UNIVERSITY OF CALIFORNIA
Los Angeles

Stronger Round-Optimal Secure Protocols without Setup

A dissertation submitted in partial satisfaction
of the requirements for the degree
Doctor of Philosophy in Computer Science

by

Rex Fernando

2022

ABSTRACT OF THE DISSERTATION

Stronger Round-Optimal Secure Protocols without Setup

by

Rex Fernando
Doctor of Philosophy in Computer Science
University of California, Los Angeles, 2022
Professor Amit Sahai, Chair

In this dissertation, we study the round complexity of cryptographic protocols, giving special attention to secure multi-party computation (MPC), which allows a group of mutually distrusting parties $P_1, \ldots, P_n$, each with private input x_i, to compute the evaluation of some function $f(x_1, \ldots, x_n)$ without revealing their inputs to each other. We study this question via a recent new strong version of MPC, identified by a recent work by Benhamouda and Lin [BL20] and termed *Multiparty reusable Non-Interactive Secure Computation* (MrNISC). MrNISC requires the following general structure:

1. *Input encoding*: at any time, a party can publish an encoding of its input noninteractively, independent of the number of parties.

2. *Computation encoding*: At any time, any subset I of parties can jointly compute a function f on their inputs $x_I = \{x_i\}_{i \in I}$ by broadcasting a single public message. Each party's message is only dependent on the input encodings of the parties in I.

Parties are allowed to join the system at any time by publishing their input encoding, even after an arbitrary number of computation sessions have occurred. MrNISC achieves

essentially the best-possible form of non-interactivity for MPC protocols without running into known impossibility results on non-interactive MPC. MrNISC is a strict generalization of two-round concurrent-secure MPC.

We give the first construction of MrNISC which satisfies the full definition of malicious security in the plain model. By doing so, we also give the first construction of concurrent-secure two-round MPC. Security is given in the super-polynomial-simulation regime, and relies on the existence of an indistinguishability obfuscation scheme along with other standard assumptions.

During the course of obtaining our main result, we also obtain new results in the areas of zero-knowledge arguments and non-malleable commitments.

First, we give a two-round zero-knowledge argument which satisfies a weak form of statistical soundness, which we call *sometimes-statistical soundness*. Previously, no two-round zero knowledge protocols satisfied any form of statistical soundness. We also are able to give such a protocol which simultaneously satisfies statistical zero-knowledge and is highly reusable.

Second, we give a new one-round non-malleable commitment which satisfies full non-malleability with respect to commitment. Our construction works in the simultaneous-message model and is based heavily on the work of (Khurana, EUROCRYPT 2021), and is notable for not relying on the "multi-collision-resistant" hash function. All previous one-round non-malleable commitments with full security have relied on this assumption.

The dissertation of Rex Fernando is approved.

Alexander Sherstov

Rafail Ostrovsky

Raghu Meka

Amit Sahai, Committee Chair

University of California, Los Angeles

2022

To my wife, my parents, my brothers and my sister.

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGMENTS

This dissertation is a version of the work "Maliciously-Secure MrNISC in the Plain Model" by the author along with Aayush Jain and Ilan Komargodski. I am very grateful for their collaboration as well as for their friendship.

During the course of my PhD, I have had the great privilege of meeting and working with an incredible group of people. First and foremost is my advisor, Amit Sahai. Amit is a phenomenal teacher and a great mentor. Many times I would go into his office feeling down about research and about my career, and would leave feeling much relieved of my burdens. I am thankful for his support, and for his encouragement and advice.

Several people have had an impact on my development as a researcher. I would especially like to thank my good friends Ilan Komargodski and Aayush Jain. I met Ilan when I was his intern at NTT Research, and since then I have spent more time working with him than perhaps anyone else. Ilan has been extremely generous with his time, and is seemingly always available to talk about research, even at odd times during the night. I admire his creativity; he is constantly coming up with interesting new problems to work on. He has been a great mentor to me, and has shown great care for my well-being. I am grateful for all the discussions we have had about cryptography and about life in general.

Aayush has become one of my closest friends during my time at UCLA. We have shared countless meals together. Aayush also invited me to stay with him for almost two months while I was in between leases. Recently we started working on research together and it has been a ridiculous amount of fun. I feel very lucky to be able to work with such a great mind.

