different protocols co-exist and are tightly inter-connected. This *concurrency* severely undermines the foundation of the traditional study of cryptography. Since the early 90’s, it has been an important theme in cryptography to address security in such concurrent setting. However, till recently, no satisfactory solutions were proposed for performing general tasks in a concurrently secure way.

In this thesis, we resolve “concurrent security”—we exhibit a construction of cryptographic protocols for *general tasks* that remain secure even in concurrent settings like the Internet. Different from previous works, our construction *does not rely on any trusted infrastructure or strong hardness assumptions*. As such, our construction broadens the applicability of cryptography by enabling it in more realistic settings and weakening the preconditions it is based on.

Beyond the general feasibility result, we also significantly improve the efficiency of secure protocols for performing general tasks even in the stand-alone setting: We construct *constant-round* secure protocols for general tasks based on enhanced trapdoor permutations; this yields the first improvement on the round-efficiency—from linear to constant—over the original construction of [GMW87] based on the same assumptions as [GMW87].

Towards our constructions, we identify the key role of “input independence” in achieving concurrent security. Intuitively, if adversaries are forced to act independently in different protocol executions, then concurrency comes for free since it is as if each execution were taking place in isolation. We study two notions of "input independence": Non-malleability and adaptive hardness. Both notions are central tools in cryptography and have been extensively studied. A main question is to determine the number of rounds needed for protocols satisfying these notions. In this thesis, we completely resolve the round-complexity of these two notions in the context of commitments: We construct constant-round non-malleable commitments—introduced by [DDN91]—and $\omega(\log n)$-round adaptively hard commitments—or CCA-secure commitments introduced in this thesis—from the minimal assumption of one-way functions without using any trusted infrastructure; the latter construction as we show is round optimal.

BIOGRAPHICAL SKETCH

Huijia grew up in Hangzhou, China. In 2004, she received a B.S. from Zhejiang University in Computer Science. She finished her Ph.D. at Cornell University in Computer Science in 2012.

ACKNOWLEDGEMENTS

Words, are not enough to express my gratitude towards RAFAEL PASS, my adviser, mentor, friend, and, deep in my heart, a family member. I wish I could thank him. But exhausting all possible means, I could not think of any words that have the slightest chance to carry the weight of the paramount help I received from him without belittling it. Perhaps, the greatest thanks can only be said in silence.

I am fortunate to have WEI-LUNG DUSTIN TSENG as my teammate for the PhD journey; we started at Cornell at the same year, grew together as researchers, explored side by side unfamiliar territories in life. Now we are both departing for our new lives, with a heart-warming friendship that will not be cut by time.

I wish to thank MUTHURAMAKRISHNAN VENKITASUBRAMANIAM, an extremely generous, supportive and inspiring colleague and friend. He helped me to gradually dive into the field of cryptography from the beginning and became one of the most pleasant collaborators of mine. He never hesitated to offer me comfort and encouragement whenever I needed. I am very grateful to that.

I am indebted to my mentor, CYNTHIA DWORK, for showing me a wonderful summer at Microsoft, sharing generously and sincerely her opinions and wisdom whenever I solicited for suggestions in research and difficult situations in life. I am especially grateful to her for exhibiting me different ways of life when I succumb to conventions and restrict myself in boxes unconsciously.

I want to thank the cryptography group at IBM for warmly taking me into their family since my summer internship there. In particular, TAL RABIN, with her mother-like caring nature and wise opinions in research, guided me during the summer and after. I enjoyed greatly my collaboration with SHAI HALIVI, whose clear views, sharp thoughts and gentle nature made our discussions both thought-stimulating and delightful. HUGO KRAWCZYK introduced me to very interesting questions that never stopped intriguing me. Special thanks to CRAIG GENTRY for riding me every day to work for the whole summer and showing me his unique Craig-style humor besides his unique Craig-style research.

I am grateful to professors at Cornell, especially THORSTEN JOACHIMS, JOHN KLEINBERG, ROBERT KLEINBERG, DEXTER KOZEN, and EVA TARDOS, for years of education and inspiration. In addition, I want to thank EVA TARDOS for listening patiently to my confusions and questions on research and life, and sharing with me her personal experience and opinions.

My research has also benefited a lot from interactions with other researchers including RAN CANETTI, KAI-MIN CHUNG, DANA DACHMAN-SOLED, HU FU, PARIKSHIT GOPALAN, TONIANN PITASSI, ELIZABETH QUAGLIA, GUY ROTHBLUM, HOTECK WEE, DANIAL WICHS, and SERGEY YEKHANIN.

I would like to thank my best friends at Cornell for their understanding, support and love: FATIMA MAHMOOD, YUE GAO, CRISTIAN DANESCU. Without them, the journey would have been monochromatic and dry.

Special thanks to RAFAEL PASS for many helpful suggestions to the writing of this thesis, and RAN CANETTI and SHAFI GOLDWASSER for their generous support during the writing of this thesis.

The last but not the least, thank my parents CUNDE LIN and WENWEI ZHOU, and my sister LIJIA LIN and brother-in-law YONG ZOU for everything.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Secure Multi-Party Computation

The notion of *secure multi-party computation* allows m mutually distrustful parties to securely compute (or, *realize*) a functionality $f(\bar{x})$ of their corresponding private inputs $\bar{x} = x_1, ..., x_m$, such that party P_i receives the i^{th} component of $f(\bar{x})$. Loosely speaking, the security requirements are that the output of each party is distributed according to the prescribed functionality—this is called *correctness*—and that even malicious parties learn nothing more from the protocol than their prescribed output—this is called *privacy*. These properties should hold even in case that an arbitrary subset of the parties maliciously deviates from the protocol.

The above intuitive security requirements of correctness and privacy is later formalized using the *simulation paradigm*, originally developed for capturing the security of encryption and then extended to Zero-Knowledge [GM84, GMR89]. The idea is to say that a protocol π securely realizes f if running π *"emulates"* an idealized process where all parties secretly provide inputs to an imaginary trusted party that computes f and returns the outputs to the parties; more precisely, any "harm" done by a polynomial-time adversary in the real execution of π, could have been done even by a polynomial-time adversary (called a *simulator*) in the ideal process. The simulation paradigm provides strong security guarantees: It ensures that running the protocols is "as good as" having a trusted third party computing the functionality for the players, and an adversary participating in the real execution of the protocols does not gain any "computational advantage" over the simulator in the ideal process (except from polynomial time advantage). We call this definition *basic security.*

Soon after the conceptualization of secure multi-party computation, strong results were established: General constructions of secure computation protocols that satisfy basic security were developed for virtually all multi-party functionalities [Yao86, GMW87]. These constructions require only authenticated communication and are based on the existence of enhanced trapdoor permutations (TDPs).

However, the original constructions of secure computation protocols take many rounds of communication, specifically $O(m)$ rounds, where m is the number of players in the execution. Subsequent works improved the round-complexity by making stronger assumptions. Katz, Ostrovsky, and Smith [KOS03] obtained a $O(\log n)$-round protocol assuming TDPs and dense-crypto systems. By additionally assuming the existence of hash-function collision-resistant against circuits of sub-exponential size (and non-black-box technique), they also obtained a $O(1)$-round protocol. The latter results was subsequently improved by Pass [Pas04], showing the existence of a $O(1)$-rounds protocol assuming only TDPs and (standard) collision resistant hash functions (but still using non-black box techniques). But so far, no (asymptotic) improvement to the round-complexity of multi-party computation has been established assuming only TDPs, leaving open the following question:

> **Question 1:** *Can we construct* sub-linear-round *secure computation protocols assuming only TDPs ?*

1.2 Concurrent security

A more fundamental problem with the traditional study on basic security is that the setting in which it investigated secure multi-party computation, however, only allowed the execution of a single instance of the analyzed protocol in isolation; this is the so-called *stand-alone* setting. However, in modern networks like the Internet, protocols are never run in isolation. A more realistic setting, is one which allows the concurrent execution of many instances of different protocols; this is the so-called *concurrent setting*. In the concurrent setting, new security risk arises: Adversary may launch a *coordinated attack* that interleaves many instances of different protocols and create "rogue connection" between them.

Unfortunately, the notion of basic security and its solution does not provide security guarantees in the concurrent setting. It is thus imperative to develop a notion of security and solutions for the concurrent setting. To prevent coordinated attacks, any meaningful notion of security should provide the following basic and, necessary, guarantee.

> **Concurrent Security:** The security properties, correctness and privacy, of the *analyzed protocol* should remain valid even when it is executed concurrently with instances of many other, potentially *unknown*, protocols and are susceptible to coordinated attacks against multiple instances.

Concurrent security considers the composition of the analyzed protocol with arbitrary other protocols in a dynamic execution environment like the Internet. A similar composition occurs in complex software systems, where different protocols co-exist and may interact with each other in a large system. A natural desideratum that arises is,

> **Modular analysis:** The notion of security should support designing composite protocols in a modular way, while preserving security. That is, there should be a way to deduce security properties of the overall protocol from security properties of its components. This is essential for asserting security of complex protocols.

In the literature, the strongest and also the most realistic formalization of concurrent security is the notion of Universal Composability (UC): It considers the concurrent execution of an unbounded number of instances of the analyzed protocol, in an arbitrary, and adversarially controlled, network environment. It also supports modular analysis of protocols. But, these strong properties come at a price: Many natural functionalities cannot be realized with UC security in the *plain model,* where players only have access to authenticated communication channels. In fact, strong impossibility result has been established showing that UC security for general functionalities is infeasible in the plain model [CF01, CKL03]; to make the state of affairs worse, even when leaving out the requirement for support to modular analysis, the infeasibility result still holds for protocols satisfying only the concurrent security guarantees of UC security [Lin04].

To circumvent the broad impossibility results, two distinct veins of research can be identified in the literature.

Trusted set-up models: A first vein of work initiated by Canetti and Fischlin [CF01] and Canetti, Lindell, Ostrovsky and Sahai [CLOS02] (see also e.g., [BCNP04, CDPW07, KLP05, CPS07]) considers constructions of UC-secure protocol using various trusted infrastructure (or set-up), where the parties have limited access to a trusted entity (e.g., a trusted entity that publishes a common reference string chosen randomly from a pre-determined distribution). A large body of works (e.g. [CLOS02, BCNP04, KLP05, CPS07, GO07, Kat07, CDPW07]) show that with the appropriate trusted set-ups, UC-security becomes feasible.

However, in many situations, trusted set-up is hard to come by (or at least expensive). An important question is to identify the weakest possible set-up that allows us to obtain general feasibility results for UC security.

Relaxed UC security: Another vein of work considers relaxed notions of security such as *super-polynomial time simulation* (SPS) [Pas03a, PS04, BS05] and *angel-based security* [PS04, MMY06][1]. These works are relaxation of the UC security and only provide weak guarantees on the "computational advantage" gained by an adversary in a concurrent execution of protocols.

Still, for many applications, these relaxed notions of security provide an adequate level of security. Furthermore, protocols satisfying these notions are very attractive in practice: They use no trusted set-ups other than authenticated communication, and provide a meaningful notion of security in the concurrent setting. The only drawback is that, known constructions are based on either strong super-polynomial time hardness assumptions or specific and non-standard hardness assumptions.

In summary, the current state-of-the-art exhibits the following curious and unsatisfactory phenomenon: Whereas basic security can be realized in the plain

[1]In the literature, there is another relaxed notion of security, *input indistinguishability*[MPR06]. Their security guarantee has the same flavor as the notion of witness-indistinguishability for interactive proof systems. In this thesis, we, however, focus only on the stronger, simulation-based security guarantees.

model under standard hardness assumptions, so far, *no notion of concurrent security is realized under the same conditions*—known constructions all have to rely on some *additional "trust"*, either in the form of trusted set-ups or strong hardness assumptions. (The latter is just another form of trust that some hypothesis is true).

This leaves open the following fundamental question:

> **Question 2:** *Is additional "trust" inherent for concurrent security, or, can we achieve a meaningful notion of concurrent security without it?*

1.3 Our Contribution

In this thesis, we answer the above two questions affirmatively.

First, we construct constant-round secure computation protocols satisfying basic security assuming only TDPs. In fact, our result applies to the stronger UC security: We construct constant-round UC-secure protocols in essentially all previously considered trusted set-up models (e.g. [CLOS02, BCNP04, KLP05, CPS07, GO07, Kat07, CDPW07]) assuming only TDPs. Furthermore, we obtain an essentially *tight* characterization on the "minimal" trusted set-up under which UC security is feasible.

Second, we propose a new notion of concurrent security, that provides meaningful security guarantees in the concurrent setting (in particular, it implies SPS security), and show that protocols satisfying this notion can be constructed in the plain model under standard hardness assumptions (specifically, the existence of constant-round stand-alone secure semi-honest Oblivious Transfer (OT) protocols).

We obtain the above results by crucially relying on two notions of commitment schemes that ensures "input independence".

1.4 Input Independence

In the context of secure computation, it is important to ensure that players behave independently of each other; particularly, the input of a player should be independent of the private inputs of other players. We study two notions of commitments that ensures input independence: Non-malleable commitments introduced by [DDN91] and chosen-ciphertext-attack secure commitments introduced in this thesis.

1.4.1 Non-Malleable Commitments

Roughly speaking, a commitment scheme is often described as the "digital" analogue of sealed envelopes that enables a *sender or committer* to commit itself to a value while keeping it secret from the *receiver*; this is called the *hiding* property. Furthermore, the commitment is *binding*, and thus 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 stage. Applications of commitments range from coin flipping [Blu83] to the secure multi-party computation [GMW87].

For many applications, however, the most basic security guarantees of commitments are not sufficient. For instance, the basic definition of commitments does not rule out an attack where an adversary, upon seeing a commitment to a specific value v, is able to commit to a related value (say, $v - 1$), even though it does not know the actual value of v. This kind of attack might have devastating consequences if the underlying application relies on the *independence* of committed values (e.g., consider a case in which the commitment scheme is used for securely implementing a contract bidding mechanism). Indeed, as we will see in Section 1.5, such independence is crucial for the task of general secure multiparty computation [GMW87]. The state of affairs is even worsened by the fact that many of the known commitment schemes are actually susceptible to this kind of attack. In order to address the above concerns, Dolev, Dwork and Naor (DDN) introduced the concept of *non-malleable commitments* [DDN91]. Loosely speaking, a commitment scheme is said to be non-malleable if it is infeasible for an adversary to "maul" a commitment to a value v into a commitment to a related value $\tilde{v}$.

More precisely, we consider a *man-in-the-middle* (MIM) attacker that participates in two concurrent execution of a commitment scheme $\langle C, R\rangle$; in the "left" execution it interacts with an honest committer (running C); in the "right" execution it interacts with an honest receiver (running R). Additionally, we assume that the players have n-bit identities (where n is polynomially related to the security parameter), and that the commitment protocol depends on the identity of the committer; we sometimes refer to this as the identity of the interaction. Intuitively, $\langle C, R\rangle$ being non-malleable means that if the identity of the right interaction is different than the identity of the left interaction (i.e., A does not use the same identity as the left committer), the value A commits to on the right does not depend on the value it receives a commitment to on the left; this is formalized by requiring that for any two values v_1, v_2, the values A commits to after receiving a left commitment to v_1 or v_2 are indistinguishable.

Dolev, Dwork and Naor [DDN91] constructed the first non-malleable commitment. Their protocol is in the plain model and does not assume any additional assumption other than the existence of commitment schemes, which is necessary. The only drawback of their construction is that the protocol takes $\Omega(\log n)$ communication rounds, where $n \in N$ is the security parameter. Since the original work [DDN91], non-malleable commitments have been extensively studied in the literature; the main question has been to determine the number of communication rounds needed for non-malleable commitments.

DiCrenenzo, Ishai and Ostrovsky [CKOS01] and follow-up works in e.g., [CKOS01, CIO01, CF01, FF09, DG03] showed how to improve the round-complexity of the DDN construction when assuming the existence of some trusted set-up; in such models *non-interactive* (i.e., single message) non-malleable commitments based on only one-way function are known [DG03]. The first improvement to the round-complexity of the DDN construction without any trusted infrastructure came more than a decade later. Following the ground-breaking work by Barak on *non-black-box simulation* [Bar01], in 2002, Barak [Bar02] presented a constant-round protocol for non-malleable commitments; the security of this protocol however relies on the existence of trapdoor permutations and hash functions that are collision-resistant against circuits of sub-exponential size. A few years later, Pass and Rosen [PR05b] (relying on a technique from [Pas04]), showed that collision resistant hash functions secure against polynomially-sized circuits are sufficient to obtain a constant-round protocol. Next, Pandey, Pass and Vaikuntanathan [PPV08] provided a construction of a non-interactive non-malleable commitment based on a new hardness assumption with a strong non-malleability flavour; in contrast to the earlier constant-round constructions, their protocol has a black-box proof of security.

But, despite two decades of research, there have been no improvements over the original DDN construction, when only assuming the existence of one-way functions, leaving open the question:

> *Does there exist a* sub-logarithmic-round *non-malleable commitment scheme, assuming only one-way functions?*

In this thesis we settle the round-complexity of non-malleable commitments: we present a constant-round protocol in the plain model that is based on the assumption of one-way functions, and has a black-box proof of security. Since the existence of commitment schemes already implies the existence of one-way functions (cl. [IL89]), we have:

Theorem. *Assume the existence of a commitment scheme. Then, there exists a constant-round non-malleable commitment scheme with a black-box proof of security.*

Concurrent Non-Malleability: The original notion of non-malleability considers a MIM attacker participating in a single execution on the left and a single execution on the right. Already the original DDN paper suggested that a stronger notion of non-malleability—*concurrent non-malleability*—where the MIM may participate in an unbounded number of executions on both the left and the right, is desirable. Pass and Rosen [PR05a] provided the first construction of a concurrently non-malleable commitment scheme; their scheme only has a constant number of rounds but relies on the existence of claw-free permutations (and non-black-box techniques). Subsequently, Lin, Pass and Venkitasubramaniam [LPV08] provided an $O(n)$-round construction based on one-way functions. As we show, our construction of non-malleable commitment is also concurrently non-malleable.

Robust Non-Malleability: In this thesis we also introduce a new notion of non-malleability: *robust non-malleability*. Roughly speaking, whereas traditional non-malleability considers a scenario where a MIM participates in the *same* commitment protocol on the left and the right, r-robust non-malleability considers a notion of non-malleability for commitments where the MIM attacker participates in an arbitrary r-round protocol on the left, and the commitment protocol on the right. Robustness is useful when using non-malleable commitments as sub-protocols within larger protocols (see Section 1.5). As we show, for any constant r, our protocol can be made r-robust while still remaining constant-round.

Thus summarizing the above discussion, we have

Theorem. *Assume the existence of a commitment scheme. Then, for any constant r, there exists a constant-round commitment scheme that is r-robust concurrently non-malleable with a black-box proof of security.*

1.4.2 Chosen-Ciphertext-Attack Secure Commitments

In this thesis, we propose a new and stronger way of capturing input independence through the notion of *adaptive hardness* [PPV08]: Namely, a primitive is *adaptively hard*, if its security holds even in face of adversaries that have adaptive access to some help that breaks security of *other instances* of the primitive for them. For instance, we may ask the question whether factoring is adaptively hard, that is, whether it is hard to factor the product of two big random primes, even if the adversary has access to some help that factors products of *other* primes of its choice. It has been observed in previous works [PS04, MMY06] that adaptive hardness is useful for achieving a relaxed notion UC security, angel-based security; but, these works all rely on certain, quite specific *adaptive hardness* assumptions. However, such assumptions appear to be qualitatively stronger than non adaptive ones. A natural question is then,

> *Can an adaptive hardness property be based on a standard, in particular non adaptive, assumption?*

We answer this question positively: We formulate a useful adaptive hardness property in the context of commitment schemes—called the chosen-ciphertext-attack (CCA) secure commitments—and show that this adaptive hardness property can be based on a standard (non adaptive) assumption, specifically the existence of one way functions.

Roughly speaking, a commitment scheme is said to be *CCA-secure* if the hiding property of the commitment holds even if the attacker receives help from a "decommitment oracle" that extracts for the attacker decommitment information of commitments of its choice, provided that the attacker does not query the oracle on the exact same commitment it is supposed to violate the hiding property of.

It is not hard to construct CCA-secure commitments using trusted set-up (by e.g., relying on known constructions of CCA-secure encryption schemes).

It is also quite simple to construct a CCA-secure commitment scheme under an adaptive hardness assumption (such as the existence of adaptively-secure one-way permutations—namely one-way permutations that remain uninvertible even if the adversary has access to a inversion oracle) [PPV08].[2]

Instead, we show how to construct a CCA-secure commitment scheme based only on one-way functions, and without any trusted set-up.

Theorem. *Assume the existence of one-way functions. Then, for any positive constant ε, there exists an $O(n^{\varepsilon})$-round CCA-secure commitment scheme.*

As far as we know this yields the first non-trivial primitive whose "adaptive hardness" can be proven based on standard assumptions without set-up. Note that many standard cryptographic primitives (such as pseudo-random functions [GGM86], signatures [GMR89], and CCA-secure encryption [RS91]) consider adaptive attacks where an adversary has access to an oracle breaking the primitive. However, all these cryptographic primitives rely on some trusted set-up (in the case of signatures and CCA-secure encryption, the public-key used needs to be well-formed, whereas in the case of pseudo-random function, the "seed" needs to be uniform and perfectly hidden from the adversary).

Robust CCA-Security: We also introduce a strengthening of the CCA-security, *robust CCA-security*: Roughly speaking, a CCA-secure commitment is said to be *t-robust*, if any *t*-round interaction with an adversary having help from the decommitment oracle can be simulated by another adversary without access to the decommitment oracle; intuitively, this means having access to the decom-

[2][PPV08] considered a variant of the notion of CCA-security for non-interactive and perfectly binding commitment scheme (called adaptively-secure commitments). We have extended this definition to general commitment schemes.

mitment oracle does not give an adversary any advantage when participating in the execution of a t-round protocol. A CCA-secure commitment is said to be *robust* if it is t-robust for all constant t. As the notion of robustness for non-malleability, robustness for CCA-security is useful when putting CCA-secure commitments as a sub-protocol in other protocols. (See Section 1.6.) Furthermore, our $O(n^{\varepsilon})$-round CCA-secure commitment scheme is robust with respect to any constant-round protocols.

1.4.3 Connection between the Two Commitments

CCA-security can be viewed as a natural strengthening of *concurrent non-malleability* [DDN00, PR03, LPV08]—roughly speaking, a commitment scheme is concurrently non-malleable if it is CCA-secure with respect to restricted classes of adversaries that only ask a single parallel—i.e., non-adaptive—query to the decommitment oracle after it has received the commitment it is supposed to violate the hiding property of.

Interestingly, we can go from non-malleability to CCA-security as well. Our construction of $O(n^{\varepsilon})$-round CCA-secure commitment shows a connection between CCA-secure commitments and non-malleable commitments, as it crucially relies on a key technique, the so-called message scheduling technique, used in the original construction of non-malleable commitment in [DDN91]. We formalize this connection and show a generic compilation technique that transforms any robust non-malleable commitment scheme into a robust CCA-secure commitment scheme, relying additionally on a concurrent extraction strategy introduced in the context of concurrent $\mathcal{ZK}$ protocols (e.g., [RK99, KP01, PRS02]).

This connection immediately yields new constructions of CCA-secure commitments. Combining our constant-round non-malleable commitments and the concurrent extraction strategy of [PRS02], we obtain a $\omega(\log n)$-round robust CCA-secure commitment scheme based on one-way functions. This is essentially *optimal*: In a companion lower bound, we show that any 4-robust CCA-secure commitments must have $\tilde{\Omega}(\log n)$ rounds. This is established by showing that the existence of k-round 4-robust CCA-secure commitments implies the existence of $k + 4$-round concurrent $\mathcal{ZK}$ arguments, which is shown to have $\tilde{\Omega}(\log n)$ rounds [CKPR01].

Theorem. *Assume the existence of one-way functions. Then, there exists an $\omega(\log n)$-round robust CCA-secure commitment scheme. Furthermore, any 4-robust CCA-secure commitments have $\tilde{\Omega}(\log n)$ rounds.*

Finally, inspired by a work of [PV08] that circumvents the lower bound on the round-complexity of concurrent $\mathcal{ZK}$ protocols by considering quasi-polynomial time adversaries, we show that when considering robust CCA-secure commitments for quasi-polynomial time adversaries, constant-round construction becomes possible. Our constant-round construction, again, follows from the generic compilation technique by plugging in our constant-round non-malleable commitments and the concurrent extraction strategy implicitly defined in [PV08].

Theorem. *Assume the existence of quasi-polynomial time hard one-way functions. Then, for any constant r, there exists an constant-round r-robust CCA-secure commitment scheme for quasi-polynomial time adversaries.*

1.5 Constant-Round Secure Multi-Party Computation

In a novel work by Lin, Pass and Venkitasubramaniam [LPV09], they present a *unified framework* for achieving UC security. Their framework put forward a template for constructing UC secure protocols from two much simpler primitives: *UC-puzzles and 4-robust non-malleable commitments*, based on the existence of enhanced trapdoor permutations. They demonstrated that *constant-round* UC-puzzles can be obtained "easily" from essentially all previous set-up assumptions (e.g. [CLOS02, BCNP04, KLP05, CPS07, GO07, Kat07, CDPW07]) as well as in relaxed security models like super-polynomial time simulation (SPS). This not only leads to conceptually simply solutions, but also allows weakening, and in some models minimizing, the set-up assumptions used in previous works (as well as the hardness assumptions these works are base on). In fact, the notion of UC-puzzles is essentially a tight characterization of the set-up assumptions needed to obtain UC security. In other words, the existence of a UC-puzzle is essentially the *"minimal"* trusted set-up for achieving UC.

Furthermore, the round-complexity of the UC secure protocols produced by the unified framework of [LPV09] depends solely on the round-complexity of the non-malleable commitments they rely on; specifically, the number of communication rounds is of the same order as that of the non-malleable commitments. Therefore, plugging our new constant-round 4-robust non-malleable commitment scheme into the unified framework, we directly obtain constant-round UC-secure protocols under "minimal" set-up.

Theorem (Informally stated)**.** *Assume the existence of enhanced trapdoor permutations. Then, for virtually every functionality $\mathcal{F}$, there exists a constant-round protocol* Π *that securely realizes $\mathcal{F}$ with UC security under "minimal" trusted set-up.*

Obtaining Constant-Round Secure Computation Protocols. The unified framework of [LPV09] also encompasses the stand-alone setting: It is easy to construct a constant-round UC puzzle in the stand-alone setting without relying on any trusted set-up. Therefore, we obtain directly as a corollary of the above theorem that there exist constant-round secure computation protocols for general functionalities satisfying basic security, based only on TDPs. This yields the first improvement on the round-complexity of secure multi-party computation over the original works of [Yao86, GMW87], without resorting to stronger assumptions.

Theorem (Informally stated)**.** *Assume the existence of enhanced trapdoor permutations. Then, for virtually every functionality $\mathcal{F}$, there exists a constant-round protocol* Π *that securely realizes $\mathcal{F}$ with basic security.*

1.6 Concurrent Security without Additional "Trust"

Our next goal is to find a notion of security that provides guarantees in the concurrent setting, and yet, is realizable in the plain model under standard assumptions. In the literature, there are two relaxed notions of UC security that are realizable in the plain model.

Super-Polynomial Time Simulation. Security with super-polynomial simulators (SPS) [Pas03a] is a relaxation of UC security. Recall that UC guarantees that any "harm" done by a polynomial-time adversary in a concurrent execution of the protocol could have been done by a polynomial-time adversary (called a *simulator*) in the ideal process. SPS allows the simulator to run in super-polynomial time. Informally, this corresponds to guaranteeing that "any poly-time attack that can be mounted against the protocol can also be mounted in the ideal execution—albeit with super-polynomial resources." For many applications, SPS provides an adequate level of security, and it guarantees concurrent security (with super-polynomial simulation). However, SPS security is not a convenient basis for modular analysis of protocols.

Angel-based UC security. Angel-based UC security [PS04] is a framework for notions of security that provides similar security guarantees as SPS and at the same supports modular analysis. Specifically, angel-based security considers a model where both the adversary and the simulator have access to an oracle (an "angel") that allows some judicious use of super-polynomial resources. Since the angels can be implemented in super-polynomial time, for any angel, angel-based security implies SPS security. Furthermore, akin to UC security, angel-based UC security, with any angel, can be used as a basis for modular analysis of protocols.

Both SPS and angel-based security implies meaningful notions of concurrent security. Unfortunately, so far, all known constructions are based on either strong super-polynomial time hardness assumption [Pas03a, BS05, LPV09]) or non-standard and specific hardness assumptions [PS04, MMY06].

UC with Super-Polynomial Time Helpers. In this work, we propose a new notion of security, called UC with super-polynomial time helpers. This notion is inspired by and very similar to the angel-based security where both the adversary and the simulator have access to a helper that provides some super-polynomial time help through a limited interface. Like angel-based security, UC security with super-polynomial time helpers implies SPS security. But, unlike angel-based security where angels are basically non-interactive and stateless, the helpers in our model are *highly interactive and stateful*. The qualitative difference between the non-interactive angels and our interactive helpers is that the former is only known to be implemented based on non-standard and specific assumptions, whereas, as we show, interactive helpers can be implemented based on the minimal assumption of one-way functions.

The notion of UC with super-polynomial time helpers is formalized within the externalized UC (EUC) framework of [CDPW07]. Roughly speaking, this framework is identical to standard UC security as in [Can00], except that all parties have access to an additional global entity. We use this global entity to model interactive helpers that, as in [PS04], interacts only with the adversary and the simulator[3]. We call the global entity, $\mathcal{H}$, a *helper functionality*, and denote the corresponding notion of security $\mathcal{H}$-EUC security.

We implement our helper functionality $\mathcal{H}$ using robust CCA-secure commitments. Given a robust CCA-secure commitment scheme, $\mathcal{H}$ is simply its decommitment oracle. In the real execution of the protocol, even though the adversary have access to the decommitment oracle, the CCA-security ensures that it still cannot break the security of commitments made by other honest players. On the other hand, having access to the decommitment oracle allows the simulator in the ideal process to extract the decommitment information of commitments made by the adversary. This disparity in capability between the adversary and

[3]More precisely, it also interacts with the environment in the EUC framework. In [CDPW07], the global entity is used to model global trusted set-up such as a reference string or strong public-key infrastructure. Here, in contrast, we use it to model an interactive helper that interacts only with the corrupted parties and the environment. This means that the actual protocol uses no trusted infrastructure, and the global entity becomes a means for relaxing the security requirement.

simulator makes simulation relatively easy, and allows us to securely implement the ideal commitment functionality $\mathcal{F}_{\mathsf{com}}$ with $\mathcal{H}$-EUC security.

At this point, if we are willing to assume super-polynomial time hardness assumptions, we can already obtain general $\mathcal{H}$-EUC security by directly combining our protocol implementing $\mathcal{F}_{\mathsf{com}}$ with previous results [CLOS02, BMR90, IPS08]. By crucially relying on the robustness of our CCA-secure commitments, we eliminate the use of such super-polynomial time hardness assumptions, and base general $\mathcal{H}$-EUC security on the existence of constant-round stand-alone secure semi-honest OT protocols. Therefore, we obtain,

Theorem (Informally Stated)**.** *Assume the existence of constant-round stand-alone secure semi-honest OT protocols. Then, there exists a sub-exponential-time computable interactive machine $\mathcal{H}$ such that for virtually every polynomial-time functionality $\mathcal{F}$, there exists a protocol that realizes $\mathcal{F}$ with $\mathcal{H}$-EUC security, in the plain model.*

This gives the first construction of protocols achieving a meaningful notion of concurrent security, in particular implying SPS-security, in the plain model based only on standard assumptions.

Furthermore, when using instead our construction of constant-round CCA-secure commitment scheme for quasi-polynomial time adversaries, we obtain constant-round protocols satisfying concurrent security and supports modular analysis, based on the existence of constant-round stand-alone semi-honest OT protocols secure for quasi-polynomial time adversaries.

Theorem (Informally Stated)**.** *Assume the existence of constant-round stand-alone semi-honest OT protocols secure for quasi-polynomial time adversaries. Then, there exists a sub-exponential-time computable interactive machine $\mathcal{H}$ such that for virtu-*

ally every polynomial-time functionality $\mathcal{F}$, there exists a constant-round protocol that realizes $\mathcal{F}$ with $\mathcal{H}$-EUC security, in the plain model.

1.7 Outline

In this thesis we investigate secure multi-party computation in the concurrent setting. Our contribution is two-fold: First, we improve the round complexity of secure computation protocols in the stand-alone setting to a constant, and second, we enable secure multi-party computation in the concurrent setting by achieving a meaningful notion of concurrent security in the plain model based on standard assumptions. In this journey, we construct round-optimal non-malleable commitments and CCA-secure commitments, which are of independent interests.

Chapter 2: Preliminaries. We introduce basic notation and recall basic definitions that will be used throughout the thesis.

Chapter 3: Non-Malleability and Constant-Round Secure Computation. We present our construction of a constant-round non-malleable commitment scheme (in the plain model) based on one-way functions. As discussed above, the construction, when plugged in the unified framework of [LPV09], yields constant-round UC secure protocols in various trusted set-up and relaxed security models.

Chapter 4: CCA-Security and Concurrent Security without Additional "Trust". We introduce the notion of robust CCA-secure commitments, and present our first construction of robust CCA-secure commitments based on one-way functions. Then, using this construction, we achieve $\mathcal{H}$-EUC security with an interactive helper acting as the decommitment oracle of our CCA-secure commitments, in the plain model based on standard assumptions.

Chapter 5: From Non-Malleability to CCA-Security. We present a generic compilation technique that transforms any non-malleable commitment scheme and a concurrent extraction strategy of a special type, into a robust CCA-secure commitment scheme. This compilation technique leads to two new constructions of robust CCA-secure commitments with improved round-efficiency: The first has $\omega(\log n)$ rounds and is based on one-way functions, and the second has only constant rounds but is based on one-way functions hard for quasi-polynomial time (and is for quasi-polynomial time adversaries). Furthermore, we show that our first construction is round optimal, by establishing a lower bound that any 4-robust CCA-secure commitment protocol (for polynomial-time adversaries) must have $\tilde{\Omega}(\log n)$ rounds.

CHAPTER 2

PRELIMINARIES

2.1 Basic Notation

2.1.1 General Notation

We employ the following general notation.

Integer and String representation. We denote by N the set of natural numbers: 0, 1, 2, ... Unless otherwise specified, a natural number is presented in its binary expansion (with no *leading* 0s) whenever given as an input to an algorithm. If $n \in N$, we denote by 1^n the unary expansion of n (i.e., the concatenation of n 1's). We denote by $\{0,1\}^n$ the set of n-bit long string, by $\{0,1\}^*$ the set of binary strings, and by $[n]$ the set $\{1,..,n\}$.

We denote the concatenation of two strings x and y by $x||y$ (or more simply by xy). If α is a binary string, then $|\alpha|$ denotes α's length and $\alpha_1 \cdots \alpha_i$ denotes α's i-bit prefix.

Probabilistic notation. We employ the following probabilistic notation from [GMR88]. We focus on probability distributions $X : S \rightarrow R^+$ over finite sets S.

Probabilistic assignments. If D is a probability distribution and p a predicate, then "$x \overset{R}{\leftarrow} D$" denotes the elementary procedure consisting of choosing an element x at random according to D and returning x.

Probabilistic experiments. Let p be a predicate and $D_1, D_2, \ldots$ probability distributions, then the notation $\Pr\left[x_1 \overset{R}{\leftarrow} D_1;\ x_2 \overset{R}{\leftarrow} D_2;\ \ldots\ :\ p(x_1, x_2, \ldots)\right]$ denotes the probability that $p(x_1, x_2, \ldots)$ will be true after the ordered execution of the probabilistic assignments $x_1 \overset{R}{\leftarrow} D_1;\ x_2 \overset{R}{\leftarrow} D_1;\ \ldots$

New probability distributions. If $D_1, D_2, \ldots$ are probability distributions, the notation $\{x \overset{R}{\leftarrow} D_1; y \overset{R}{\leftarrow} D_2; \cdots\ :\ (x, y, \cdots)\}$ denotes the new probability distribution over $\{(x, y, \cdots)\}$ generated by the ordered execution of the probabilistic assignments $x \overset{R}{\leftarrow} D_1,\ y \overset{R}{\leftarrow} D_2, \cdots$.

Probability ensembles. Let I be a countable index set. A *probability ensemble indexed by I* is a vector of random variables indexed by I: $X = \{X_i\}_{i \in I}$.

In order to simplify notation, we sometimes abuse notation and employ the following "short-cut": Given a probability distribution X, we let X denote the random variable obtained by selecting $x \leftarrow X$ and outputting x.

Algorithms. We employ the following notation for algorithms.

Deterministic algorithms. By an algorithm we mean a Turing machine. We only consider *finite* algorithms, i.e., machines that have some fixed upper-bound on their running-time (and thus always halt). If M is a deterministic algorithm, we denote by $\text{STEPS}_{M(x)}$ the number of computational steps taken by M on input x. We say that an algorithm M has time-complexity $\text{TIME}_M(n) = t(n)$, if $\forall x \in \{0, 1\}^*\ \text{STEPS}_{M(x)} \leq t(|x|)$. (Note that time complexity is defined as an upper-bound on the running time of M *independently* of its input.) We write $M(x\|*)$, or $M(x, *)$ to denote the machine that proceeds identically to M but with a part of its input fixed to x.

Probabilistic algorithms. By a probabilistic algorithms we mean a Turing machine that receives an auxiliary random tape as input. If M is a probabilistic algorithm, then for any input x, the notation "$M_r(x)$" denotes the output of the M on input x when receiving r as random tape. We let the notation "$M(x)$" denote the probability distribution over the outputs of M on input x where each bit of the random tape r is selected at random and independently. (Bote that this is a well-defined probability distribution since we only consider algorithms with finite running-time.) We sometimes abuse the notation and use $M(x)$ to refer to the probablstic procedure of running machine M on input x with uniformly and randomly chosen random tape.

We denote by $\mathcal{PPT}$ probabilistic polynomial time Turing machines and $\mathcal{PQT}$ probabilistic quasi-polynomial time Turing machines.

Oracle algorithms. Given two algorithms M, A, we let $M^A(x)$ denote the output of the algorithm M on input x, when given oracle access to $A(\cdot)$. We denote by $\mathsf{query}(M^A(x))$ the random variable representing the set of queries made by oracle machine M to oracle A on input x.

Negligible functions. The term "negligible" is used for denoting functions that are asymptotically smaller than the inverse of any fixed polynomial. More precisely, a function $\mu(\cdot)$ from non-negative integers to reals is called *negligible* if for every constant $c > 0$ and all sufficiently large n, it holds that $\mu(n) < n^{-c}$.

2.1.2 Protocol Notation

We assume familiarity with the basic notions of an *Interactive Turing Machine* [GMR89] (ITM for brevity) and a *protocol*. Briefly, an ITM is a Turing Machine with a read-only *input* tape, a read-only *auxiliary input* tape, a read-only *random* tape, a read/write *work-tape*, a read-only communication tape (for receiving messages) a write-only communication tape (for sending messages) and finally an *output* tape. The content of the input (respectively auxiliary input) tape of an ITM A is called *the input* (respectively *auxiliary input*) *of* A and the content of the output tape of A, upon halting, is called *the output of* A.

A protocol $\langle A, B\rangle$ is a pair of ITMs that share communication tapes so that the (write-only) send-tape of the first ITM is the (read-only) receive-tape of the second, and vice versa. The computation of such a pair consists of a sequence of rounds $1, 2, ...$ In each round only one ITM is active, and the other is idle. A round ends with the active machine either halting —in which case the protocol ends— or by it entering a special *idle* state. The string m written on the communication tape in a round is called the *message sent* by the active machine to the idle machine.

In this thesis we consider protocols $\langle A, B\rangle$ where the inputs to A and B are of the form $(x||x_A)$ and $(x||x_B)$, which we also write as (x, x_A) and (x, x_B). The part of input x that is shared between A and B is called the *common input* of A and B, and the parts that are not shared x_A and x_B are called the *private inputs* of A and B. We make use of the following notation for protocol executions.

Rounds. We say that a protocol has $r(n)$ rounds or $r(n)$ messages (or simply is an $r(n)$-round protocol) if the protocol $\langle A, B\rangle$ consists of $r(n)$-rounds of communication between A and B when executed on common input $x \in \{0,1\}^n$.

Executions, transcripts and views. Let M_A, M_B be vectors of strings $M_A = \{m_A^1, m_A^2, ...\}$, $M_B = \{m_B^1, m_B^2, ...\}$ and let $x, x_1, x_2, r_1, r_2, z_1, r_2 \in \{0,1\}^*$. We say that the pair $((x, x_1, z_1, r_1, M_A), (x, x_2, z_2, r_2, M_B))$ is an execution of the protocol $\langle A, B \rangle$ if, running ITM A on common input x, private input x_1, auxiliary input z_1 and random tape r_1 with ITM B on x, x_2, z_2 and r_2, results in m_A^i being the i'th message received by A and in m_B^i being the i'th message received by B. We also denote such an execution by $\langle A_{r_1}(x_1, z_1), B_{r_2}(x_2, z_2) \rangle(x)$.

In an execution $((x, x_1, z_1, r_1, M_A), (x, x_2, z_2, r_2, M_B)) = (V_A, V_B)$ of the protocol $\langle A, B \rangle$, we call V_A the *view of A* (in the execution), and V_B the *view of B*. We let $\mathsf{view}_A[\langle A_{r_1}(x_1, z_1), B_{r_2}(x_2, z_2) \rangle(x)]$ denote A's view in the execution $\langle A_{r_1}(x_1, z_1), B_{r_2}(x_2, z_2) \rangle(x)$, $\mathsf{view}_B[\langle A_{r_1}(x_1, z_1), B_{r_2}(x_2, z_2) \rangle(x)]$ B's view in the same execution, and $\mathsf{view}_{A,B}[\langle A_{r_1}(x_1, z_1), B_{r_2}(x_2, z_2) \rangle(x)]$ the joint view of A and B in the execution.