I would also like to thank Alon Rosen and Elaine Shi. Alon was my host during a lovely summer in Herzliya. His encouragement was crucial for me in developing self-confidence in doing research. He is also a legendary storyteller. I visited Elaine at CMU for six months, and she was an incredibly gracious host. Her deep mastery of so many areas, from theoretical

cryptography to security to distributed systems, is so inspiring.

Saikrishna Badrinarayanan was a third-year student in Amit's lab when I started, and he almost immediately took me under his wing. I owe him a lot for his patience in explaining to me the basic definitions and proof techniques and for giving me some of my first problems to work on.

I met many other friends during my time at UCLA, without which my time here would not have been the same. I thank Peter Rasmussen, Shirley Chen, Ruchi Jain, Prabhanjan Ananth, Alain Passelègue, Fermi Ma, Sam Kim, Paul Lou, Nathan Manohar, Ashutosh Kumar, Alexis Korb, and Riddhi Gosal for making my time at UCLA much more memorable. I especially have great memories with Peter writing both of our first result in cryptography together!

There are many people outside of UCLA and outside of cryptography who have helped me to get where I am today. Eric Bach was my unofficial advisor during my masters degree at UW-Madison, and my first paper in theoretical computer science is with him. I am grateful for all the time he spent with me. I am also grateful to Jack Lutz, Pavan Aduri and Jonathan Smith; during my time at ISU, their courses caused me to fall in love with math and theoretical computer science.

Last but not least, I would like to thank my wonderful family. My uncles and aunties Ravi and Sharmi Fernando, and Nirmalie Fernando, have made their houses homes away from home for me. Aunty Nirmalie in particular opened her home to me for almost eight months during the pandemic, while I was working at NTT in the Bay Area. I met my wife Elodie during my first year in Los Angeles, and throughout our time together, she has constantly supported, encouraged, and taken care of me while I have worked toward my PhD. Words cannot express how much I owe her for her patience and love. Finally, I thank my parents for loving me unconditionally and for being my greatest role models. Much of my enthusiasm for research comes from them: both are intensely curious about the world. I am so thankful to both of them for molding me into the person I am, and for always being there for me.

VITA

2013 B.S. (Computer Science and Applied Mathematics), Iowa State University.

2016 M.S. (Computer Sciences), University of Wisconsin–Madison.

2019 Intern hosted by Prof. Alon Rosen, IDC Herzliya.

2020 Intern hosted by Dr. Ilan Komargodski, NTT Research, Inc.

2021 Visiting Researcher hosted by Prof. Elaine Shi, Carnegie Mellon University.

PUBLICATIONS

Maliciously Secure Massively Parallel Computation for All-but-One Corruptions. Rex Fernando, Yuval Gelles, Ilan Komargodski, and Elaine Shi. CRYPTO 2022.

Secure Massively Parallel Computation for Dishonest Majority. Rex Fernando, Ilan Komargodski, Yanyi Liu, and Elaine Shi. TCC 2020.

Statistical Zap Arguments. Saikrishna Badrinarayanan, Rex Fernando, Aayush Jain, Dakshita Khurana, and Amit Sahai. Eurocrypt 2020.

Output Compression, MPC, and iO for Turing Machines. Saikrishna Badrinarayanan, Rex Fernando, Venkata Koppula, Amit Sahai, and Brent Waters. Asiacrypt 2019.

Preventing CLT Attacks on Obfuscation with Linear Overhead. Rex Fernando, Peter Rasmussen, and Amit Sahai. Asiacrypt 2017.

Infinitely Many Carmichaels for a Modified Miller-Rabin Prime Test. Eric Bach and Rex Fernando. ISSAC 2016.