In an execution $((x, x_1, z_1 r_1, M_A), (x, x_2, z_2, r_2, M_B))$, the pair (M_A, M_B) is called the transcript of the execution.

Outputs of executions and views. If e is an execution of a protocol (A_1, A_2) we denote by $\mathsf{out}_C[e]$ the output of C, where C can be either A_1 or A_2, and by $\mathsf{out}_{A_1,A_2}[e]$ the joint output of or A_1, A_2. Analogously, if v is the view of A, we denote by $\mathsf{out}[v]$ the output of A in v.

Random executions. We denote by $\langle A(x_1, z_1), B_{r_2}(x_2, z_2) \rangle(x)$, $\langle A_{r_1}(x_1, z_1), B(x_2, z_2) \rangle(x)$ and $\langle A(x_1, z_1), B(x_2, z_2) \rangle(x)$ the probability distribution of the random variable obtained by selecting each bit of r_1 (respectively, each bit of r_2, and each bit of r_1 and r_2) randomly and independently, and then outputting $\langle A_{r_1}(x_1, z_1), B_{r_2}(x_2, z_2) \rangle(x)$. The corresponding probability distributions for view and out are analogously defined.

Counting ITM steps. Let A be an ITM and $v = (x, x', z, r, (m_1, m_2, ..m_k))$. Then by $\text{STEPS}_A(v)$ we denote the number of computational steps taken by A running on common input x, private input x', auxiliary input z, random tape r, and letting the i^{th} message received be m_i.

Time Complexity of ITMs. We say that an ITM A has time-complexity $\text{TIME}_A(n) = t(n)$, if for every ITM B, every common input x, every private inputs x_1, x_2 and every auxiliary inputs z_1, z_2, it holds that $A(x, x_1, z_1)$ *always* halts within $t(|x|)$ steps in an interaction with $B(x, x_2, z_2)$, regardless of the content of A and B's random tapes). Note that time complexity is defined as an upper-bound on the running time of A *independently* of the content of the messages it receives. In other words, the time complexity of A is the *worst-case* running time of A in *any* interaction.

2.2 Basic Notion

2.2.1 Basic Complexity Classes

We recall the definitions of the basic complexity classes $\mathcal{P}$, $\mathcal{NP}$ and $\mathcal{BPP}$.

The Complexity Class $\mathcal{P}$. We start by recalling the definition of the class $\mathcal{P}$, i.e., the class of languages that can be decided in (deterministic) polynomial-time.

Definition 1 (Complexity Class $\mathcal{P}$). *A language L is recognizable in (deterministic) polynomial-time if there exists a deterministic polynomial-time algorithm M such* *that $M(x) = 1$ if and only if $x \in L$. $\mathcal{P}$ is the class of languages recognizable in polynomial time.*

The Complexity Class $\mathcal{NP}$. We recall the class $\mathcal{NP}$, i.e., the class of languages for which there exists a proof of membership that can be verified in polynomial-time.

Definition 2 (Complexity Class $\mathcal{NP}$). *A language L is in $\mathcal{NP}$ if there exists a Boolean relation $R_L \subseteq \{0,1\}^* \times \{0,1\}^*$ and a polynomial $p(\cdot)$ such that R_L is recognizable in polynomial-time, and $x \in L$ if and only if there exists a string $y \in \{0,1\}^*$ such that $|y| \leq p(|x|)$ and $(x,y) \in R_L$.*

The relation R_L is called a *witness relation* for L. We say that y is a witness for the membership $x \in L$ if $(x,y) \in R_L$. We will also let $R_L(x)$ denote the set of witnesses for the membership $x \in L$, i.e.,

$$R_L(x) = \{y : (x,y) \in L\}$$

We let co-$\mathcal{NP}$ denote the complement of the class $\mathcal{NP}$, i.e., a language L is in co-$\mathcal{NP}$ if the complement to L is in $\mathcal{NP}$.

The Complexity Class $\mathcal{BPP}$. We recall the class $\mathcal{BPP}$, i.e., the class of languages that can be decided in *probabilistic* polynomial-time (with two-sided error).

Definition 3 (Complexity Class $\mathcal{BPP}$). *A language L is* recognizable in probabilistic polynomial-time *if there exists a probabilistic polynomial-time algorithm M such that*

- $\forall x \in L, \Pr[M(x) = 1] \geq 2/3$
- $\forall x \notin L, \Pr[M(x) = 0] \geq 2/3$

$\mathcal{BPP}$ is the class of languages recognizable in probabilistic polynomial time.

2.2.2 Indistinguishability

The following definition of (computational) indistinguishability originates in the seminal paper of Goldwasser and Micali [GM84].

Definition 4 (Indistinguishability)**.** *Let X and Y be countable sets. Two ensembles $\{A_{x,y}\}_{x\in X,y\in Y}$ and $\{B_{x,y}\}_{x\in X,y\in Y}$ are said to be **computationally indistinguishable over** X, if for every probabilistic "distinguishing" algorithm D whose running time is polynomial in its first input, there exists a negligible function $\nu(\cdot)$ so that for every $x \in X, y \in Y$:*

$$\left|\Pr\left[a \leftarrow A_{x,y} : D(x,y,a) = 1\right] - \Pr\left[a \leftarrow B_{x,y} : D(x,y,b) = 1\right]\right| < \nu(|x|)$$

*$\{A_{x,y}\}_{x\in X,y\in Y}$ and $\{B_{x,y}\}_{x\in X,y\in Y}$ are said to be **statistically close** over X if the above condition holds for all (possibly unbounded) algorithms D.*

2.3 Interactive Proofs and Arguments

We state the standard definitions of interactive proofs (introduced by Goldwasser, Micali and Rackoff [GMR89]) and arguments (introduced by Brassard, Chaum and Crepeau [BCC88]).

Definition 5 (Interactive Proof (Argument) System)**.** *A pair of interactive machines (P, V) is called an **interactive proof system** for a language L if machine V is* *polynomial-time and the following two conditions hold with respect to some negligible function $\nu(\cdot)$:*

- Completeness: *For every $x \in L$ there exists a (witness) string y such that*

$$\Pr\left[\mathsf{out}_V[\langle P(y), V\rangle(x)] = 1\right] = 1$$

- Soundness: *For every $x \notin L$, every interactive machine B and every $y \in \{0,1\}^*$*

$$\Pr\left[\mathsf{out}_V[\langle B(y), V\rangle(x)] = 1\right] \leq \nu(|x|)$$

In case that the soundness condition is required to hold only with respect to a computationally bounded prover, the pair (P, V) is called an interactive argument *system.*

Public-Coin Protocols. In certain applications of interactive proofs/ arguments it is desirable that the verifier only uses *public* random coins (i.e., all its random coins are also revealed the prover; such proofs/arguments are called public-coin or Arthur-Merlin[BM88]. Due to the fact that the verifier in a public-coin protocol only uses public random coins, we can without loss of generality only consider public-coin proofs/arguments having the following canonical structure:

- The verifier only sends random messages to the prover, and
- at the end of the interaction, the verifier determines whether to accept or not by applying a *deterministic* predicate to the transcript of all messages in the interaction.

Interactive proofs with efficient provers. For cryptographic applications it is necessary that the prover strategy can be implemented efficiently when given a witness.

Definition 6 (Efficient Provers). *Let (P, V) be an interactive proof (argument) system for the language $L \in \mathcal{NP}$ with the witness relation R_L. We say that (P, V) has an* **efficient prover** *if P is a probabilistic polynomial-time algorithm and the completeness condition of Definition 5 holds for every $x \in L$ and every $y \in R_L(x)$.*

In this thesis, when referring to an interactive proof/ argument system for a language $L \in \mathcal{NP}$, by default we mean that the system has efficient prover.

As an example of an interactive proof system with an efficient prover, consider the following "trivial" interactive proof system for a language $L \in \mathcal{NP}$, with witness relation R_L: on common input x, and auxiliary input $w \in R_L(x)$, the prover P simply send the witness w to verifier V. V accepts if and only if $(x, w) \in R_L$. Below we provide less "trivial" examples of interactive proof/argument systems that have other properties like witness indistinguishability, proof/argument of knowledge, special-soundness, zero-knowledge and concurrent zero-knowledge.

2.3.1 Witness Indistinguishable Proofs

The notion of *witness indistinguishability* ($\mathcal{WI}$) was introduced by Feige and Shamir in [FS90]. Roughly speaking, an interactive proof is said to be $\mathcal{WI}$ if the verifier's output is "computationally independent" of the witness used by the prover for proving the statement. In this context, we focus on languages $L \in \mathcal{NP}$ with a corresponding witness relation R_L, and consider interactions in which, on common input x, the prover is given a witness in $R_L(x)$. By saying that the output is computationally independent of the witness, we mean that for any two possible $\mathcal{NP}$-witnesses that could be used by the prover to prove the statement $x \in L$, the corresponding outputs are computationally indistinguishable.

Definition 7 (Witness-indistinguishability). *Let $\langle P, V\rangle$ be an interactive proof system for a language $L \in \mathcal{NP}$. We say that $\langle P, V\rangle$ is* witness-indistinguishable *for R_L, if for every $\mathcal{PPT}$ ITM V^* and for every two sequences $\{w_x^1\}_{n\in N, x\in L\cap\{0,1\}^n}$ and $\{w_x^2\}_{n\in N, x\in L\cap\{0,1\}^n}$, such that $w_x^1, w_x^2 \in R_L(x)$ for every x, the following probability ensembles are computationally indistinguishable.*

- $\left\{\mathsf{view}_2[\langle P(w_x^1), V^*(z)\rangle(x)]\right\}_{n\in N, x\in L\cap\{0,1\}^n, z\in\{0,1\}^*}$
- $\left\{\mathsf{view}_2[\langle P(w_x^2), V^*(z)\rangle(x)]\right\}_{n\in N, x\in L\cap\{0,1\}^n, z\in\{0,1\}^*}$

2.3.2 Proofs and Arguments of Knowledge

Given a language $L \in \mathcal{NP}$ and an instance x, a *proof or argument of knowledge* ($\mathcal{POK}$ or $\mathcal{AOK}$) not only convinces the verifier that $x \in L$, but also to demonstrate that the prover possesses an $\mathcal{NP}$-witness for x. This is formalized by the existence of an extractor: given black-box access to a machine that can successfully complete the proof or argument of knowledge on input x, the extractor can compute a witness for x.

Definition 8 (Proofs and arguments of knowledge [FS90, BG92]). *An interactive protocol $\Pi = (P, V)$ is a* proof of knowledge *(resp.* argument of knowledge*) of $\mathcal{NP}$-language L with respect to witness relation R_L if Π is indeed an interactive proof (resp. argument) for L. Additionally, there exists a polynomial q, a negligible function ν, and a probabilistic oracle machine E, such that for every interactive machine P^* (resp. for every polynomially-sized machine P^*), every $x \in L$ and every auxiliary input $z \in \{0,1\}^*$, the following holds:*

If $\Pr[\mathsf{out}_2[\langle P^*(z), V\rangle(x)] = 1] > \nu(|x|)$, *then on input x and oracle access to $P^*(x, z)$,* *machine E outputs a string from the $R_L(x)$ within an expected number of steps bounded by*

$$\frac{q(|x|)}{\Pr\left[\langle P^*(z), V\rangle(x) = 1\right] - \nu(|x|)}$$

The machine E is called the knowledge extractor.

Special Soundness

Special-sound protocols are proofs of knowledge with a very rigid and useful structure.

Definition 9 (Special soundness [CDS94]). *A 4-round interactive proof (P, V) for $\mathcal{NP}$-language L with witness relation R_L is special sound ($\mathcal{SS}$) with respect to R_L if (P, V) is public-coin, and on input x, all verifier messages have length $g(|x|) \geq |x|$.*

Moreover, there exists a deterministic polynomial-time extraction procedure X such that on input x, with all but negligible probability in $|x|$ over the choice of a uniform $\rho \in \{0,1\}^{g(|x|)}$, for all $\alpha, \beta, \beta', \gamma, \gamma'$ such that $\beta \neq \beta'$, and $(\rho, \alpha, \beta, \gamma)$ and $(\rho, \alpha, \beta', \gamma')$ are both accepting transcripts of (P, V) on input x, $X(x, (\rho, \alpha, \beta, \gamma), (\rho, \alpha, \beta', \gamma'))$ outputs a witness $w \in R_L(x)$.

2.3.3 Zero Knowledge

An interactive proof is said to be *zero-knowledge* if it yields nothing beyond the validity of the statement being proved. Formally, zero-knowledge requires that the view of any adversarial verifier can be reconstructed by an efficient simulator.

Definition 10 (Zero-Knowledge [GMR89]). *An interactive protocol (P, V) for $\mathcal{NP}$-language L with a witness relation R_L is* zero-knowledge *if for every $\mathcal{PPT}$ adversarial verifier V^*, there exists an expected $\mathcal{PPT}$ simulator S such that the following two ensembles are computationally indistinguishable over $x \in L$:*

$$\{\mathsf{view}_2[\langle P(w), V^*(z)\rangle (x)]\}_{x\in L, w\in R_L(x), z\in\{0,1\}^*} \approx \{S(x,z)\}_{x\in L, w\in R_L(x), z\in\{0,1\}^*}$$

A stronger definition is that of *black-box zero-knowledge*, in which there is one universal simulator S that must generate the view of any adversarial V^*, given only black-box access to V^*.

Definition 11 (Black-Box Zero-Knowledge [GO94]). *An interactive protocol (P, V) for $\mathcal{NP}$-language L with a witness relation R_L is* black-box zero-knowledge *if there exists an expected $\mathcal{PPT}$ simulator S such that for every $\mathcal{PPT}$ adversarial verifier V^*, the following two ensembles are computationally indistinguishable over $x \in L$:*

$$\{\mathsf{view}_2[\langle P(w), V^*(z)\rangle (x)]\}_{x\in L, w\in R_L(x), z\in\{0,1\}^*} \approx \left\{S^{V^*(x,z)}(x)\right\}_{x\in L, w\in R_L(x), z\in\{0,1\}^*}$$

Note that the simulator S does not have access to z, the auxiliary input of V^*. As a result, we may assume that V^* is deterministic, because V^* can treat part of z as its random tape.

Concurrent Zero-Knowledge

Given an interactive protocol (P, V) and a polynomial m, an m-session concurrent adversarial verifier V^* is a $\mathcal{PPT}$ machine that, on common input x and auxiliary input z, interacts with up to $m(|x|)$ independent copies of P concurrently. The different interactions are called sessions. There are no restrictions on how V^*

schedules the messages among the different sessions, and V^* may choose to abort some sessions but not others (unless if the protocol is public-coin).

Definition 12 (Concurrent Zero-Knowledge [DNS04]). *An interactive protocol* (P, V) *for* $\mathcal{NP}$*-language* L *with a witness relation* R_L *is* concurrent zero-knowledge *if for every* $\mathcal{PPT}$ *concurrent adversarial verifier* V^* *(i.e., any* m*-session concurrent adversarial verifier for any polynomial* m*), there exists an expected* $\mathcal{PPT}$ *simulator* S *such that the following two ensembles are computationally indistinguishable over* $x \in L$

$$\{\mathsf{view}_2[\langle P(w), V^*(z)\rangle(x)]\}_{x\in L, w\in R_L(x), z\in\{0,1\}^*} \approx \{S(x,z)\}_{x\in L, w\in R_L(x), z\in\{0,1\}^*}$$

We may also extend the definition of *black-box* zero-knowledge to the concurrent setting.

Definition 13 (Black-Box Concurrent Zero-Knowledge [DNS04]). *An interactive protocol* (P, V) *for language* L *is is* black-box concurrent zero-knowledge *if for all polynomials* m*, there exists an expected* $\mathcal{PPT}$ *black-box simulator* S_m *such that for every* $\mathcal{PPT}$ m*-session concurrent adversarial verifier* V^**, the following two ensembles are computationally indistinguishable over* $x \in L$*:*

$$\{\mathsf{view}_2[\langle P(w), V^*(z)\rangle(x)]\}_{x\in L, w\in R_L(x), z\in\{0,1\}^*} \approx \left\{S_m^{V^*(x,z)}(x)\right\}_{x\in L, w\in R_L(x), z\in\{0,1\}^*}$$

As before, in the case of black-box simulation, we may assume that V^* is deterministic. Note that in order to simulate the view of an m-session concurrent adversarial verifier, the simulator must have running time at least $m(|x|)$. This is why we allow a different simulator S_m for each polynomial m.

2.4 Signature Schemes

Signature schemes enables a sender or a signer to generete a *digital signature* for any message it wishes to authenticate, so that the signature can be universally verfied as a certificate for the "authenticity" of the message. More precisely,

Definition 14. *A signature scheme is a triplet* $(Gen, Sign, Ver)$ *of* $\mathcal{PPT}$ *algorithms satisfying the following conditions:*

- *On input* 1^n*, the key generation algorthm* Gen *outputs a pair of strings* $sk, pk \in \{0,1\}^*$*.* vk *is called the verification key and* sk *is called the signing key.*
- *For every pair* (sk, vk) *in the range of* $Gen(1^n)$*, and for every* $\alpha \in \{0,1\}^*$*, algorithm* $Sign$ *and* Ver *satisfy*

$$\Pr\left[Ver(vk, \alpha, Sign(sk, \alpha)) = 1\right] = 1$$

  We call $Sign$ *the signing algorithm and* Ver *the verification algorithm.*

- *For every* $\mathcal{PPT}$ *oracle machine* A*, there exists a neglgible function* μ*, such that, for all* $n \in N$*,*

$$Pr\Big[(sk, vk) \leftarrow Gen(1^n), (\alpha, \sigma) \leftarrow A^{Sign(sk,*)}(1^n) \ : \\ Ver(vk, \alpha, \sigma) = 1 \ \wedge \ \alpha \notin \mathsf{query}(A^{Sign(sk,*)}(1^n))\Big] \leq \mu(n)$$

In this thesis, we focus on signature schemes $\Pi = (Gen, Sign, Ver)$ that have *fixed-length*, that is, the signing algorithm $Sign$ on input 1^n, a public key pk and a message $m \in \{0,1\}^*$, always outputs a signature of length n. We call such signature schemes *fixed-length signature schemes*.

2.5 Commitments

In this thesis, the central tool that we study is *Commitment Schemes*. Roughly speaking, A commitment scheme enables a party, called the committer, to commit itself to a value to another party, the receiver. At first the value is hidden from the receiver; this property is called *hiding*. At a later stage when the commitment is opened, it can only reveal a single value as determined in the committing phase; this property is called *binding*. First we define the structure of a commitment scheme.

Definition 15 (Commitment Schemes). *A* commitment scheme *is an interactive protocol $\langle C, R\rangle$ with the following properties:*

1. *Both the committer C and the receiver R are $\mathcal{PPT}$ machines.*
2. *The commitment scheme has two stages: a commit stage and a reveal stage. In both stages, C and R receive a security parameter 1^n as common input. C additionally receives a private input $v \in \{0,1\}^n$ that is the string to be committed.*
3. *The commit stage results in a joint output c, called the commitment, a private output for C, d, called the decommitment string. Without loss of generality, c can be the full transcript of the interaction between C and R.*
4. *In the reveal stage, committer C sends the pair (v, d) to the receiver R, and decides to accept or reject the decommitment (c, v, d) deterministically.*

If C and R do not deviate from the protocol, then R should accept (with probability 1) during the reveal stage.

Next we define the binding and hiding property of a commitment scheme.

Definition 16 (Binding). *A commitment scheme $\langle C, R\rangle$ is* statistically (resp. computationally) binding *if for every machine (resp. non-uniform $\mathcal{PPT}$ machine) C^* (a malicious committer), there exists a negligible function ν such that C^* succeeds in the following game with probability at most $\nu(n)$:*

> *On security parameter 1^n, C^* first interacts with R in the commit stage to produce commitment c. Then C^* outputs two decommitments (c, v_0, d_0) and (c, v_1, d_1), and succeeds if $v_1 \in \{0,1\}^n$, $v_2 \in \{0,1\}^n$, $v_1 \neq v_2$ and R accepts both decommitments.*

The commitment scheme is perfectly binding if no machine C^* can ever succeed at the above game.

Definition 17 (Hiding). *A commitment scheme $\langle C, R\rangle$ is* computationally (resp. statistically) *hiding if for every non-uniform $\mathcal{PPT}$ machine (resp. every machine) R^* (a malicious receiver), the following ensembles are computationally indistinguishable over $n \in N$ (resp. statistically indistinguishable over $n \in N$):*

$$\{\mathsf{view}_2[\langle C(v_0), R^*(z)\rangle(1^n)]\}_{n\in N, v_0\in\{0,1\}^n, z\in\{0,1\}^*} \approx \{\mathsf{view}_2[\langle C(v_1), R^*(z)\rangle(1^n)]\}_{n\in N, v_1\in\{0,1\}^n, z\in\{0,1\}^*}$$

In this thesis, we focus on statistically binding and computationally hiding commitment schemes, in short, statistically binding commitments. Additionally, we will consider the following two properties:

Definition 18. *Let $\langle C, R\rangle$ be a statistically binding commitment scheme.*

Tag-based commitment scheme [PR05a, DDN91]: *$\langle C, R\rangle$ is a* tag-based scheme with $l(n)$-bit identities *if, in addition to the security parameter 1^n, the committer and the receiver also receive a "tag"—a.k.a. identity—*id *of length $l(n)$ as common input.*

2.6 Useful Known Constructions

The following protocols are useful in our various constructions:

- 2-round statistically binding commitments constructed from one-way functions ([Nao91, HILL99]).
- 4-round computational $\mathcal{WI}$ and $\mathcal{SS}$ proofs based on one-way functions. This can be instantiated with a parallel repetition of the Blum Hamiltonicity protocol [Blu86] with 2-round statistically binding commitments constructed from one-way functions.
- $\omega(1)$ round $\mathcal{ZK}$ proof system based on one-way functions [GMR89].
- A fixed-length signature scheme. Such signature schemes can be constructed relying on universal one-way hash functions [NY89], which in turn can be based on any one-way function [Rom90]; see [Gol04].

CHAPTER 3

NON-MALLEABILITY AND CONSTANT-ROUND SECURE COMPUTATION

Non-malleable commitments provide strong guarantees on the "independence" of inputs across different sessions; such input independence is crucial for many applications and in particular, the task of secure multi-party computation for general functionalities. In this chapter, we settle the round complexity of non-malleable commitments: We construct a constant-round non-malleable commitment scheme based on one-way functions. A a direct application of this result enables us to obtain constant-round secure multi-party computation based on TDPs. Both our constructions of non-malleable commitments and secure multi-party computation protocols yield the first improvement on round-efficiency over the original works by [DDN91] and [GMW87] under the same assumptions they are based on.

3.1 Definition of Non-Malleable Commitments

3.1.1 Concurrent Non-Malleability

We recall the definition of concurrent non-malleability from [LPV08]. For convenience, we use a slightly different presentation (based on indistinguishability rather than simulation); equivalence follows using a standard argument (c.f. [GM84, PR05a]).

Let $n \in N$ be a security parameter, and let $\langle C, R\rangle$ be a tag-based commitment scheme with n-bit identities. Consider a man-in-the-middle adversary A (as shown in Figure 3.1) that, on inputs n and z (where z is received as an auxiliary input), participates in m left and right interactions simultaneously. In the left interactions the man-in-the-middle adversary A interacts with C, receiving commitments to values $v_1, \ldots, v_m$, using identities of length n, $\mathsf{id}_1, \ldots, \mathsf{id}_m \in \{0,1\}^n$, of its choice. In the right interactions A interacts with R attempting to commit to a sequence of related values $\tilde{v}_1, \ldots, \tilde{v}_m$, again using identities of length n $\tilde{\mathsf{id}}_1, \ldots, \tilde{\mathsf{id}}_m$ of its choice. If any of the right commitments are invalid, or undefined, its value is set to $\perp$. For any i such that $\tilde{\mathsf{id}}_i = \mathsf{id}_j$ for some j, set $\tilde{v}_i = \perp$—i.e., any commitment where the adversary uses the same identity as one of the left interactions is considered invalid. Let $\mathsf{mim}^A_{\langle C,R \rangle} v_1, \ldots, v_m, z$ denote a random variable that describes the values $\tilde{v}_1, \ldots, \tilde{v}_m$ and the view of A, in the above experiment.

Definition 19. *A commitment scheme $\langle C, R \rangle$ is said to be* concurrent non-malleable (with respect to itself) *if for every polynomial $p(\cdot)$, and every $\mathcal{PPT}$ man-in-the-middle adversary A that participates in at most $m = p(n)$ concurrent executions, the following ensembles are computationally indistinguishable.*

$$\left\{\mathsf{mim}^A_{\langle C,R \rangle}(v_1, \ldots, v_m, z)\right\}_{n \in N, v_1, \ldots, v_m \in \{0,1\}^n, v'_1, \ldots, v'_m \in \{0,1\}^n, z \in \{0,1\}^*}$$
$$\left\{\mathsf{mim}^A_{\langle C,R \rangle}(v'_1, \ldots, v'_m, z)\right\}_{n \in N, v_1, \ldots, v_m \in \{0,1\}^n, v'_1, \ldots, v'_m \in \{0,1\}^n, z \in \{0,1\}^*}$$

We also consider relaxed notions of concurrent non-malleability: one-one, one-many, and many-one secure non-malleable commitments (See Figure 3.2 below.) In a one-one (a.k.a., a stand-alone secure) non-malleable commitment, we consider only adversaries A that participate in one left and one right interaction; in one-many, A participates in one left and many right, and in many-one, A participates in many left and one right.

As shown in [LPV08], any protocol that is one-many non-malleable is also concurrent non-malleable.

$$\begin{tikzcd}
C & & & R \\
Com(v_1) \arrow[ddr] & & & Com(\tilde{v}_1) \\
\cdots & & & \cdots \\
Com(v_i) \arrow[r] & A \arrow[rr] \arrow[uurr] \arrow[ddrr] & & Com(\tilde{v}_i) \\
\cdots & & & \cdots \\
Com(v_m) \arrow[uur] & & & Com(\tilde{v}_m)
\end{tikzcd}$$

Figure 3.1: A concurrent man-in-the-middle adversary.

$$\begin{tikzcd}
Com(v) \arrow[r] & A \arrow[r] & Com(\tilde{v})
\end{tikzcd}$$

(i) one-one

$$\begin{tikzcd}
 & & Com(\tilde{v}_1) \\
 & & \cdots \\
Com(v) \arrow[r] & A \arrow[r] \arrow[uur] \arrow[ddr] & Com(\tilde{v}_i) \\
 & & \cdots \\
 & & Com(\tilde{v}_m)
\end{tikzcd}
\qquad
\begin{tikzcd}
Com(v_1) \arrow[ddr] & & \\
\cdots & & \\
Com(v_i) \arrow[r] & A \arrow[r] & Com(\tilde{v}) \\
\cdots & & \\
Com(v_m) \arrow[uur] & &
\end{tikzcd}$$

(ii) one-many (iii) many-one

Figure 3.2: Restricted man-in-the-middle adversaries.

Proposition 1 ([LPV08])**.** *Let $\langle C, R\rangle$ be a one-many concurrent non-malleable commitment. Then, $\langle C, R\rangle$ is also a concurrent non-malleable commitment.*

3.1.2 *k*-Robustness: Non-Malleability w.r.t. *k*-round Protocols

The concept of non-malleability is traditionally only considered in a setting where a man-in-the middle adversary is participating in two (or more) executions of the *same* protocol. We here consider a new notion of non-malleability with respect to arbitrary k-round protocols.

Consider a one-many man-in-the-middle adversary A (as shown in Figure 3.3) that participates in one left interaction—communicating with a machine B—and many right interactions—acting as a committer using the commitment scheme $\langle C, R\rangle$. As in the standard definition of non-malleability, A can adaptively choose the identities in the right interactions. We denote by $\mathsf{mim}^{B,A}_{\langle C,R\rangle}(y, z)$ the random variable consisting of the view of $A(z)$ in a man-in-the-middle execution when communicating with $B(y)$ on the left and honest receivers on the right, combined with the values $A(z)$ commits to on the right. Intuitively, we say that $\langle C, R\rangle$ is one-many non-malleable w.r.t B if $\mathsf{mim}^{B,A}_{\langle C,R\rangle}(y_1, z)$ and $\mathsf{mim}^{B,A}_{\langle C,R\rangle}(y_2, z)$ are indistinguishable, whenever interactions with $B(y_1)$ and $B(y_2)$ cannot be distinguished.

$$\begin{tikzcd}
 & & Com(\tilde{v}_1) \\
 & & \cdots \\
B(y) \arrow[r, shift left=4] \arrow[r, phantom, shift right=2, "\cdots"] \arrow[r, shift right=4] & A \arrow[l, shift right=1] \arrow[uur] \arrow[r] \arrow[ddr] & Com(\tilde{v}_i) \\
 & & \cdots \\
 & & Com(\tilde{v}_m)
\end{tikzcd}$$

Figure 3.3: A concurrent man-in-the-middle adversary with respect to protocol B on input y.

Definition 20. *Let $\langle C, R\rangle$ be a commitment scheme, and B a $\mathcal{PPT}$ ITM. We say the commitment scheme $\langle C, R\rangle$ is* one-many non-malleable w.r.t. *B, if for every two sequences $\{y^1_n\}_{n\in N}$ and $\{y^2_n\}_{n\in N}$, $y^1_n, y^2_n \in \{0, 1\}^n$, such that, for all $\mathcal{PPT}$ ITM $\tilde{A}$, it holds that*

$$\left\{\mathsf{out}_{\tilde{A}}[\langle B(y^1_n), \tilde{A}(z)\rangle(1^n)]\right\}_{n\in N, z\in\{0,1\}^*} \approx \left\{\mathsf{out}_{\tilde{A}}[\langle B(y^2_n), \tilde{A}(z)\rangle(1^n)]\right\}_{n\in N, z\in\{0,1\}^*}$$

then it also holds that, for every $\mathcal{PPT}$ one-many man-in-the-middle adversary A,

$$\left\{\mathsf{mim}^{B,A}_{\langle C,R\rangle}(y^1_n, z)\right\}_{n\in N, z\in\{0,1\}^*} \approx \left\{\mathsf{mim}^{B,A}_{\langle C,R\rangle}(y^2_n, z)\right\}_{n\in N, z\in\{0,1\}^*}$$

We say that $\langle C, R\rangle$ is one-many k-robust if $\langle C, R\rangle$ is one-many non-malleable w.r.t. any machine B that interacts with the man-in-the-middle adversary in k rounds.

3.2 Overview of Our Construction

The main idea underlying all non-malleable commitment schemes is to "encode" the identity of the committer into the protocol. At the very least, this ensure that unless the attacker copies the identity of the left committer, the attacker cannot simply forward messages between the left and the right executions. But we also need to ensure that the attacker cannot in a clever way maul the messages it receives on the left so they become useful on the right. For instance, in the original DDN construction, the identity is encoded into the scheduling of messages in the protocol; on a very high-level (and oversimplifying), the idea is to ensure that at some point in the execution, the MIM must "speak" while only receiving "useless" messages. The problem with this approach is that it requires a high round-complexity.

We will revisit the DDN approach. The main idea behind our scheme is to perform the message scheduling "in the head". A bit more precisely, our protocol follows the "simulation-soundness" paradigm of Sahai [Sah99], first used in the context of CCA-secure encryption, and next used by Pass and Rosen [PR05a] in the context of non-malleable commitments; that is, the main component of our construction is a method for enabling us to "simulate" the left interaction, while ensure that the right interaction remains "sounds". Towards this, we embed a "trapdoor" into the protocol which depends on the identity of the interaction; proving simulation-soundness then essentially amounts to showing that there exists a way to recover the trapdoor for the left interaction, while ensuring that the adversary does not recover the trapdoor for the right interaction (as long as the right interaction has a different identity than the left interaction).

The idea is to have a protocol where the trapdoor can be recovered by "rewinding" some specific messages in the protocol—called "slots"—in *a specific order which depends on the identity of the interaction*. Furthermore, the protocol should have the property that if this specific rewinding order is not the rewinding order actually used, then a trapdoor cannot be recovered. So, if we rewind the left interaction according to the rewinding order corresponding to the identity of the left interaction, this will still not enable the adversary to recover the trapdoor corresponding to the right interaction (unless the identity of the right interaction is the same as the identity of the left interaction). In our particular instantiation of this idea, the trapdoor will be a "signature-chain" (i.e., a signature on a signature on a signature, etc.) of length n (i.e., the identity length) using different keys; the choice of the keys in the signature chain are determined by the identity of the interaction. Next, the protocol will have a "slot" for each of the keys where the receiver is willing to sign a *single* message for the committer using the key corresponding to the slot. The key point is that the simulator is able to rewind the slots in an appropriate order to recover a signature-chain corresponding to the identity of the left interaction; but the rewindings will still not enable the adversary to recover a signature-chain corresponding to any other identity.

To explain the main ideas in more details, we here first focus on outlining the construction of a constant-round non-malleable commitment scheme that is secure for *synchronizing* and *non-aborting* adversaries; we next comment on how to deal with general adversaries. An adversary is said to be synchronizing if it "aligns" the left and the right executions; that is, whenever it receives message i on the left, it directly sends message i on the right, and vice versa. An adversary is said to be non-aborting if it never sends any invalid messages in the left interaction (where it is acting as a receiver); it might still send invalid messages on the right.

As mentioned above, the idea is to have a protocol with an "identity-based trapdoor" embedded into it. The trapdoor will be a "signature-chain" using a sequence of keys that are determined by the identity of the protocol. More precisely, we say that $(\sigma_0, \sigma_1, \ldots, \sigma_n)$ is a *plain signature chain*[1] with respect to the signature scheme Π, the verification keys vk_0, vk_1 and the pattern $\psi \in \{0,1\}^n$ if $\sigma_0 = 0$ and for all $0 \leq i < n$, σ_{i+1} is a signature on the message (i, σ_i) with respect to the key $vk_{\psi_{i+1}}$. For convenience of notation, for the remainder of this section we fix a particular signature scheme Π; all signatures we use are with respect to this this particular scheme.

The following simple claim regarding signature chains will be useful. Consider a "signature game" where an adversary A gets access to two randomly chosen verification keys vk_0, vk_1 and additionally has access to signature oracles with respect to vk_0 and vk_1; let φ denote the "access pattern" of the adversary to the signature oracle (that is, if the i'th oracle call is to the signature oracle w.r.t. vk_b, then $\varphi_i = b$). The claim now is that, with overwhelming probability, if in the signature game, A manages to output a plain signature chain with respect to vk_0, vk_1 and pattern ψ, then ψ is a sub-string of φ.

[1] We use the name "plain signature chain" (instead of just "signature chain"), since the actual signature chains we will use in the final construction will be a bit more complicated.

The protocol for committing to a string v with identity id proceeds as follows:

- **Slot 1**: The receiver R generates a key-pair (sk_0, vk_0) for the signature scheme Π, and sends vk_0 to the committer C. C next send a random message r_0 to R who signs r_0 and then returns the signature to C.
- **Slot 2**: R generates another key-pair (sk_1, vk_1) and sends vk_0 to the committer C. As in Slot 1, C next send a random message r_1 to R who signs r_1 and then returns the signature to C.
- **Commit phase**: C commits to v using a standard statistically binding commitment.
- **Proof phase**: C gives R a "special-purpose"[2] witness indistinguishable argument of knowledge of the fact that it either knows the value committed to in the commit phase, or that it knows a plain signature chain with respect to vk_0, vk_1 and id.

We now turn to argue that this protocol is non malleable with respect to non-aborting and synchronizing adversaries. For simplicity, we here focus only on one-one (i.e., stand-alone) non-malleability (but the same proof actually also works for concurrent non-malleability). Consider a man-in-the-middle adversary A that uses identity id on the left and identity $\tilde{\mathsf{id}} \neq \mathsf{id}$ on the right, and receives a commitment to the value v on the left. We will argue that no matter what the value of v is, the value it commits to on the right will be indistinguishable. Towards this goal, consider a hybrid experiment where the left interaction is simulated by acting honestly in Slot 1 and 2, next committing to 0, and finally using a "fake-witness"—namely a signature chain—in the proof phase; the simulator obtains this fake witness by simply rewinding Slot 1 and 2 (that is, to

[2]We will shortly explain what makes this proof special.

rewinding slot b, we restore the state of A after vk_b has been sent, and send a new message to be signed) in the appropriate order to obtain a signature chain with respect to id (note that since A is non-aborting, each time the simulator asks it to sign a message, it does). To show the above claim, we now argue that no matter what the value of v is, the value A commits to on the right in the real execution (when receiving a commitment to v), is indistinguishable from the value it commits to on the right when the left interaction instead is simulated.

The key-point of the proof is the claim that even in the simulation, A cannot use a fake-witness in the right interaction. This follows from the fact that since A is synchronizing, when we rewind Slot 1 and 2 on the left, the same slots are rewound on the right *in exactly the same order*. Thus, by the signature-game claim, if A manages to get a signature chain it must be a subset of the pattern $01\mathsf{id}$ (the reason we need to append 01 is that A gets two signatures in the honest emulation of Slot 1 and 2, already before we start the rewindings). So, if we appropriately restrict the identity set (for instance, by requiring that all identities start with 10) then the only valid identity that is a sub-string of $01\mathsf{id}$ is id, and thus $\tilde{\mathsf{id}} = \mathsf{id}$, which is a contradiction.

To argue that the value committed to on the right does not change when we move from the real interaction to the simulation, consider an intermediary hybrid where we only change the witness used in the proof phase (but keep the value committed to in the commit phase to v). Note that since the adversary is synchronizing, the proof phase of the left interaction appears completely after the commitment (in Stage 2) in the right interaction. Therefore, the right value does not change at all when switching the the witness used in the proof phase on the left.

Finally, we simply have to argue that the value on the right does not change once we change the value committed to in the commit phase on the left. By the hiding property of the left commitment, the view of the adversary does not change when the left committed value switches. But since the value committed to on the right cannot be efficiently recovered, this does not directly imply that the committed value also is indistinguishable. To resolve this problem, we rely on the argument of knowledge property of the proof phase: A witness on the right can be extracted efficiently from the proof phase. Since the witness used in the right interaction cannot be a fake witness (by the key-claim above), it must be the value committed to in the commit phase, so indistinguishability of the committed value follows from the hiding property of the the left commitment.

Dealing with aborting adversaries: When considering aborting adversaries, we run into two obstacles:

- The adversary might notice that the simulator is feeding it signature chains to sign (instead of random messages) and thus decide to abort the left execution. We handle this by adapting the definition of a signature chain: instead of requiring the chain to be "a signature on a signature on a signature... etc", we require a signature-chain to be a signature on "a *commitment* of a signature on a commitment of a signature... etc". And next, in the protocol, we let C send commitments to 0 instead of random strings. To be able to establish an analog of the above signature-game claim, we additionally require C to give a zero-knowledge argument of knowledge of the value it committed to before R agrees to sign it.
- Another problem is that A might abort the left execution with some proba-

bility p. This means that we might have to rewind the left execution many times (roughly $1/p$ times) before getting the signature we are looking for. As a consequence, the "access pattern" on the right will be a sub-string of $01\mathsf{id}_1^*\mathsf{id}_2^* \ldots \mathsf{id}_n^*$. To get around this problem, we add an additional slot (and a corresponding signature key). Next, we require that the signature-chain corresponding to the identity id to be with respect to the pattern $2\mathsf{id}_1 2\mathsf{id}_2 2\mathsf{id}_3 \ldots 2\mathsf{id}_n$.

Dealing with non-synchronizing adversaries: As is usually the case, synchronizing adversaries are the "hardest" to deal with. To prove security against non-synchronizing adversaries, we follow basically the same argument: First, if A is not synchronizing there exists some slot that is never rewound and so if the identity of the right interaction contains at least two 0's and two 1's, we can still establish the above key-claim. Next, to argue that the committed value on the right does not change, we consider again the intermediary hybrid above. However, when the adversary is not synchronizing, it may choose to interleave messages in the proof phase of the left interaction and the commitment of the right interaction, and thus the right committed value may change when the witness on the left changes. To overcome this problem, we again rely on the argument of knowledge property of the proof phase to extract a witness from the right interaction. Since the witness cannot be a signature-chain (by the key-claim), it must be the committed value; then the indistinguishability of the committed value follows from the witness-indistinguishability of the left proof phase. However, one problem is that extraction on the right may rewind the left proof phase and thus break the witness indistinguishability property. One way of resolving this problem would be to (in analogy with [PR05a]) have the proof phase be statistically witness indistinguishable; but this requires additional assumptions (to keep it constant-round). Instead, we here introduce a different technique to overcome this problem: we let the proof phase consist of *multiple sequentially ordered* witness indistinguishable special-sound proofs.[3] This allows us to change the witness in each of the proofs, one by one, while ensuring that the witness on the right can be extracted from some other proof, without rewinding the left proof where the witness currently is being changed.

3.3 Signature Chains and Games

Let $\Pi = (Gen, Sign, Ver)$ be a fixed-length signature scheme, and com a statistically-binding commitment scheme. For simplicity of notation, we keep these schemes fixed, and provide our definitions and protocols with respect to those particular schemes. Furthermore, for simplicity of exposition, we assume that com that is non-interactive; however, all of our definitions and protocols can be easily modified to work with any two-round statistically-binding commitment schemes; see Remark 1 for further details.

We now turn to formally defining the notion of a *signature-chain* and then proceed to defining *signature-games*.

Definition 21 (Signature-Chain). *Let $\ell \in N$, $\psi \in \{0,1,2\}^\ell$ and $vk_0, vk_1, vk_2 \in \{0,1\}^*$ be three verification keys for the signature scheme Π. We say that a triplet $\delta = (\bar{\sigma}, \bar{c}, \bar{r})$ is a* signature-chain w.r.t. keys *vk_0, vk_1, vk_2* and pattern *ψ, if $\bar{\sigma}$, $\bar{c}$, and $\bar{r}$ are vectors of length ℓ satisfying the following properties.*

[3]This method was originally used by us in the amplification procedure of [LP09]; this "trick" is also a central component enabling the works of [Wee10, Goy11].

- *For all $i \in [\ell]$, $\bar{\sigma}_i$ is a valid signature of the message $\bar{c}_i$ under key vk_{ψ_i}, that is, $Ver(vk_{\psi_i}, \bar{c}_i, \bar{\sigma}_i) = 1$.*
- *For all $1 < i \le \ell$, $\bar{c}_i$ is a commitment to the tuple $(i-1, \bar{\sigma}_{i-1})$ using com and randomness $\bar{r}_i$; and $\bar{c}_1$ is a commitment to 0^m using com and randomness $\bar{r}_1$, where $m = \log \ell + n$.*

We say that a signature-chain $\delta = (\bar{\sigma}, \bar{c}, \bar{r})$ has length ℓ if $|\bar{\sigma}| = l$.

We proceed to define a signature-game $\mathrm{SG}^{A,\ell}(n, z)$, where A on input $1^n, z$ interacts with a *Challenger* in the following three stages:

Stage 1: the Challenger samples three pairs of signing and verification keys at random, $(sk_b, vk_b) \leftarrow Gen(1^n)$, where $b \in \{0, 1, 2\}$, and sends A the verification keys, vk_0, vk_1, and vk_2.

Stage 2: A interacts with the Challenger in a sequence of iterations for as long as it wishes. Iteration i proceeds as follows:

- A sends the Challenger a tuple (φ_i, c), where $\varphi_i \in \{0, 1, 2\}$, followed by a 5-round $\mathcal{WISSP}$ proof of the statement that c is a valid commitment of com.
- if the proof is convincing, the Challenger signs the commitment c using the signing key s_{φ_i} and returns the signature to A; otherwise, it aborts the iteration (without giving back a signature).

Stage 3: Finally, A outputs the tuple (δ, ψ).

We call the sequence $\varphi = \varphi_1, \varphi_2, \ldots$ of signing request, the "access pattern" of A. We say that the output of A is *well-formed* if δ is a length $l(n)$ signature-chain with respect to vk_0, vk_1, vk_2 and ψ. Finally, we say that A *wins* if its output is well-formed at ψ is not a sub-string of its access pattern φ (and *looses* otherwise).

Lemma 1. *For every $\mathcal{PPT}$ adversary A and every polynomial ℓ, there exists a negligible function μ, such that for every $n \in N, z \in \{0,1\}^*$, the probability that A wins in $SG^{A,\ell}(n,z)$ is at most $\mu(n)$.*

Proof. Consider any adversary A, polynomial ℓ, $n \in N$, and $z \in \{0,1\}^*$. Without loss of generality, we can assume that A always outputs tuples $(\delta = (\bar{\sigma}, \bar{c}, \bar{r}), \psi)$ such that $|\bar{\sigma}| = |\bar{c}| = |\bar{r}| = \psi| = l(n)$ (since whenever it doesn't it loses). For each $i \in [l(n)]$, define the random variable $\mathcal{I}_i$ to be the index of the *first* iteration (in Stage 2 of the game $\mathrm{SG}^{A,\ell}(n,z)$) in which A queries the Challenger for a signature of the commitment $\bar{c}_i$ under key vk_{ψ_i}; if A never queries the Challenger for a signature of $\bar{c}_i$, $\mathcal{I}_i$ is set to $\perp$.

Note that if the output of A is well-formed, it contains a signature-chain $\delta = (\bar{\sigma}, \bar{c}, \bar{r})$ w.r.t. pattern ψ, such that for every i, $\bar{\sigma}_i$ is a valid signature of $\bar{c}_i$ under key vk_{ψ_i}. It thus follows from the unforgibility of the signature scheme that, except with negligible probability, for each i, A must have queried $\bar{c}_i$ for a signature of vk_{ψ_i} in some iteration. We thus have the following claim.

Claim 1. *For every $\mathcal{PPT}$ adversary A and polynomial ℓ, there exists a negligible function μ_1, such that for all $n \in N, z \in \{0,1\}^*$, the probability that the output of A in $SG^{A,\ell}(n,z)$ is of A is well-formed and there exists an $i \in [\ell(n)]$ such that $\mathcal{I}_i = \perp$, is smaller than $\mu_1(n)$.*

We also have the following claim.

Claim 2. *For every $\mathcal{PPT}$ adversary A and polynomial ℓ, there exists a negligible function μ_2, such that, for all $n \in N, z \in \{0,1\}^*$, the probability that the output of A in*

$SG^{A,\ell}(n, z)$ is well-formed and there exists an $i \in [\ell(n) - 1]$ such that , $\mathcal{I}_i \neq \perp$, $\mathcal{I}_{i+1} \neq \perp$ and $\mathcal{I}_i \geq \mathcal{I}_{i+1}$, is smaller than $\mu_2(n)$.

Before proceeding to the proof of Claim 2, we let us first prove Lemma 1 using Claim 1 and 2. It follows from the two claims that, except with negligible probability, *either* the output of A is not well-formed, *or* the output is well-formed and for all i, $\mathcal{I}_i \neq \perp$ and $\mathcal{I}_i < \mathcal{I}_{i+1}$. In the former case, the adversary loses the game. In the latter case, as $\mathcal{I}_i \neq \perp$ for all i, A must have asked for a signature using key vk_{ψ_i} in the $\mathcal{I}_i^{\text{th}}$ iteration, which means $\varphi_{\mathcal{I}_i} = \psi_i$. Furthermore, as $\mathcal{I}_i < \mathcal{I}_{i+1}$ for all i, it follows that ψ is a sub-string of φ. Therefore, A loses in this case as well. Thus, except with negligible probability, A looses.

Proof of Claim 2. First notice that it follows from the (statistical) binding property of com, that except with negligible probability[4], if the output $(\delta = (\bar{\sigma}, \bar{c}, \bar{r}), \psi)$ of A is well-formed, then for all i, $\bar{c}_i \neq \bar{c}_{i+1}$, since $\bar{c}_i, \bar{c}_{i+1}$ are respectively commitments to tuples of the form $(i, \cdot)$ and $(i + 1, \cdot)$. It follows that, except with negligible probability, if the output of A is well-formed, there doesn't exists some i such that $\mathcal{I}_i, \mathcal{I}_{i+1} \neq \perp$ but $\mathcal{I}_i = \mathcal{I}_{i+1}$. Thus, it suffices to bound the probability that the output of A is well formed and there exists some i such that $\mathcal{I}_i, \mathcal{I}_{i+1} \neq \perp$ and $\mathcal{I}_i > \mathcal{I}_{i+1}$.

Towards this, assume for contradiction that there exists an adversary A and a polynomial ℓ, such that there exists a function $i : N \to N$ and a polynomial p, such that for infinitely many $n \in N, z \in \{0, 1\}^*$, the probability that the output of A in the game $\mathrm{SG}^{A,\ell}(n, z)$ is well-formed, $\mathcal{I}_i, \mathcal{I}_{i+1} \neq \perp$, and $\mathcal{I}_i > \mathcal{I}_{i+1}$ for $i = i(n)$, is

[4]Since we assume that com is non-interactive, we actually have perfect binding, but given that we want an analysis that works also for two-round commitments, we here directly consider the more general case of statistical binding.

at least $1/p(n)$. We can construct a machine B that violate the unforgibility of the signature scheme Π.

B, on input $1^n, z$ and a randomly generated verification key vk, has access to the signing oracle corresponding to vk, and tries to forge a signature (of vk) as follows: it internally emulates an execution of the signature game $\mathrm{SG}^{A,\ell}(n, z)$ with A honestly, with the following exceptions:

- In Stage 1, it picks an index $t \in \{0, 1, 2\}$ at random and forwards the verification key vk to the adversary as the t^{th} verification key.
- In Stage 2, whenever A requests a signature of a message m under key vk, it obtains such a signature from the signing oracle and forwards it to A.

  Furthermore, it guesses that $\mathcal{I}_i = u$ and $\mathcal{I}_{i+1} = k$, for random $u > k$. Then, in the k^{th} iteration (in Stage 2 of $\mathrm{SG}^{A,\ell}(n, z)$), after receiving a request from A to sign the commitment c, it extracts out the value (j, σ^*) committed to in c from the $\mathcal{WISSP}$ that A provides following the signing request. Later, in the u^{th} iteration, when A submits a query c^* to the Challenger, it checks whether σ^* is a valid signature of c^* under key vk. If so, it halts and outputs the message-signature pair (c^*, σ^*); otherwise, it halts and outputs fail.

By construction, B emulates the view of A in the signature game $\mathrm{SG}^{A,\ell}(n, z)$ perfectly before it halts. Therefore, by our hypothesis, with probability at least $1/p(n)$, in emulation by B, A would query for the first time the commitments $\bar{c}_i$ and $\bar{c}_{i+1}$ in iterations $\mathcal{I}_i$ and $\mathcal{I}_{i+1}$ respectively, such that $\mathcal{I}_{i+1} < \mathcal{I}_i$ and $\bar{c}_{i+1}$ is a commitment to a tuple $(i + 1, \bar{\sigma}_{i+1})$, where $\bar{\sigma}_{i+1}$ is a signature of $\bar{c}_i$ under the verification key vk_{ψ_i}. Let $M(n)$ be the maximum number of iterations in the game; M is polynomially bounded since the running-time of A is. With probability at least $\frac{1}{q(n)} = \frac{1}{3M(n)^2 p(n)}$, it holds that (1) the above event occurs in the emulation by B and (2) B correctly guesses the values of $\mathcal{I}_i$, $\mathcal{I}_{i+1}$ and vk_{ψ_i}. In this case, except with negligible probability, the committed value σ^* that B extracts out from the $\mathcal{WISSP}$ following $c = \bar{c}_{i+1}$ contains a valid signature of $\bar{c}_i$, which is queried *for the first time* in the uth iteration for a signature using key vk. Hence B will output a valid message-signature pair $(\bar{c}_i, \sigma^*)$ for vk, without querying the signing oracle $\bar{c}_i$ (since once the query $\bar{c}_i$ is submitted for the first time in iteration u, B halts immediately and outputs the pair); this violates the unforgibility of the signature scheme Π. □

□

3.4 A Constant-Round Non-Malleable Commitment Scheme

Let $\Pi = (Gen, Sign, Ver)$ be a fixed-length signature scheme, and com a non-interactive statistically-binding commitment scheme as defined in the last section. To simplify the presentation of the proof, we assume that both Π and com can be "broken"—i.e., signatures can be generated for *any* message, and the value committed to can be recovered for *any* commitment—in time $2^{n/2}$ where n is the security parameter; this is without loss of generality since we can always appropriately "scale-down" the security parameter in Π and com (and make sure that com commits to values "bit-by-bit"). To further simplify the presentation, we provide the construction of a non-malleable commitment $\langle C, R\rangle$ that works assuming player identities are ℓ-bit binary strings that contains at least two 0-bits and two 1-bits; any such scheme can trivially be turned into one that works for arbitrary identities (by simply appending two 0's and two 1's to the identity).

To commit to a value v, the Committer and the Receiver of $\langle C, R\rangle$, on common input a security parameter 1^n (in unary) and an identity $\mathsf{id} \in D^\ell$, proceed in the following three stages:

Stage 1: The receiver interacts with the Committer in three iterations, where iteration $i \in \{0, 1, 2\}$ proceeds in the following steps:

1. The Receiver generates a pair of signing and verification keys of the signature scheme Π, $(sk_i, vk_i) \leftarrow Gen(1^n)$, and sends the verification key vk_i.
2. The Committer commits to 0^m, where $m = \log \ell + n$, using com. Let c_i be the commitment sent to the Receiver.
3. The Committer proves that c_i is a valid com commitment using a 5-round $\mathcal{WISSP}$ protocol.
4. The Receiver signs the commitment c_i using the signing key sk_i, and sends the generated signature θ_i to the Committer.

Stage 2: The Committer commits to the value v using com. Let c' be the commitment generated.

Stage 3: The Committer proves that

- *either* c' is a valid com commitment,
- *or* there exists a signature-chain δ w.r.t. vk_0, vk_1, vk_2 and pattern $\mathsf{pattern}(\mathsf{id})$, where the function $\mathsf{pattern} : \{0, 1\}^* \rightarrow \{0, 1, 2\}^*$ maps a (binary) identity id of length ℓ to a trinary string of length 2ℓ as follows:

$$\mathsf{pattern}(\mathsf{id}) = 2, \mathsf{id}_1, 2, \ldots, \mathsf{id}_i, 2, \ldots, \mathsf{id}_\ell$$

This statement is proved using $k + 5$ sequential invocations of a 4-round $\mathcal{WI}$ special sound proof system, where k is the number of messages in Stage 1 of the protocol; we additionally require that the length of the "challenge" in each special-sound proof is n.

We refer to the last three steps of an iteration in Stage 1 as a slot, which *opens* when the Committer send the com commitment to 0^m, and *closes* when the Receiver returns a signature to the commitment. We call the slot in iteration i, the i'th slot.

It is easy to see that the protocol $\langle C, R \rangle$ consists of a constant number of messages. Furthermore, it follows using standard techniques that $\langle C, R \rangle$ is a valid commitment scheme.

Proposition 2. *$\langle C, R \rangle$ is a commitment scheme.*

Proof. We show that $\langle C, R \rangle$ satisfies the binding and hiding properties.

Binding: The binding property follows directly from the statistically binding property of com used in Stage 2.

Hiding: The hiding property essentially follows from the hiding property of com and the fact that Stage 3 of the protocol is $\mathcal{WI}$ (since $\mathcal{WI}$ proofs are closed under concurrent composition [FS90]). For completeness, we provide the proof. We show that any adversary R^* that violates the hiding property of $\langle C, R \rangle$ can be used to violate the hiding property of com. More precisely, given any adversary R^*, such that, for infinitely many $n \in N$, and $v_1, v_2 \in \{0, 1\}^n$, R^* distinguishes commitments to v_1 and v_2 made using $\langle C, R \rangle$, we construct a machine R' that distinguishes commitments to v_1 and

v_2 made using com. Note that the execution of a commitment of $\langle C, R\rangle$ to v_1 proceeds identically as that of a commitment to v_2 before the Stage 2 commitment of com is sent. Then by our hypothesis, there must exist a partial joint view ρ of the committer and R^* that determines the execution of the commitment before Stage 2, such that, conditioned on ρ occurring, R^* distinguishes commitments to v_1 and v_2. Let δ be a valid signature-chain corresponding to the transcript of Stage 1 in ρ. R' on auxiliary input ρ and δ proceeds as follows: it internally incorporates R^*, and feed R^* its part of view in ρ; it then forwards the external commitment made using com to R^* in Stage 2; in Stage 3, it gives $\mathcal{WI}$ proofs using δ as a "fake witness". Finally, it outputs whatever R^* outputs. From the $\mathcal{WI}$ property of Stage 3, it follows that R' distinguishes the commitment made using com, if R^* distinguishes the commitment made using $\langle C, R\rangle$ conditioned on ρ occurring.

□

Remark 1. *Both the definition of of signature-games and our non-malleable commitment protocols makes use of a* non-interactive *statistically-binding commitment scheme* com*. Both can be easily modified to work also with any two-round statistically binding commitment schemes* $\overline{\mathsf{com}}$*. In both cases, we the first message r of a commitment of* $\overline{\mathsf{com}}$ *is sent at the beginning of the execution, and then the rest of the execution proceeds just as if* $\overline{\mathsf{com}}$ *had been non-interactive. (Additionally, in the last stage of the protocol* $\langle C, R\rangle$*, the sender proves that either the Stage 2 message is the second message of a valid* $\overline{\mathsf{com}}$ *commitment with first message r, or it knows a signature-chain $\delta = (\bar{\sigma}, \bar{c}, \bar{r})$, such that, δ is well-formed, except that, for all i, $\bar{c}_i$ is the second message of a* $\overline{\mathsf{com}}$ *commitment to $\bar{\sigma}_{i-1}$, generated in responding to the first message r using randomness $\bar{r}_i$). Exactly the same proof as in Section 3.3 and Section 3.5 still go through using these modified* *construction, since commitments of $\overline{\mathsf{com}}$ are hiding, no matter what the first message is, and even if the first message is reused.*

3.5 Proof of Non-Malleability

In this section, we first show that $\langle C, R\rangle$ is stand-alone non-malleable. Then, in Sections 3.5.3 and 3.5.2, we extend the proof to show that $\langle C, R\rangle$ is also robust and concurrent non-malleable.

Theorem 1. *$\langle C, R\rangle$ is (one-one) non-malleable.*

Proof. The goal is to show that for every one-one man-in-the-middle adversary A that participates in one left and one right execution, the following ensembles are indistinguishable:

$$\left\{\mathsf{mim}^A_{\langle C,R\rangle}(v_1, z)\right\}_{n\in N, v_1, v_2\in\{0,1\}^n, z\in\{0,1\}^*}$$
$$\left\{\mathsf{mim}^A_{\langle C,R\rangle}(v_2, z)\right\}_{n\in N, v_1, v_2\in\{0,1\}^n, z\in\{0,1\}^*}$$

Towards this, we define a series of hybrid experiments $H_0, \ldots, H_{k+6}$. In each of these experiments, we show that the view of A, combined with the value that A commits to on the right, are indistinguishable. Let $\mathsf{hyb}_i(v, z)$ denote the random variable describing the view of $A(z)$, combined with the value it commits to in the right interaction in hybrid H_i (as usual, the committed value is replaced with $\perp$ if the right interaction fails or if A has copied the identity of the left interaction).

Hybrid H_0: In H_0 we first perfectly emulate a real execution of $\mathsf{mim}^A_{\langle C,R\rangle}(v, z)$—we call this the *Main Execution*—and next, if A successfully completed Stage 1 in the Main Execution, we try extract a "fake-witnesses" (i.e., a signature-chain) for the left interaction. More precisely, let id_l, vk_0, vk_1, vk_2, respectively be the identity and the verification keys of the left interaction in the Main Execution, and let $\psi = \mathsf{pattern}(\mathsf{id}_l)$; the *Extraction Procedure* now proceeds in $|\psi| = 2\ell$ iterations described below.

Iteration 1: If A successfully completes Stage 1 of the left interaction in the Main Execution, it must have provided three valid signatures $\theta_0, \theta_1, \theta_2$ of commitments to 0^m, where $m = \log \ell + n$, under keys vk_0, vk_1, vk_2 respectively. Since a signature-chain with pattern ψ starts off with a signature $\bar{\sigma}_1$ of a commitment to 0^m under key $vk_{\psi_1} = vk_2$, the procedure simply sets $\bar{\sigma}_1 = \theta_2$, $\bar{c}_1$ to be the transcript of the commitment to 0^m generated in iteration 2 (in Stage 1) of the left interaction, and $\bar{r}_1$ to be the randomness used in the commitment.

Iteration $i + 1$: Assume that at the end of the i^{th} iteration, for $i \in [2\ell - 1]$, the procedure has obtained a signature-chain δ_i of length i w.r.t. (keys vk_0, vk_1, vk_2 and) pattern $[\psi]_1^i$, containing signatures $\bar{\sigma}_1, \ldots, \bar{\sigma}_i$. Then, in iteration $i + 1$, we obtain a signature-chain δ_{i+1} of length $i + 1$, w.r.t. pattern $[\psi]_1^{i+1}$ by rewinding the appropriate slot in Stage 1 of the left interaction. More precisely, the procedure repeatedly rewinds A from where the slot ψ_{i+1} opens on the left in the Main Execution, and commits to the tuple $(i, \bar{\sigma}_i)$ (instead of 0^m) in the rewindings, until this left-slot closes successfully (i.e., A returns a valid signature on the commitment under key $vk_{\psi_{i+1}}$). In each of these rewindings, the right executions are emulated using fresh randomness; in particular, this means that whenever a rewinding goes beyond the point when a verification key is sent in the right interaction, in each such rewinding a fresh verification key is picked. Then the extraction procedure simply sets $\bar{\sigma}_{i+1}$ to be this signature, and again sets $\bar{c}_{i+1}$ and $\bar{r}_{i+1}$ to be the commitment and randomness used.

If the extraction procedure takes more than $2^{n/2}$ steps, it is "cut-off"; in this case, a signature chain can be recovered in time poly($2^{n/2}$) by our assumption on the signature scheme Π. The extraction procedure thus always terminates and always recovers a valid signature chain for the left interaction.

Since the view of A in the Main Execution in H_0 is perfectly emulated as in $\mathsf{mim}^A_{\langle C,R\rangle}(v,z)$, we trivially have that the view and value A commits to in H_0 is identically distributed to that in the real execution.

Claim 3. *For every $\mathcal{PPT}$ adversary A, it holds that:*

$$\left\{\mathsf{mim}^A_{\langle C,R\rangle}(v,z)\right\}_{n\in N,v\in\{0,1\}^n,z\in\{0,1\}^*} = \left\{\mathsf{hyb}_0(v,z)\right\}_{n\in N,v\in\{0,1\}^n,z\in\{0,1\}^*}$$

Hybrid H_1 to H_{k+5}: In hybrids H_1 to H_{k+5}, we change the witness used in the $k+5$ $\mathcal{WISSP}$ proofs in Stage 3 of the left interaction. More specifically, experiment H_i proceeds identically to H_{i-1}, except that in the first i proofs in Stage 3 of the left interaction, we prove that there exists a signature-chain w.r.t. vk_0, vk_1, vk_2 and pattern $\mathsf{pattern}(\mathsf{id}_l)$, by using the extracted signature-chain δ as a "fake-witness". We show that the view and value committed to on the right interaction in H_{i-1} and H_i are indistinguishable.

Proposition 3. *For every $\mathcal{PPT}$ adversary A, and every function $i : N \to N$, it holds that:*

$$\left\{\mathsf{hyb}_{i(n)-1}(v,z)\right\}_{n\in N,v\in\{0,1\}^n,z\in\{0,1\}^*} \approx \left\{\mathsf{hyb}_{i(n)}(v,z)\right\}_{n\in N,v\in\{0,1\}^n,z\in\{0,1\}^*}$$

Towards this, we reduce the indistinguishability of $\{\mathsf{hyb}_{i(n)-1}(v, z)\}$ and $\{\mathsf{hyb}_{i(n)}(v, z)\}$ to the witness indistinguishability of the Stage 3. More specifically, consider some adversary A, a function i, and a polynomial p, such that (for infinitely many $n \in N$, inputs $v \in \{0, 1\}^n$ and $z \in \{0, 1\}^*$,) $\mathsf{hyb}^{i(n)-1}(v, z)$ and $\mathsf{hyb}^i(n)(v, z)$ are distinguishable with probability $1/p(n)$. We show that there exists a $\mathcal{PPT}$ machine B that can violate the $\mathcal{WI}$ property of the $\mathcal{WISSP}$ protocol $\langle P, V\rangle$ used in Stage 3 of the protocol.

On a high-level, the machine B, on common input 1^n and auxiliary input v, z, externally interacts with a machine C that proceeds as follows: Machine C on input $b \in \{0, 1\}$, after receiving in the first round a statement x and two valid witnesses w_0 and w_1, gives an honest proof of $\langle P, V\rangle$ of the statement x using witness w_b. The statement x that B sends C corresponds to a left-interaction Stage 3 statement and the two witnesses w_0 and w_1 are respectively the real witness—the decommitment of the Stage 2 commitment in the left interaction—or the fake witness w_1—a signature chain for the left interaction.

Internally, B emulates an execution of either hyb^{i-1} or hyb^i with A (depending on the witness used in the external proof), except that, for the ith proof in Stage 3 of the left interactions, it forwards the statement, the real and fake witnesses, as well as messages in that proof externally to C. Furthermore, if the right interaction is successful and has a different identity from the left, B attempts to extract the value committed to on the right by repeatedly rewinding the $\mathcal{WISSP}$ proofs in Stage 3 of the right interaction by sending new challenge messages in this proof. Since the ith left-proof is forwarded externally, the rewinding has to be done in a manner that does not "affect" the ith left-proof. Roughly speaking, this is possible since

Description of B

Input: B receives a security parameter 1^n and v and z as auxiliary input.
Procedure: B externally interacts with a prover P of the $\mathcal{WISSP}$ protocol $\langle P, V\rangle$, receiving a proof of a statement x using witness w_0 or w_1, where x, w_0 and w_1 are chosen by B. Internally, it proceeds in the following three phases:

Simulation Phase: B internally emulates an execution of the experiment $\mathsf{hyb}_i(v, z)$ with A, with the exception that messages in the i^{th} left-proof of the Main Execution are forwarded externally to P. More precisely, at the beginning of the i^{th} left-proof, B sends the external prover P the statement x of the i^{th} proof, together with the "real witness" $w_0 = (v, r)$ (the decommitment of the Stage 2 commitment of the left interaction) and the "fake witness" $w_1 = \delta$ (the signature-chain of the left interaction extracted from A); B next forwards the proof of x generated by P (using either w_0 or w_1) to A as the i^{th} left-proof. Let Δ be the simulated view of A in the Main Execution.

Rewinding Phase: If the right interaction is successful and has a different identity from the left interaction in Δ, B extracts the value committed to in this interaction as follow:

- Find the first $\mathcal{WISSP}$ proof $(\alpha_1, \alpha_2, \beta, \gamma)$ in Δ, such that, during its the execution, no messages belonging to Stage 1 or the i^{th} proof of the left interaction are exchanged. (Such a $\mathcal{WISSP}$ proof must exist since there are $k+5$ $\mathcal{WISSP}$ proofs, whereas only $k+4$ messages in Stage 1 and the i^{th} proof of the left interaction.)
- Rewinds the proof by sending new random challenges β' until a second transcript $(\alpha_1, \alpha_2, \beta', \gamma')$ is obtained.

  In the rewindings, emulate the left and right interaction for A in identically the same way as in the Main Execution, except that, whenever A expects a new message in Stage 1 or the i^{th} proof of the left interaction, cancel the execution and start a new rewinding again.
- If $\beta_\rho \neq \beta'_\rho$, extract witness w from $(\alpha_1, \alpha_2, \beta, \gamma)$ and $(\alpha_1, \alpha_2, \beta', \gamma')$. Otherwise halt and output fail_1.
- If $w = (v, r)$ is valid decommitment for the right interaction, then set $\hat{v} = v$. Otherwise halt and output fail_2.

Output Phase: If the right interaction that is not convincing or the identity of the right interaction is the same as the left interaction, set $\hat{v} = \perp$. Output $\hat{v}$ and Δ.

Figure 3.4: The construction of B

there are more $\mathcal{WISSP}$ proofs in Stage 3 of the right interaction, than the number of messages in the ith left-proof. Therefore, in the right interaction, there exist some $\mathcal{WISSP}$ proofs that does not interleave with any messages in the ith left-proof, and B can use rewindings to extract a witness *without rewinding* the left-proof. Our actual rewinding strategy also avoids rewinding Stage 1 of the left interaction, so that the fake-witness δ of the left interaction remains a valid signature chain also in the rewindings, and thus can be reused to simulate the left interaction also in the rewindings. This is again possible since there are more right-proofs than the number of messages in Stage 1 and the ith proof in the left interaction. To slightly simplify the analysis, we additionally "cut-off" the rewindings if B takes more than $2^{n/2}$ steps and simply recover the value committed to in time poly($2^{n/2}$); recall that this is possible due to our assumption on com.

If during the rewindings, B sends the same challenge message twice, it aborts outputting fail_1. Additionally, if the witness extracted from the right interaction is not a valid decommitment (it could also be a fake-witness), B aborts outputting fail_2. Otherwise, B outputs the emulated view of A, together with the value committed to in the right interaction.

See Figure 3.4 for a formal description of B. Let $\mathsf{STA}_b(\langle P, V\rangle, B, v, z)$ denote the output of B in the above experiment with machine C on input b. Below, in Lemma 2, we show that the running-time of machine B is "bounded", in the sense that the probability that B runs for super-polynomial time is negligible.

Lemma 2. *There exists a polynomial function T, such that for every polynomial function q, every $b \in \{0, 1\}$, every sufficiently large $n \in N$, and inputs $v \in \{0, 1\}^n$ and $z \in \{0, 1\}^*$, the probability that machine B runs for more than $q(n)T(n)$ steps* *in an execution of the experiment* $\mathsf{STA}_b(\langle P, V\rangle, B, v, z)$ *is smaller than* $1/q(n)$.

Roughly speaking, the Lemma is proven by first bounding the running-time of a "hypothetical procedure" which perform all the same rewindings, but otherwise acts honestly (i.e., always commits to 0^m in Stage 1, and always uses the honest witness in Stage 3); it follows using a simple "$p \times 1/p$" argument (similar to those in [LPV08]) that the expected running-time of this procedure is polynomial. Next we show that, with high probability, the running-time of the actual procedure is not too far off. We note that due to reasons similar to those in [GK96] we are not able to bound the expected running-time of B. Additionally, it seems unclear if the methods of [GK96] could be applicable to obtains a simulation with an expected polynomial running-time. Fortunately, in our application, since we do not actually per se care about the running time of the simulation (but only care about breaking some specific security property, namely witness indistinguishability) our weaker bound suffices.

A formal proof of Lemma 2 can be found in Section 3.5.1.

The following lemma is the core of our analysis.

Lemma 3. *The following holds.*

$$\begin{aligned}
\{\mathsf{STA}_0(\langle P, V\rangle, B, v, z)\}_{n\in N, v\in\{0,1\}^n, z\in\{0,1\}^*} &\approx \left\{\mathsf{hyb}^{i-1}(v, z)\right\}_{n\in N, v\in\{0,1\}^n, z\in\{0,1\}^*}\\
\{\mathsf{STA}_1(\langle P, V\rangle, B, v, z)\}_{n\in N, v\in\{0,1\}^n, z\in\{0,1\}^*} &\approx \left\{\mathsf{hyb}^{i}(v, z)\right\}_{n\in N, v\in\{0,1\}^n, z\in\{0,1\}^*}
\end{aligned}$$

Before proceeding to the proof of 3, let us see how Lemma 3 and 2 together violate the $\mathcal{WI}$ property of the Stage 3 proofs. Recall that by our assumption, $\mathsf{hyb}^i(v, z)$ and $\mathsf{hyb}^i - 1(v, z)$ can be distinguished with probability $1/p(n)$; by Lemma 3, $\mathsf{STA}_0(\langle P, V\rangle, B, v, z)$ and $\mathsf{STA}_0(\langle P, V\rangle, B, v, z)$ can thus be distinguished with probability at least, say, $3/4p(n)$. By Lemma 2, the probability that B runs for more than, say, $4p(n)T(n)$ steps in either experiment is at most $1/4p(n)$. Therefore, by the union bound, the outputs of B (in STA_0 and STA_1) are still distinguishable with probability at least $1/4p(n)$, even if we cut-off the execution of B after $4p(n)T(n)$ steps (and output $\perp$ if B fails to complete), which is a contradiction.

Let us now turn to proving Lemma 3.

Proof. (of Lemma 3) By construction, B perfectly emulates the view of A in $\mathsf{hyb}^{i-1}(v, z)$ when receiving an external proof generated using the real witness w_0, and that in $\mathsf{hyb}^{i}(v, z)$ when receiving a proof generated using the fake witness w_1. Therefore, to show Lemma 3, it suffices to show that B (almost) always extracts a valid decommitment for the right interaction if it is successful and has a different identity from the left interaction (recall that by statistical binding of $\langle C, R\rangle$, the committed value is unique with overwhelming probability). In other words, showing Lemma 3 amounts to showing that the probability that B outputs fail_1 or fail_2 is negligible.

Claim 4. *There exists a negligible function μ, such that for every $b \in \{0, 1\}$, every sufficiently large $n \in N$, and inputs $v \in \{0, 1\}^n$ and $z \in \{0, 1\}^*$, the probability that B outputs fail_1 in $\mathsf{STA}_b(\langle P, V\rangle, B, v, z)$ is smaller than $\mu(n)$.*

Proof. Recall that B outputs fail_1 only if in some rewinding it picks the same challenge β' as the challenge β used in the same proof in the Main Execution. Since the number of rewindings by B is bounded by $2^{n/2}$ and the length of each challenge is n, by the union bound, the probability that this happens is negligible. □

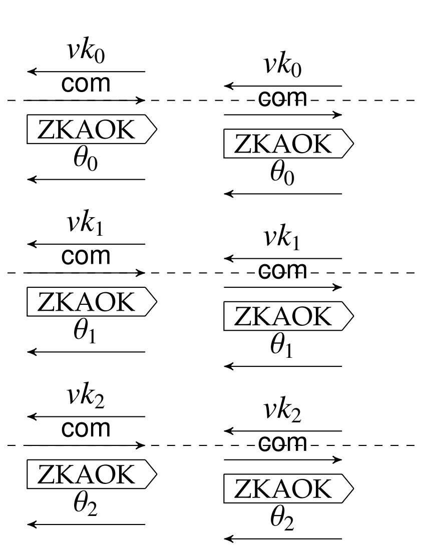
Claim 5. *There exists a negligible function μ, such that, for every $b \in \{0, 1\}$, every sufficiently large $n \in N$, and inputs $v \in \{0, 1\}^n$ and $z \in \{0, 1\}^*$, the probability that B outputs fail_2 in $\mathsf{STA}_b(\langle P, V\rangle, B, v, z)$ is smaller than $\mu(n)$.*

Proof. Assume for contradiction that there exists a polynomial $g(n)$, such that, with probability $1/g(n)$, B extracts an invalid decommitment from the right interaction. Towards reaching a contradiction, we consider another machine B', which proceeds identically to B except that it cuts-off the execution after $g(n)T(n)$ steps (and outputs $\perp$ in this case). It follows from Lemma 2 that the probability that B runs for more than $g(n)T(n)$ steps is at most $1/2g(n)$. Therefore, the probability that B' extracts out an invalid decommitment from the right interaction k is at least $1/2g(n)$. Furthermore, by the special-soundness property of the right-proofs, if the witness is not a valid decommitment, it must be a signature-chain δ w.r.t. the right-interaction keys vk'_0, vk'_1, vk'_2 and pattern $\mathsf{pattern}(\mathsf{id}_r)$. Consider the following two possible adversarial schedulings w.r.t. the left and the k^{th} right interactions in the Main Execution:

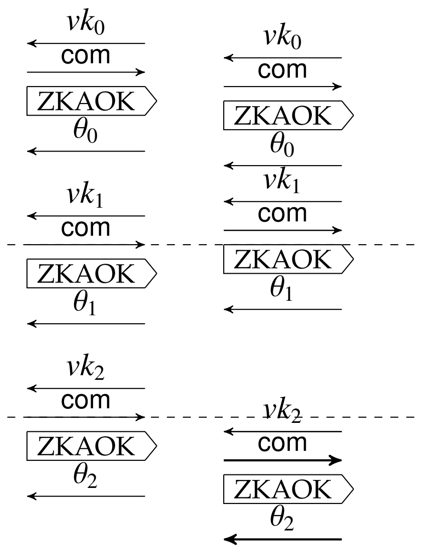
Scheduling 1: A "aligns" the slots in the left and right interactions *one by one*: a right-slot is said to be *aligned* with a left-slot if (1) its corresponding verification key is sent before the left-slot opens, and (2) its opening message (i.e., the commitment from A) is sent after the left-slot opens; see Figure 3.5 (i).

Scheduling 2: A does not align the slots in the left and right interactions; see Figure 3.5 (ii). This means that there exists some right-interaction slot that is not aligned with *any* left-interaction slot.

Since Scheduling 1 and 2 are the only two possible schedulings, by our



(i) Scheduling 1



(ii) Scheduling 2

Figure 3.5: The two schedulings of the messages in Stage 1 of the left and right interactions.

hypothesis, at least one of the following two conditions holds.

Condition 1: The probability that Scheduling 1 occurs in the Main Execution and that B' extracts an invalid decommitment from the right interaction is non-negligible.

Condition 2: The probability that Scheduling 2 occurs in the Main Execution and that B' extracts an invalid decommitment from the right interaction is non-negligible.

We show that neither condition can hold.

Assume Condition 1 holds. We reach a contradiction by constructing a machine C that externally participates in the signature game, while internally emulating an execution of $\mathsf{STA}_b(\langle P, V\rangle, B', v, z)$ except that messages in Stage 1 of the right interaction are emulated by forwarding the appropriate messages from the signature games to A. More precisely, C forwards the three verification keys vk_0, vk_1, vk_2 in the signature game to A as the verification keys in Stage 1 of the right interaction in the Main Execution. If Schedule 1 does not occur in the Main Execution, C simply aborts. Otherwise, whenever during some rewinding, A requests another signature in one of the slots on the Main Execution (and thus using one of vk_0, vk_1, vk_2), C obtains such a signature by accessing the appropriate signature oracle in the game and forwards it to A. Recall that whenever we rewind beyond the point where a verification key is sent, a new verification key is generated by B and thus B can obtain the appropriate signatures without querying the oracle. Since the left and right slots in the Main Execution are aligned one by one, we have that whenever the t^{th} left-slot is rewound, the adversary A may only request new signatures using key v'_t on the right. It follows that the "access-pattern" of the signatures requested is a sub-string of

$$\varphi = 012\|(\mathsf{id}_l)^*_1, 2^*, \ldots, (\mathsf{id}_l)^*_i, 2^*, \ldots, (id_l)^*_\ell$$

So, whenever B' extracts out a signature-chain δ w.r.t. (keys vk'_0, vk'_1, vk'_2 and pattern $\mathsf{pattern}(\mathsf{id}_r)$), C wins in the signature game since $\mathsf{pattern}(\mathsf{id}_r)$ is not a sub-string of φ (as $\mathsf{id}_r \neq \mathsf{id}_l$). Since the running-time of C is polynomial this contradicts Lemma 1.

Assume Condition 2 holds. We construct a machine C' just as in the previous case, except that C' abort whenever Schedule 2 does not happen in the Main Execution. When Scheduling 2 does occurs in the Main Execution, there exists a right-slot t that is not aligned with any left-slots; in other words, in all the rewindings where A gets to request a new signature in Slot t on the right, the rewinding goes beyond the point where the verification key for slot t is sent (and so new keys gets generated in each rewinding) and thus the t'th oracle is *never* used during the extraction phase. It follows that the access pattern in the signature game has a single character t, but the signature extracted is with respect to a pattern with two of each character. So, as in Condition 1, whenever B' extracts out a signature-chain δ, C' wins in the signature game. There is just one slight complication with the implementation of C': in the rewindings, B might rewind A in the middle of one of the $\mathcal{WISSP}$ in Stage 1, and since the $\mathcal{WISSP}$'s are not public-coin, we might not be able to emulate the continuation of the verifier strategy for this protocol. Note, however, that Lemma 1 still holds even if consider a slight variant of the signature game where after each $\mathcal{WISSP}$ the verifier reveals all of its random coins; this follows since this adjusted protocol would still be an $\mathcal{WISSP}$ and Lemma 1 no mayyer what $\mathcal{WISSP}$ we use in the signature game. C' can now easily be implemented as an adversary for this modified signature game. □

□

Hybrid H_{k+6} : Hybrid H_{k+6} proceeds identically to H_{k+5} except that the Stage 2 commitment of the left execution is emulated by committing to 0^n. It follows using the same argument as in hybrids H_i, for $i \in [k+5]$, that the value committed in the right interaction can be extracted without rewinding Stage 2 of the left interaction. It then follows from the hiding property of the Stage 2 commitment that the combined view and values committed to by A in H_{k+5} are indistinguishable from that in H_{k+6}.

It follows by a hybrid argument that,

$$\left\{\mathsf{mim}^{A}_{\langle C,R\rangle}(v,z)\right\}_{n\in N, v\in\{0,1\}^n, z\in\{0,1\}^*} \approx \left\{\mathsf{hyb}_{k+6}(v,z)\right\}_{n\in N, v\in\{0,1\}^n, z\in\{0,1\}^*}$$

Since the above holds for every value v, we have

$$\left\{\mathsf{mim}^{A}_{\langle C,R\rangle}(v_1, z)\right\}_{n\in N, v_1, v_2\in\{0,1\}^n, z\in\{0,1\}^*} \approx \left\{\mathsf{hyb}_{k+6}(v_1, z)\right\}_{n\in N, v_1, v_2\in\{0,1\}^n, z\in\{0,1\}^*}, \text{ and}$$
$$\left\{\mathsf{mim}^{A}_{\langle C,R\rangle}(v_2, z)\right\}_{n\in N, v_1, v_2\in\{0,1\}^n, z\in\{0,1\}^*} \approx \left\{\mathsf{hyb}_{k+6}(v_2, z)\right\}_{n\in N, v_1, v_2\in\{0,1\}^n, z\in\{0,1\}^*}$$

Finally, since by the definition of hyb_{k+6}, it holds that for every v_1, v_2 and z, $\mathsf{hyb}_{k+6}(v_1, z) = \mathsf{hyb}_{k+6}(v_2, z)$, we conclude that,

$$\left\{\mathsf{mim}^{A}_{\langle C,R\rangle}(v_1, z)\right\}_{n\in N, v_1, v_2\in\{0,1\}^n, z\in\{0,1\}^*} \approx \left\{\mathsf{mim}^{A}_{\langle C,R\rangle}(v_2, z)\right\}_{n\in N, v_1, v_2\in\{0,1\}^n, z\in\{0,1\}^*}$$

□

3.5.1 Proof of Lemma 2

Proof of Lemma 2. The running-time of B consists of three parts:

Part 1—Time spent simulating the Main Execution: Since A runs in strict polynomial time, the time $T_1(n)$ that B spends in the Main Execution polynomially bounded.

Part 2—Time spent extracting the left "fake-witness": We show that there exists a polynomial $T_2(n)$ such that for every polynomial q_2, (every $b \in \{0,1\}$, every sufficiently large $n \in N$, and inputs $v \in \{0,1\}^n$, $z \in \{0,1\}^*$,) the probability that B spends more than $q_2(n)T_2(n)$ steps extracting the left "fake-witness" in $\mathsf{STA}_b(\langle P, V\rangle, B, v, z)$ is smaller than $1/q_2(n)$.

Part 3—Time spent extracting the committed value on the right: We show that there exists a polynomial $T_3(n)$ such that for every polynomial q_3, the probability that B spends more than $q_3(n)T_3(n)$ steps extracting the right committed value in $\mathsf{STA}_b(\langle P, V\rangle, B, v, z)$ is smaller than $1/q_3(n)$.

So given an arbitrary polynomial q, we get by the union bound that, the probability B spends more than $2q(n)T_2(n)$ step in part 2, or more than $2q(n)T_3(n)$ steps in part 3, is smaller than $1/q(n)$. We conclude that there exists some sufficiently big polynomial $T(n) \geq T_1(n) + 2T_2(n) + 2T_3(n)$ such that for every polynomial q, the probability that B takes more than $q(n)T(n)$ steps is smaller than $1/q(n)$.

Analysis of Part 2: Recall that in an execution of $\mathsf{STA}_b(\langle P, V\rangle, B, v, z)$, the extraction of the left "fake-witness" proceeds in 2ℓ iterations. The running-time of the first iteration is clearly polynomial; we proceed to analyze the time spent in the remainder of the iterations. Recall that in an iteration $i > 1$, B takes the signature-chain $\delta_{i-1} = ([\bar{\sigma}]_1^{i-1}, [\bar{c}]_1^{i-1}, [\bar{r}]_1^{i-1})$ of length $i-1$, w.r.t. (keys vk_0, vk_1, vk_2 and) pattern $[\psi]_1^{i-1}$, (where $\psi = \mathsf{pattern}(\mathsf{id}_l)$,) obtained in the previous iteration, and extends it to a signature-chain $\bar{\sigma}_i$ of length i w.r.t. pattern $[\psi]_1^i$. This is done by repeatedly rewinding A from the start of the left-slot ψ_i and committing to $(i-1, \bar{\sigma}_{i-1})$ in the rewindings, until A closes this left-slot successfully. (Below we assume for simplicity that the extraction procedure is never cut-off and may run for more than $2^{n/2}$ steps, since this only increases the running time). Towards bounding the running-time of this extraction procedure, we first consider a *hypothetical procedure*, which proceeds almost the same as the actual extraction procedure, except that in the rewindings in iteration $i > 1$, instead of committing to $(i-1, \bar{\sigma}_{i-1})$, it commits to 0^m. In other words, the hypothetical procedure simulates the view of A in the rewindings using identically the same distribution as in the Main Execution. We show that the expected running-time of this hypothetical procedure is *poly*(n); we next bound the running-time of the actual extraction procedure.

Running-time Analysis of the Hypothetical Procedure: Let ψ = $\mathsf{pattern}(\mathsf{id}_l)$ be the pattern of the "fake-witness" of the left interaction. In iteration $i > 1$, the hypothetical extraction procedure repeatedly rewinds the left-slot ψ_i; let $\mathcal{T}^i$ be the random variable that describes the time spent in rewinding the left-slot ψ_i in iteration $i > 1$. We show that $E[T^i] \leq poly(n)$ and then by linearity of expectation, we conclude that the expected running-time of the hypothetical procedure is

$$\sum_{i=2}^{2\ell} E[T^i] \leq \sum_{i=2}^{2\ell} poly(n) \leq poly(n),$$

since the number of iterations is $poly(n)$.

Let us turn to bounding $E[T^i]$. Let Γ_{ψ_i} denote the set of prefixes ρ—i.e., partial transcripts of the Main Execution—from where the left-slot ψ_i opens. Given a prefix $\rho \in \Gamma_{\psi_i}$, we introduce the following notations:

- let $\Pr[\rho]$ denote the probability that ρ occurs as a prefix in the Main Execution;
- let p_ρ denote the probability that, conditioned on the prefix ρ occurring (in the Main Execution), the left-slot ψ_i closes successfully in the Main Execution.

Take any ρ from Γ_{ψ_i}. We claim that conditioned on ρ occurring, the expected value of T^i—denoted $E[T^i|\rho]$—is $poly(n)$. This follows since, first, the hypothetical procedure starts rewinding the left-slot ψ_i in iteration i only if this slot closes successfully in the Main Execution; hence, (conditioned on ρ occurring,) the probability the left-slot ψ_i is rewound is at most p_ρ. Secondly, once it starts rewinding the left-slot ψ_i, it continues until the slot closes successfully again;

since the hypothetical procedure proceeds identically in the rewindings as in the Main Execution, the probability that the left-slot ψ_i closes successfully in any rewinding is also p_ρ, and thus, (conditioned on ρ occurring,) the expected number of rewindings performed before this happens is $1/p_\rho$. Therefore, the overall expected number of rewindings from ρ is $p_\rho \times \frac{1}{p_\rho} = 1$. As each rewinding takes at most $poly(n)$ steps, we conclude that $E[T^i|\rho] \le poly(n)$. Thus,

$$E[T^i] = \sum_{\rho \in \Gamma_{\psi_i}} E[T^i|\rho] \Pr[\rho] \ \le \ poly(n) \times \sum_{\rho \in \Gamma_{\psi_i}} \Pr[\rho] \le \ poly(n)$$

Running-time Analysis of the Actual Extraction Procedure: Given that the expected running time of the hypothetical procedure is bounded by a polynomial $\tilde{T}(n)$, it follows using the Markov inequality that, for every polynomial q_2, (every b, every $n \in N$, and inputs v, z,) the probability that the hypothetical procedure takes more than $q_2(n)\tilde{T}(n)/2$ steps is smaller than $2/q_2(n)$. Then we claim that the probability that actual extraction procedure takes more than $q_2(n)\tilde{T}(n)/2$ steps is smaller than $1/q_2(n)$. This follows since the only difference between the hypothetical and the actual extraction procedures is that, in the former the rewindings are simulated by committing to 0^m using com, whereas in the latter rewindings are simulated by committing to a tuple that contains a signature. Since the $\mathcal{WISSP}$ proof following the commitment is never rewound, it follows directly from the hiding property of com and the zero knowledge property of the $\mathcal{WISSP}$ proof that, the probability that the actual extraction procedure runs for more than $q_2(n)\tilde{T}(n)/2$ steps differs from that of the hypothetical procedure by at most a negligible amount. Thus, for sufficiently large n, we have that the probability B spends more than $T_2(n) = q_2(n)\tilde{T}(n)/2$ steps is smaller than $1/q_2(n)$.

Analysis of Part 3: We show that the time that B spends in the Rewinding Phase is bounded by a polynomial $T_3(n)$ in expectation. It then follows by the Markov inequality that, for every polynomial q_3, the probability that B takes more than $q_3(n)T_3(n)$ steps is smaller than $1/q_3(n)$.

It follows from the same argument as in the above "running-time analysis of the hypothetical procedure" that to bound the expected time spent extracting the right committed value (also here, we consider the running-time without cut-offs), it suffices to bound the expected time spent in rewinding each right $\mathcal{WISSP}$ proof, since the total number of right-proofs is $poly(n)$. Then recall that a right-proof is rewound only if the proof completes successfully in the Main Execution, without interleaving with any message in Stage 1 or the i^{th} proof of the left interaction. On the other hand, once the rewinding starts, it continues until this right-proof completes successfully again, while cancelling every rewinding in which the proof interleaves with any message in Stage 1 or the i^{th} proof of the left interaction. Furthermore, as every rewinding is simulated exactly the same as in the Main Execution, it follows using the same "p times $1/p$ argument" as in the analysis of part 2 that the expected number of rewindings for every right-proof is 1, and hence the expected time spent in extracting the right committed value is bounded by a polynomial $T_3(n)$. □

3.5.2 Proof of Concurrent Non-Malleability

Let us turn to proving that $\langle C, R\rangle$ is also concurrently non-malleable. Recall that in Proposition 1, to show concurrent non-malleability, it suffices to prove that $\langle C, R\rangle$ is one-many non-malleable; that is, for every one-many man-in-the-middle adversary A, that participates in one left and many right interactions, the view of A and the values it commits to on the right are indistinguishable, no matter what value it is receiving a commitment to on the left. Towards this, we consider the same hybrid experiments H_0 to H_{k+6} as in the proof of stand-alone non-malleability. It follows from almost the same proof as before that the view of A and the values it commits to on the right are indistinguishable in sequential hybrids, except that, in hybrids H_1 to H_{k+6}, we (or more precisely, the simulator B) now need to extract out the values that A commits to in *all* the right interactions (recall that the proof relies on the fact that the value that A commits to in the right interaction can be extracted "efficiently", to show the indistinguishability of hybrid H_i and H_{i+1} for $1 \leq i \leq k + 6$). This is easy to achieve, since we can simply extract the values that A commits to in each right interaction *one by one*, after the Main Execution completes. More precisely, in the Rewinding Phase, for *every* successful right interaction that has a different identity from the left interaction in the Main Execution, B finds a $\mathcal{WISSP}$ proof in Stage 3 of this right interaction that does not interleave with any message in Stage 1 and the i^{th} proof (or Stage 2 for hybrid H_{k+6}) of that left interaction, and repeatedly rewinds the proof until a second transcript is obtain; it then computes a witness, if the two transcripts are different. Since there are only polynomial number of right interactions, it follows using almost the same proof of Lemma 2 that the running time of B is "bounded", and further using exactly the same proof of Lemma 3 that, except with negligible probability, the witnesses that B extracts out are indeed the values committed to in the right interactions. Thus by the $\mathcal{WI}$ property of the Stage 3 proofs (or the hiding property of Stage 2 resp.), the view and the values committed to by A are indistinguishable in hybrids H_i and H_{i+1} for $1 \leq i \leq k + 4$ (or in H_{k+5} and H_{k+6} resp.). We thus have:

Theorem 2. *$\langle C, R\rangle$ is concurrent non-malleable.*

3.5.3 Proof of Robust Non-Malleability

In this section, we show that, for any $r \in N$, $\langle C, R\rangle$ can be easily modified into a $O(r)$-round (concurrent) non-malleable commitment scheme $\langle \tilde{C}, \tilde{R}\rangle$ that is additionally one-many r-robust.

At a very high-level, one-many r-robust commitment schemes are easy to construct: any commitment scheme that is "extractable" and has more than r "rewinding slots" is directly one-many non-malleable w.r.t. r-round protocols, since the committed values of the right interactions can be extracted out from these rewinding slots, without hurting the indistinguishability property of the r-round left interaction (since there are more slots than the number of messages in the left), from which r-robustness follows. Therefore, to make our constant-round non-malleable commitment scheme $\langle C, R\rangle$ one-many r-robust, we simply add more $\mathcal{WISSP}$ proofs in Stage 3 of the protocol. More precisely, the commitment scheme $\langle C_r, R_r\rangle$ proceeds identically to $\langle C, R\rangle$, except that in Stage 3 of the protocol, the Committer $\tilde{C}$ needs to provide $\max(r+1, l)$ $\mathcal{WISSP}$ proofs (of the statement that either the Stage 2 message is a valid commitment or that it knows a "trapdoor"), where l is the number of $\mathcal{WISSP}$ proofs in Stage 3 of the original protocol $\langle C, R\rangle$.

Next we show that the modified protocol $\langle C_r, R_r\rangle$ is indeed r-robust. The main idea of the proof is to reduce the one-many r-robustness to the indistinguishability of the interaction with machine $B(y_n^1)$ or $B(y_n^2)$, by extracting the value committed to in the right interactions from the $\mathcal{WISSP}$ proofs in Stage 3 of the protocol, *without rewinding the left interactions*. This is achievable, (similar to the proof of the indistinguishability of Hybrid H_i and H_{i+1} in Section 3.5,) as there are more $\mathcal{WISSP}$ proofs in Stage 3 than the number of messages in the left interaction, and one can always find a $\mathcal{WISSP}$ proof that does not interleave with the left interaction and extract a witness from this proof, without rewinding the left interactions. The witness extracted must be a valid decommitment, as otherwise, by the special-soundness of the proof, it must be a valid signature-chain, which violates the soundness of the signature-game (since the adversary here is never rewound and obtains only three signatures during the straight-line execution of the right interaction). Therefore, we conclude that $\langle C_r, R_r \rangle$ is one-many r-robust. It follows using the same proof in Section 3.5 that $\langle C_r, R_r \rangle$ is stand-alone non-malleable; and it further follows using the same proof as in Section 3.5.2 that it is, in fact, also concurrent non-malleable.

Lemma 4. *For every $r \in N$, the protocol $\langle C_r, R_r \rangle$ has $O(r)$-round, and is concurrently non-malleable and one-many r-robust.*

The following theorem follows directly from Lemma 4.

Theorem 3. *Assume the existence of a commitment scheme. Then, for any constant r, there exists a constant-round commitment scheme that is r-robust concurrently non-malleable.*

Furthermore, for $r < l$, the protocol $\langle C_r, R_r \rangle$ is the same as $\langle C, R \rangle$; thus,

Corollary 1. *For any $r < l$, $\langle C, R \rangle$ is concurrently non-malleable and one-many r-robust.*

3.6 Constant-Round Secure Computation

As mentioned, "independence" of inputs is crucial for secure multi-party computation protocols. In the stand-alone setting, there has been a tight interplay between work on the round-complexity of multi-party computation (MPC) and work on non-malleable commitments.

Goldreich, Micali and Wigderson's [GMW87] original work on secure multi-party computation showed a $\Omega(m)$-round multi-party computation protocol based on the existence of enhanced trapdoor permutations (TDPs), where m is the number of players in the execution; implicit in their work is a $O(n)$-round non-malleable commitment for the special case of so-called "synchronizing" adversaries that have identities of length $\log n$. Subsequent works improved the round-complexity by making stronger assumptions. Following the work by Chor and Rabin [CR87], Katz, Ostrovsky and Smith [KOS03] obtained a $O(\log m)$-round MPC protocol assuming TDPs and dense-crypto systems by relying on the non-malleable commitments from [DDN00]. By additionally assuming the existence of hash-function collision-resistant against circuits of sub-exponential size (and non-black-box techniques), they also obtained a $O(1)$-round MPC protocol by instead relying on the non-malleable commitment from [Bar02]. More recently, Pass [Pas04], showed the existence of a $O(1)$-rounds MPC protocol assuming only TDPs and (standard) collision resistant hash functions (but still using non-black box techniques); this technique in turned was used in the non-malleable commitment of [PR05a].

The implicit connection between the round-complexity of non-malleable commitments and secure multi-party was formalized by Lin, Pass and Venkitasubramaniam in [LPV09]: they show that *the existence of k-round 4-robust non-malleable commitments and the existence of TDPs implies the existence of $O(k)$-round secure multi-party computation*. Combining the result of [LPV09] with Theorem 3, we get that secure multi-party computations can be performed in a constant number of round based on only TDPs.

Theorem 4. *Assume the existence of enhanced trapdoor permutations. Then there exists a constant-round protocol for secure multi-party computation.*

In fact, the result of [LPV09] applies to the much more general model of UC security, in various trusted set-up and relaxed security models including the CRS, URS, sun-spot, key registration, timing, tamper-proof hardware, and quasi-polynomial time simulation models. They also provide an essentially tight characterization on the models of computation that implies UC-security: They show that UC is feasible in any model of computation that implies the existence of a *UC-puzzle*—which, intuitively, is a protocol with the property that no adversary can successfully complete the puzzle and also obtain a trapdoor (or an answer to the puzzle), but there exists a simulator who can generate (correctly distributed) puzzles together with trapdoors. More precisely, they show that *in any model of computation that implies the existence of a t-round UC-puzzle, the existence of k-round 4-robust non-malleable commitments and the existence of TDPs implies the existence of $O(k + t)$-round UC-secure computation in that model.*

Then since it is relatively "easy" to construct a constant-round UC-puzzle in essentially all previously considered trusted set-up and relaxed security models, combining the result in [LPV09] with Theorem 3, we get that UC-security can be achieved in a constant number of round based on only TDPs, in essentially all previously considered trusted set-up and relaxed security models.

Theorem 5. *Assume the existence of enhanced trapdoor permutations. Then there exists a constant-round protocol for UC-secure computation, in the CRS, URS, sun-spot, key registration, timing, tamper-proof hardware, and quasi-polynomial time simulation models.*

3.7 Historical Notes

This result in this chapter is a combined version of results in two papers of [LP09] and [LP11b]. In [LP09], we first showed the existence of a $O(1)^{\log^* n}$-round protocol that is based on the existence of one-way functions and uses a black-box proof of security. On a high-level, the main technique of [LP09] was a method for *amplifying non-malleability*: that is, we presented a method for transforming a non-malleable commitment scheme that handles identities of length t into one that handles identities of length $O(2^t)$. The $O(1)^{\log^* n}$-round protocol was finally obtained starting off with the protocol of DDN for constant length identities and next iteratively amplifying it. The notion of robust non-malleability was also first defined in [LP09] and was an integral part of the amplification procedure: in fact, our amplification procedure could only be applied to robust non-malleable commitments.

In [LP09], we additionally pointed out that amplification procedure also yield a natural route towards constant-round non-malleable commitments: it suffices to come with a constant-round protocol that handles identities of length $\log^k n = \log\log\ldots\log n$, where k is a constant; any such protocol can be amplified to a full-fledged non-malleable commitment while still remaining constant round. Subsequent work by Pass and Wee [PW10] obtained a constant-round protocol based on sub-exponentially hard one-way functions (again using a black-box proof of security), by following this paradigm: sub-exponential one-way functions were used to construct a constant-round non-malleable commitment for "small" identities, and the protocol can then be amplified into a full-fledged one. An elegant work by Wee [Wee10] later simplified an improved the amplification procedure of [LP09], leading to a protocol using $O(\log^* n)$-rounds, based on one-way functions.Finally, independently of [LP11b], a beautiful work by Goyal also obtains a constant-round non-malleable commitment based on one-way functions; the construction of [Goy11] also follows the above amplification paradigm by Goyal instead constant-round robust non-malleable commitment protocol for small identities based on one-way functions.

The construction from [LP11b] is direct: we no longer require amplification; instead we directly construct a full-fledged robust non-malleable protocol. Nevertheless, some of the ideas used in the amplification procedure are helpful when analyzing our protocol.

CHAPTER 4

CCA-SECURITY AND CONCURRENT SECURITY WITHOUT ADDITIONAL "TRUST"

In this chapter, we introduce a new and stronger way of capturing input independence than non-malleability—the notion of robust CCA-security. We show that robust CCA-secure commitments can be constructed in the plain model from any one-way functions. This gives the first implementation of a primitive with adaptive hardness property based on standard non adaptive assumption. Then, by using our robust CCA-secure commitments as a key tool, we achieve a meaningful notion of concurrent security in the plain model from standard hardness assumptions, enabling, for the first time, secure multi-party computation in the concurrent setting without additional "trust".

4.1 Definition of CCA-Secure Commitments

4.1.1 Efficiently Verifiability

When considering robust CCA-security, we focus on commitment schemes that are *efficiently verifiable*. Roughly speaking, a commitment scheme $\langle C, R\rangle$ is efficiently verifiable if the honest receiver, after participating in an interaction of the commit stage of $\langle C, R\rangle$ with an arbitrary (potentially unbounded) committer, can already efficiently verify whether the commitment produced at the end of the commit stage has a valid decommitment or not. Formally,

Definition 22. *A statistically binding commitment scheme $\langle C, R\rangle$ is* efficiently verifiable, *if it has the following structure:*

- *At the end of the commit stage, the committer C and the receiver R produce a joint output, the commitment c; additionally R outputs a bit b. In an honest execution between C and R, the receiver R always outputs 1. We say that the receiver R accepts in this case and rejects if it outputs 0.*

  We say that a commitment is accepting *if R outputs 1 at the end of the commit stage, and a commitment is* valid *if there exists a valid decommitment (v, d) that makes R accept in the reveal stage.*

- *For every machine A, the probability that A in an interaction with the honest receiver R in the commit stage of the $\langle C, R\rangle$, produces a commitment that is accepting but not valid is negligible.*

4.1.2 CCA-Security

Security under chosen-ciphertext-attacks (CCA security) [RS91] has been studied extensively in the context of encryption schemes, where the confidentiality of encrypted messages is guaranteed even in the presence of a decryption oracle. We here define an analogous notion for commitment schemes. Roughly speaking, a commitment scheme is CCA (chosen-commitment-attack) secure if the commitment scheme retains its hiding property even if the receiver has access to a "decommiment oracle".

Let $\langle C, R\rangle$ be a tag-based efficiently verifiable commitment scheme with $l(n)$-bit identities. A decommitment oracle O of $\langle C, R\rangle$ acts as follows in interaction with an adversary A: it participates with A in many sessions of the commit phase of $\langle C, R\rangle$ as an honest receiver, using identities of length n, chosen adaptively by A. At the end of each session, if the session is *accepting and valid*, it reveals a decommitment of that session to A; otherwise, it sends $\perp$. (Note that when a session has multiple decommitments[1], the decommitment oracle only returns one of them. Hence, there might exist many valid decommitment oracles.)

Loosely speaking, a tag-based efficiently verifiable commitment scheme $\langle C, R\rangle$ is said to be CCA-secure, if there *exists* a decommitment oracle $\mathcal{O}$ for $\langle C, R\rangle$, such that the hiding property of the commitment holds even with respect to adversaries with access to $\mathcal{O}$. More precisely, denote by $A^{\mathcal{O}}$ the adversary A with access to the decommitment oracle $\mathcal{O}$. Let $\mathsf{IND}_b(\langle C, R\rangle, \mathcal{O}, A, n, z)$, where $b \in \{0, 1\}$, denote the output of the following probabilistic experiment: on common input 1^n and auxiliary input z, $A^{\mathcal{O}}$ (adaptively) chooses a pair of challenge values (v_0, v_1) $\in \{0, 1\}^n$—the values to be committed to—and an identity $\mathsf{id} \in \{0, 1\}^{l(n)}$, and receives a commitment to v_b using identity id. Finally, the experiment outputs the output y of $A^{\mathcal{O}}$; the output y is replaced by $\perp$ if during the execution A sends $\mathcal{O}$ any commitment using identity id (that is, any execution where the adversary queries the decommitment oracle on a commitment using the same identity as the commitment it receives, is considered invalid).

Definition 23 (CCA-secure Commitments.)**.** *Let $\langle C, R\rangle$ be a tag-based efficiently verifiable 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. *$\mathcal{O}$, if for every $\mathcal{PPT}$ ITM A, the following ensembles are computationally indistinguishable:*

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

[1]Note that the statistically binding property only guarantees that, with overwhelming probability, the committed value is unique. However, there may still exist many different decommitments.

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

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

As mentioned in the introduction, CCA-security easily implies concurrent non-malleability [DDN00, PR03, LPV08]. The proof is standard and omitted. Note that in the definition of CCA-security, we could also have considered an adversary that can select a pair of *sequences* $\vec{v}_0, \vec{v}_1$ of challenge messages, and receives commitments to one of the sequence $\vec{v}_b$ using a sequence $\vec{\mathsf{id}}$ of identities of its own choice. (The output of the experiment IND_b is now replaced with $\perp$ if A copies any of the left identities on the right). It follows using a standard hybrid argument that CCA-security implies security also in this setting; we omit the proof.

4.1.3 k-Robust CCA-Security

We introduce a strengthening of the notion of CCA-security analogously to the notion of *robust non-malleable commitments* introduced in Chapter 3. We here consider a man-in-the-middle adversary that participates in an *arbitrary* left interaction with a *limited number of rounds*, while having access to a decommitment oracle.

Definition 24. *Let $\langle C,R\rangle$ be a tag-based efficiently verifiable commitment scheme, with $l(n)$-bit identities, and O a decommitment oracle for it. We say that $\langle C,R\rangle$ is* k-robust CCA-secure w.r.t. *O, if $\langle C,R\rangle$ is CCA-secure w.r.t. O, there exists a $\mathcal{PPT}$ simulator S, such that, for every $\mathcal{PPT}$ adversary A, and every $\mathcal{PPT}$ k-round ITM B, the following* *two ensembles are computationally indistinguishable.*

- $\left\{\mathsf{out}_{B,A^O}[\langle B(y), A^O(z)\rangle(1^n)]\right\}_{n\in N, y\in\{0,1\}^*, z\in\{0,1\}^*}$
- $\left\{\mathsf{out}_{B,S^A}[\langle B(y), S^{A(z)}\rangle(1^n)]\right\}_{n\in N, y\in\{0,1\}^*, z\in\{0,1\}^*}$

Thus, roughly speaking, $\langle C, R\rangle$ is k-robust if the (joint) output of every k-round interaction, with an adversary having access to the oracle $\mathcal{O}$, can be simulated without the oracle. In other words, having access to the oracle does not help the adversary in participating in any k-round protocols.

We say that a tag-based efficiently verifiable commitment $\langle C, R\rangle$ with $l(n)$-bit identities is robust CCA-secure if there exists a a decommitment oracle $\mathcal{O}$, such that, $\langle C, R\rangle$ is k-robust CCA-secure w.r.t. $\mathcal{O}$, for every constant k.

On the identity length Recall that in the definition of CCA-security, the adversary can pick arbitrary identities id of length n for both the left and the right interaction. We may also consider a restricted notion of (robust) CCA-security where the adversary is restricted to use identities of some bounded length. As we show below, standard techniques [DDN00] can be used to show that any *robust* CCA-secure commitment that is secure for identities of length $\ell(n) = n^\varepsilon$ can be turned into a robust CCA-secure commitment (that is secure for identities of length n), by adding one extra message at the beginning of the protocol.

Proposition 4. *Let ε be any constant such that $0 < \varepsilon < 1$, ℓ a polynomial such that $\ell(n) = n^\varepsilon$, and $\langle C, R\rangle$ a k-round robust CCA-secure commitment scheme with ℓ-bit identities. Then assuming the existence of one-way functions, there exists a robust $k+1$-round CCA-secure commitment scheme $\langle\hat{C}, \hat{R}\rangle$ with n-bit identities.*

Proof. The transformation from a robust CCA-secure commitment $\langle C, R\rangle$ for short (n^{ε}-bit) identities into one for long identities (n-bit) uses a standard technique from [DDN00] as follows: to commit to a message $v \in \{0,1\}^n$, the committer $\hat{C}$ on common input a security parameter $n \in N$ and an identity $\mathsf{id} \in \{0,1\}^n$, first generates a key pair (sk, vk) of a signature scheme, such that the verification-key vk is of length $n^{\varepsilon 2}$, and sends vk and a signature of the n-bit identity id (using sk) to the receiver in the first stage. (The receiver verifies that the signature is valid; it aborts otherwise). Then, in the second stage, it simply commits to v using $\langle C, R\rangle$ and the verification key vk as the identity.

Let O be a decommitment oracle of $\langle C, R\rangle$, with respect to which $\langle C, R\rangle$ is robust CCA-secure. Then consider a decommitment oracle $\hat{O}$ of $\langle \hat{C}, \hat{R}\rangle$ that extends O in the following straightforward way: to decommit an accepting commitment of $\langle \hat{C}, \hat{R}\rangle$, $\hat{O}$ returns the decommitment that O returns for the Stage 2 commitment of $\langle C, R\rangle$. Below we show that $\langle \hat{C}, \hat{R}\rangle$ is robust CCA-secure w.r.t. $\hat{O}$. First, it follows directly from the robustness of $\langle C, R\rangle$ w.r.t. O (and the fact that, given access to O, the decommitment oracle $\hat{O}$ can be emulated perfectly) that $\langle \hat{C}, \hat{R}\rangle$ is robust w.r.t. $\hat{O}$. Then, for CCA security, consider an arbitrary adversary A that participates in an experiment of $\mathsf{IND}_b(\langle \hat{C}, \hat{R}\rangle, A, n, z)$. We claim that, except with negligible probability, A never picks the same verification key vk, as that picked in the left interaction (by the left committer), in any accepting right interaction that has a different identity from the left interaction. Otherwise, we could construct an adversary, who with access to $\hat{O}$, is able to forges a signature of a randomly chosen key (corresponding to the key chosen by the left committer); then, by the robustness of $\langle \hat{C}, \hat{R}\rangle$ w.r.t. $\hat{O}$, there exists a simulator B that is able

[2]The existence of signature schemes is implied by the existence of one-way functions [Rom90]. To get a signature scheme with a "short" verification-key, simply "scale-down" the security parameter.

to forge a signature even without access to $\hat{O}$, which violates the unforgibility of the signature scheme. In other words, every successful right interaction that has a different identity from the left, also has a different verification key from the left interactions. It thus follows from the CCA-security of $\langle C, R\rangle$ that $\langle \hat{C}, \hat{R}\rangle$ is CCA-secure w.r.t. $\hat{O}$. □

4.2 A CCA-Secure Commitment Scheme

In this section, we show the following theorem.

Theorem 6. *Assume the existence of one-way functions. Then, for every $\delta > 0$, there exists an $O(n^\delta)$-round robust CCA-secure commitment scheme (where n is the security parameter).*

Our CCA-secure commitment $\langle C, R\rangle$ is based on a variant of the concurrent non-malleable commitment protocol of [LPV08], which in turn is based on the message scheduling technique of [DDN00]. However, here we use a slightly different message schedule in order to provide more "safe" rewinding slots. For simplicity of exposition, below we describe out protocol assuming the existence of one-way functions with efficiently recognizable range, but the protocol can be easily modified to work with any arbitrary one-way function (see Remark 2 for more details). Furthermore, we also rely on 3-round special-sound proofs in our protocol, but the analysis also works also with 4-round proofs (see Remark 3 for more details). Additionally, we will rely on a statistically binding two-round commitment scheme com with binding error 2^{-2n}; the construction in [HILL99, Nao91] gives such a commitment scheme. [3]

[3]Any statistically binding commitment scheme with exponentially small binding error can

Let ℓ and η be polynomials in the security parameter n. To commit to a value v, the Committer C and the Receiver R, on common input 1^n and the identity $\mathsf{id} \in \{0,1\}^{\ell(n)}$, proceed in the following three stages in the commit phase.

- *Stage 1:* the Receiver picks a random string $r \in \{0,1\}^n$, and sends its image $s = f(r)$, through a one-way function f with an efficiently recognizable range to the Committer. The Committer checks that s is in the range of f and aborts otherwise. Additionally, the receiver also sends the first messages r_1, r_2 for two commitments, using a two-round statistically binding string commitment com.
- *Stage 2:* The Committer provides a commitment c_1 to v using the commitment scheme com and r_1 as the first message; let (v, d) denote the decommitment information. Next, it provides a commitment c_2 to (v, d) using com and r_2 as first message. We refer to (r_1, c_1) as the *first commitment* and (r_2, c_2) as the *second commitment*[4].
- *Stage 3:* The Committer proves that
  - (r_1, c_1) is a valid commitment to v and (r_2, c_2) is a valid commitment to a decommitment pair for (r_1, c_1),
  - *or* s is in the image set of f.

  This is proved using $4\ell(n)\eta(n)$ invocations of a special-sound $\mathcal{WI}$ proof system where the verifier query has length $3n$.[5] The messages in the proofs are scheduled based on the identity id and relies on scheduling pairs of

be turned into a scheme with binding error 2^{-2n} by using a scaled-up security parameter. As we shall see later, this property will be used later in the security proof.

[4]A statistically binding commitment may have multiple decommitments. However, here the second commitment uniquely decides one decommitment of the first commitment. As we shall see later in the security proof, this helps us define a decommitment oracle with respect to which the protocol is robust CCA secure.

[5]As we shall see later on, the length restriction will facilitate the security proof.

proofs according to schedules design_0 and design_1 depicted in Figure 4.1. More precisely, the proof stage consist of $\ell(n)$ phases. In phase i, the committer provides $\eta(n)$ sequential $\mathsf{design}_{\mathsf{id}_i}$ pairs of proofs, followed by $\eta(n)$ sequential $\mathsf{design}_{1-\mathsf{id}_i}$ pairs of proofs.

In the reveal phase, the Committer simply decommits to the first commitment (r_1, c_1). The Receiver accepts if the decommitment is valid and rejects otherwise.

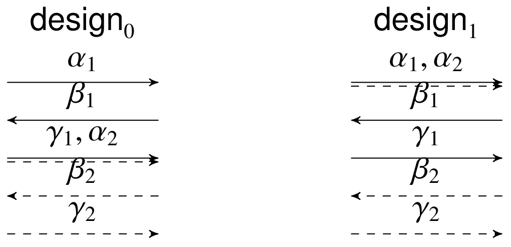


Figure 4.1: Description of the schedules used in Stage 3 of the protocol. $(\alpha_1,\beta_1,\gamma_1)$ and $(\alpha_2,\beta_2,\gamma_2)$ are respectively the transcripts of a pair of 3-round special-sound proofs.

On the round complexity of $\langle C, R\rangle$: The round complexity of $\langle C, R\rangle$ is $O(\ell(n)\eta(n))$. We will show that $\langle C, R\rangle$ is robust CCA-secure when $\eta(n) = n^{\epsilon}$ for any constant $\epsilon > 0$. Furthermore, by Proposition 4, it suffices to consider identities of length $\ell(n) = n^{\varepsilon'}$ for any constant $\varepsilon' > 0$. Thus by setting ε and ε' properly so that $\delta = \varepsilon + \varepsilon'$, we obtain a $O(n^{\delta})$-round protocol.

Proposition 5. *$\langle C, R\rangle$ is a statistically binding commitment scheme with efficient verifiability.*

Proof. It follows using essentially the same proof as in [LPV08] that the protocol is statistically binding and computationally hiding; for completeness, we provide the proof below.

Binding: The binding property follows directly from the binding property of com.

Hiding: The hiding property essentially follows from the hiding property of com and the fact that Stage 3 of the protocol is $\mathcal{WI}$ (since $\mathcal{WI}$ proofs are closed under concurrent composition [FS90]). More precisely, we show that any adversary R^* that violates the hiding property of $\langle C, R\rangle$ can be used to violate the hiding property of com. More precisely, given any adversary R^* (without loss of generality, deterministic) that distinguishes a commitment made using $\langle C, R\rangle$, we construct a machine R' that distinguishes commitments—two commitments constructed in the same way as that in Stage 2 of $\langle C, R\rangle$—made using com. Let s be the value in the range of one way function f sent by R^* in the first message. R' on auxiliary-input a "fake" witness r such that $s = f(r)$, proceeds as follows. It internally incorporates R^* and forwards the external commitments to R^* in Stage 2. In Stage 3, R' gives $\mathcal{WI}$ proofs using the "fake witness" r. Finally, it outputs whatever R^* outputs. From the $\mathcal{WI}$ property of Stage 3, it follows that R' distinguishes the commitments made using com, if R^* distinguishes the commitment made using $\langle C, R\rangle$. On the other hand, it follows from the hiding property of com that no adversary can distinguish commitments of com even if it receives an arbitrary number of commitments. This gives a contradiction.

Efficient Verifiability: It remains to show that the validity of an $\langle C, R\rangle$ commitment is efficiently checkable. By construction, an $\langle C, R\rangle$ commitment is valid if and only if the "first commitment" is valid. It then follows from the soundness of Stage 3 of the protocol that, whenever the receiver is accepting (at the end of the commit phase), the first commitment is valid except with negligible probability. Thus, the validity of $\langle C, R\rangle$ is also efficiently checkable.

$\square$

We turn to show that $\langle C, R\rangle$ is a robust CCA-secure commitment when $\eta(n) = n^\epsilon$ for any constant $\epsilon > 0$.

Theorem 7. *Let $\epsilon > 0$ be a constant, and let $\eta(n) = n^\epsilon$. Then $\langle C, R\rangle$ is a robust CCA-secure commitment.*

To show that $\langle C, R\rangle$ is a robust CCA-secure commitment, we need to exhibit a decommitment oracle O for $\langle C, R\rangle$ such that $\langle C, R\rangle$ is robust CCA-secure w.r.t. O. Consider the following decommitment oracle O: O acts just as an honest receiver during the commit phase. At the end of every accepting interaction, if the "second commitment" (r_2, c_2) defines a unique value (v, d) such that (v, d) is a valid decommitment for the "first commitment" (r_1, c_2), O returns (v, d). Otherwise, O returns the lexicographically first decommitment, or $\perp$ if there does not exists a valid decommitment. The main technical challenge consists of proving the following proposition.

Proposition 6. *$\langle C, R\rangle$ is robust CCA-secure w.r.t O.*

We provide a high-level overview of the proof below. The formal proof can be found in Second 4.3: we will first introduce the notion of safe-points, and then provide the proofs of CCA security and robustness, which rely on the notion of safe-points, in Section 4.3.2 and 4.3.3 respectively.

4.2.1 Proof Overview of Robust CCA-Security

Towards proposition 6, we first argue that $\langle C, R\rangle$ is CCA-secure w.r.t. $\mathcal{O}$. Consider the CCA-experiment IND_b, where the adversary A interacts with an honest committer C, and is given access to $\mathcal{O}$. We refer to its interaction with C as the *left interaction*, and its interactions with $\mathcal{O}$ as the *right interactions*. Recall that proving CCA-security w.r.t. $\mathcal{O}$ amounts to showing that the views of A in experiments IND_0 and IND_1 are indistinguishable (when A has oracle access to $\mathcal{O}$). The main hurdle in showing this is that the oracle $\mathcal{O}$ is not efficiently computable; if it were, indistinguishability would directly follow from the hiding property of the left interaction. However, since $\langle C, R\rangle$ consists of a sequence of special-sound proofs of the committed value, the oracle $\mathcal{O}$ can be efficiently implemented by "rewinding" the special-sound proofs in the right interactions. But, the problem is that once we start rewinding the right interactions, A might send new messages also in the left interaction. So, if done naively, this would require us to also rewind the left interaction, which could violate its hiding property.

The crux of the proof is showing how to appropriately rewind the right interactions (so as to extract out the right committed values), without violating the hiding property of the left interaction. This implies that the view of A in IND_b can be efficiently emulated, without the oracle $\mathcal{O}$, and hence by the hiding property of the left interaction, A's view is indistinguishable. Towards doing this, we rewind the right interaction only at special points in the interaction; we call such points $\mathsf{safe\ points}$.

Note that the hiding property of the left interaction remains intact if during the rewinding, one of the following two cases occurs.

- *Case 1:* A does not request any new messages in the left interaction.
- *Case 2:* The only new messages A requests in the left interaction are *complete $\mathcal{WISSP}$* proofs.

The fact that the left interaction is still hiding even if case 2 happens follows using exactly the same proof as the proof of hiding of $\langle C, R\rangle$. Obviously, the left interaction remains hiding if case 1 happens.

Roughly speaking, we now say that a prefix ρ of a transcript Δ (which consists of one left interaction and many right interactions) is a safe-point for the k^{th} right interaction if ρ "lies in between" the first two messages α_r and β_r of a $\mathcal{WISSP}$ proof $(\alpha_r, \beta_r, \gamma_r)$ in the k^{th} interaction (i.e., it contains α_r but not β_r), and satisfies that from ρ to the point when γ_r is sent, one of the two cases above occurs in the left interaction. We call $(\alpha_r, \beta_r, \gamma_r)$ the *$\mathcal{WISSP}$ associated* with the safe point ρ. It follows using a combinatorial argument (similar in spirit, but more complicated than, [DDN00, LPV08]) that the message scheduling in Stage 3 of $\langle C, R\rangle$ guarantees the following key property:

> Let Δ be any transcript of one left interaction, and many right interactions, of $\langle C, R\rangle$. Then, any right interaction k that 1) has completed, and 2) uses a different identity from the left interaction, has $\Omega(n^\epsilon)$ *non-overlapping $\mathcal{WISSP}$* (i.e., $\mathcal{WISSP}$'s that are sequentially arranged) that are associated with a safe point in Δ.

We will now use these safe-points to construct a simulator that can efficiently emulates the oracle $\mathcal{O}$. (As mentioned in the Introduction, the reason we need "many" safe-points is to ensure that we can extract out the committed values in all the right interactions while ensuring an expected polynomial running-time.) On a high-level, the simulation emulates O by following the honest receiver strategy of $\langle C, R\rangle$, until it encounters a "good" safe-point ρ. It then keeps rewinding the execution back to ρ until it obtains another accepting transcript of the right proof associated with ρ. Once two accepting proof transcripts are obtained, the special-soundness property allows the simulator to extract the decommitment information.

More precisely, the simulation is defined recursively in the following manner: On recursion level d, we say that a safe-point ρ of a transcript Δ is "good" (we call this a $d+1$-good safe-point) if the number of right-execution $\mathcal{WISSP}$ proofs—possibly from different interactions—starting in between ρ and the point where γ_ρ is sent in Δ, is smaller than $k_d = M/\eta'^{d+1}$, where M is a (polynomial) upper-bound on the total number of messages in Δ, and $\eta' = n^{\varepsilon'}$ for some constant ε' such that $0 < \varepsilon' < \varepsilon$.

Then, on recursion level d, the simulator emulates every right interaction honestly, but as soon as it encounters a $d+1$-good safe-point in the current transcript, it begins "rewinding" the execution back to the safe-point, and invokes itself recursively at level $d+1$. In each rewinding, if it notices that ρ might no longer be a $d+1$-good safe-point, it cancels the rewinding and starts a new rewinding. It continues the process until it gets another transcript where ρ is a $d+1$-good safe-point again; from this second transcript we can extract out (and store) the decommitment information for the right interaction associated with the safe-point. Finally, whenever in the emulation a right interaction completes, the simulator provides A with the decommitment information extracted out (or outputs `fail` if the decommitment information has not been recovered). By cancelling "bad" rewindings (i.e., rewindings that are not $d+1$-good safe-points) we are guaranteeing two properties: 1) we *never* violate the hiding property of the left interaction, and 2) the running-time of the simulation does not blow up.

Let us now briefly argue that the simulator indeed emulates O both correctly and efficiently, without violating the hiding property of the left interaction.

Hiding property of the left interaction: Since the simulator only rewinds the right interactions from safe-points, and cancels every rewinding in which the point is no longer a safe-point, it follows that the left interaction remains hiding.

Correctness: We argue that for each right interaction that uses an identity that is different from the left interaction, we extract out the decommitment information before the interaction completes successfully. First, note that the recursion level is bounded by $c = \log_{\eta'} M$, which is a constant (since is M is polynomial in n and $\eta' = n^{\epsilon'}$). Since each successful right interaction that uses a different identity than the left interaction has n^{ϵ} safe-points (by the key property above), it follows that for each such interaction, there exists some recursive level d, such that the right interaction has at least $n^{\epsilon}/c > \eta'$ safe-points on level d. But, as the total number of right-proofs that start on level d is bounded by $k_d = M/\eta'^d$ (otherwise, the simulation at this recursive level is cancelled), there must exist one right-proof with an associated safe-point ρ, such that less than M/η'^{d+1} right-proofs start in between ρ and the last message of the proof. Therefore ρ is a $d+1$-good safe-point and will be rewound. Finally, since we continue rewinding until the decommitment information is found, it follows that for each successful right interaction that uses a different identity than the left interaction, the decommitment information is recovered by the simulator.

Efficiency: To prove that the simulation is efficient, consider a simplified scenario where A *never* sends a com commitment that can be decommitted to two different values—i.e., A never manages to violate the statistical binding property of com. We argue that the simulation is efficient in this case. (It suffices to consider this case, since the probability that A violates the statistical binding property of com is negligible, and thus we can always "cut-off" the simulation without loosing "too much".) To prove that the expected running-time of the simulation (in the simplified scenario) is polynomially bounded, first recall that the recursive depth is a constant c. Secondly, at each recursive level d, there are at most M possible points from which we can rewind and from each of these points, the expected number of rewindings is 1. The latter follows since the simulator only starts rewinds from a point ρ if it is a $d+1$-good safe-point, and it continues rewinding until ρ becomes a $d+1$-good safe-point again; furthermore, in each of the rewindings the simulated view of the adversary is identically distributed to its view in the first execution (this fact relies on us considering the case when all com commitments are well-defined). Thus, the probability that a point is a $d+1$-good safe-point (conditioned on it occurring as a prefix in the execution) is the same in the first execution and in the rewindings. Therefore, the expected number of recursive calls starting from ρ is 1. We now conclude that the expected total number of rewindings is bounded by $O(M)^c$.

The robustness w.r.t. $\mathcal{O}$ property of $\langle C, R\rangle$ follows using essentially the same proof as above: the key step here is, again, to show that the decommitment oracle $\mathcal{O}$ can be emulated efficiently, without "affecting" the security of the left interaction. Now, however, a rewinding is "safe" only if the adversary does not request *any* new message in the left (constant-round) interaction. That is, the the left interaction is never rewound during the extractions on the right. Roughly speaking, this is achieved by rewinding only those $\mathcal{WISSP}$ proofs that do not interleave with any message in the left interaction (and cancelling every rewinding in which the $\mathcal{WISSP}$ proof interleaves with a left-message).

4.3 Proof of Robust CCA-Security

In this section, we formally prove Proposition 6. Below we first provide the definition of safe-points. The formal proof of Proposition 6 consists of two parts: in Section 4.3.2, we show that $\langle C, R\rangle$ is CCA-secure w.r.t $\mathcal{O}$; and in section 4.3.3, we show that it is also robust w.r.t. $\mathcal{O}$.

4.3.1 Safe-Points

Our notion of safe-points is almost the same as that in [LPV08] (which in turn is based on the notion of safe rewinding block of [DDN00]), with the only exception that our definition also considers the Stage 1 and 2 messages of the protocol, whereas the definition in [LPV08] only concerns messages in the $\mathcal{WISSP}$ proofs.

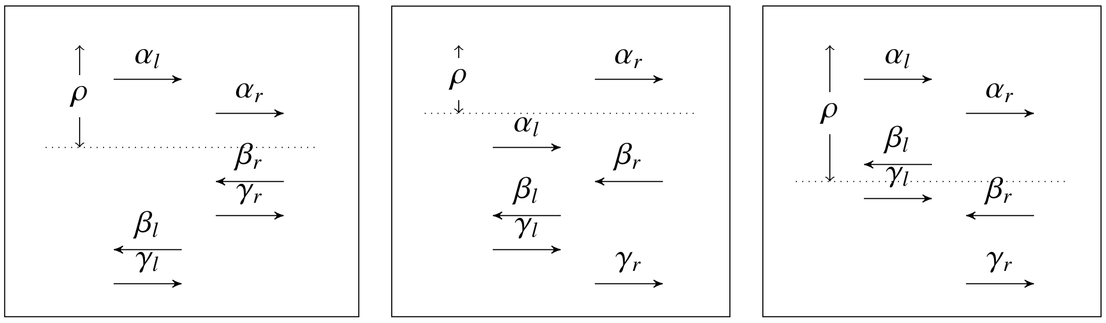


Figure 4.2: Three characteristic safe-points.

Intuitively, a safe-point ρ of a right interaction k is a prefix of a transcript Δ that lies in between the first two messages α_r and β_r of a $\mathcal{WISSP}$ proof $(\alpha_r, \beta_r, \gamma_r)$ in interaction k, such that, when rewinding from ρ to γ_r, if A uses the *same* "scheduling of messages" as in Δ, then the left interaction can be emulated without affecting the hiding property. This holds, if in Δ from ρ to where γ_r is sent, A expects either no message or only complete $\mathcal{WISSP}$ proofs in the left interaction, as shown in Figure 4.2 (i) and (ii) respectively, (Additionally, in both cases, A may request the reply message of some $\mathcal{WISSP}$ proof, as shown in Figure 4.2 (iii). This is because, given the first two messages of a $\mathcal{WISSP}$ proof, the reply message is deterministic, and hence can be emulated in the rewinding by replaying the reply in Δ.)

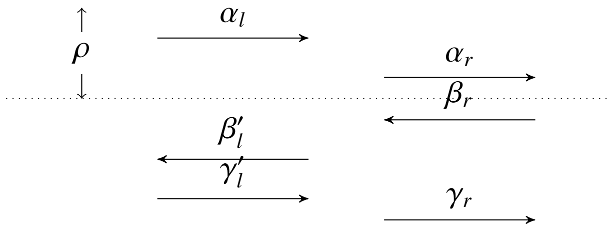


Figure 4.3: Prefix ρ that is not a safe point.

Definition 25. *Let Δ be any transcript of one left interaction, and many right interactions, of $\langle C, R\rangle$. A prefix ρ of a transcript Δ is called a* safe-point *for right interaction k, if there exists an accepting proof $(\alpha_r, \beta_r, \gamma_r)$ in the right interaction k, such that:*

1. *α_r occurs in ρ, but not β_r (and γ_r).*
2. *for any proof $(\alpha_l, \beta_l, \gamma_l)$ in the left interaction, if α_l occurs in ρ, then β_l occurs after γ_r.*
3. *messages in Stage 1 and 2 of the left interaction occur either before ρ or after γ_r.*

If ρ is a safe-point, let $(\alpha_\rho, \beta_\rho, \gamma_\rho)$ denote the canonical "safe" right proof associated with ρ. Note that the only case a right-interaction proof is not associated with any safe-point is if it is "aligned" with a left-execution proof, as shown in Figure 4.3. In contrast, in all other cases, a right-interaction proof has a safe-point, as shown in Figure 4.2. Below we show in Lemma 5 that in any transcript of one left and many right interactions of $\langle C, R\rangle$, every accepting right interaction that has a different identity from the left interaction, has at least $\eta(n)$ safe-points. This technical lemma will be very instrumental in the proof of CCA-security in the next section.

Lemma 5 (Safe-point Lemma)**.** *Let Δ be any transcript of one left interaction, and many right interactions, of $\langle C, R\rangle$. Then, in Δ, for every successful right interaction that has a different identity from the left interaction, there exist at least a number of* $\Omega(\eta(n))$ non-overlapping *$\mathcal{WISSP}$ proofs that are associated with a* safe-point*.*

Proof. Consider any transcript Δ of one left and many right interactions of $\langle C, R\rangle$ with A, and any right interaction k in Δ that is accepting and has a different identity from the left interaction. We show that there are at least $\Omega(\eta(n))$ $\mathcal{WISSP}$ proofs $(\alpha_r, \beta_r, \gamma_r)$ in the right interaction k, for which there exists a point ρ—a prefix of Δ—satisfying the following two properties:

- ρ contains α_r, but not β_r and γ_r.
- For any proof $(\alpha_l, \beta_l, \gamma_l)$ in the left interaction, if α_l occurs in ρ, then β_l occurs after γ_r.

We call ρ a "weak" safe-point for $(\alpha_r, \beta_r, \gamma_r)$. It is easy to see that, if a right interaction has a number of $\mu = \Omega(\eta(n))$ proofs that are associated with a weak safe-point, then at least a number of $\mu/2$ of these proofs are completely sequentially arranged (as all the designs are sequentially arranged), and hence non-overlapping. Furthermore, since there are only a constant number of messages in Stage 1 and 2 of the protocol, at least $\mu/2 - c$ of these weak safe-points are actually full safe-points. Hence, we can conclude that there are $\Omega(\eta(n))$ non-overlapping proofs in this right interaction that are associated with a safe-point.

Now it only remains to show that there are $\Omega(\eta(n))$ proofs with a weak safe-point. Consider the following two adversarial schedulings in Δ with respect to the left and the k^{th} right interaction.

- The adversary "aligns" the proofs in the left and right interactions *one by one*, where a left proof is aligned with a right proof if its challenge message lies in between the challenge and reply messages of the right proof in Δ. (Similarly, we say that two designs are aligned with each other if the two proofs in the left design are aligned with the two in the right design respectively.)
- The adversary does not "align" the proofs in the left and right interactions.

We show that in the first case, there exist $\eta(n)$ weak safe-points. Then we show that, starting from an aligned scheduling, no matter how the adversary changes the message scheduling, the number of weak safe-points never decreases; hence, the claim also holds in the second case.

Assume that Case 1 holds. Since the identities of the left and right interactions, id^l and id^r, are different, they must exist one bit, i, on which they differ (i.e., $\mathsf{id}^l_i \neq \mathsf{id}^r_i$). Then each of the $\eta(n)$ design_1's in the i^{th} iteration (in Stage 3) of the right interaction k is aligned with a design_0 on the left. Next we show that whenever a design_1 is aligned with a design_0 on the left, there exists a weak safe-point. Let $(\alpha^r_b, \beta^r_b, \gamma^r_b)$ for $b = 1, 2$ be the two proofs in the right design_1, and $(\alpha^l_b, \beta^l_b, \gamma^l_b)$ for $b = 1, 2$ the two proofs in the left design_0 that are aligned with them, as shown in Figure 4.4. Then consider the prefix ρ of Δ that includes all the messages up to β_1^l. As β_1^l lies in between β_1^r and γ_1^r, so does ρ. Hence, ρ is associated with the right-proof $(\alpha_2^r, \beta_2^r, \gamma_2^r)$, and every left proof either has its first two messages inside ρ, or after, which means ρ is a weak safe-point. Therefore each of the $\eta(n)$ design_1's in the ith iteration of the right interaction has a weak safe-point.

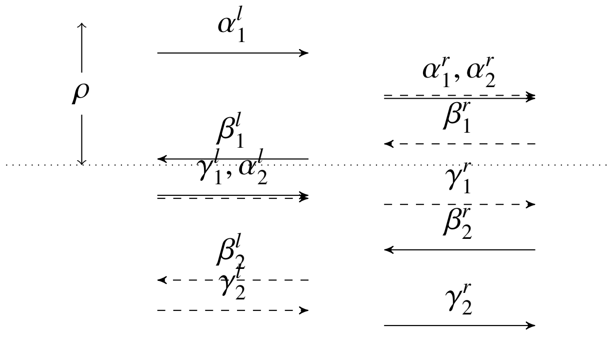


Figure 4.4: A design_0 matches up with design_1.

Next consider a game in which the adversary tries to decrease the number of weak safe-points to below $\eta(n)$, by changing the message-scheduling. By the argument above, we know that if the $\eta(n)$ design_1's in the ith iteration of interaction j are aligned with design_0's, then there are at least $\eta(n)$ weak safe-points, with one for each design_1. Then to succeed, the adversary must manage to eliminate the weak safe-points associated with at least one design_1 (in iteration i). Take any design_1 on the right.

- If one of the two proofs in the design_1 is not aligned with any proof on the left, then there exists a weak safe-point associated with that proof.
- If the two proofs in the design_1 are aligned with the two proofs in a design_0 on the left, then as argued above, there exists a weak safe-point.
- If the two proofs in the design_1 are aligned with two left-proofs belonging to two different designs, then since the two left-proofs are sequentially

arranged as in a design_0, it follows using the same argument as in the second case that there exists a weak safe-point.

Therefore, the only way to ensure that a design_1 is not associated with any weak safe-point is to align it with a design_1 on the left. Then if the adversary wants to eliminate the weak safe-points associated with j design_1's in the i$^{\text{th}}$ iteration on the right, say right proofs $k^r_1 < k^r_2 < \ldots < k^r_j$, it must align them with j design_1's on the left, say designs $k^l_1 < k^l_2 < \ldots < k^l_j$. Since the $\eta(n)$ design_1's in the i'th iterations corresponds to design_0's on the left, it holds that for every k^l_h, either $k^l_h < k^r_1$ or $k^l_h > k^r_j$. There are only three possibilities:

The adversary shifts j left-designs down, i.e., $k^l_j < k^r_1$. Then the first $k^r_1 - 1$ right-designs can only be aligned with the first $k^l_1 - 1$ left-designs. Since $k^l_1 - 1 \leq k^l_j - j \leq (k^r_1 - 1) - j$, there are at least $2j$ proofs on the right (belonging to the first $k^r_l - 1$ right-designs) that are not aligned with any proofs on the left. Then each of them is associated with a weak safe-point.

The adversary shifts j left-designs up, i.e., $k^l_1 > k^r_j$. It follows from the same argument as above that there are at least $2j$ weak safe-points in the last $2\ell(n)\eta(n) - k^r_j$ designs on the right.

The adversary shifts some left-designs down and some up, i.e., $k^l_1 < k^r_1$ and $k^l_j > k^r_j$. Then every right-design k, such that $k < k^r_1$ or $k > k^r_j$, can only be aligned with a left-design k', such that $k' < k^l_1$ or $k' > k^l_j$. Since there are at least $2\ell(n)\eta(n) - \eta(n)$ right-designs with indexes smaller than k^r_1 or greater than k^r_j, but at most $2\ell(n)\eta(n) - \eta(n) - j$ left-designs with indexes smaller than k^l_1 or greater than k^l_j, there are (again) at least $2j$ proofs on the right that are not aligned with any proofs on the left, which gives $2j$ weak safe-points.

Therefore no matter how the adversary changes the message scheduling, there always exist at least $\eta(n)$ weak safe-points in the right interaction. Hence we conclude the Lemma. □

4.3.2 Proof of CCA Security

We show that for every $\mathcal{PPT}$ adversary A, the following ensembles are computationally indistinguishable.

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

Towards this, we consider new commitment scheme $\langle \hat{C}, \hat{R}\rangle$ (similar to the "adaptor" schemes of [DDN00, LPV08]), which is a variant of $\langle C, R\rangle$ where the receiver can ask for an arbitrary number of special-sound $\mathcal{WI}$ designs in Stage 3. Furthermore, $\langle \hat{C}, \hat{R}\rangle$ does not have a fixed scheduling in Stage 3; the receiver instead gets to choose which design to execute in each iteration (by sending bit b to select design_b). Note that, clearly, any execution of $\langle C, R\rangle$ can be emulated by an execution of $\langle \hat{C}, \hat{R}\rangle$ by simply requesting the appropriate designs. It follows using essentially the same proof for the hiding property of $\langle C, R\rangle$ in Proposition 5 that $\langle \hat{C}, \hat{R}\rangle$ is computationally hiding; we omit the proof here.

Now assume for contradiction that there exists an adversary A, a distinguisher D, and a polynomial p, such that for infinitely many $n \in N$, there exists $z \in \{0, 1\}^*$, such that,

$$\left|\Pr\left[D(\mathsf{IND}_0(\langle C, R\rangle, \mathcal{O}, A, n, z)) = 1\right] - \Pr\left[D(\mathsf{IND}_1(\langle C, R\rangle, \mathcal{O}, A, n, z)) = 1\right]\right| \geq \frac{1}{p(n)}$$

We reach a contradiction by exhibiting a (stand-alone) adversary B^* that distinguishes commitments using $\langle \hat{C}, \hat{R} \rangle$. Let $\mathsf{STA}_b(\langle \hat{C}, \hat{R} \rangle, B^*, n, z)$ denote the output of $B^*(1^n, z)$ after receiving a commitment of $\langle \hat{C}, \hat{R} \rangle$ to value v_b, where as in experiment IND_b the challenges v_0 and v_1 are chosen adaptively by B^*. We show that the following two claims hold w.r.t B^*.

Claim 6. *There exists a polynomial function t, such that for every $b \in \{0, 1\}$, $n \in N$ and $z \in \{0, 1\}^*$, B^* in experiment $\mathsf{STA}_b(\langle \hat{C}, \hat{R} \rangle, B^*, n, z)$ takes $t(n)$ steps in expectation.*

Claim 7. *Let $b \in \{0, 1\}$. The following ensembles are computationally indistinguishable.*

- $\left\{\mathsf{STA}_b(\langle \hat{C}, \hat{R} \rangle, B^*, n, z)\right\}_{n \in N, z \in \{0,1\}^*}$
- $\left\{\mathsf{IND}_b(\langle C, R \rangle, \mathcal{O}, A, n, z)\right\}_{n \in N, z \in \{0,1\}^*}$

By Claim 7, it thus follows that for infinitely many $n \in N$, there exists $z \in \{0, 1\}^*$, such that,

$$\left|\Pr\left[D(\mathsf{STA}_0(\langle \hat{C}, \hat{R} \rangle, B^*, n, z)) = 1\right] - \Pr\left[D(\mathsf{STA}_1(\langle \hat{C}, \hat{R} \rangle, B^*, n, z)) = 1\right]\right| \geq \frac{3}{4p(n)}$$

Furthermore, by Claim 6, the probability that B^* runs for more than $T(n) = 4t(n)p(n)$ steps is smaller than $1/4p(n)$. Therefore, the execution of B^* can be truncated after $T(n)$ steps, while only affecting the distinguishing probability by at most $\frac{1}{4p(n)}$, which means there exists a $\mathcal{PPT}$ machine that distinguishes commitments with probability $\frac{1}{2p(n)}$; this contradicts the hiding property of $\langle \hat{C}, \hat{R} \rangle$.

Construction of B^*. On a high-level, B^* in interaction with an honest committer $\hat{C}$ on the left emulates the decommitment oracle $\mathcal{O}$ for A by extracting the decommitments of the right interactions from the $\mathcal{WISSP}$ proofs in $\langle C, R \rangle$. It does this by rewinding A only from safe-points. This ensures that we do not have to rewind the external left execution; rather, it suffices to request an additional design on the left to handle these rewindings. But, as the simulator needs to provide the decommitments in a "on-line" fashion (i.e., as soon as a right-interaction completes, the simulator needs to provide A with decommitment information for this interaction), these rewindings might become recursive (if the right interactions are nested). And, if we were to perform these rewindings naively, the running-time quickly grows exponentially (just as in the context of *concurrent zero knowledge* [DNS04]). To make sure that the recursion depth is constant, we instead only rewind from safe-points ρ such that the number of new right-proofs that start between ρ and the last message γ_ρ of the right-proof associated with ρ, is "small"; here, "small" is defined appropriately based on the recursion level. More precisely, we say that a safe-point ρ is $d + 1$-*good* for a transcript Δ if less than $k_d = M/\eta'^d$ right-proofs start between ρ and γ_ρ, where M is an upper-bound on the total number of messages that A sends or receives, and $\eta' = n^{\varepsilon'}$ for some constant ε' such that $0 < \varepsilon' < \varepsilon$. On recursion level d, B^* then only rewinds A from $d + 1$-good safe-points.

Formally, we describe B^* using a recursive helper procedure EXT. EXT, on input an integer d (the recursion level), a partial joint view $\mathcal{V}$ of A and the (emulated) right receivers, the index s of a right-proof, a "repository" $\mathcal{R}$ of decommitments of the right interactions that have been previously collected, proceeds as follows.

Procedure EXT$(d, \mathcal{V}, s, \mathcal{R})$**:** Let ρ be the (partial) transcript contained in $\mathcal{V}$. If $d = 0$, EXT will emulates a complete execution of IND with the adversary A. If $d > 0$, it will instead extends the partial view $\mathcal{V}$ to the completion of the right-proof s; if at any point in the emulation, ρ is not a $d+1$-good safe-point for s, EXT aborts and returns $\perp$. Finally, EXT returns the the view $\mathcal{V}_A$ of A in the emulation (generated so far). We now turn to describe how EXT emulates the left and the right interactions.

The left interaction is emulated by simply requesting the appropriate messages from the external committer. At the top level (i.e., $d = 0$), A participates in a *complete* $\langle C, R\rangle$ commitment on the left, which can be easily emulated by simply requesting the appropriate designs from $\hat{C}$. At lower levels (i.e., $d > 0$), EXT cancels every execution in which ρ is not a safe-point. Hence it only needs to emulate the left interaction when ρ is a safe-point. In this case, as previously discussed, A either does not request any new messages on the left, or only asks for complete new $\mathcal{WISSP}$ proofs; the former case can be trivially emulated (by simply doing nothing or replaying old messages if A asks for the third message of a left $\mathcal{WISSP}$ proof again), in the latter case, EXT emulates the left interaction by asking for more designs from $\hat{C}$.

On the other hand, in the right interactions EXT follows the honest receiver strategy of $\langle C, R\rangle$. Furthermore, whenever A completes a proof $(\alpha_r, \beta_r, \gamma_r)$ in a right interaction j, EXT attempts to extract a decommitment for this interaction, if the proof $(\alpha_r, \beta_r, \gamma_r)$ is associated with a $d+1$-good safe-point ρ'. To extract, EXT invokes itself recursively on input $(d+1, \mathcal{V}', s', \mathcal{R})$, where $\mathcal{V}'$ is the (partial) joint view of A and the right receivers corresponding to the transcript ρ', and s' is the index of the right-proof $(\alpha_r, \beta_r, \gamma_r)$. It continues invoking itself recursively until one of the recursive invocations returns a view containing another accepting transcript $(\alpha_r, \beta'_r, \gamma'_r)$ of the s'^{th} proof. When this happens, if $\beta_r \neq \beta'_r$, EXTcomputes a witness w by the special-soundness property of the $\mathcal{WISSP}$. Furthermore, if w contains a valid decommitment (v, d) of the right interaction

j (i.e., (v, d) is the value committed to in the second commitment and is a valid decommitment of the first commitment), EXT records (v, d) in the repository $\mathcal{R}$. Later, whenever A expects a decommitment for a right interaction j, it simply checks the repository $\mathcal{R}$ for a matching decommitment; it aborts and outputs fail if no valid decommitment is available—we say that EXT "gets stuck" on interaction j in this case. (If A expects the decommitment of a right interaction that fails or has the same identity as the left, it simply sends $\perp$ to A.)

We now return to the description of B^*. B^* in interaction with $\hat{C}$, simply invokes EXT on inputs $(0, \mathcal{V}, \mathsf{null}, \emptyset)$, where $\mathcal{V}$ is the initial joint states of A and honest right receivers. Once EXT returns a view $\mathcal{V}_A$ of A, B^* return the output of A in this view if A never used the identity of the left interaction in any of the right interactions, and returns $\perp$ otherwise. Furthermore, to simplify our analysis, B^* cuts-off EXT whenever it runs for more than 2^n steps. If this happens, B^* halts and outputs fail.

Proof of Claim 6—Running-time Analysis of B^*. Let $\overline{\mathsf{Bind}}$ denote the event that in an execution between B^* and $\hat{C}$, the adversary A (in simulation by B^*) provides a com commitment that can be decommitted to more than one values. Then we claim the following:

Subclaim 1. *For every $b \in \{0, 1\}$, $n \in N$ and $z \in \{0, 1\}^*$, the probability that event $\overline{\mathsf{Bind}}$ occurs in experiment $\mathsf{STA}_b(\langle \hat{C}, \hat{R} \rangle, B^*, n, z)$ is at most $1/2^n$.*

Proof. It follows from the statistically binding property of com that the probability that A sends a commitment with more than one valid committed value is bounded by $1/2^{2n}$. Since B^* never runs for more than 2^n steps, by union bound, the probability that A sends *any* commitment that can be decommitted to more than one values in the execution of B^* is bounded by $2^n \frac{1}{2^{2n}} = \frac{1}{2^n}$. □

Next we bound the expected running time of B^* by analyzing separately the running time of B^* conditioned on $\overline{\mathsf{Bind}}$ occurring and not. In the first case, by construction, the running time of B^* is bounded by 2^n. Below we show that in the second case the expected running time of B^* is bounded by a polynomial $t'(n)$. Then overall the expected running of B^* is also polynomially bounded and Claim 6 follows.

$$\begin{aligned}
& E[\text{running time of } B^*] \\
= \; & \Pr\left[\overline{\mathsf{Bind}} \text{ occurs}\right] E[\text{running time of } B^* \mid \overline{\mathsf{Bind}} \text{ occurs}] \\
& + \Pr\left[\overline{\mathsf{Bind}} \text{ not occur}\right] E[\text{running time of } B^* \mid \overline{\mathsf{Bind}} \text{ not occur}] \\
\leq \; & \frac{1}{2^n} 2^n + (1 - \frac{1}{2^n}) t'(n) \leq poly(n)
\end{aligned}$$

Now it only remains to show that the expected running time of B^* conditioned on $\overline{\mathsf{Bind}}$ not occurring is bounded by a polynomial, or equivalently, the expected running time of the procedure EXT is bounded by a polynomial (conditioned on $\overline{\mathsf{Bind}}$ not occurring). Below in Subclaim 2 we first show that the recursive depth of EXT is always a constant, and then bound the running time of EXT in Subclaim 3.

Subclaim 2. *There exists a constant D such that for every $n \in N$, and every $\mathcal{V}$, s, and $\mathcal{R}$, $\mathsf{EXT}(D, \mathcal{V}, s, \mathcal{R})$ does not perform any recursive calls.*

Proof. Recall that at recursion level d, the procedure EXT terminates and returns $\perp$ whenever more than $k_d = M(n)/\eta'(n)^d$ new right-proofs has started in its execution, where $M(n)$ is an upper bound on the total number of messages that the adversary A may send and receive, and $\eta'(n)$ equals to $n^{\varepsilon'}$ for some constant $0 < \varepsilon' < \varepsilon < 1$. Let n^c be an upper bound on $M(n)$; set D to $\lceil \log_{\eta'(n)} n^c \rceil$, which is a constant. When $d = D$, $k_D < 1$, which means the execution terminates whenever A starts a new right-proof. On the other hand, EXT only makes a recursive call at the completion of a new right-proof. Therefore at recursion level D, EXT never makes any recursive calls. $\square$

Next, we show that the expected number of queries that EXT makes to A at every recursion level $d \leq D$ is always bounded by a polynomial, conditioned on the event that A never sends a com commitment that can be decommitted to two different values.

Subclaim 3. *For every $0 \leq d \leq D$, it holds that for every $n \in N$, $\mathcal{V}$, s, and $\mathcal{R}$, the expected number of queries that* EXT$(d, \mathcal{V}, s, \mathcal{R})$ *makes to A is bounded by $\theta(d) = M(n)^{3(D-d+1)}$, conditioned on that every* com *commitment that A sends has a unique committed value in* EXT*.*

Proof. Consider some fixed $\mathcal{V}$, s and $\mathcal{R}$. We prove the subclaim by induction on d. When $d = D$, the claim follows, since EXT does not perform any recursive calls and the number of queries made by EXT can be at most the total number of messages, which is $M = M(n)$.

Assume the claim is true for $d = d' + 1$. We show that it holds also for $d = d'$. The procedure EXT simulates an execution with A in a straight-line on recursion level d', until it encounters the completion of a right-proof s that has a $d' + 1$-good safe-point ρ, then it tries to extract a witness of s, by repeatedly invoking EXT on recursion level $d' + 1$ from (the partial transcript) ρ. Hence, the number of queries made by EXT is bounded by the sum of the number of queries made on level d', and the queries made by the recursive calls: the former is at most the total number of messages, that is, M, while the latter is bounded by the sum of the queries made by those recursive calls invoked for every right-proof s. Furthermore we compute the expected number of queries made by the recursive calls for a right-proof s by taking expectation over all partial transcript that is potentially a d'-good safe-point for s. let Γ_i denote the set of all partial transcripts of length i that are consistent with $\mathcal{V}$; for every $\rho \in \Gamma_i$, we denote by $\Pr[\rho$ occurs on level $d']$ the probability that ρ occurs (in the simulation) on level d', and $E[Q^s_{d'}(\rho)|\rho]$ the expected number of queries made by the recursive calls started from ρ for the right-proof s, conditioned on ρ occurring on level d'. Then

$$E[\text{number of queries by EXT}] = M + \sum_s \sum_i \sum_{\rho \in \Gamma_i} Pr[\rho \text{ occurs on level } d']\, E[Q^s_{d'}(\rho)|\rho]$$

Next we bound $E[Q^s_{d'}(\rho)|\rho]$ in two steps: the first step bounds the expected number of recursive calls started from ρ for proof s, and the second step uses the induction hypothesis to derive a bound on $E[Q^s_{d'}(\rho)|\rho]$.

Step 1: Given a partial transcript ρ from Γ_i, let $p^s_{d'}(\rho)$ denote the probability that conditioned on ρ occurring on level d', EXT starts recursive calls from ρ for the right-proof s, which happens if and only if EXT does not output fail (before the proof s completes), *and* ρ is a $d'+1$-good safe-point for proof s, that is,

$$p^s_{d'}(\rho) = \Pr\left[\text{EXT at level } d' \text{ does not output fail} \wedge \rho \text{ is } d'+1\text{-good at level } d' \mid \rho\right]$$

When this happens, EXT repeatedly calls itself on recursion level $d'+1$, until an invocation succeeds without cancelling, or it outputs fail. Let $q^s_{d'}(\rho)$ denote the probability that conditioned on ρ occurring on level d', a recursive call to EXT on level $d'+1$ succeeds without cancelling or outputs fail. Since an invocation is cancelled if and only if ρ fails to be a $d'+1$-good safe-point for s in the invocation on level $d'+1$, we have

$$q^s_{d'}(\rho) = \Pr\left[\mathsf{EXT}\text{ at level } d'+1 \text{ outputs } \mathsf{fail} \lor \rho \text{ is } d'+1\text{-good at level } d'+1 \mid \rho\right]$$

We claim that $q^s_{d'}(\rho) \geq p^s_{d'}(\rho)$. This is because,

$$\begin{aligned}
q^s_{d'}(\rho) &= \Pr\left[\mathsf{EXT}\text{ at level } d'+1 \text{ outputs } \mathsf{fail} \mid \rho\right] + \\
&\quad \Pr\left[\rho \text{ is } (d'+1)\text{-good at level } d'+1 \mid \mathsf{EXT}\text{ at level } d'+1 \text{ not outputs } \mathsf{fail} \land \rho\right] \\
&\geq \Pr\left[\rho \text{ is } (d'+1)\text{-good at level } d'+1 \mid \mathsf{EXT}\text{ at level } d'+1 \text{ not outputs } \mathsf{fail} \land \rho\right] \\
&= \Pr\left[\rho \text{ is } (d'+1)\text{-good at level } d' \mid \mathsf{EXT}\text{ at level } d' \text{ not outputs } \mathsf{fail} \land \rho\right] \\
&\geq \Pr\left[\rho \text{ is } (d'+1)\text{-good at level } d' \land \mathsf{EXT}\text{ at level } d' \text{ not outputs } \mathsf{fail} \mid \rho\right] \\
&= p^s_{d'}(\rho)
\end{aligned}$$

The second last equality in the above derivation holds because conditioned on ρ occurring and that EXT does not output fail, the view of A on levels d' and $d'+1$ are simulated identically: on both levels d' and $d'+1$, EXT emulates messages in the commitments of $\langle C, R\rangle$ for A perfectly; and furthermore, whenever A expects a decommitment of a right interaction, EXT sends it the value committed to in the second com commitment obtained through recursive calls; conditioned on EXT not outputting fail and $\overline{\mathsf{Bind}}$ not occurring, A always receives the same value on both level d' and $d'+1$.

Then conditioned on ρ occurring on level d', the expected number of recursive invocations to level $d'+1$ before encountering a successful one or aborting is $1/q^s_{d'}(\rho)$. Since EXT only starts recursive invocations from ρ with probability $p^s_{d'}(\rho)$, we have that the expected number of recursive calls from ρ for proof s, conditioned on ρ occurring on level d', is at most $p^s_{d'}(\rho)/q^s_{d'}(\rho) \leq 1$.

Step 2: From the induction hypothesis, we know that the expected number of queries made by an invocation of EXT on level $d' + 1$ is at most $\theta(d' + 1)$. Therefore, if u recursive invocations are made from ρ for a right proof s, the expected number of queries made is bounded by $u\theta(d' + 1)$. Then we bound $E[Q^s_{d'}(\rho)|\rho]$ as follow:

$$\begin{aligned} E[Q^s_{d'}(\rho)|\rho] &\leq \sum_{u \in N} \Pr[u \text{ recursive calls are made from } \rho \text{ for } s]\ u\,\theta(d' + 1) \\ &= \theta(d' + 1) \sum_{u \in N} \Pr[u \text{ recursive calls are made from } \rho \text{ for } s]\ u \\ &\leq \theta(d' + 1) \end{aligned}$$

Therefore,

$$\begin{aligned} & E[\text{number of queries by } \mathsf{EXT}] \\ \leq\ & M + \sum_s \sum_i \sum_{\rho \in \Gamma_i} \Pr[\rho \text{ occurs on level } d']\ \theta(d' + 1) \\ =\ & M + \theta(d' + 1) \sum_s \sum_i \sum_{\rho \in \Gamma_i} \Pr[\rho \text{ occurs on level } d'] \\ =\ & M + \theta(d' + 1)M^2 \\ \leq\ & M^{3(D - d' + 1)} = \theta(d') \end{aligned}$$

□

Combining Subclaim 2 and 3, we conclude that conditioned on $\overline{\mathsf{Bind}}$ not occurring (that is, A never sending a com commitment that can be decommitted to different values), the expected running time of machine B^* is bounded by a polynomial $t'(n)$. This concludes Claim 6.

Proof of Claim 7—Correctness of the Output distribution of B^*. We proceed to show that the output distribution of B^* is correct. Claim 7 follows from the following two subclaims:

Subclaim 4. *For every $b \in \{0,1\}$, $n \in N$ and $z \in \{0,1\}^*$, it holds that the view of A in simulation by B^* in experiment $\mathsf{STA}_b(\langle \hat{C}, \hat{R} \rangle, B^*, n, z)$ is identical to that in experiment $\mathsf{IND}_b(\langle C, R \rangle, O^*, A, n, z)$, conditioned on that A never sends a commitment of com that can be decommitted to two different values* and *that B^* does not output* fail *in $\mathsf{STA}_b(\langle \hat{C}, \hat{R} \rangle, B^*, n, z)$.*

Subclaim 5. *For every $b \in \{0,1\}$, $n \in N$ and $z \in \{0,1\}^*$, the probability that A sends a commitment of com that can be decommitted to two different values* or *that B^* outputs* fail *in $\mathsf{STA}_b(\langle \hat{C}, \hat{R} \rangle, B^*, n, z)$ is negligible.*

Proof of Subclaim 4. First note that, in $\mathsf{STA}_b(\langle \hat{C}, \hat{R} \rangle, B^*, n, z)$, B^* returns the output of A contained in the simulated view $\mathcal{V}_A$ returned by EXT (at the top recursion level $d = 0$). As in $\mathsf{IND}_b(\langle C, R \rangle, O^*, A, n, z)$, the output is replaced with $\perp$ if A copies the identity of the left interaction in any right interaction. Hence it suffices to show that in the case where A does not copy the identity of the left interaction, EXT simulates the messages in the left and right interactions for A perfectly. By construction of EXT, all the messages belonging to the commitments of $\langle C, R \rangle$ (both on the left and right) are simulated perfectly. Furthermore, conditioned on that B^* does not output fail, whenever A expects a decommitment of a right interaction, B^* must have previously extracted a valid decommitment of that right interaction (otherwise, B^* would abort and output fail). Finally, conditioned on that every commitment of com sent by A has a unique decommitment, the decommitment extracted by B^* must be the same as that provided by the oracle O in experiment IND. Therefore the simulated view of A in B^* is identical to that in IND. □

Proof of Subclaim 5. Consider a fixed $b \in \{0,1\}$. Since B^* runs for at most 2^n steps, it follows directly from the statistically binding property of com that the probability that A sends a commitment of com that can be decommitted to two different values is negligible. (The argument is exactly the same as that in the proof of Claim 6.) Therefore, it suffices to bound the probability that B^* outputs fail. By construction, B^* outputs fail if it runs for more than 2^n steps, or "gets stuck" on a right interaction j that is accepting and has a different identity from the left interaction. By Claim 6, the probability that B^* runs for more than 2^n steps is negligible. Thus the former case happens with only negligible probability. On the other hand, the latter happens only if one of the following three cases occurs.

Case 1: None of the $\mathcal{WISSP}$ proofs in the right interaction is rewound.

Case 2: Some proof is rewound but the recursive calls invoked for this proof generates the same proof transcript as in the original execution of the interaction.

Case 3: A witness is extracted from one of the proofs in the interaction, but it is not a valid decommitment of the right interaction.

Below we analyze the probabilities that each of the above cases occurs. We show that all these events occur with only negligible probability. Therefore, overall the probability that B^* outputs fail is negligible.

Analysis of Case 1: We show that Case 1 never happens. More precisely, for every accepting right interaction j with a different identity from the left interaction, one of its proofs must be rewound. By Lemma 5, there exist a number of $\Omega(\eta(n))$ non-overlapping proofs in the right interaction j that has a safe-point. Recall that in B^* (more precisely, in EXT), a right interaction may be carried out at multiple different recursion levels (through recursive calls); and at level d, B^* rewinds every proof in this interaction that has a $d+1$-good safe-point. By Subclaim 2, the recursion depth is only a constant; hence there must be a level d, on which a number of $\Omega(\eta(n))$ non-overlapping proofs with a safe-point start in interaction j. Since the total number of right-proofs that start on level d is bounded by $k_d = M/\eta'(n)^d$ (otherwise, the simulation is cancelled) and $\eta'(n) = o(\eta(n))$, there must exist one right-proof that has a safe-point ρ, such that there are less than $M/\eta'(n)^{d+1}$ right-proofs starting in between ρ and the last message of the proof. Therefore ρ is a $d+1$-good safe-point for this right-proof, and will be rewound.

Analysis of Case 2: We bound the probability that any challenge message is picked twice in whole execution of B^* to be negligible. Since B^* runs for at most 2^n steps, it picks at most 2^n challenges during the whole execution. Furthermore, the length of each challenge is $3n$. By applying the union bound, we obtain that, the probability that a challenge β is picked again is at most $\frac{2^n}{2^{3n}}$, and hence, using the union bound again, the probability that *any* challenge in the execution is picked twice is at most $2^n \frac{2^n}{2^{3n}}$. Hence, overall, the probability that this case occurs is negligible.

Analysis of Case 3: Suppose for contradiction that, there exists a polynomial g, such that for infinitely many $n \in N$ and z, case 3 occurs with probability at least $1/g(n)$ during the execution of B^*. By Claim 6, the probability that B^* runs for more than $g'(n) = 2t(n)g(n)$ steps is at most $1/2g(n)$. Therefore, the probability that B^* extracts out a witness w in some right interaction j that is not a valid decommitment of that interaction *in the first* $g'(n)$ *steps* is at least $1/2g(n)$. It follows from the special-soundness property that the witness w extracted must be a value r such that $f(r) = s$, where s is the first message in the right interaction j. Then we show that we can invert the one-way function f. More precisely, given A, n and z, we construct A^* that inverts f. A^* on input $y = f(r')$, emulates an execution of $A(z)$ internally exactly as B^* does, except that (1) it "cuts" the execution off after $g'(n)$ steps, and (2) it picks a random right interaction started in the first $g'(n)$ steps, and feeds y as the Stage 1 message in that interaction; finally, A^* outputs all the witnesses extracted from this interaction. Since A^* proceeds identically as B^* in the first $g'(n)$ steps, the probability that it inverts f in some right interaction is exactly the same as the probability that B^* does in the first $g'(n)$ steps, which is at least $1/2g(n)$. Furthermore, with probability at least $1/g'(n)$, A^* guesses correctly the right interaction in which this happens, and thus it inverts y with probability at least $1/2g(n)g'(n)$.

□

4.3.3 Proof of Robustness

In this section, we extend the proof in the last section to show that $\langle C, R\rangle$ is also *robust* CCA-secure w.r.t. O. Towards this, we need to show (in addition to that $\langle C, R\rangle$ is CCA-secure w.r.t. O) that for every constant k, and every $\mathcal{PPT}$ adversary A, there exists a simulator S, such that, for every $\mathcal{PPT}$ k-round ITM B, the interaction between B and A with access to O is indistinguishable from that between B and S.

Given an adversary A, and a constant k, the construction of the simulator S is very similar to that of B^* in the last section. On a high-level, S externally interacts with an arbitrary k-round ITM B, and internally simulates an execution between B and A^O, by forwarding messages from B internally to A, while concurrently extracting the decommitments of the right interactions from A to simulate $\mathcal{O}$. The extraction strategy of S is essentially the same as that used by B^*: it recursively rewinds A over the $\mathcal{WISSP}$ proofs in Stage 3 of the protocol to extract the decommitments, except that, here the goal is to make sure that the left interaction with B is *never* rewound, (instead of the goal of ensuring that the left interaction remains hiding (in B^*)). This is achieved by rewinding only those $\mathcal{WISSP}$ proofs that do not interleave with any messages in the left interaction, and cancelling every rewinding in which the $\mathcal{WISSP}$ proof interleaves with a left-message. More precisely, consider the notion of R-safe-point (which is in analogous to the notion of safe-point)—a prefix ρ of a transcript Δ is a R-safe-point for a right-proof (α, β, γ) if it includes all the messages in Δ up to α (inclusive), and that no left-message is exchanged in between ρ and γ. Then S simply runs the procedure EXTdefined in the last section internally, except that it replaces the notion of safe-point with R-safe-point, and that it simulates the left interaction with A by forwarding the messages between A and B; everything else remains the same. Then it follows from the fact that S always rewinds A from a R-safe-point ρ, and cancels every rewindings in which ρ is not a R-safe-point, the left interaction is never rewound. Furthermore, since the left interaction with B consists of only k rounds, there exist $\Omega(n^{\varepsilon})$ R-safe-point in every successful right interaction. Then, it follows from the same proof as in Claim 6 and Claim 7 that, except from negligible probability, S runs in expected polynomial time, and that the joint output of S and B is indistinguishable from that of $A^{\mathcal{O}}$ and B.

Remark 2. *The protocol $\langle C, R \rangle$ described in Section 4.2 uses a one-way function with efficiently recognizable range in its first stage. It can be modified to work with any arbitrary one-way function f as follows: in Stage 1, the receiver sends the images of* two *secrets, i.e.,* $s_1 = f(r_1)$ *and* $s_2 = f(r_2)$*, followed by a proof that either* s_1 *or* s_2 *is in the range of* f*, using a resettable* $\mathcal{WI}$ *proof system [CGGM00] (the committer verifies the proof and aborts if it is not convincing); and in Stage 3 the committer proves that either it has committed to* v *honestly, or that one of* s_1 *and* s_2 *is in the range of* f*. It follows using almost the same proofs as above that the modified protocol is a robust CCA-secure commitment scheme, except that it relies on the one-wayness of* f *and, additionally, the resettable* $\mathcal{WI}$ *property of the proof in Stage 1 that, the value extracted from* A *is the desired decommitment, despite that* A *is rewound during the extraction. (See Case 3 in the proof of Claim 5.)*

Remark 3. *We further modify the protocol to work with 4-round special-sound proofs instead of 3-round special-sound proofs: in Stage 1, the receiver sends, in addition to the images of two secrets, the first message* r *of a 4-round special-sound proof; then in Stage 3, the committer and the receiver simply use the last three messages of a 4-round special-sound proof with the first message fixed to* r*, as a 3-round special-sound proof. It follows from the same proof as described above that the modified protocol is robust CCA secure, as both the* $\mathcal{WI}$ *and the special-soundness properties hold even if the special-sound proofs share the same first message.*

4.4 UC and Global UC security

We briefly review UC and externalized UC (EUC) security. For full details see [Can00, CDPW07]. The original motivation to define EUC security was to capture settings where all ITMs in the system have access to some global, potentially trusted information (such as a globally available public key infrastructure or a bulletin board) [CDPW07]. Here however we use the EUC formalism to capture the notion of global helper functionalities that are available only to the corrupted parties.

We first review the model of computation, ideal protocols, and the general definition of securely realizing an ideal functionality. Next we present hybrid protocols and the composition theorem.

The basic model of execution. Following [GMR89, Gol01], a protocol is represented as an interactive Turing machine (ITM), which represents the program to be run within each participant. Specifically, an ITM has three tapes that can be written to by other ITMs: the input and subroutine output tapes model the inputs from and the outputs to other programs running within the same "entity" (say, the same physical computer), and the incoming communication tapes and outgoing communication tapes model messages received from and to be sent to the network. It also has an identity tape that cannot be written to by the ITM itself. The identity tape contains the program of the ITM (in some standard encoding) plus additional identifying information specified below. Adversarial entities are also modeled as ITMs.

We distinguish between ITMs (which represent static objects, or programs) and *instances of ITMs*, or *ITI*s, that represent interacting processes in a running system. Specifically, an ITI is an ITM along with an identifier that distinguishes it from other ITIs in the same system. The identifier consists of two parts: A session-identifier (SID) which identifies which protocol instance the ITI belongs to, and a party identifier (PID) that distinguishes among the parties in a protocol instance. Typically the PID is also used to associate ITIs with "parties", or clusters, that represent some administrative domains or physical computers.

The model of computation consists of a number of ITIs that can write on each others tapes in certain ways (specified in the model). The pair (SID,PID) is a unique identifier of the ITI in the system. With one exception (discussed within) we assume that all ITMs are probabilistic polynomial time.[6]

Security of protocols. Protocols that securely carry out a given task (or, protocol problem) are defined in three steps, as follows. First, the process of executing a protocol in an adversarial environment is formalized. Next, an "ideal process" for carrying out the task at hand is formalized. In the ideal process the parties do not communicate with each other. Instead they have access to an "ideal functionality," which is essentially an incorruptible "trusted party" that is programmed to capture the desired functionality of the task at hand. A protocol is said to securely realize an ideal functionality if the process of running the protocol amounts to "emulating" the ideal process for that ideal functionality. Below we overview the model of protocol execution (called the *real-life model*), the ideal process, and the notion of protocol emulation.

The model for protocol execution. The model of computation consists of the parties running an instance of a protocol π, an adversary A that controls the communication among the parties, and an environment Z that controls the inputs to the parties and sees their outputs. We assume that all parties have a security parameter $k \in N$. (We remark that this is done merely for convenience and is not essential for the model to make sense). The execution consists of a sequence of *activations*, where in each activation a single participant (either Z, A, or some

[6]An ITM is $\mathcal{PPT}$ if there exists a constant $c > 0$ such that, at any point during its run, the overall number of steps taken by M is at most n^c, where n is the overall number of bits written on the *input tape* of M in this run. In fact, in order to guarantee that the overall protocol execution process is bounded by a polynomial, we define n as the total number of bits written to the input tape of M, *minus the overall number of bits written by M to input tapes of other ITMs*; see [Can01].

other ITM) is activated, and may write on a tape of at most *one* other participant, subject to the rules below. Once the activation of a participant is complete (i.e., once it enters a special waiting state), the participant whose tape was written on is activated next. (If no such party exists then the environment is activated next.)

The environment is given an external input z and is the first to be activated. In its first activation, the environment invokes the adversary A, providing it with some arbitrary input. In the context of UC security, the environment can from now on invoke (namely, provide input to) only ITMs that consist of a single instance of protocol π. That is, all the ITMs invoked by the environment must have the same SID and the code of π. In the context of EUC security the environment can in addition invoke an additional ITI that interacts with all parties. We call this ITI the helper functionality, denoted $\mathcal{H}$.

Once the adversary is activated, it may read its own tapes and the outgoing communication tapes of all parties. It may either deliver a message to some party by writing this message on the party's incoming communication tape or report information to Z by writing this information on the subroutine output tape of Z. For simplicity of exposition, in the rest of this paper we assume authenticated communication; that is, the adversary may deliver only messages that were actually sent. (This is however not essential since authentication can be realized via a protocol, given standard authentication infrastructure [Can04].)

Once a protocol party (i.e., an ITI running π) is activated, either due to an input given by the environment or due to a message delivered by the adversary, it follows its code and possibly writes a local output on the subroutine output tape of the environment, or an outgoing message on the adversary's incoming communication tape.

The protocol execution ends when the environment halts. The output of the protocol execution is the output of the environment. Without loss of generality we assume that this output consists of only a single bit.

Let $\text{EXEC}_{\pi,A,Z}(k,z,r)$ denote the output of the environment Z when interacting with parties running protocol π on security parameter k, input z and random input $r = r_Z, r_A, r_1, r_2, \ldots$ as described above (z and r_Z for Z; r_A for A, r_i for party P_i). Let $\text{EXEC}_{\pi,A,Z}(k,z)$ denote the random variable describing $\text{EXEC}_{\pi,A,Z}(k,z,r)$ when r is uniformly chosen. Let $\text{EXEC}_{\pi,A,Z}$ denote the ensemble $\{\text{EXEC}_{\pi,A,Z}(k,z)\}_{k\in N, z\in\{0,1\}^*}$.

Ideal functionalities and ideal protocols. Security of protocols is defined via comparing the protocol execution to an *ideal protocol* for carrying out the task at hand. A key ingredient in the ideal protocol is the *ideal functionality* that captures the desired functionality, or the specification, of that task. The ideal functionality is modeled as another ITM (representing a "trusted party") that interacts with the parties and the adversary. More specifically, in the ideal protocol for functionality $\mathcal{F}$ all parties simply hand their inputs to an ITI running $\mathcal{F}$. (We will simply call this ITI $\mathcal{F}$. The SID of $\mathcal{F}$ is the same as the SID of the ITIs running the ideal protocol. (the PID of $\mathcal{F}$ is null.)) In addition, $\mathcal{F}$ can interact with the adversary according to its code. Whenever $\mathcal{F}$ outputs a value to a party, the party immediately copies this value to its own output tape. We call the parties in the ideal protocol dummy parties. Let $\pi(\mathcal{F})$ denote the ideal protocol for functionality $\mathcal{F}$.

Securely realizing an ideal functionality. We say that a protocol π *emulates* protocol ϕ if for any adversary A there exists an adversary $\mathcal{S}$ such that no environment Z, on any input, can tell with non-negligible probability whether it is interacting with A and parties running π, or it is interacting with $\mathcal{S}$ and parties running ϕ. This means that, from the point of view of the environment, running protocol π is 'just as good' as interacting with ϕ. We say that π *securely realizes* an ideal functionality $\mathcal{F}$ if it emulates the ideal protocol $\pi(\mathcal{F})$. More precise definitions follow. A distribution ensemble is called *binary* if it consists of distributions over $\{0, 1\}$.

Definition 26. *Let π and ϕ be protocols. We say that π* UC-emulates *(resp.,* EUC-emulates*) ϕ if for any adversary A there exists an adversary $\mathcal{S}$ such that for any environment Z that obeys the rules of interaction for UC (resp., EUC) security we have* $\text{EXEC}_{\phi,\mathcal{S},Z} \approx \text{EXEC}_{\pi,A,Z}$.

Definition 27. *Let $\mathcal{F}$ be an ideal functionality and let π be a protocol. We say that π* UC-realizes *(resp.,* EUC-realizes*) $\mathcal{F}$ if π* UC-emulates *(resp.,* EUC-emulates*) the ideal protocol $\pi(\mathcal{F})$.*

Security with dummy adversaries. Consider the adversary $\mathcal{D}$ that simply follows the instructions of the environment. That is, any message coming from one of the ITIs running the protocol is forwarded to the environment, and any input coming from the environment is interpreted as a message to be delivered to the ITI specified in the input. We call this adversary the dummy adversary. A convenient lemma is that UC security with respect to the dummy adversary is equivalent to standard UC security. That is:

Definition 28. *Let π and ϕ be protocols. We say that π* UC-emulates *(resp.,* EUC-emulates*) ϕ* w.r.t the dummy adversary *$\mathcal{D}$ if there exists an adversary $\mathcal{S}$ such that* *for any environment Z that obeys the rules of interaction for UC (resp., EUC) security we have* $\mathrm{EXEC}_{\phi,\mathcal{S},Z} \approx \mathrm{EXEC}_{\pi,\mathcal{D},Z}$.

Theorem 8. *Let π and ϕ be protocols. Then π* UC-emulates *(resp.,* EUC-emulates*) ϕ if and only if π* UC-emulates *(resp.,* EUC-emulates*) ϕ* with respect to the dummy adversary.

Hybrid protocols. Hybrid protocols are protocols where, in addition to communicating as usual as in the standard model of execution, the parties also have access to (multiple copies of) an ideal functionality. Hybrid protocols represent protocols that use idealizations of underlying primitives, or alternatively make *trust assumptions* on the underlying network. They are also instrumental in stating the universal composition theorem. Specifically, in an $\mathcal{F}$-hybrid protocol (i.e., in a hybrid protocol with access to an ideal functionality $\mathcal{F}$), the parties may give inputs to and receive outputs from an unbounded number of copies of $\mathcal{F}$.

The communication between the parties and each one of the copies of $\mathcal{F}$ mimics the ideal process. That is, giving input to a copy of $\mathcal{F}$ is done by writing the input value on the input tape of that copy. Similarly, each copy of $\mathcal{F}$ writes the output values to the subroutine output tape of the corresponding party. It is stressed that the adversary does not see the interaction between the copies of $\mathcal{F}$ and the honest parties.

The copies of $\mathcal{F}$ are differentiated using their SIDs. All inputs to each copy and all outputs from each copy carry the corresponding SID. The model does not specify how the SIDs are generated, nor does it specify how parties "agree" on the SID of a certain protocol copy that is to be run by them. These tasks are left to the protocol. This convention seems to simplify formulating ideal functionalities, and designing protocols that securely realize them, by freeing the functionality from the need to choose the SIDs and guarantee their uniqueness. In addition, it seems to reflect common practice of protocol design in existing networks.

The definition of a protocol securely realizing an ideal functionality is extended to hybrid protocols in the natural way.

The universal composition operation. We define the universal composition operation and state the universal composition theorem. Let ρ be an $\mathcal{F}$-hybrid protocol, and let π be a protocol that securely realizes $\mathcal{F}$. The composed protocol ρ^{π} is constructed by modifying the code of each ITM in ρ so that the first message sent to each copy of $\mathcal{F}$ is replaced with an invocation of a new copy of π with fresh random input, with the same SID, and with the contents of that message as input. Each subsequent message to that copy of $\mathcal{F}$ is replaced with an activation of the corresponding copy of π, with the contents of that message given to π as new input. Each output value generated by a copy of π is treated as a message received from the corresponding copy of $\mathcal{F}$. The copy of π will start sending and receiving messages as specified in its code. Notice that if π is a $\mathcal{G}$-hybrid protocol (i.e., ρ uses ideal evaluation calls to some functionality $\mathcal{G}$) then so is ρ^{π}.

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 A there exists a simulator $\mathcal{S}$ such that no environment machine Z can tell with non-negligible probability whether it is interacting with 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.[7]

Theorem 9 (Universal Composition [Can01, CDPW07])**.** *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-emulates ρ (resp., EUC-emulates ρ).*

An immediate corollary of this theorem is that if the protocol ρ UC-realizes (resp., EUC-realizes) some functionality $\mathcal{G}$, then so does ρ^π.

4.5 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'll 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 Z is called *aided by $\mathcal{H}$* if: (a) Z invokes

[7]The universal composition theorem in [Can01] applies only to "subroutine respecting protocols", namely protocols that do not share subroutines with any other protocol in the system. In [CDPW07] the theorem is extended to protocols that share subroutines with arbitrary other protocols, as long as the composed protocol, ρ^π, realizes $\mathcal{F}$ with EUC security.

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), Z lets $\mathcal{H}$ know of this fact; (c) $\mathcal{H}$ interacts only with the corrupted parties. Then:

Definition 29. *Let π and ϕ be protocols, and let $\mathcal{H}$ be a helper functionality (i.e., an ITM). We say that π $\mathcal{H}$-EUC-emulates ϕ if for any adversary A there exists an adversary $\mathcal{S}$ such that for any environment Z that's aided by $\mathcal{H}$ we have $\text{EXEC}_{\phi,\mathcal{S},Z} \approx \text{EXEC}_{\pi,A,Z}$.*

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 $\mathcal{H}$-EUC-emulates protocol ϕ where $\mathcal{H}$ obeys the above rules and runs in time $T(n)$, then π UC-emulates ϕ according to a relaxed notion where the adversary $\mathcal{S}$ can run in time $poly(T(n))$. As noted in the past, for many protocols and ideal functionalities, this relaxed notion of security suffices even when $T(n) = exp(n)$ [Pas03b, PS04, BS05, MMY06].

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

Theorem (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 $\mathcal{H}$-EUC-realizes $\mathcal{F}$. Then protocol π^ρ $\mathcal{H}$-EUC-emulates π.*

Proof. The proof of Theorem 4.5 follows the same steps as the proof of Theorem 9 (see e.g. the proof in [Can00]). The only difference is in the construction of the distinguishing environment Z_π (see there). Recall that Z_π takes an environment Z that distinguishes between an execution of π and an execution of π^ρ, and uses it to distinguish between an execution of ρ and an ideal evaluation of $\mathcal{F}$. For this purpose, Z_π emulates for Z an execution of π^ρ.

Now, in the presence of the helper $\mathcal{H}$, Z_ρ must emulate for Z also the interaction with $\mathcal{H}$. Note that Z_π cannot run $\mathcal{H}$ on its own, since $\mathcal{H}$ may well be super-polynomial in complexity. Instead, Z_π will forward to the external instance of $\mathcal{H}$ each message sent to $\mathcal{H}$ by Z. Similarly, whenever any of the corrupted parties that Z_π locally runs sends a message to $\mathcal{H}$, Z_π externally invokes a party with the same ID and code, corrupts it, and instructs it to send the query to the external instance of $\mathcal{H}$. The responses of $\mathcal{H}$ are handled analogously.

Note that the proof uses the fact that the helper functionality $\mathcal{H}$ does not take messages directly from the adversary. Indeed, Z_π cannot emulate for the external instance of $\mathcal{H}$ messages coming from the adversary. $\square$

4.6 UC with Super-Polynomial Helpers from CCA-Security

We here show how to realize any functionality by relying on our construction of CCA-secure commitments. Let $\langle C, R\rangle$ be a commitment scheme that is robust CCA-secure w.r.t. a decommitment oracle O. Furthermore, assume that O can be computed in sub-exponential time; note that this property can be ensured by using our construction of CCA secure commitments with a "scaled-down" security parameter. Consider a helper functionality $\mathcal{H}$ that "breaks" commitments of $\langle C, R\rangle$ in the same way as O does, subject to the condition that player P_i in a protocol instance *sid* can only query the functionality on commitments that uses identity (P_i, sid). More precisely, every party P_i in a secure computation can simultaneously engage with $\mathcal{H}$ in multiple sessions of the commit phase of

$\langle C, R\rangle$ as a committer using identity P_i, where the functionality simply forwards all the messages internally to the decommitment oracle $\mathcal{O}$, and forwards P_i the decommitment pair returned from $\mathcal{O}$ at the end of each session. See Figure 4.5 for a formal description of the functionality. Clearly this functionality can also be implemented in sub-exponential time.

Functionality $\mathcal{H}$

Corrupted Parties: Upon receiving an input (`Corrupt`, P_i, sid) from the environment, record (`Corrupt`, P_i, sid).

Initialization: Upon receiving an input (`Init`,P_i, sid, k) from party P_i in the protocol instance sid, if there is no previously recorded tuple (`Corrupt`, P_i, sid) or there is a previously recorded session (P_i, sid, k), ignore this message; otherwise, initialize a session of $\langle C, R\rangle$ with $\mathcal{O}$ using identity (P_i, sid), and record session (P_i, sid, k).

Accessing $\mathcal{O}$: Upon receiving an input (`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 (`Mesg`,P_i, sid, k, m') to P_i.

Figure 4.5: The ideal functionality $\mathcal{H}$

We have now the following theorem:

Theorem 10. *Let ε be any positive constant. Assume the existence of enhanced trapdoor permutations. Then for every well-formed functionality*[8] *$\mathcal{F}$, there exists a $O(n^{\varepsilon})$-round protocol Π that $\mathcal{H}$-EUC-emulates $\mathcal{F}$.*

Towards proving the theorem, we first show how to implement the *ideal commitment functionality $\mathcal{F}_{\mathsf{com}}$* in the $\mathcal{H}$-EUC-model, and then show how to realize any functionality using $\mathcal{F}_{\mathsf{com}}$.

[8]See [CLOS02] for a definition of well-formed functionalities.

Realizing $\mathcal{F}_{\text{com}}$: The ideal commitment functionality $\mathcal{F}_{\text{com}}$, as presented in Figure 4.6, acts as a physical "lock-box" for the players in a secure computation: it enables a player (P_i) to commit to a receiver (P_j) a string m in a perfectly hiding way, by simply having P_i send m to the functionality in a commit phase; and later, in a reveal phase, it ensures that the commitment is opened in a perfectly binding way, by sending the unique previously recorded string m to P_j.

Functionality $\mathcal{F}_{\text{com}}$

Commit Phase: Upon receiving an input $(\texttt{Commit}, sid, x)$ from C, verify that $sid = (C, R, sid')$ for some R, else ignore the input. Next, record x and generate a public delayed output $(\texttt{Receipt}, sid)$ to R. Once x is recorded, ignore any subsequent `Commit` inputs.

Reveal Phase: Upon receiving an input $(\texttt{Open}, sid)$ from C, proceed as follows: If there is a recorded value x then generate a public delayed output $(\texttt{Open}, sid, x)$ to R. Otherwise, do nothing.

Figure 4.6: The ideal commitment functionality $\mathcal{F}_{\text{com}}$

Lemma 6. *Let ε be any positive constant. There exists a $O(n^{\varepsilon})$-round protocol π_{com} that $\mathcal{H}$-EUC-emulates $\mathcal{F}_{com}$.*

Proof. The committer P_i and the receiver P_j, on input $(\texttt{Commit}, \ (P_i, P_j, sid), v)$ to P_i and $(\texttt{Receipt}, \ (P_i, P_j, sid))$ to P_j, proceed as follows:

Stage 1: the receiver P_j picks a random secret $r \in \{0,1\}^n$, and commits to r using the CCA-secure commitment scheme $\langle C, R \rangle$, and using (P_j, sid) as the identity of the interaction.

Stage 2: the committer P_i commits to the value v using $\langle C, R \rangle$, and using (P_i, sid) as the identity of the interaction.

Stage 3: the committer P_i commits to 0^n using $\langle C, R \rangle$, and using (P_i, sid) as the identity of the interaction.

Finally, on input $\texttt{(Open,}(P_i, P_j, sid)\texttt{)}$ to both P_i and P_j, the committer P_i decommits to the value v in a reveal phase, by sending v to P_j and then providing a strongly $\mathcal{WI}$ proof of the statement that either it has committed to v in Stage 2, or that it has committed to a valid decommitment pair (r, d) of the Stage 1 commitment, in Stage 3. The receiver P_j accepts if the proof is convincing, and rejects otherwise.

Next we proceed to show that π is indeed a secure realization of $\mathcal{F}_{\mathsf{com}}$. Below we describe the technique for simulating the protocol execution of π in the ideal-world, where parties have access to the ideal commitment functionality $\mathcal{F}_{\mathsf{com}}$, and give a proof that the simulation in the ideal-world setting is indistinguishable from a real-world execution of π. Recall that we only need to prove that π $\mathcal{H}$-EUC-emulates $\mathcal{F}_{\mathsf{com}}$; hence in both the ideal and real worlds, the environment and the adversary have access to the $\mathcal{H}$ functionality.

Let A be any $\mathcal{PPT}$ adversary. The simulator S for A in the ideal world internally simulates a real-world execution with A: it simulates A's interaction with the environment Z and the functionality $\mathcal{H}$, by simply forwarding the communications between A and Z or $\mathcal{H}$; furthermore, it simulates the commit and reveal phases of the commitment $\langle C, R\rangle$ for A as follows:

Strategy 1: If the Committer (P_i) is honest and the Receiver (P_j) is corrupted, the simulator need to be able to complete the commit phase of the protocol on behalf of the committer without actually knowing the committed value v (which $\mathcal{F}_{\mathsf{com}}$ will not disclose until the reveal phase is initiated). Thus, the simulator needs to be able equivocate the commitment so that it can be opened to the proper value v in the subsequent reveal phase.

Towards this, in the commit phase, the simulator first forwards the $\langle C, R\rangle$

commitment from A (controlling P_j) in Stage 1 to the functionality $\mathcal{H}$. Since the receiver P_j is corrupted, $\mathcal{H}$ accepts commitments with identity P_j from S, and returns S a valid decommitment pair (r, d) if the commitment is accepting. (Note that it follows from the efficient verifiability property of $\langle C, R\rangle$ that if a commitment of $\langle C, R\rangle$ is accepting, then except from negligible probability, it is valid, and further by the statistically binding property, has a unique committed value, in which case, $\mathcal{H}$ would return a valid decommitment pair to this value.) Next the simulator completes the commit stage by committing to 0^n in Stage 2, and committing to the decommitment pair (r, d) in Stage 3. Then, later, in the reveal phase, S can open to any value v, by sending v to A and proving in the strongly $\mathcal{WI}$ proof that it has committed to a decommitment of the Stage 1 commitment in Stage 3.

Strategy 2: If the Committer (P_i) is corrupted and the Receiver (P_j) is honest, the simulator will need to learn the committed value v from the committer in order to correctly provide the corresponding commit phase input to $\mathcal{F}_{\mathsf{com}}$.

The simulator S emulates the messages from the receiver P_j for A (controlling P_i), by following the honest receiver strategy; additionally, it forwards the Stage 2 commitment of $\langle C, R\rangle$ from A to the decommitment functionality $\mathcal{H}$, and uses the committed value returned from the functionality as the commit phase input. More precisely, let (v, d) be the decommitment pair returned from $\mathcal{H}$; S then sends the message $(\texttt{Commit}, (P_i, P_j, sid), v)$ to $\mathcal{F}_{\mathsf{com}}$, if the Stage 2 commitment from A is accepting; otherwise, it does nothing.

Strategy 3: If both the Committer (P_i) and the Receiver (P_j) are honest, since the adversary A reads the communications between all parties, S needs to be able to simulate the commit phase between two honest parties for A, without knowing the value committed to by P_i, and later equivocate the simulated commitment to open to any value that P_i reveals. This task is almost the same as that in the first case, except that the receiver now is honest. Then S simulates the messages from the receiver P_j by following the honest receiver strategy; and simulates the messages from the committer P_i by applies Strategy 1 against the honest receiver strategy.

Below we analyze each of the simulation strategies above, and show that the environment Z's interactions with S in the ideal-world is indistinguishable from that with A in the real-world in each of the cases.

Analysis of the first case: Consider the following three hybrids:

Hybrid H_1 proceeds identically to the ideal-execution, except that in H_1 the ideal commitment functionality $\mathcal{F}_{\mathsf{com}}$ discloses the committed value v to S, once it receives it from P_i; S then commits to v honestly, instead of 0^n, in Stage 2 of the commit phase simulated for A, using $\langle C, R \rangle$ and identity (P_i, sid). Then the only difference between the ideal-execution and H_1 lies in the value committed to in Stage 2 of the simulation of π. Since $\mathcal{H}$ "breaks" commitments of $\langle C, R \rangle$ in the same way as O does, it follows from the CCA-security w.r.t. O of $\langle C, R \rangle$ that, the execution in H_1 is indistinguishable from that in the ideal-world, provided that $\mathcal{H}$ does not "break" any commitments of $\langle C, R \rangle$ using identity (P_i, sid). (Recall that CCA-security holds only if the adversary does not query the decommitment oracle on any commitment using the same identity as the left interaction.) The last requirement holds, since $\mathcal{H}$ only accepts queries on commitments from corrupted parties using their own identities; therefore, $\mathcal{H}$ never "breaks" any commitment with identity (P_i, sid) (which belongs to an honest party). Hence we conclude that the ideal-execution is indistinguishable from H_1.

Hybrid H_2 proceeds the same as H_1 does, except that S further emulates the reveal phase honestly for A, i.e., it proves in the strongly $\mathcal{WI}$ proof in the reveal phase that it has committed to v honestly in Stage 2 of the commit phase. Since the only difference between H_1 and H_2 lies in the witness used in the proof in the reveal phase, it follows from the robustness w.r.t. O property of $\langle C, R\rangle$ and the strongly $\mathcal{WI}$ property of the proof that, executions in H_1 and H_2 are indistinguishable.

Hybrid H_3 proceeds the same as H_2 does, except that S commits to 0^n, instead of the secret r, in Stage 3 of the commit phase. Since the only difference between H_2 and H_3 lies in the value committed to (using $\langle C, R\rangle$) in Stage 3 of the commit phase simulated for A, it follows from the same argument as in H_1 that the executions in H_2 and H_3 are indistinguishable.

Finally, as the view of A in H_3 is emulated perfectly as in the real-execution, we have that the output of Z in H_3 is identical to that in the real-execution. It then follows using a hybrid argument that the ideal and the real executions are indistinguishable in the first case.

Analysis of the second case: In this case, since the simulator S emulates the honest receiver P_j perfectly for A (controlling the committer P_i), the views of A in the ideal and real worlds are identically distributed. Furthermore, we show that the committed value that S extracts from A is almost always the same as the value that A opens to in the reveal phase. Hence the outputs of the honest receiver P_j in the ideal and real woulds are (almost always) identical, and thus so are the outputs of the environment.

Recall that in this case, S sends the value v that A commits to in Stage 2 (obtained from $\mathcal{H}$) as the commit phase input to $\mathcal{F}_{\mathsf{com}}$. Assume for contradiction that v is not the value A opens to later in the reveal stage, with non-negligible probability. Then by the soundness of the proof in the reveal stage, A must have committed to the secret r (i.e., the committed value of Stage 1) in Stage 3, with non-negligible probability. Then we can construct an adversary A' that violates the CCA-security w.r.t. O of $\langle C, R\rangle$. In the experiment IND_b, The adversary A'^{O}, internally emulates an ideal-execution with A and Z, by emulating the functionality $\mathcal{H}$ using O; it emulates the commit and reveal phases for A as S does, except that it forwards the commitment of $\langle C, R\rangle$ it receives from the external committer to A as Stage 1—in IND_b, the external commitment is a commitment to value r_b, from $\{r_0, r_1\}$ chosen by A', and uses identity id again chosen by A'; here A' selects r_0 and r_1 at random and sets id to be (P_j, sid)—furthermore, A' also forwards the Stage 3 commitment from A to O to obtain a decommitment pair (u, d), and outputs u at the end of the execution. Since A' emulates the ideal-execution perfectly for A and Z, the probability that A, in emulation by A', commits to the secret in Stage 3 of the commit phase is identical to that in the ideal-execution. Therefore, by our hypothesis, in IND_0, where the secret is set to r_0, the probability that A commits to r_0 in Stage 3 is non-negligible. However, in IND_1, the probability that A commits to r_0 in Stage 3 is at most $1/2^n$, as the ideal-execution is simulated completely without using r_0. Therefore the outputs of A' in IND_0 and IND_1 are distinguishable. Furthermore, since A' only forwards O the commitments from A and Z, which use identities of the corrupted parties, A' never queries O on any commitment that uses the identity of the left interaction (P_j, sid). Hence A' violates the CCA-security w.r.t. O of $\langle C, R\rangle$.

Analysis of the third case: In this case, S simulates the interaction between two honest parties for A. Since the simulation strategy is the same as that in the first case, it follows from the same proof that the real and ideal executions are indistinguishable in this case. $\square$

Realizing Any Functionality: First note that to realize any well-formed functionality, it suffices to realize the *ideal oblivious transfer functionality* $\mathcal{F}_{\mathsf{OT}}$ [Rab05, EGL85], which allows a receiver to obtain one out of two bits held by a sender, without revealing to the sender the identity of its selection, as presented in Figure 4.7. By previous works [Kil92, BOGW88, GMW87, IPS08], this suffices for *unconditionally* implementing any functionality.

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

1. Upon receiving input $(\texttt{Sender}, sid, b_0, b_1)$ from party S, verify that $sid = (S, R, sid')$ for some identity R and $b_0, b_1 \in \{0,1\}$; else ignore the input. Next, record b_0, b_1 and generate a public delayed output $(\texttt{Sender}, sid)$ to R. Ignore further $(\texttt{Sender}, ...)$ inputs.

2. Upon receiving input $(\texttt{Receiver}, sid, i)$ from R, where $i \in \{0,1\}$, wait until a value b_0, b_1 is recorded, and then send a private delayed output $(\texttt{Output}, sid, b_i)$ to R and halt.

3. Upon receiving a message $(\texttt{Corrupt}, sid, P)$ from the adversary, where $P \in \{S, R\}$, send b_0, b_1 to the adversary. Furthermore, if S is corrupted, the adversary now provides values b'_0, b'_1 with each $b'_j \in \{0,1\}$, and no output was yet written to R, then output $(\texttt{Output}, sid, b'_i)$ to R and halt.

Figure 4.7: The oblivious transfer functionality, $\mathcal{F}_{\mathsf{OT}}$

Towards securely realizing $\mathcal{F}_{\mathsf{OT}}$, we would like to rely on the the CLOS-BMR protocol [CLOS02, BMR90]—a *constant-round* protocol that UC-realize $\mathcal{F}_{\mathsf{OT}}$ in the $\mathcal{F}_{\mathsf{com}}$-hybrid model—and simply realize $\mathcal{F}_{\mathsf{com}}$ using our $\mathcal{H}$-EUC secure implementation (described above). If the CLOS-BMR protocol had been a $\mathcal{H}$-EUC realization of $\mathcal{F}_{\mathsf{OT}}$, then the resulting composed protocol would also be $\mathcal{H}$-EUC secure (and we would be done). But recall that UC-security does not necessarily imply $\mathcal{H}$-EUC security (since now the environment is endowed by the super-polynomial oracle $\mathcal{H}$). One way around this problem would be to rely on sub-exponentially-hard trapdoor permutations in the CLOS-BMR protocol. We take a different route (which dispenses of the extra assumptions): Since the CLOS-BMR protocol is constant-round, we can rely on the *robust* CCA-security property of O to prove that CLOS-BMR (relying on standard, polynomially-hard, trapdoor permutations) in fact is secure also w.r.t $\mathcal{H}$. More precisely,

Lemma 7. *Assume the existence of enhanced trapdoor permutations. Then, there exists a constant-round $\mathcal{F}_{com}$-hybrid protocol π_{OT} that $\mathcal{H}$-EUC-emulates $\mathcal{F}_{OT}$.*

Proof. To prove this lemma, we rely on the previous results from [CLOS02, BMR90][9].

Theorem 11 ([CLOS02, BMR90]). *Assume the existence of enhanced trapdoor permutations. Then, there exists a constant-round $\mathcal{F}_{com}$-hybrid protocol ρ_{OT} that UC-realizes $\mathcal{F}_{OT}$, with a black-box security proof.*

We show that the protocol ρ_{OT} that UC-emulates $\mathcal{F}_{\mathsf{OT}}$ also $\mathcal{H}$-EUC-emulates $\mathcal{F}_{\mathsf{OT}}$. By Theorem 8, it suffices to show that, for the dummy adversary $\mathcal{D}$, (which simply forwards messages between the honest parties, using ρ_{OT}, and the environment,) there exists a simulator S, such that no environment Z, with access to the helper functionality $\mathcal{H}$, can distinguish its interactions with $\mathcal{D}$ or S. Note that since the dummy adversary $\mathcal{D}$ never accesses $\mathcal{H}$, it is also a valid real-world

[9]Previous results in [CLOS02, BMR90] showed that, assuming the existence of enhanced trapdoor permutations, for every (non-reactive) function g, there exists a constant-round $\mathcal{F}_{\mathsf{com}}$-hybrid protocol that UC-securely evaluates this function. Here we only relies on this result applied to a specific function, that is, the oblivious transfer function.

adversary in the UC-model. Then by the definition of UC-security, there exists a simulator S', such that no environment Z' that obeys the rules of interaction for UC, can distinguish its interaction with $\mathcal{D}$ (and the honest parties using the $\mathcal{F}_{\mathsf{com}}$-hybrid protocol ρ_{OT} in the real-world), from its interaction with S' (and the honest parties using $\mathcal{F}_{\mathsf{OT}}$ in the ideal-world). Then let B_1 be the compound machine that contains $\mathcal{D}$, the honest parties, and $\mathcal{F}_{\mathsf{com}}$ in the real-world, and B_2 the compound machine that contains S', the honest parties, and $\mathcal{F}_{\mathsf{OT}}$ in the ideal-world. We have that:

- No UC-environment Z' can tell apart its interaction with B_1 or B_2, and
- both B_1 and B_2 are constant-round ITMs—since the dummy adversary $\mathcal{D}$ simply forwards messages between the honest parties and the environment, the interaction with B_1 consists of only messages in the protocol ρ_{OT}, and hence has some constant number of rounds k; furthermore, by the indistinguishability of the interaction with B_1 and B_2, the interaction with B_2 also contains only k rounds.

Given the two properties above, we show that S' is also a valid simulator for $\mathcal{D}$ in the $\mathcal{H}$-EUC-model, that is, no environment Z, in the $\mathcal{H}$-EUC-model, can tell apart its interactions with $\mathcal{D}$ (and the honest parties using ρ_{OT}), from that with S' (and honest parties using $\mathcal{F}_{\mathsf{OT}}$). In other words, no environment Z, having access to $\mathcal{H}$, can tell apart its interaction with B_1 or B_2. Suppose not, and that there is an environment Z that distinguishes interactions with B_1 and B_2. Then by the robustness w.r.t. O of $\langle C, R\rangle$, and the fact that both B_1 and B_2 are constant-round ITMs, there exists a simulator Z'', such that, the interacion between B_i and $Z^{\mathcal{H}}$, is indistinguishable from that between B_i and Z''. Therefore the environment Z'', without access to $\mathcal{H}$, also distinguishes the interactions with B_1 and

B_2. However, this contradicts with the first property above. Hence, we conclude the lemma. □

Finally, given $\mathcal{F}_{\mathsf{OT}}$, we can securely realize any well-formed functionalities.

Lemma 8. *For every well-formed functionality $\mathcal{F}$, there exists a constant-round $\mathcal{F}_{OT}$-hybrid protocol ρ that $\mathcal{H}$-EUC-emulates $\mathcal{F}$.*

Proof. This lemma follows essentially from the previous works [Kil92, BOGW88, GMW87, IPS08], which showed that for any well-formed functionality $\mathcal{F}$, there exists a constant-round protocol π that UC-securely realizes $\mathcal{F}$ in the $\mathcal{F}_{\mathsf{OT}}$-hybrid model, with a black-box security proof. In fact, it follows syntactically from the same proof as in [Kil92, BOGW88, GMW87, IPS08] that this result holds even for environments that run in sub-exponential time. In particular, for the dummy adversary $\mathcal{D}$ in the $\mathcal{H}$-EUC model, (who, as argued in the proof of Lemma 7, does not accesses the helper functionality and thus is also a valid UC-adversary), there exists a simulator S in the ideal world, such that no sub-exponential time environment can tell part its interactions with $\mathcal{D}$ or S. Since the helper functionality $\mathcal{H}$ can be implemented in sub-exponential time, we can view environments in the $\mathcal{H}$-EUC model as sub-exponential time machines. Thus, combining Theorem 8, we have that π also $\mathcal{H}$-EUC-emulates $\mathcal{F}$. □

CHAPTER 5

FROM NON-MALLEABILITY TO CCA-SECURITY

We show that there is a tight connection between the two notions of input independence: Already we know that CCA-security is a natural strengthening of non-malleability; in this chapter, we show that there is a generic compilation technique that transforms any robust non-malleable commitment scheme into a t-robust CCA secure commitment scheme for any constant t, by additionally relying on a concurrent extraction strategy of a special type. This leads to new, round optimal, constructions of CCA-secure commitments. By plugging these new constructions in the construction of $\mathcal{H}$-EUC secure protocols in the last chapter, we immediately improve the round-complexity of $\mathcal{H}$-EUC security.

5.1 Generic Construction of Robust CCA-Secure Commitments

In this section we provide a construction of t-robust CCA-secure commitments using a robust non-malleable commitment scheme and a concurrent extraction strategy. As shown in Chapter 3, the former can be based on one-way functions. The latter lies in the heart of constructions of concurrent $\mathcal{ZK}$ protocols. Almost all constructions of concurrent $\mathcal{ZK}$ protocols implicitly define a sub-protocol, which as shown in their security proofs, has a corresponding concurrent extraction strategy—that is, a rewinding extractor—that can extract out secrets defined in the sub-protocols when executed in a concurrent setting. For example, the preamble stage of the concurrent $\mathcal{ZK}$ protocol of [PRS02] is such a concurrently-extractable sub-protocol, which consists of $\omega(\log n)$ rounds; so is the preamble stage of the concurrent $\mathcal{ZK}$ protocol of [PV08], which contains only constant rounds but exhibits a quasi-polynomial time extraction strategy. We note that Micciancio, Ong, Sahai and Vadhan introduced the notion of *concurrently-extractable commitment schemes* in [MOSV06], which is an abstraction of the preamble stage of the protocol in [PRS02]. We here, however, are unable to employ their abstraction. Instead, we will directly define a restricted type of concurrent extraction strategies that are "conforming" for our CCA-secure commitments, and then provide a modular security analysis, based on properties of the conforming concurrent extraction strategy and the robust non-malleable commitment scheme. It turns out that many previous concurrent extraction strategies [RK99, KP01, PRS02, PTV08] are in fact conforming. Furthermore, the round-complexity of our CCA-secure commitments is dominated by the round-complexity required by the conforming concurrent extraction strategy. Therefore, by employing the strategy in [PTV08] (which in turn is based on [KP01, PRS02], we obtain a $\omega(\log n)$ robust CCA-secure commitment scheme based on one-way functions. That is,

Theorem 12. *Assume the existence of one-way functions. Then, there exists a $\omega(\log n)$-round robust CCA-secure commitment scheme.*

Furthermore, by employing the concurrent extraction strategy in [PV08] for quasi-polynomial time adversaries (or $\mathcal{PQT}$), we obtain a constant-round t-robust CCA-secure commitment scheme w.r.t. $\mathcal{PQT}$ based on one-way functions hard for quasi-polynomial time.

Theorem 13. *Assume the existence of one-way functions hard for quasi-polynomial time. Then, for every constant t, there exists a constant-round t-robust CCA-secure commitment scheme w.r.t. quasi-polynomial time adversaries.*

Below we first describe our constructions in a unified way and specify the concrete instantiations in Section 5.2. For simplicity of exposition, our description below relies on the following primitives:

1. *One-way functions f with efficiently recognizable range.*

   But, the protocol can be easily modified to work with any arbitrary one-way function using the same technique used in Chapter 4 of providing a witness hiding proof that an element is in the range of the one-way function; see Remark 2 for more details.

2. *A 2-round statistically-binding commitment scheme* com *that has* unique decommitment *(i.e., for every machine A, it holds that in an interaction with the honest receiver, the probability that A produces a commitment c, such that, there exist two decommitments $(v_1, d_1) \neq (v_2, d_2)$ that make the honest receiver accept in the reveal stage is negligible.*

   The requirement of unique decommiment is not necessary and can be avoided by using the same technique introduced in our first construction of robust CCA-secure commitments in chapter 4 (see Remark 4 for more details.)

Furthermore, our construction makes use of an α-round $\mathcal{ZK}$ proof system $\langle P_z, V_z \rangle$ for $\mathcal{NP}$, and an $O(\alpha)$-round α-robust non-malleable commitment scheme $\langle \tilde{C}, \tilde{R} \rangle$; such a non-malleable commitment scheme exits by By Theorem 3.

Given the above primitives, for any positive integer t, we construct a t-robust CCA-secure commitment scheme $\langle C_t, R_t \rangle$ as follows. To commit to a value v, the Committer and the Receiver, on common input 1^n and the identity $\mathsf{id} \in \{0,1\}^n$, where n is the security parameter, proceed in six stages:

Stage 1 the Receiver picks a random string $r \in \{0,1\}^n$, and sends its image $s = f(r)$, through the one-way function f with an efficiently recognizable range, to the Committer. The Committer checks that s is in the range of f and aborts otherwise. Furthermore, the receiver also sends in parallel the first message c_1 of a commitment of com.

Stage 2 the Committer sends the second message c_2 of a commitment of com to v, followed by a proof of the statement that (c_1, c_2) is a valid commitment of com using the α-round $\mathcal{ZK}$ proof system $\langle P_z, V_z \rangle$.

Stage 3 the Committer commits to v (again) using the $O(\alpha)$-round α-robust non-malleable commitment scheme $\langle \tilde{C}, \tilde{R} \rangle$ and identity id.

Stage 4 the Committer commits to 0^n using $\langle \tilde{C}, \tilde{R} \rangle$ and identity id.

Stage 5 the Committer performs a 4-round $\mathcal{WISSP}$ proof $\langle P_w, V_w \rangle$ of the statement that it has committed to 0^n in Stage 4, *or* it knows a pre-image of s.

Stage 6 the Committer proves that it has committed to value v in both Stage 2 and 3, or a pre-image of s in Stage 4. This is proved in λ iterations, where iteration $i \in [\lambda]$ consists of l invocations of the 4-round $\mathcal{WISSP}$ protocol $\langle P_w, V_w \rangle$ arranged as follows: The first two messages of the l invocations are exchanged in parallel in the first two rounds, followed by l message exchanges. In the j^{th} round, the prover sends $\beta_j \leftarrow \{0,1\}^*$, a random second last message for the j^{th} $\mathcal{WISSP}$ invocation, and the verifier replies with the last message γ_j of the proof. Each message exchange is called a *(rewinding) slot* and the l $\mathcal{WISSP}$ proofs in an iteration is called a *block*. A slot is *convincing* if the verifier produces an accepting proof; a block is *convincing* if every slot in the block is convincing. If there is ever an *unconvincing* slot, the prover aborts the whole session.

Finally, the parameter λ is set to $\max(2k+1, t)$ where k is the maximum number of messages exchanged in each of Stage 2 to 5, and l will be defined later. (Jumping ahead, $l(n)$ is the number of "rewinding slots" required by the concurrent extraction strategy. For instance, l is any super-logarithmic polynomial if we employ the concurrent extraction strategy of [PTV08]). Let $\ell = \lambda l$ be the total number of $\mathcal{WISSP}$ proofs in Stage 6.

Let Δ be a transcript of the protocol $\langle \tilde{C}, \tilde{R} \rangle$. We define the value committed to in the transcript to be the value committed to in Stage 2 (using com) of Δ. That is, in the reveal stage, the committer needs to reveal the committed value and a corresponding decommitment string of Stage 2 of the protocol as the decommitment.

On the round complexity of $\langle C_t, R_t \rangle$: It is easy to see that the round complexity of the first 5 stages of $\langle C_t, R_t \rangle$ is bounded by $4k + 1$ and that of Stage 6 is bounded by $4\ell(n) = 4\lambda l$. Since t is a constant, the overall round complexity of $\langle C_t, R_t \rangle$ is $O(kl)$. Furthermore, as k is the maximum number of messages exchanged in Stage 2 to 5, and hence $O(\alpha)$, the round complexity of $\langle C_t, R_t \rangle$ is $O(\alpha l)$. Therefore, the round complexity is decided by our instantiations of the $\mathcal{ZK}$ proof system $\langle P_z, V_z \rangle$ (that decides α), and the concurrent extraction strategy (that decides l). We defer the concrete instantiations to Section 5.2.2 and 5.2.3.

Proposition 7. *$\langle C_t, R_t \rangle$ is binding with efficient verifiability.*

Proof. **Binding:** The binding property follows directly from the statistically binding property of com used in Stage 2.

Efficient Verifiability: It follows from the soundness of the Stage 2 zero-knowledge proof of $\langle P_z, V_z \rangle$ that, except with negligible probability, whenever a commitment of $\langle C_t, R_t \rangle$ is accepting, the Stage 2 commitment of com must be valid. Then since the decommiment of a commitment of $\langle C_t, R_t \rangle$ is simply the decommitment of its Stage 2 commitment, we have that the whole $\langle C_t, R_t \rangle$ commitment is also valid.

$\square$

Next we proceed to show that $\langle C_t, R_t \rangle$ is a t-robust CCA-secure commitment. Before providing the formal proof, we first sketch the high-level idea of the proof of CCA-security; the proof of t-robustness uses very similar ideas. Towards this, recall that we need to exhibit a decommitment oracle O for $\langle C_t, R_t \rangle$ such that $\langle C_t, R_t \rangle$ is CCA-secure and t-robust w.r.t. O. Since the decommitment of a $\langle C_t, R_t \rangle$ commitment is the *unique* decommitment of its Stage 2 commitment of com. Consider the following oracle O: O acts as an honest receiver during the commit stages; at the end of every accepting interaction of commit stage, if the Stage 2 com commitment is valid, O returns the unique decommitment of Stage 2; otherwise, it returns $\perp$.

5.1.1 Proof Overview of CCA-security

To show that $\langle C_t, R_t \rangle$ is CCA-secure w.r.t. O, we follow the same high-level idea used in our first construction of robust CCA-secure commitment scheme in Chapter 4. Recall that proving CCA-security w.r.t. O amounts to showing that the views of A in experiments IND_0 and IND_1 are indistinguishable (when A has oracle access to O). The main hurdle in showing this is that the oracle O is not efficiently computable; if it were, indistinguishability would directly follow from the hiding property of the left interaction. However, since $\langle C_t, R_t \rangle$ consists of a sequence of special-sound proofs of the committed value in the last stage, the oracle O can be implemented by extracting the decommitments from the adversary, via "rewinding" the special-sound proofs in the right interactions. However, two problems arise when trying to simulate the oracle O:

- First, once we start rewinding the right interactions, A might send new messages also in the left interaction. So, if done naively, this would require us to also rewind the left interaction, which could violate its hiding property.
- Second, in the experiment IND_b, the adversary A expects to receive the decommitment information at the very moment it completes a commitment to its oracle. If the adversary "nests" its oracle calls, these rewindings become recursive and the running-time of the extraction quickly becomes exponential.

Previous solution: To overcome the two problems, our first construction relies crucially on two techniques: A special message scheduling technique based on the message scheduling technique for constructing non-malleable commitments in [DDN00, LPV08], and a concurrent extraction strategy based on that of Richardson and Kilian (RK) [RK99]. The special message scheduling ensures that for every accepting right interaction with an identity that is different from the left interaction, there exists $\Omega(n^\varepsilon)$ safe-points in the interaction, from which one can rewind the right interaction (and extract out the value committed to) without requesting any *new* message in the left interaction. Using these safe-points, the oracle O can then be simulated without violating the hiding property of the left interaction. Furthermore, since there are $\Omega(n^{\varepsilon})$ safe-points in every right interaction, there is always at least one safe-point where the number of nested executions inside is "small", and if rewinding only from such safe-points according to a RK-type extraction strategy, the overall running-time of extraction is bounded by a polynomial in expectation.

Our solution: The drawback of relying on message scheduling technique and RK-type concurrent extraction strategy is that both of them require the protocol to have at least $\Omega(n^{\varepsilon})$ "rewinding" slots. To improve the round complexity, we seek different solutions to the two above-mentioned problems.

To overcome the first problem, instead of considering a single extraction strategy that "preserves" the hiding property of the "whole" left interaction as in our first construction, we consider *different* extraction strategies that "preserve" the hiding property of different "parts" of the left interaction. Therefore, the view of A is indistinguishable if each time we only change a small "part" of the left interaction. Then by gradually changing different parts of the left interaction, it follow from a hybrid argument that the view of A is indistinguishable in IND_0 and IND_1. More precisely, consider the following sequence of hybrids H_0 to $H_{\ell+4}$, $\ell = \ell(n)$, where H_0 is simply the honest execution of experiment IND_0, and every two subsequent hybrids differ at a small "part" of the left interaction.

Hybrid H_0: Hybrid H_0 consists of an honest emulation of the experiment IND_0.

Hybrid H_1: Hybrid H_1 proceeds identically to H_0 except that the left execution is emulated by finding a fake witness r, using a brute-force search, and next using this r as a witness to succeed in the $\mathcal{WISSP}$ proof in Stage 5; everything else remains the same.

Hybrid H_2: Hybrid H_2 proceeds identically to H_1 except that the left execution is emulated by committing to the fake witness r using $\langle \tilde{C}, \tilde{R} \rangle$ in Stage 4.

Hybrids H_3 to $H_{\ell+2}$: In hybrids H_3 to $H_{\ell(n)+2}$, we change the witness used in the ℓ $\mathcal{WISSP}$ proofs in Stage 6 of the left interaction. More specifically, the experiment H_{i+2} proceeds identically to H_2, except that in the first i proofs in Stage 6 of the left interaction, we use the fact that we have committed to a fake witness r in Stage 4.

Hybrid $H_{\ell+3}$: Hybrid $H_{\ell+3}$ proceeds identically to $H_{\ell+2}$ except that Stage 3 of the left execution is emulated by committing to 0 using $\langle \tilde{C}, \tilde{R} \rangle$.

Hybrid $H_{\ell+4}$: Hybrid $H_{\ell+4}$ proceeds identically to $H_{\ell+3}$ except that the Stage 2 commitment of the left execution is emulated by committing to 0 followed by an honest $\mathcal{ZK}$ proof of the validity of the commitment.

By construction, every two subsequent hybrids H_i and H_{i+1} differ only at a small "part" of the left interaction: Either one of Stage 2 to 5, or one of the $\mathcal{WISSP}$ proofs in Stage 6. Furthermore, the difference in the "part" changed in H_i and H_{i+1} is computationally indistinguishable. More precisely, from hybrid H_0 to H_1 and from H_j to H_{j+1} for $2 \leq j \leq \ell + 1$, the difference lies in how a $\mathcal{WISSP}$ proof (in Stage 5 or Stage 6 respectively) in the left interaction is simulated, it thus follows from the witness indistinguishability of the $\mathcal{WISSP}$ protocol $\langle P_w, V_w \rangle$ that the difference is indistinguishable. Additionally, from hybrid H_1 to H_2 and from $H_{\ell+2}$ to $H_{\ell+3}$, the indistinguishability follows from the hiding property of the commitments $\langle \tilde{C}, \tilde{R} \rangle$, and from $H_{\ell+3}$ to $H_{\ell+4}$, it follows from the hiding property of com together with the $\mathcal{ZK}$ property of the protocol $\langle P_z, V_z \rangle$.

To show CCA-security, it essentially amounts to show that every two subsequent hybrids H_i and H_{i+1} are even indistinguishable to adversaries with access to the decommitment oracle $\mathcal{O}$. As in the proof for our first construction of CCA-secure commitments, we prove this by showing that the decommitment oracle $\mathcal{O}$ can be simulated via rewindings (so as to extract out the committed values) *without violating the indistinguishability of the "part" (in the left interaction) that is being changed from H_i to H_{i+1}*. This is the main technical content of the proof. Compared with the task in our first construction of designing a concurrent extraction strategy that retains the hiding property of the *whole* left interaction, here, since every two subsequent hybrids differ at only at a small part of the left interaction consisting of a few—bounded by k—messages, it becomes much easier to devise an extraction strategy that retains the indistinguishability of these k messages. More precisely, we employ an idea similar to that used in the context of concurrent non-malleable $\mathcal{ZK}$ with adaptive input selection of [LP11a]. They presented a concurrent extraction strategy that avoids rewinding a small number γ of "critical messages", by using the PRS extraction strategy in a modular way. Intuitively, the basic idea is to provide more, say $\gamma + 1$, "blocks" of special-sound proofs in the protocol than the number of critical messages, where each block contains many special-sound proofs adequate for performing the concurrent extraction strategy (for example, $\omega(\log n)$ proofs when using the PRS concurrent extraction strategy). Therefore in every right interaction, there must exist one complete "block" of special-sound proofs such that during its execution, no critical message is being exchanged; then a witness can be extracted via rewinding the special-sound proofs in that block without rewinding any critical message. In the protocol $\langle C_t, R_t \rangle$, the last stage contains at least $(2k + 1)$ blocks of special-sound proofs, each consisting of l proofs. By setting l appropriately according to the concurrent extraction strategy we choose, we can extract out a witness for every right interaction without rewinding the critical messages—at most k of them—being changed from H_i to H_{i+1}. In other words, the oracle O can be emulated efficiently without rewinding the "part" being changed from H_i to H_{i+1}.

There is one more problem to be solved before the above argument goes through: We need to show that the witnesses extracted from the right interactions are indeed the valid decommitments. This essentially boils down to show that the adversary A never commits to a "trapdoor"—a pre-image of the Stage 1 message—in Stage 4 of any right interaction, even though the left interaction it participates in is simulated using a "trapdoor". It seems that this should follow from the robust non-malleability of the Stage 4 commitments of $\langle \tilde{C}, \tilde{R} \rangle$. However, when showing this, we run into the same problem as we encounter when showing the indistinguishability of the view of A: Namely, the robust non-malleability property needs to hold even to an adversary with access to the decommitment oracle O; or essentially, it needs to hold under rewindings (for simulating O). We use the same idea as above: as long as there are sufficiently many blocks of special-sound proofs, we can describe a concurrent extraction strategy that avoids rewinding the "part" being changed in H_i and H_{i+1} together with the commitment of $\langle C_t, R_t \rangle$ that we want to violate robust non-malleability of

5.2 Proof of t-Robust CCA-Security

In this section, we provide the formal proof of the t-robust CCA-security of $\langle C_t, R_t \rangle$. As discussed above, the proof crucially relies on different concurrent extraction strategies for the protocol $\langle C_t, R_t \rangle$; let us start by introducing them.

5.2.1 Concurrent Extraction Strategy for $\langle C_t, R_t\rangle$

Our basic definition of a concurrent extraction strategy for $\langle C_t, R_t\rangle$ is very similar to that of concurrently extractable commitments introduced in [MOSV06]. Consider a (potentially unbounded) probabilistic adversary A that, on receiving a security parameter 1^n and an auxiliary input z, sends $m = m(n)$ commitments of $\langle C_t, R_t\rangle$ to values $v_1, \ldots, v_m$ of its choice, using identities $\mathsf{id}_1, \ldots, \mathsf{id}_m$ of its choice again. Let $\mathcal{V}$ be a prefix of the view of A in such an experiment, which consists of a list messages that A receives and the random coins it tosses. Let $\mathsf{view}_A(n, z, \mathcal{V})$ denote the view of A continuing from $\mathcal{V}$ in the experiment. Loosely speaking, a basic concurrent extraction strategy of $\langle C_t, R_t\rangle$ is a probabilistic machine E (called the extractor) that, on input the security parameter 1^n, $m(n)$ and a partial view $\mathcal{V}$, with black-box access to $A(1^n, z, \mathcal{V}\|*)$, emulates the view of A continuing from $\mathcal{V}$ perfectly; furthermore, for every block of special-sound proofs that A completes successfully, E outputs a valid witness of the statement proved in that block.

Definition 30 (Basic Concurrent Extraction Strategy for $\langle C_t, R_t\rangle$). *A concurrent extraction strategy for $\langle C_t, R_t\rangle$ is a probabilistic machine E, such that, for every polynomial m, and every probabilistic machine A that on common input 1^n and auxiliary input z opens up at most $m(n)$ commitments of $\langle C_t, R_t\rangle$, the following two properties holds.*

Statistical Simulation: *For every partial view $\mathcal{V}$, the following ensembles are statistically close.*

- $\{\mathsf{view}_A(n, z, \mathcal{V})\}_{n\in N, z\in\{0,1\}^*}$
- $\left\{E^{A(1^n, z, \mathcal{V}\|*)}(1^n, m(n), \mathcal{V})\right\}_{n\in N, z\in\{0,1\}^*}$

Concurrent Extraction from Blocks: *In the execution of $E^{A(1^n, z, \mathcal{V}\|*)}(1^n, m(n), \mathcal{V})$, it holds that except with negligible probability, whenever E makes a query Q to A,* *for every convincing block of special sound proofs contained completely in Q, E outputs on a special output tape a witness w of the statement proved in that block.*

Conforming Strategies

In this work, we focus on concurrent extraction strategies that follow certain restrictions: Intuitively, an extractor E uses rewindings to extract the witnesses, and during rewindings, it can feed the adversary any messages and random inputs of its choice. We say that a concurrent extractor is "conforming" if in rewindings, it always emulates messages from the receiver for A honestly and independently (in different rewindings) and supply A with independent truly random inputs. More precisely, for every query $Q = (T; r)$ made by E to A that consists of a list of input messages $T = (a_1, \ldots, a_k)$ and random input r, let $(b_1, \ldots, b_k)$ be the list of messages returned by A_r on input messages in T, and $\mathsf{mesg}(T) = (a_1, b_1, a_2, b_2, \ldots, a_k, b_k)$ the complete transcript of messages; furthermore, denote by $(T; R)\|(a; r)$ the query $(T\|a; R\|r)$.

Definition 31 (Conforming Concurrent Extraction Strategy for $\langle C_t, R_t \rangle$)**.** *A concurrent extraction strategy E is* $\mathsf{conforming}$ *if it satisfies the following properties:*

Random Queries: *It holds that before E makes a query $((T\|a); (R\|r))$ to A, it has already made query $(T; R)$ to A. Furthermore, r is a truly random string and the distribution of a is identical to the distribution of the next message from the honest receiver $\hat{R}$ conditioned on that* $\mathsf{mesg}(T)$ *has occurred in an interaction with $\hat{R}$.*

Independent Queries: *Every pair of queries (T_0, R_0) and (T_1, R_1) that E makes to A may share a prefix, but are otherwise uncorrelated. That is, there exists a query (T, R), such that, it holds that for every $b \in \{0, 1\}$, $T_b = T\|T'_b$ and $R_b = R\|R'_b$, and* *the distributions of* (T'_0, R'_0) *and* (T'_1, R'_1) *are independent.*

Furthermore, it is without loss of generality to assume that the emulated view $\tilde{T}$ of A that E outputs in the end is queried to A at some point.

Intuitively, a conforming concurrent extractor E only leverage the power of rewinding to extract witnesses: It internally emulates executions between A (on truly random input) and the honest receiver $\hat{R}$; it may rewind these executions, and in rewindings, messages to A are emulated again honestly and independently and A is supplied with truly random inputs. Finally, it outputs one complete execution with A that it emulates. In fact, almost all previous works on concurrent $\mathcal{ZK}$ implicitly defines a concurrent extraction strategy, and many of them are conforming and can be adapted to work on protocol $\langle C_t, R_t \rangle$. Following terminologies used in previous works, we call a continuous execution between A and $\hat{R}$ emulated by E internally a *thread*, and the execution that E outputs in the end the *main thread*.

In this work, we consider two conforming concurrent extraction strategies adapted respectively from the strategies defined implicitly in [PTV08] and [PV08]. In [PTV08], Pass, Tseng, and Venkitasubramaniam present a concurrent $\mathcal{ZK}$ protocol with a preamble phase consisting of $\omega(\log n)$ special sound proofs scheduled in the same as in a block in protocol $\langle C_t, R_t \rangle$. The $\mathcal{ZK}$ property of their protocol is based on a more general analysis of the oblivious simulation technique of Kilian and Petrank [KP01]. The core of their analysis is a concurrent extraction strategy over the $\omega(\log n)$ special sound proofs; furthermore, their strategy is "conforming". Recall that each block of the protocol $\langle C_t, R_t \rangle$ consists of l special sound proofs. Then, using the analysis of [PTV08], we obtain that:

Proposition 8 ([PTV08])**.** *Set the polynomial* l *to* $\omega(\log n)$*. Then, there exists a con-*

forming concurrent extraction strategy for $\langle C_t, R_t \rangle$ that runs in polynomial time.

In another work [PV08], Pass and Venkitasubramaniam presented a constant-round concurrent $\mathcal{ZK}$ protocol for quasi-polynomial time adversaries based on the simulation technique of Richardson and Kilian [RK99]. Their analysis exhibits a concurrent extraction strategy over 3 special-sound proofs (scheduled as in a block of $\langle C_t, R_t \rangle$) that takes quasi-polynomial time, which is, again, conforming. Therefore,

Proposition 9 ([PV08])**.** *Set the polynomial l to 3. Then there exists a conforming concurrent extraction strategy for $\langle C_t, R_t \rangle$ that runs in quasi-polynomial time.*

Robust Concurrent Extraction Strategy for $\langle C_t, R_t \rangle$

In this section, we define robust concurrent extraction strategy. Roughly speaking, it is a concurrent extraction strategy that can avoid rewinding a small number of "critical" messages. Formally, consider a (potentially unbounded) deterministic adversary A that, on receiving a security parameter 1^n and auxiliary input z, participates in one left and many right interactions. On the left, it interacts with a machine B, and on the right it sends $m = m(n)$ commitments of $\langle C_t, R_t \rangle$ as defined above. Loosely speaking, a concurrent extraction strategy E of $\langle C_t, R_t \rangle$ is robust w.r.t. a machine B, if E on input 1^n and $m(n)$, with black-box access to $A(1^n, z)$, emulates the execution between A and B (almost) perfectly *without rewinding B*; furthermore, for every accepting *commitment* from A, E outputs a valid witness of the statement proved in Stage 6 of that commitment. (Note that different from the basic concurrent extraction strategy, the robust extractor only needs to simulate the execution between A and B from the very beginning, instead of from some partial execution, and only need to output a witness for every accepting commitment, instead of every convincing block.)

Definition 32 (Robust Concurrent Extraction Strategy for $\langle C_t, R_t\rangle$). *Let B be a interactive Turing machine. A concurrent extraction strategy E for $\langle C_t, R_t\rangle$ is* robust w.r.t. *B if for every polynomial m, and every deterministic machine A that on common input 1^n and auxiliary input z, interacts with B and opens up at most $m(n)$ commitments of $\langle C_t, R_t\rangle$, the following holds.*

Statistical Simulation: *The following ensembles are statistically close.*

- $\{\mathsf{out}_{B,A}[\langle B, A(z)\rangle(1^n)]\}_{n\in N, z\in\{0,1\}^*}$
- $\left\{\mathsf{out}_{B,E^A}[\langle B, E^{A(1^n,z)}(m(n))\rangle(1^n)]\right\}_{n\in N, z\in\{0,1\}^*}$

Robust Concurrent Extraction: *In the experiment $\langle B, E^{A(1^n,z)}(m(n))\rangle(1^n)$, it holds that except with negligible probability, whenever E makes a query Q to A, for every accepting commitment in Q, E outputs on a special output tape a witness w of the statement proved in Stage 6 of that commitment.*

We say that a concurrent extraction strategy E for $\langle C_t, R_t\rangle$ is β-robust if it is robust w.r.t. every machine B that sends at most β messages.

In [LP11a], Lin and Pass presented a transformation that turns specifically the [PTV08] concurrent extraction strategy into another strategy that avoids rewinding a small number β of critical messages. We show that their transformation can actually be applied as a generic transformation turning any concurrent extraction strategy for $\langle C_t, R_t\rangle$ into a β-robust concurrent extraction strategy, provided that the number of blocks of special sound proofs in $\langle C_t, R_t\rangle$ is greater than β. Recall that in Stage 6 of $\langle C_t, R_t\rangle$ there are λ blocks. Then,

Proposition 10. *Let β be any positive integer smaller than λ. Let $\tilde{E}$ be a (conforming) concurrent extraction strategy for $\langle C_t, R_t\rangle$. Then there exists a (conforming) concurrent extraction strategy E_β for $\langle C_t, R_t\rangle$ that is β-robust. Furthermore, the running time of E_β is polynomial in the running time of E.*

Proof. Given a concurrent extraction strategy $\tilde{E}$ for $\langle C_t, R_t\rangle$, our goal is to construct another strategy E_β, such that, for every deterministic machine A and machine B that sends at most β messages, E_β emulates the interaction between A and B without rewinding B and extracts out a witness for every accepting commitment from A.

We assume without loss of generality that the interaction between A and B consists of β rounds of message exchanges where A sends the first message. Let $(m_1, \dots, m_\beta)$ denote the messages that A sends to B and a_i for $i \in [\beta]$ the message that B sends to A in reply to m_i. Then the execution with A can be emulated by the sequential execution of the following $\beta + 1$ machines $A_1, \dots, A_{\beta+1}$, where machine A_i produces a partial view $\mathcal{V}_i$ of A up until the message m_i is sent.

Machine A_i: A_i on input the partial view $\mathcal{V}_{i-1}$ produced by its predecessor A_{i-1} and the reply a_{i-1} ($\mathcal{V}_0 = \varepsilon$, $a_0 = \varepsilon$) from B, continue the execution of A from $\mathcal{V}_{i-1} \| a_{i-1}$, by feeding $\mathcal{V}_{i-1}$ and a_{i-1} to A, and forwarding every message from A externally; finally, it aborts when A terminates or sends the message m_i, and output the newly generated view $\mathcal{V}_i$ of A.

Note that each machine A_i with message a_{i-1} and partial view $\mathcal{V}_{i-1}$ hard-coded is a machine that interacts only with $\hat{R}$. Therefore, when applying the the concurrent extraction strategy $\tilde{E}$ over A_i, E emulates the view of A_i continuing from

$\mathcal{V}_{i-1}\|a_{j-1}$ and extracts a witness from every convincing block of special-sound proofs that starts completely outside $\mathcal{V}_{i-1}$.

The concurrent extraction strategy E_β emulates the view of A in interaction with B in a "progressive" way by emulating the views of $A_1, \ldots, A_{\beta+1}$ using $\tilde{E}$ in sequence in $\beta + 1$ iterations. More precisely, let $\mathcal{V}_j$ for $j \in [\beta + 1]$ (and $\mathcal{V}_0 = \varepsilon$) denote the partial view of A that E_β generates after j steps.

In iteration j**:** E_β sends the last message m_{j-1} in $\mathcal{V}_{j-1}$ to B and obtains a reply a_{j-1}.

It then simulates the view of A_j continuing from $\mathcal{V}_{j-1}\|a_{j-1}$ using $\tilde{E}$. Whenever $\tilde{E}$ makes a query Q to $A_j(1^n, z, \mathcal{V}_{j-1}\|a_{j-1}\|*)$, E_β emulates the answer by making a query $Q' = (\mathcal{V}_{j-1}\|a_{j-1}\|Q)$ to its own oracle $A(1^n, z, *)$. Furthermore, E_β copies whatever $\tilde{E}$ outputs on the special output tape to its own special output tape.

Finally, it sets $\mathcal{V}_j$ to the output of $\tilde{E}$, that is,

$$\mathcal{V}_j = E^{A_j(1^n, z, \mathcal{V}_{j-1}\|a_{j-1}\|*)}(1^n, \mathcal{V}_{j-1}\|a_{j-1})$$

After $\beta + 1$ steps, E_β returns the output of A after receiving messages in $\mathcal{V}_{t+1}$.

By construction, E_β never rewinds machine B. Next we show that E_β satisfies the statistical simulation and concurrent extraction properties. For the former, since E_β emulates the oracle $A_j(1^n, z, \mathcal{V}_{j-1}\|a_{j-1}\|*)$ perfectly for $\tilde{E}$, it follows from the statistical simulation property of $\tilde{E}$ that every partial view $\mathcal{V}_j$ is emulated (almost) perfectly, thus the joint output of E_β and B distributes statistically close to the joint output of A and B. For the latter, we need to show that whenever E_β makes a query $(\mathcal{V}_{j-1}\|a_{j-1}\|Q)$ (in some iteration $j \leq [\beta + 1]$) to A that contains an accepting commitment, it has already output on its special output tape a matching witness. Consider the following two scenarios: the view Q contains a complete block of special sound proofs in that commitment, or not. In this former case, it follows directly from the concurrent extraction property of $\tilde{E}$ that when $\tilde{E}$ makes the query Q to A_j, it also outputs on its special output tape a matching witness; thus E_β copies that witness to its special output tape. In the latter case, the prefix $\mathcal{V}_{j-1}$ must contain at least $\lambda - 1$ blocks of special sound proofs in that commitment. Consider all the partial views $\mathcal{V}_1 \dots \mathcal{V}_{j-1}$ generated in previous steps. Since for every i, $\mathcal{V}_{i-1}$ is a prefix of $\mathcal{V}_i$, we have that one of $\mathcal{V}_1, (\mathcal{V}_2 - \mathcal{V}_1), \dots, (\mathcal{V}_{j-1} - \mathcal{V}_{j-2})$ (where $V_2 - V_1 = V$ if $V_2 = V_1 \| V$) must contain a complete block of special sound proofs, say it is $\mathcal{V}_i - \mathcal{V}_{i-1}$. Then in step i, $\tilde{E}$ must have queried A_i on transcript $\mathcal{V}_i - \mathcal{V}_{i-1}$ (recall that we assume without loss of generality that the concurrent extraction strategy always queries the adversary on the view it finally outputs). Therefore, by the concurrent extraction property of $\tilde{E}$ again that $\tilde{E}$ must have output a matching witness in step i, and thus so has E_β. In conclusion, E_β is a β-robust concurrent extraction strategy.

Finally, it is easy to see that above transformation also preserves the "conforming" property, and the running time of E_β is polynomial in the running time of E. □

5.2.2 $\omega(\log n)$-Round Robust CCA-Secure Protocol

We prove in this section the following proposition:

Proposition 11. *Set the polynomial l to $\omega(\log n)$. Then $\langle C_t, R_t \rangle$ is t-robust CCA secure w.r.t. O.*

Then by instantiating $\langle C_t, R_t \rangle$ with a $\omega(1)$ round $\mathcal{ZK}$ proof system (in Stage 2), we obtain a $\omega(\log n)$-round t-robust CCA-secure commitment scheme based on one-way functions. Furthermore, since when l is set to $\omega(\log n)$, for all constant t, the protocol $\langle C_t, R_t \rangle$ is the same, call it $\langle C, R \rangle$. We obtain that, $\langle C, R \rangle$ is CCA-secure and robust to any constant-round protocols, and conclude Theorem 12

Proof of CCA-Security

Consider the sequence of hybrid $H_1, \ldots, H_{\ell+4}$ defined in Section 5.1.1. Let $\mathsf{hyb}_i(A, n, z)$ denote the view of the adversary $A(z)$ (with access to the decommitment oracle O) in the hybrid H_i. Furthermore, in a commitment of $\langle C_t, R_t \rangle$, we call the pre-image of the Stage 1 message a "trapdoor" of that commitment; we say that A "cheats" in a hybrid H_i, if in an accepting right commitment A commits to the trapdoor of that commitment in Stage 4. To show that $\langle C_t, R_t \rangle$ is CCA-secure w.r.t. O, it essentially amounts to prove the following lemma.

Lemma 9. *For every $\mathcal{PPT}$ adversary A, it holds that*

$$\left\{\mathsf{hyb}_0(A, n, z)\right\}_{n \in N, z \in \{0,1\}^*} \approx \left\{\mathsf{hyb}_{\ell(n)+4}(A, n, z)\right\}_{n \in N, z \in \{0,1\}^*}$$

Furthermore, for every function $i : N \to N$, the probability that A cheats in $H_{i(n)}$ is negligible.

Proof. We show that the lemma follows from the following two claims:

Claim 8. *For every $\mathcal{PPT}$ adversary A, the probability that A cheats in H_0 is negligible.*

Claim 9. *For every $\mathcal{PPT}$ adversary A, and every function $i : N \to N$, it holds that, if the probability that A cheats in $H_{i(n)}$ is negligible, then it holds that:*

- $\left\{\mathsf{hyb}_{i(n)}(A, n, z)\right\}_{n \in N, z \in \{0,1\}^*} \approx \left\{\mathsf{hyb}_{i(n)+1}(A, n, z)\right\}_{n \in N, z \in \{0,1\}^*}$
- *The probability that A cheats in $H_{i(n)+1}$ is negligible.*

Assume for contradiction that the lemma is false, that is, there exists an adversary A such that

a) *Either* there exists a distinguisher D and a polynomial p_1, such that, for infinitely many $n \in N$ and $z \in \{0, 1\}^*$, it holds that D distinguishes $\mathsf{hyb}_0(A, n, z)$ and $\mathsf{hyb}_{\ell+4}(A, n, z)$ with probability $1/p_1(n)$,

b) *Or* there exists a function i and a polynomial p_2, such that, A cheats in $H_{i(n)}$ with probability greater than $1/p_2(n)$.

Then consider the following two possible scenarios:

Condition b) holds. In this case, by Claim 8, there must exist a j such that the probability of A cheating jumps from being negligible to non-negligible. Formally, there exists a function j, a negligible function μ' and a polynomial q, such that for infinitely many n and z, A cheats in $H_{j(n)}$ with negligible probability $\mu'(n)$, but cheats in $H_{j(n)+1}$ with much higher probability $\mu'(n) + 1/q(n)$.

Assume for contradiction that for every function j, it holds that either the probability of cheating in $H_{j(n)}$ is non-negligible, or it is, but the probability of cheating in $H_{j(n)+1}$ is non-negligible. By Claim 8, the probability of cheating in H_0 is negligible. Thus by our hypothesis, it follows from a hybrid argument that the probability that A cheats in all hybrids are negligible, which contradicts with condition b).

Otherwise, condition b) does not hold, but condition a) holds. In this case, for every function i, the probability that A cheats in $H_{i(n)}$ is negligible, and for infinitely many n and z, D distinguishes $\mathsf{hyb}_0(A, n, z)$ and $\mathsf{hyb}_{\ell+4}(A, n, z)$ with probability $1/p_1(n)$. This implies there exists a function j, such that, the probability that A cheats in $H_{j(n)}$ is negligible and D distinguishes $\mathsf{hyb}_{j(n)}(A, n, z)$ and $\mathsf{hyb}_{j(n)+1}(A, n, z)$ with probability at least $1/p_1(n)(\ell(n) + 4)$.

Both of the above cases contradicts with Claim 9; we thus conclude the lemma.

□

Since hybrid H_0 is identical to the real experiment IND_0, Lemma 9 shows that the view of the adversary in IND_0 is indistinguishable to that in $H_{\ell+4}$. Similarly, we can consider another sequence of hybrids H'_0 to $H'_{\ell+4}$ starting from IND_1; it follows from the same proof for showing the indistinguishability of IND_0 and $H_{\ell+4}$ that the view of A in IND_1 is indistinguishable to that in $H'_{\ell+4}$. Since in both $H_{\ell+4}$ and $H'_{\ell+4}$, the left interaction is simulated completely independently of the two challenge messages chosen by the adversary, the two hybrids $H_{\ell+4}$ and $H'_{\ell+4}$ are identical. Therefore, we obtain that the views of A in IND_0 and IND_1 are indistinguishable, and thus $\langle C_t, R_t \rangle$ is CCA-secure w.r.t. $\mathcal{O}$. Next we complete the proof by providing a formal proof of Claim 8 and 9.

Proof of Claim 8 At a high-level, Claim 8 essentially follows from the one-wayness of the one-way function f (and the special-soundness of the Stage 5 $\mathcal{WISSP}$ proof): If a computationally bounded adversary can commit to a "trapdoor" in Stage 4 of the commitment, then by the special-soundness of Stage 5, we can extract out a "trapdoor", which is a pre-image of the Stage 1 message, and thus violates the one-wayness of function f. However, this argument does not go through directly, since the adversary A in hybrid H_0 has access to the oracle O, which is computationally unbounded. To resolve this, we show that the oracle O can be simulated efficiently via rewindings. By Proposition 8, when l is set to $\omega(\log n)$, there exists a conforming concurrent extraction strategy E for $\langle C_t, R_t \rangle$. Fix any adversary A. Consider the following two experiments.

Experiment EXP_0**:** This experiment emulates an execution of H_0, by viewing A together with the left committer $\hat{C}$ as a concurrent committer $\tilde{A}$ of $\langle C_t, R_t \rangle$ (that receives some exponential time help to obtain the decommitment of each commitment), and applying E on the concurrent committer.

More formally, let $\tilde{O}$ be a decommitment oracle of com that on input the transcript of a com commitment, return the unique decommitment if there is any, and $\perp$ otherwise. Consider a machine $\tilde{A}$ that with access to $\tilde{O}$, externally acts as a committer of $\langle C_t, R_t \rangle$ in many concurrent interactions, and internally emulates an execution of H_0 with A as follows: It emulates messages in the left interaction for A by running the honest committer strategy; it emulates the right interactions by first forwarding commitments of $\langle C_t, R_t \rangle$ externally, and whenever A completes a right commitment successfully and expects a decommitment from O, $\tilde{A}$ queries the oracle $\tilde{O}$ on the transcript of the com commitment in Stage 2 of that commitment, and returns to A the answer from $\tilde{O}$; finally, $\tilde{A}$ outputs the view of A. By construction, $\tilde{A}^{\tilde{O}}$ acts externally as a concurrent committer of $\langle C_t, R_t \rangle$ and emulates internally the view of A perfectly. Experiment EXP_0 invokes the concurrent extraction strategy E over machine $\tilde{A}^{\tilde{O}}$, that is, executing $E^{\tilde{A}^{\tilde{O}}(1^n,z,*)}(1^n, m)$ (where $m = m(n)$ is the maximum number of right interactions that A opens up), and outputs what E outputs (i.e., an emulated view of $\tilde{A}^{\tilde{O}}$).

Experiment EXP_1**:** This experiment proceeds identically to EXP_0, except that it emulates the decommitment oracle $\tilde{O}$ for $\tilde{A}$ using the witnesses that E extracts during its execution.

Recall that during the execution $E^{\tilde{A}^{\tilde{O}}(1^n,z,*)}(1^n, m)$, whenever E makes a query Q to $\tilde{A}^{\tilde{O}}$, for every accepting commitment of $\langle C_t, R_t \rangle$ in Q, E outputs on its special output tape a witness of the statement proved in Stage 6 of that commitment[1]. On the other hand, when $\tilde{A}^{\tilde{O}}$ is invoked with a query Q, for every accepting commitment of $\langle C_t, R_t \rangle$ in Q, it queries $\tilde{O}$ in order to obtain a decommitment; here, experiment EXP_1 emulates $\tilde{O}$ by returning the appropriate witness that E outputs on its special output tape.

Intuitively, both experiments EXP_0 and EXP_1 uses the concurrent extraction strategy E to emulate the view of A in hybrid H_0. Since E is conforming, it runs internally many threads of executions with A (incorporated in $\tilde{A}$) and may rewind A in order to extract witnesses; in each rewinding, messages from the right receiver are emulated honestly and independently (by E), as well as that from the left committer (by $\tilde{A}$). The only difference between EXP_0 and EXP_1 is that in the former, the decommitments of the right commitments are obtained using brute force search (with help from $\tilde{O}$), whereas in the latter, they are emulated using the witnesses extracted by E. Let $\mathsf{EXP}_b(A, n, z)$ describe emulated view of A in EXP_b (formally, it is the view of A embedded in the view of $\tilde{A}^{\tilde{O}}$ output by E). By construction and the statistical simulation property of E, the output of $E^{\tilde{A}^{\tilde{O}}}$ is statistically close the view of $\tilde{A}^{\tilde{O}}$. Then since $\tilde{A}^{\tilde{O}}$ emulates the real view of A in H_0 perfectly, we have $\{\mathsf{EXP}_0(A, n, z)\}$ and $\{\mathsf{hyb}_0(A, n, z)\}$ are statistically close. Furthermore, we show below in Subclaim 6 that the probability of A cheating in the view emulated by EXP_1 is negligible; in fact Subclaim 6

[1]More accurately, E outputs a witness for every block in Stage 6 of the commitment; since all blocks have the same statement, it suffices to consider one of the witnesses as the witness of the statement proved in Stage 6.

proves something even stronger that the probability of A cheating in any thread emulated by E in EXP_1 is negligible. In Subclaim 7, we show that the emulated view of A in EXP_0 and EXP_1 are statistically close. Therefore, we obtain that the probability that A cheats in the view emulated by EXP_0 is negligible. Thus the probability of cheating in H_0 is also negligible.

Subclaim 6. *The probability that A cheats in any thread in the emulation by E in* EXP_1 *is negligible.*

Subclaim 7. *Ensembles* $\{\mathsf{EXP}_0(A, n, z)\}_{n\in N, z\in\{0,1\}^*}$ *and* $\{\mathsf{EXP}_1(A, n, z)\}_{n\in N, z\in\{0,1\}^*}$ *are statistically close.*

Proof of Subclaim 6. Assume for contradiction that there is a polynomial p, such that, for infinitely many $n \in N$ and $z \in \{0, 1\}^*$, A cheats in some thread in EXP_1 with probability $1/p(n)$. Then we construct another $\mathcal{PPT}$ machine B that can invert the one-way function f.

Towards this, we first construct a $\mathcal{PPT}$ machine B' that on input a value y (together with n, z) gives a $\mathcal{WISSP}$ proof of a statement x, such that, with high probability, the only valid witnesses of x are pre-images of y through f. Then machine B can be constructed by incorporating B' and extracting a witness from the $\mathcal{WISSP}$ proof; by the special-soundness property, the extracted witness must be a pre-image of y, and thus B violates the one-wayness of f.

Machine B' proceeds identically to EXP_1, except the following: It picks at random a thread and a commitment in that thread; in that commitment, it feeds A the value y in Stage 1, and forwards the statement and messages of the $\mathcal{WISSP}$ proof in Stage 5 externally. Since B' emulates EXP_1 perfectly, it follows from our hypothesis that the probability that A cheats in some thread in B'

is $1/p(n)$. Then, since there are at most a polynomial number of threads and a polynomial number of commitments in each thread, with non-negligible probability, B' guesses correctly the commitment in which A cheats. In that commitment, A commits to a "trapdoor" in Stage 4; as a result, the only valid witnesses of the Stage 5 $\mathcal{WISSP}$ proof are pre-images of y. Therefore, with non-negligible probability, B' gives a $\mathcal{WISSP}$ proof whose witnesses are pre-images of y. $\square$

Proof of Subclaim 7. The only difference between EXP_0 and EXP_1 lies in how the decommitments of the right commitments are emulated. Therefore, to show that $\{\mathsf{EXP}_1(A, n, z)\}$ is statistically close to $\{\mathsf{EXP}_0(A, n, z)\}$, it suffices to show that EXP_1 emulates the decommitments in a way statistically close to that in EXP_0. That is, it suffices to show that in every thread (in emulation by E) in EXP_1, whenever a right commitment completes successfully, E already extracts a witness and the witness is a valid decommitment. By Subclaim 6, A (almost) never cheats in any thread in EXP_1. Therefore, if E succeeds in extracting a witness for a right commitment, the witness must be a valid decommitment. Thus it boils down to show that E always extracts successfully in EXP_0^1.

By the concurrent extraction property, the extractor E (almost) always extracts successfully in EXP_0. Assume for contradiction that this is not the case in EXP_1, that is, in some thread, when a right commitment completes successfully, E fails to output a valid witness for it. Consider the first commitment for which this happens, it follows from the argument above that, before this commitment completes, EXP_1 emulates all the decommitments (almost) perfectly, and thus the view of E in EXP_1 is (almost) identical to that in EXP_0. Then the probability that E extracts a witness for that commitment in EXP_1 is (almost) the same as that in EXP_0 which is overwhelming. This gives a contradiction. Therefore, we conclude the subclaim. □

Proof of Claim 9 Fix any adversary A and any $i = i(n)$. We want to show that if the probability that A cheats in H_i is negligible, then (1) the views of A in H_i and H_{i+1} are indistinguishable and (2) the probability that A cheats in H_{i+1} is also negligible. Below we first prove the second condition, and then show that following almost the same proof the first condition also holds.

PROVING (2)—*The probability that A cheats in H_{i+1} is negligible.*
Recall that hybrid H_{i+1} proceeds almost identically to H_i, except that it simulates a small "part" of the left interaction in a different but still *indistinguishable* way. By construction, the "part" where H_i and H_{i+1} differ consists of either a commitment of $\langle \tilde{C}, \tilde{R} \rangle$ (when $i = 1$ or $i = \ell + 2$) or a "short" sub-protocol with at most α rounds (otherwise). Then, provided that the decommitment oracle O in H_i and H_{i+1} can be emulated efficiently, it essentially follows from respectively the non-malleability and α-robustness of Stage 4 that the probability A cheats in H_i and H_{i+1} differ by at most a negligible amount, and hence the second condition holds. More precisely, let $\mathsf{part}(i)$ denote the set of left-interaction messages where H_i and H_{i+1} differ. Assume for contradiction that there exists a polynomial p such that for infinitely many $n \in N$ and $z \in \{0,1\}^*$, A cheats in H_{i+1} with probability $1/p(n)$. We show how to derive a contradiction by relying on the α-robust non-malleability of $\langle \tilde{C}, \tilde{R} \rangle$ and a robust concurrent extraction strategy of $\langle C_t, R_t \rangle$.

By our hypothesis, there must exists a $j = j(n)$ (in $[m(n)]$) such that in H_{i+1}, A cheats in the first j right interactions with a negligible probability $\mu(n)$, but cheats in the first $j + 1$ right interactions with a polynomial probability $1/p'(n)$, (where the right interactions are ordered by the completion of their Stage 4 commitment.) Let $\tilde{H}_i$ and $\tilde{H}_{i+1}$ denote respectively the executions of H_i and H_{i+1} that are cut-off immediately after A completes Stage 4 of the j^{th} right interaction. Consider the following two experiments that emulates the executions of $\tilde{H}_i$ and $\tilde{H}_{i+1}$ using a robust concurrent extraction strategy. Recall that protocol $\langle C_t, R_t \rangle$ contains λ blocks of special-sound proofs, where $\lambda = \max(2k + 1, t)$ is greater than $2k$. Therefore by Proposition 8 and 10, there exists a conforming $2k$-robust concurrent extraction strategy E for $\langle C_t, R_t \rangle$.

Experiment $\mathsf{EXP}^i_0[u]$: This experiment emulates an execution of $\tilde{H}_i$ or $\tilde{H}_{i+1}$, by viewing A together with the left committer $\hat{C}$ as a concurrent committer $\tilde{A}$ of $\langle C_t, R_t \rangle$ (as in EXP_0), that additionally interacts with a $2k$*-round* machine B, and then applying E to emulate the interaction between $\tilde{A}$ and B.

Formally, $\tilde{A}$ (as $\tilde{A}$ in EXP_0,) internally emulates an execution of $\tilde{H}_i$ with A, by forwarding the right commitments from A externally to an honest receiver $\hat{R}$, obtaining decommitment information through the decommitment oracle $\tilde{O}$ of com, and emulating messages in the left interaction for A as the left committer in $\tilde{H}_i$ does; in the end, $\tilde{A}$ outputs the emulated view of A. Furthermore, $\tilde{A}$ (different from $\tilde{A}$ in EXP_0,) cuts off the execution immediately after the completion of Stage 4 of the j^{th} right interaction, and it additionally forwards messages in $\mathsf{part}(i)$ of the left interaction and that in Stage 4 of the j^{th} right interaction externally to machine B. Machine B, on input $u \in \{0, 1\}$, responds to messages in $\mathsf{part}(i)$ of the left interaction as the left committer in H_{i+u} does, and to that in Stage 4 of the right interaction as the honest receiver $\tilde{R}$ of $\langle \tilde{C}, \tilde{R} \rangle$ does; finally, it outputs the transcript of messages it sends/receives. By construction, $\tilde{A}^{\tilde{O}}$ acts externally as a concurrent committer of $\langle C_t, R_t\rangle$ and interacts with B in at most $2k$ rounds; internally, it emulates internally the view of A in $\tilde{H}_i$ or $\tilde{H}_{i+1}$ perfectly, depending on the input u of B.

Experiment $\mathsf{EXP}^i_0[u]$ emulates an interaction between $B(u)$ and $\tilde{A}^{\tilde{O}}$ by running $B(u)$ with $E^{\tilde{A}^{\tilde{O}}}$. Since $\tilde{A}$ outputs an emulated view $\mathcal{V}$ of A and forwards Stage 4 of the j^{th} right interaction in $\mathcal{V}$ to B, so does $E^{\tilde{A}^{\tilde{O}}}$.

Experiment $\mathsf{EXP}^i_1[u]$**:** This experiment proceeds identically to $\mathsf{EXP}^i_0[u]$, except that it emulates the decommitment oracle $\tilde{O}$ for $\tilde{A}$ using the witnesses that E extracts during its execution as in EXP_1.

Let $\mathsf{EXP}^i_b[u](A, n, z)$ describe the joint distribution of the output of E together with the value it commits to $B(u)$ using $\langle\tilde{C}, \tilde{R}\rangle$ in $\mathsf{EXP}^i_b[u]$. By the statistical simulation property of E, the joint output of E and B in $\mathsf{EXP}^i_0[u]$ is statistically close to that of $\tilde{A}^{\tilde{O}}$ and B. Since the output of $\tilde{A}^{\tilde{O}}$ is an emulated view of A that distributes identically to the real view of A in $\tilde{H}_{i+u}$, and the outputs of B is the transcript of messages it sends/receives, which decides the value that $\tilde{A}$ commits to using $\langle\tilde{C}, \tilde{R}\rangle$, we have that $\mathsf{EXP}^i_0[u](A, n, z)$ is statistically close to the emulated view of A by $\tilde{A}^{\tilde{O}}$ and the value it commits to B. Since the commitment that $\tilde{A}$ sends to B is forwarded from Stage 4 of the j^{th} interaction in the emulated view of A, we derive that $\mathsf{EXP}^i_0[u](A, n, z)$ is statistically close to $\tilde{\mathsf{hyb}}_{i+u}(A, n, z)$, where $\tilde{\mathsf{hyb}}_{i+u}(A, n, z)$ denotes the view of A in $\tilde{H}_{i+u}$ and the value it commits to in Stage 4 of right interaction j. Formally

Subclaim 8. *For every $u \in \{0, 1\}$, it holds that the ensembles $\left\{\tilde{\mathsf{hyb}}_{i+u}(A, n, z)\right\}_{n\in N, z\in\{0,1\}^*}$ and $\left\{\mathsf{EXP}^i_0[u](A, n, z)\right\}_{n\in N, z\in\{0,1\}^*}$ are statistically close.*

Furthermore, below we show that the emulated view of A together with the value E commits to in experiment $\mathsf{EXP}^i_0[u]$ are statistically close to that in $\mathsf{EXP}^i_1[u]$.

Subclaim 9. *For every u, it holds that the two ensembles $\left\{\mathsf{EXP}^i_0[u](A,n,z)\right\}_{n\in N, z\in\{0,1\}^*}$ and $\left\{\mathsf{EXP}^i_1[u](A,n,z)\right\}_{n\in N, z\in\{0,1\}^*}$ are statistically close.*

Finally, since in $\mathsf{EXP}^i_1[u]$ the decommitment oracle $\tilde{O}$ is emulated efficiently using the witnesses extracted by E, machine E with black-box access to $\tilde{A}$ runs in polynomial time. Therefore, it follows from the α-robust non-malleability of $\langle \tilde{C}, \tilde{R}\rangle$ that the view of $E^{\tilde{A}}$ and the value it commits to using $\langle \tilde{C}, \tilde{R}\rangle$ (to B), after receiving messages in $\mathsf{part}(i)$ of the left interaction in $\mathsf{EXP}^i_1[0]$ and $\mathsf{EXP}^i_1[1]$ (simulated indistinguishably by $B(0)$ and $B(1)$) are indistinguishable. That is,

Subclaim 10. *Ensembles $\left\{\mathsf{EXP}^i_1[0](A,n,z)\right\}_{n\in N, z\in\{0,1\}^*}$ and $\left\{\mathsf{EXP}^i_1[1](A,n,z)\right\}_{n\in N, z\in\{0,1\}^*}$ are computationally indistinguishable.*

Therefore, it follows from a hybrid argument that $\tilde{\mathsf{hyb}}_i(A,n,z)$ and $\tilde{\mathsf{hyb}}_{i+1}(A,n,z)$ are computationally indistinguishable, implying that the value A commits to in the j^{th} right interaction is indistinguishable in $\tilde{H}_i$ and $\tilde{H}_{i+1}$. This contradicts with our hypothesis and we conclude that the probability that A cheats in H_{i+1} is negligible. Below for completeness, we provide the formal proof of Subclaim 9.

Proof of Subclaim 9. Fix any u. This subclaim essentially follows from the fact that A (almost) never cheats in $\tilde{H}_i$ and $\tilde{H}_{i+1}$. (Note that, though A may commit to a trapdoor in the j^{th} right interaction in $\tilde{H}_{i+1}$, this right interaction is never completed successfully, since the execution is cut off immediately after Stage 4 of the j^{th} right interaction.) Then we claim that in experiment $\mathsf{EXP}^i_0[u]$, the probability that A cheats in any accepting commitment in any thread emulated by E is negligible. This follows from the fact that E is conforming, meaning that in every thread, it emulates messages to A perfectly as in $\tilde{H}_{i+u}$; then if in $\mathsf{EXP}^{i}_{0}[u]$, E, with probability p, can generate a thread in which A cheats in an accepting commitment, then with at least probability p/w, such a thread appears in $\tilde{H}_{i+u}$, where w is an upper bound on the number of threads that E generates. Therefore the probability that A cheats in any accepting commitment in $\mathsf{EXP}^{i}_{0}[u]$ is negligible.

Furthermore, it follows from the concurrent extraction property of E that, E always succeeds in extracting a valid witness for every accepting right interaction in $\mathsf{EXP}^{i}_{0}[u]$. Then it follows from the fact that A never cheats in any accepting commitments in $\mathsf{EXP}^{i}_{0}[u]$, we have that the witnesses extracted by E are indeed the unique decommitments of these accepting commitments. Therefore, it is equivalent to replace the decommitment from the oracle $\tilde{O}$ used in $\mathsf{EXP}^{i}_{0}[u]$ with the witnesses extracted by E used in $\mathsf{EXP}^{i}_{1}[u]$. Therefore $\left\{\mathsf{EXP}^{i}_{0}[u](A,n,z)\right\}$ and $\left\{\mathsf{EXP}^{i}_{1}[u](A,n,z)\right\}$ are statistically close. □

PROVING (1)—*The view of A is indistinguishable in H_i and H_{i+1}.*
Recall that H_i and H_{i+1} only differ at $\mathsf{part}(i)$ of the left interaction. Provided that the decommitment oracle O can be emulated efficiently without violating the indistinguishability of $\mathsf{part}(i)$ in H_i and H_{i+1}, then the indistinguishability of the view of A in H_i and H_{i+1} would simply follow from the indistinguishability of $\mathsf{part}(i)$. Towards this, we follow the same proof idea as above, since $\mathsf{part}(i)$ of the left interaction consists of at most k messages, we can use the same conforming $2k$-robust concurrent extraction strategy E as above to simulate the decommitment oracle O, and show the correctness of the emulation by considering the following experiments.

Experiment $\hat{\mathsf{EXP}}^i_0[u]$: This experiment proceeds identically to $\mathsf{EXP}^i_0[u]$ except the following: Consider machines $\hat{A}$ and $\hat{B}$ that proceed identically to $\tilde{A}$ and B, except that $\hat{A}$ only forwards $\mathsf{part}(i)$ of the left interaction (without Stage 4 of the j^{th} right interaction) in the emulated view of A to $\hat{B}$. Then $\hat{\mathsf{EXP}}^i_0[u]$ emulates the execution between $\hat{B}$ and $\hat{A}$ by running $\hat{B}$ with $E^{\hat{A}^{\tilde{O}}}$ as $\mathsf{EXP}^i_0[u]$ does.

Experiment $\hat{\mathsf{EXP}}^i_1[u]$: Similar to $\mathsf{EXP}^i_1[u]$, this experiment emulates the execution of $\hat{\mathsf{EXP}}^i_0[u]$, by replacing answers from the decommitment oracle $\tilde{O}$ with the appropriate witnesses that E extracts during its execution.

Let $\hat{\mathsf{EXP}}^i_b[u](A, n, z)$ describe the emulated view of A in $\hat{\mathsf{EXP}}^i_b[u]$. It follows from the statistical simulation property of E that $\hat{\mathsf{EXP}}^i_b[u](A, n, z)$ (i.e., the output of E) is statistically close the emulated view of A in $\hat{A}$ (i.e., the output of $\hat{A}$), which in turn, by construction, is identical to the real view of A in H_{i+u}. That is,

Subclaim 11. *For every $u \in \{0, 1\}$, it holds that the ensembles $\{\mathsf{hyb}_{i+u}(A, n, z)\}_{n\in N, z\in\{0,1\}^*}$ and $\left\{\hat{\mathsf{EXP}}^i_0[u](A, n, z)\right\}_{n\in N, z\in\{0,1\}^*}$ are statistically close.*

Then, towards establishing the indistinguishability of the view of A in H_i and H_{i+1}, it suffices to show the same claims as Subclaim 9 and 10 with respect to $\hat{\mathsf{EXP}}^i_b[u](A, n, z)$.

Subclaim 12. *For every u, it holds that the two ensembles $\left\{\hat{\mathsf{EXP}}^i_0[u](A, n, z)\right\}_{n\in N, z\in\{0,1\}^*}$ and $\left\{\hat{\mathsf{EXP}}^i_1[u](A, n, z)\right\}_{n\in N, z\in\{0,1\}^*}$ are statistically close.*

Subclaim 13. *Ensembles $\left\{\hat{\mathsf{EXP}}^i_1[0](A, n, z)\right\}_{n\in N, z\in\{0,1\}^*}$ and $\left\{\hat{\mathsf{EXP}}^i_1[1](A, n, z)\right\}_{n\in N, z\in\{0,1\}^*}$ are computationally indistinguishable.*

Subclaim 12 essentially follows from the same proof as in Subclaim 9, except that now we rely on the fact that A never cheats in H_{i+1} (proved in the previous step) to establish that the witnesses extracted by E must be valid decommitments in $\hat{\mathsf{EXP}}^i_0[1]$ (whereas in the proof of Subclaim 9, we rely on the hypothesis that A never cheats in the first j right interactions). Subclaim 13 follows from a similar argument as in Subclaim 10: Since the decommitment oracle is emulated efficiently in $\hat{\mathsf{EXP}}^i_1[u]$, the views of E after receiving $\mathsf{part}(i)$ of the left interaction in $\hat{\mathsf{EXP}}^i_1[u]$ and $\hat{\mathsf{EXP}}^i_1[u]$ (simulated indistinguishably by $\hat{B}(0)$ and $\hat{B}(1)$), are indistinguishable; therefore so are the outputs $\left\{\hat{\mathsf{EXP}}^i_1[0](A,n,z)\right\}$ and $\left\{\hat{\mathsf{EXP}}^i_1[0](A,n,z)\right\}$ of E. Finally, by a hybrid argument, we conclude that the view of A in H_i and H_{i+1} is indistinguishable.

Proof of *t*-Robustness

In this section, we extend the proof in the last section to show that $\langle C_t, R_t\rangle$ is also *robust* CCA-secure w.r.t. $\mathcal{O}$. Towards this, we need to show that there exists a simulator S, such that, for every $\mathcal{PPT}$ adversary A and every $\mathcal{PPT}$ t-round ITM B, the joint output in the interaction between B and A with access to $\mathcal{O}$ is indistinguishable from that between B and S^A.

The construction of the simulator S is very similar to the construction of experiment $\hat{\mathsf{EXP}}^i_1[u]$ in the last section. Recall that protocol $\langle C_t, R_t\rangle$ contains λ blocks of special-sound proofs, where $\lambda = \max(2k+1, t)$ is greater than t. Therefore by Proposition 8 and 10, there exists a conforming t-robust concurrent extraction strategy E for $\langle C_t, R_t\rangle$. In analogy to the proof in the last section, we first construct a simulator S' that is similar to experiment $\hat{\mathsf{EXP}}^i_0[u]$ and may take exponential time. On a high-level, S', with black-box oracle access to an adversary A, externally interacts with an arbitrary t-round ITM B, and internally simulates an execution between B and $A^{\mathcal{O}}$ as follows: Consider machine $\bar{A}$ (similar to $\hat{A}$

and $\tilde{A}$) that when interacting with B on the left and the honest receiver $\hat{R}$ on the right, internally emulates an execution with A by forwarding messages from B to A, messages from A belonging to right commitments of $\langle C_t, R_t \rangle$ to $\hat{R}$, and emulating the decommitment messages from O using a decommitment oracle $\tilde{O}$ of com; finally $\bar{A}$ returns the output of A; then S' simply invokes the t-robust conforming extraction strategy E over $\bar{A}^{\tilde{O}}$ and outputs whatever E outputs (during the execution of E, S' emulates the oracle $\bar{A}$ for E using its own oracle A). It follows from the statistical simulation property of E that the joint output in the interaction between A^O and B is statistically close to that in the interaction between S' and B.

The only problem is that S' runs in exponential time since the decommitment oracle $\tilde{O}$ does. Then using the same idea as in $\hat{\mathsf{EXP}}^i_1[u]$, the actual simulator S proceeds identically to S' except that it emulates $\tilde{O}$ using the witnesses extracted by E. To show that the joint output in the interaction between S and B distributes correctly, it suffices to show that it is statistically close to the joint output in the interaction between S' and B. As in the proof of Subclaim 9 and 12, since the only difference between S and S' lies in how the decommitment messages are emulated, this boils down to show that the witnesses that E extracts in S' are always valid decommitment. It follows from the same proof as in Claim 8 that the probability that A with access to O cheats in an interaction with B is negligible. Therefore, since E is conforming, the probability that A cheats in any thread emulated by E in S' is also negligible. Hence, the witnesses that E extracts in S' must be valid decommitment, and S is a valid simulator.

Remark 4. *The protocol $\langle C_t, R_t \rangle$ described in Section 5.1 uses a statistically binding commitment scheme with unique decommitment. It can be modified to work with any arbitrary statistically binding commitment com as follows: In Stage 2, the committer* *commits to the value v using* com *once—referred to as the first commitment—and additionally commits to a decommitment of the first commitment using* com*—referred to as the second commitment. In the rest of the protocol, use the decommitment of the first commitment decided uniquely by the statistical binding property of the second commitment as the unique decommitment of Stage 2. It follows using almost the same proofs as above that the modified protocol is a t-robust CCA-secure commitment scheme.*

5.2.3 Constant-Round Robust CCA-Secure Protocol w.r.t. $\mathcal{PQT}$

A tag-based efficiently verifiable commitment is t-robust CCA-secure w.r.t. quasi-polynomial time Turing machines, or $\mathcal{PQT}$ if it satisfies the Definitions 23 and 24 in chapter 4 with the notion of feasible computation changed from $\mathcal{PPT}$ to $\mathcal{PQT}$. Similarly, for other cryptographic primitives, we can also consider a version of the primitive w.r.t. $\mathcal{PQT}$.

Let $\langle \bar{C}_t, \bar{R}_t \rangle$ be a protocol that proceeds identically to $\langle C_t, R_t \rangle$ except that all the primitives used in $\langle C_t, R_t \rangle$ (including one-way functions, statistically binding commitments, $\mathcal{ZK}$ proof, non-malleable commitments and $\mathcal{WISSP}$ proofs) are replaced with corresponding primitives secure w.r.t. quasi-polynomial time adversaries. It follows from Proposition 9 and syntactically the same proof as Proposition 11 that the following holds.

Proposition 12. *Set the polynomial l to 3. Then $\langle \bar{C}_t, \bar{R}_t \rangle$ is t-robust CCA secure w.r.t. $\mathcal{O}$ for quasi-polynomial time adversaries.*

Assuming the existence of one-way functions hard for quasi-polynomial time adversaries, there exist constant-round statistically binding commitments, $\mathcal{ZK}$

proof, non-malleable commitments and $\mathcal{WISSP}$ proofs hard for quasi-polynomial time adversaries. (A $\mathcal{ZK}$ proof secure against quasi-polynomial time adversaries can be obtained by instantiating the 3-round GMW protocol [GMW91] with a statistically binding commitment that is hiding for quasi-polynomial time adversaries and repeating it in parallel for $\omega(\log n)$ times). Instantiating $\langle \bar{C}_t, \bar{R}_t \rangle$ with these constant-round primitives, we obtain a $O(t)$-round t-robust CCA secure commitment scheme based on one-way functions hard for quasi-polynomial time. Therefore, for every constant t, we obtain a constant-round t-robust CCA-secure commitment scheme w.r.t. $\mathcal{PQT}$ based on one-way functions hard for $\mathcal{PQT}$. This concludes Theorem 12

5.3 4-Robust Commitment Has $\tilde{\Omega}(\log n)$ Rounds

In this section, we show the following theorem.

Theorem 14. *Any efficiently verifiable commitment scheme that is 4-robust has $\tilde{\Omega}(logn)$ rounds.*

Therefore our $\omega(\log n)$-round CCA secure commitment scheme that is t-robust for any constant t has optimal round complexity (up to a logarithmic factor).

Proof. Towards the theorem, we show that the existence of a k-round 4-robust commitment scheme implies the existence of a $(k + 4)$-round black-box concurrent $\mathcal{ZK}$ argument. Then the theorem simply follows from the lower bound that black-box concurrent $\mathcal{ZK}$ protocols require $\tilde{\Omega}(\log n)$ rounds [CKPR01].

Given a k-round commitment scheme $\langle C, R \rangle$ that is 4-robust w.r.t. a decommitment oracle O, we construct the black-box concurrent $\mathcal{ZK}$ argument $\langle P, V \rangle$ as follows. To prove a statement $x \in \{0,1\}^n$, the Prover on private input a witness w and common input 1^n, interacts with the Verifier in the following two stages:

Preamble: the Verifier commits to a random value r using $\langle C, R \rangle$. The Prover aborts if the commitment is not accepting. Let $\mathcal{T}$ be the transcript of the commitment.

Proof Phase: the Prover proves using a 4-round $\mathcal{WISSP}$ proof system the statement that either x is true or $\mathcal{T}$ is a valid commitment of $\langle C, R \rangle$.

By construction, the protocol $\langle P, V \rangle$ consists of $k+4$ messages. It follows from standard technique that this protocol is complete and sound. To show that the protocol is also black-box concurrent $\mathcal{ZK}$, we need to establish that there exists a simulator S, such that for every malicious verifier V^*, inputs n, x and z, the simulator on input 1^n and x, with black-box access to $V^*(z)$, outputs a view that is indistinguishable from the real view of V^*. Towards this we first construct a simulator S' that has access to the decommitment oracle O, and then remove the use of the oracle by relying on the 4-robustness property of $\langle C, R \rangle$.

Simulator S' with black box access to a verifier V^* and O, internally simulates an execution with V^*, by forwarding all the commitments of $\langle C, R \rangle$ in the Preamble Phase from V^* to O, which returns a decommitment for every accepting commitment, and then simulating the $\mathcal{WISSP}$ proofs in the Proof Stage using the decommitment as a "trapdoor". Finally S' outputs the view of V^* in the simulation. To show that the output view of S' distributes correctly, consider the following sequence of hybrids H_0 to $H_{m(n)}$, where $m(n)$ is an upper bound on the total number of sessions that V^* opens up:

Hybrid H_i: H_0 proceeds identically to S', except that it only simulates the $\mathcal{WISSP}$

proofs in the first i interactions, using the decommitments returned by O; the rest of interactions are emulated using an honest witness.

Since the output of H_0 distributes identically to the real view of V^* and that of $H_{m(n)}$ distributes identically to the output of S'. To show the correctness of S', it suffices to show that the outputs of H_i and H_{i+1} are indistinguishable. Towards this, consider the following machines A and B: Machine A with oracle access to O, internally proceeds identically to H_i, except that, it forwards the statement x and messages of the $\mathcal{WISSP}$ proof in the $i+1^{\text{th}}$ interaction externally to B, together with the valid witness w and the trapdoor σ obtained from O; in the end, A outputs what H_i outputs; machine B on input b, acts as an honest prover of the $\mathcal{WISSP}$ proof using witness w if $b = 0$ and σ if $b = 1$. By construction, when the input of B is 0, A^O internally emulates an execution of H_i, and when the input is 1, it emulates an execution of H_{i+1}. Since machine B only interacts in four rounds, it follows from the 4-robustness of $\langle C, R\rangle$ that there exists a simulator C such that C with black box access to A, when interacting with $B(b)$, emulates the interaction between A^O and $B(b)$. That is, the joint output of C^A and $B(b)$ is indistinguishable to that of A^O and $B(b)$. Then since the outputs of C^A when interacting with B on input 0 or 1 are indistinguishable (by the $\mathcal{WI}$ property of the proof), the outputs of A^O when interacting with B on input 0 or 1 are also indistinguishable, which respectively distributes identically to the output of H_i and H_{i+1}. Thus we conclude that the outputs of H_i and H_{i+1} are indistinguishable, and the output of S' is correctly distributed.

Using S', we construct the actual simulator S that does not have access to the decommitment oracle O, again by replying on the 4-robustness of $\langle C, R\rangle$. Consider an protocol $\langle A, B\rangle$ where A and B does not interact with each other and simply output an empty string. Since the protocol $\langle A, B\rangle$ is empty, it follows from the 4-robustness of $\langle C, R\rangle$ that, there exists a simulator C, such that C with black box access to S'^{V^*} (when interacting with A) emulates the execution of $S'^{V^*,O}$ (when interacting with A). That is, the output of C, with black box access to S'^{V^*} is indistinguishable to the output of S'^{V^*} with access to O. Then the simulator S simply internally incorporates C and S', and emulates the execution of $C^{S'^{V*}}$, using its oracle access to V^*, and outputs what C outputs. Then since the output of S' distributes correctly, we have that the output of S also distributes correctly. Thus S is a valid simulator for $\langle P, V\rangle$ □

5.4 Improving Efficiency of UC with Super-Polynomial Helpers

Given our new constructions of robust CCA-secure commitment schemes, we immediately obtain more round-efficient protocols satisfying UC-security with super-polynomial time helpers.

By Theorem 12, there exists a $\omega(\log n)$-round commitment scheme $\langle C, R\rangle$ that is robust CCA-secure w.r.t. a decommitment oracle O, based on one-way functions. Then, let $\mathcal{H}'$ be a helper functionality that acts as the decommitment oracle O, as $\mathcal{H}$ in Figure 4.5 in Section 4.6. Then it follows from the same proof as in Section 4.6 that

Theorem 15. *Assume the existence of enhanced trapdoor permutations. Then for every well-formed functionality $\mathcal{F}$, there exists a $\omega(\log n)$-round protocol Π that $\mathcal{H}'$-EUC-emulates $\mathcal{F}$.*

Furthermore, since the construction in Section 4.6 actually only requires a

CCA-secure commitment scheme that is t-robust for a fixed constant t. By Theorem 13, there exists a constant-round t-robust CCA-secure commitment scheme $\langle \bar{C}_t, \bar{R}_t \rangle$, based on owe-way functions hard for $\mathcal{PQT}$. Then, let $\mathcal{H}''$ be a helper functionality acting as the decommiment oracle corresponding to the robust CCA-security property of $\langle \bar{C}_t, \bar{R}_t \rangle$. Again, it follows syntactically the same proof as in Section 4.6 that

Theorem 16. *Assume the existence of enhanced trapdoor permutations secure w.r.t. quasi-polynomial time adversaries. Then for every well-formed functionality $\mathcal{F}$, there exists a constant-round protocol Π that $\mathcal{H}''$-EUC-emulates $\mathcal{F}$.*

BIBLIOGRAPHY

[Bar01] Boaz Barak. How to go beyond the black-box simulation barrier. In *FOCS*, volume 0, pages 106–115, 2001.

[Bar02] Boaz Barak. Constant-round coin-tossing with a man in the middle or realizing the shared random string model. In *FOCS*, pages 345–355, Washington, DC, USA, 2002. IEEE Computer Society.

[BCC88] Gilles Brassard, David Chaum, and Claude Crépeau. Minimum disclosure proofs of knowledge. *J. Comput. Syst. Sci.*, 37(2):156–189, 1988.

[BCNP04] Boaz Barak, Ran Canetti, Jesper Buus Nielsen, and Rafael Pass. Universally composable protocols with relaxed set-up assumptions. In *FOCS*, pages 186–195, 2004.

[BG92] Mihir Bellare and Oded Goldreich. On defining proofs of knowledge. In *CRYPTO '92*, pages 390–420, 1992.

[Blu83] Manuel Blum. Coin flipping by telephone a protocol for solving impossible problems. *SIGACT News*, 15(1):23–27, 1983.

[Blu86] M. Blum. How to prove a theorem so no one else can claim it. *Proc. of the International Congress of Mathematicians*, pages 1444–1451, 1986.

[BM88] László Babai and Shlomo Moran. Arthur-merlin games: a randomized proof system, and a hierarchy of complexity class. *J. Comput. Syst. Sci.*, 36(2):254–276, 1988.

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

[BOGW88] Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In *STOC*, pages 1–10, 1988.

[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.

[Can00] Ran Canetti. Security and composition of multiparty cryptographic protocols. *Journal of Cryptology*, pages 143–202, 2000.

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

[Can04] Ran Canetti. Universally composable signature, certification, and authentication. In *CSFW*, pages 219–, 2004.

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

[CDS94] Ronald Cramer, Ivan Damgrd, and Berry Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In *CRYPTO*, pages 174–187, 1994.

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

[CGGM00] Ran Canetti, Oded Goldreich, Shafi Goldwasser, and Silvio Micali. Resettable zero-knowledge (extended abstract). In *STOC*, pages 235–244, 2000.

[CIO01] Giovanni Di Crescenzo, Yuval Ishai, and Rafail Ostrovsky. Universal service-providers for private information retrieval. *J. Cryptology*, 14(1):37–74, 2001.

[CKL03] Ran Canetti, Eyal Kushilevitz, and Yehuda Lindell. On the limitations of universally composable two-party computation without set-up assumptions. In *EUROCRYPT*, pages 68–86, 2003.

[CKOS01] Giovanni Di Crescenzo, Jonathan Katz, Rafail Ostrovsky, and Adam Smith. Efficient and non-interactive non-malleable commitment. In *EUROCRYPT*, pages 40–59, 2001.

[CKPR01] Ran Canetti, Joe Kilian, Erez Petrank, and Alon Rosen. Black-box concurrent zero-knowledge requires $\tilde{\omega}(\log n)$ rounds. In *STOC*, pages 570–579, 2001.

[CLOS02] Ran Canetti, Yehuda Lindell, Rafail Ostrovsky, and Amit Sahai.

Universally composable two-party and multi-party secure computation. In *STOC*, pages 494–503, 2002.

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

[CR87] Benny Chor and Michael O. Rabin. Achieving independence in logarithmic number of rounds. In *PODC*, pages 260–268, 1987.

[DDN91] Danny Dolev, Cynthia Dwork, and Moni Naor. Non-malleable cryptography (extended abstract). In *STOC*, pages 542–552, 1991.

[DDN00] Danny Dolev, Cynthia Dwork, and Moni Naor. Nonmalleable cryptography. *SIAM Journal on Computing*, 30(2):391–437, 2000.

[DG03] Ivan Damgrd and Jens Groth. Non-interactive and reusable non-malleable commitment schemes. In *STOC*, pages 426–437, 2003.

[DNS04] Cynthia Dwork, Moni Naor, and Amit Sahai. Concurrent zero-knowledge. *J. ACM*, 51(6):851–898, 2004.

[EGL85] Shimon Even, Oded Goldreich, and Abraham Lempel. A randomized protocol for signing contracts. *Commun. ACM*, 28(6):637–647, 1985.

[FF09] Marc Fischlin and Roger Fischlin. Efficient non-malleable commitment schemes. *J. Cryptology*, 22(4):530–571, 2009.

[FS90] Uriel Feige and Adi Shamir. Witness indistinguishable and witness hiding protocols. In *STOC*, pages 416–426, 1990.

[GGM86] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *J. ACM*, 33(4):792–807, 1986.

[GK96] Oded Goldreich and Ariel Kahan. How to construct constant-round zero-knowledge proof systems for NP. *Journal of Cryptology*, 9(3):167–190, 1996.

[GM84] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *J. Comput. Syst. Sci.*, 28(2):270–299, 1984.

[GMR88] Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM J. Comput.*, 17(2):281–308, 1988.

[GMR89] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on Computing*, 18(1):186–208, 1989.

[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.

[GMW91] Oded Goldreich, Silvio Micali, and Avi Wigderson. Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems. *J. ACM*, 38(3):690–728, 1991.

[GO94] Oded Goldreich and Yair Oren. Definitions and properties of zero-knowledge proof systems. *Journal of Cryptology*, 7:1–32, 1994.

[GO07] Jens Groth and Rafail Ostrovsky. Cryptography in the multi-string model. In *CRYPTO*, pages 323–341, 2007.

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

[Gol04] Oded Goldreich. *Foundations of Cryptography — Basic Applications*. Cambridge University Press, 2004.

[Goy11] Vipul Goyal. Constant round non-malleable protocols using one way functions. In *STOC*, pages 695–704, 2011.

[HILL99] Johan Hstad, Russell Impagliazzo, Leonid Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28:12–24, 1999.

[IL89] Russell Impagliazzo and Michael Luby. One-way functions are essential for complexity based cryptography (extended abstract). In *FOCS*, pages 230–235, 1989.

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

[Kat07] Jonathan Katz. Universally composable multi-party computation using tamper-proof hardware. In *EUROCRYPT*, pages 115–128, 2007.

[Kil92] Joe Kilian. A note on efficient zero-knowledge proofs and arguments (extended abstract). In *STOC*, pages 723–732, 1992.

[KLP05] Yael Tauman Kalai, Yehuda Lindell, and Manoj Prabhakaran. Concurrent general composition of secure protocols in the timing model. In *STOC*, pages 644–653, 2005.

[KOS03] Jonathan Katz, Rafail Ostrovsky, and Adam Smith. Round efficiency of multi-party computation with a dishonest majority. In *EUROCRYPT*, pages 578–595, 2003.

[KP01] Joe Kilian and Erez Petrank. Concurrent and resettable zero-knowledge in poly-loalgorithm rounds. In *STOC*, pages 560–569, 2001.

[Lin04] Yehuda Lindell. Lower bounds for concurrent self composition. In *TCC*, pages 203–222, 2004.

[LP09] Huijia Lin and Rafael Pass. Non-malleability amplification. In *STOC*, pages 189–198, 2009.

[LP11a] Huijia Lin and Rafael Pass. Concurrent non-malleable zero knowledge with adaptive inputs. In *TCC*, pages 274–292, 2011.

[LP11b] Huijia Lin and Rafael Pass. Constant-round non-malleable commitments from any one-way function. In *STOC*, pages 705–714, 2011.

[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.

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

[MOSV06] Daniele Micciancio, Shien Jin J Ong, Amit Sahai, and Salil Vadhan. Concurrent zero knowledge without complexity assumptions. *TCC*, pages 1–20, 2006.

[MPR06] Silvio Micali, Rafael Pass, and Alon Rosen. Input-indistinguishable computation. In *FOCS*, pages 367–378, 2006.

[Nao91] Moni Naor. Bit commitment using pseudorandomness. *Journal of Cryptology*, 4:151–158, 1991.

[NY89] Moni Naor and Moti Yung. Universal one-way hash functions and their cryptographic applications. In *STOC*, pages 33–43, 1989.

[Pas03a] Rafael Pass. Simulation in quasi-polynomial time, and its application to protocol composition. In *EUROCRYPT*, pages 160–176, 2003.

[Pas03b] Rafael Pass. Simulation in quasi-polynomial time, and its application to protocol composition. In *EUROCRYPT*, pages 160–176, 2003.

[Pas04] Rafael Pass. Bounded-concurrent secure multi-party computation with a dishonest majority. In *STOC*, pages 232–241, New York, NY, USA, 2004. ACM.

[PPV08] Omkant Pandey, Rafael Pass, and Vinod Vaikuntanathan. Adaptive one-way functions and applications. In *CRYPTO*, pages 57–74, 2008.

[PR03] Rafael Pass and Alon Rosen. Bounded-concurrent secure two-party computation in a constant number of rounds. In *FOCS*, pages 404–, 2003.

[PR05a] Rafael Pass and Alon Rosen. Concurrent non-malleable commitments. In *FOCS*, pages 563–572, 2005.

[PR05b] Rafael Pass and Alon Rosen. New and improved constructions of non-malleable cryptographic protocols. In *STOC*, pages 533–542, 2005.

[PRS02] Manoj Prabhakaran, Alon Rosen, and Amit Sahai. Concurrent zero knowledge with logarithmic round-complexity. In *FOCS*, pages 366–375, 2002.

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

[PTV08] Rafael Pass, Wei-Lung Dustin Tseng, and Muthuramakrishan Venkitasubramaniam. Concurrent zero knowledge: Simplifications and generalizations. Manuscript, 2008. `http://hdl.handle.net/1813/10772`.

[PV08] Rafael Pass and Muthuramakrishnan Venkitasubramaniam. On constant-round concurrent zero-knowledge. In *TCC*, pages 553–570, 2008.

[PW10] Rafael Pass and Hoeteck Wee. Constant-round non-malleable commitment from strong one-way functions. In *Eurocrypt*, 2010.

[Rab05] Michael O. Rabin. How to exchange secrets with oblivious transfer. Cryptology ePrint Archive, Report 2005/187, 2005.

[RK99] Ransom Richardson and Joe Kilian. On the concurrent composition of zero-knowledge proofs. In *Eurocrypt*, pages 415–432, 1999.

[Rom90] John Rompel. One-way functions are necessary and sufficient for secure signatures. In *STOC*, pages 387–394, 1990.

[RS91] Charles Rackoff and Daniel R. Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In *CRYPTO*, pages 433–444, 1991.

[Sah99] Amit Sahai. Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. In *FOCS*, pages 543–553, 1999.

[Wee10] Hoeteck Wee. Black-box, round-efficient secure computation via non-malleability amplification. To appear in FOCS 2010, 2010.

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