CHAPTER 1

Introduction

One fundamental measure of cryptographic protocols, such as zero knowledge arguments, non-malleable commitments, and secure multi-party computation, is the amount of interaction they require. The importance of this measure is strongly grounded in practice: while the bandwidth of modern networks has constantly been increasing, there is a physical lower bound on their latency, imposed by distance and the speed of light. For instance, since the distance from Los Angeles to Paris is over $9,000$ km, and the speed of light is approximately 300 km/ms, it is impossible to achieve better than approximately 30 ms latency between these two cities. The round complexity of a protocol can also affect its security properties. One very useful property of fully non-interactive and quasi-non-interactive[1] arguments is that proofs can be posted to some public bulletin board, like a blockchain, and then any party can later independently verify its validity, even if the original prover is offline. This enables arguments to be recursively composed, which has been used to achieve fundamental new results in the areas of succinct arguments [BCC13], and also to achieve new space and communication efficient secure multi-party computation protocols [FGK22]. It is also crucial in enabling anonymous cryptocurrency protocols such as in [BCG14], since non-mining parties would otherwise be forced to always remain online in order to enable the system to work.

In this dissertation, we study the round complexity of cryptographic protocols, giving special attention to secure multi-party computation (MPC), which allows a group of mutually distrusting parties $P_1, \ldots, P_n$, each with private input x_i, to compute the evaluation of some

[1] By *quasi-non-interactive* we refer to "non-interactive" protocols that require a trusted setup such as a common reference string.

function $f(x_1, \ldots, x_n)$ without revealing their inputs to each other. Security is shown via the notion of *simulation*, which compares the behavior of the system in the "real world" to its behavior in an "ideal world." This ideal world is defined with respect to a trusted party, which receives x_i from each party P_i, and computes and delivers $f(x_1, \ldots, x_n)$ to all the parties. Since the parties do not interact with each other directly, security holds in this world by definition. An MPC protocol is then said to be secure if for any efficient adversary in the real world which learns something after an execution of the protocol, there exists an efficient *simulator* who can learn the same thing.

The round complexity of MPC protocols has been well-studied over the last few decades. The original MPC construction of [GMW87] was highly round-inefficient, taking a number of rounds proportional to the depth of the circuit for the functionality being computed. Since then, a long line of work [BMR90, KOS03, KO04, Wee10, GMP16, ACJ17, BHP17, COS17b, CCG20] has made dramatic improvements, with recent works finally achieving four rounds [COS17b, CCG20, ACJ17, BHP17]. This was shown to be optimal by the works of [KO04, GMP16], which showed that achieving secure computation in three rounds within the standard regime of black-box polynomial-time simulation is impossible.

In the classical definition of simulation security for MPC protocols, the parties are assumed to run the protocol in an isolated environment, separate from other parties and other executions of protocols. While this definition is simple and elegant, the ubiquity of the internet means that this assumption is not very realistic. The notion of *concurrent security* fixes this by allowing an adversary to spawn an arbitrary number of parties and executions of a protocol. Unfortunately, the work of [BPS06] showed that concurrent security is impossible *in any number of rounds* within the standard regime of black-box polynomial-time simulation.

The exciting work of [Pas03] introduced a very useful relaxation of standard polynomial-time simulation, called *super-polynomial-time simulation*. In this new definition, the simulator is allowed to run for slightly longer than polynomial-time. This has been used, among other things, to achieve concurrent security for MPC protocols by the works of [CLP10,

GGJ12, KMO14], sidestepping the impossibility result of [KO04, GMP16]. In 2017, the work of [BGJ17] constructed a concurrent MPC protocol in three rounds, thus bypassing both the lower bounds of [KO04, GMP16] and [BPS06] at once. For several years, this has been the state of the art in terms of the round complexity of both MPC and concurrent-secure MPC in the plain model.

An important question, then, is whether concurrent-secure MPC, or even standalone MPC, can be achieved in two rounds in the plain model, without setup. In this dissertation, we study this question.

It is natural to ask whether MPC can be done in one round, with each party sending a single simultaneous message. However, one can very easily show that this is impossible, via the following argument, commonly referred to as the *residual function attack*. Consider the case of two parties P_1 and P_2, and say that P_1 sends its message m_1. Then P_2 should be able to compute and send her message m_2, so that both parties learn $f(x_1, x_2)$. However, this means that P_2 can compute m_2' for any other x_2' in her head, and learn $f(x_1, x_2)$ as well. She can do this for arbitrarily many x_2'. This means that parties are able to learn much more than is allowed by a secure MPC protocol. This simple argument also extends to the case of protocols with trusted setup, showing that one-round protocols are also impossible in this case.

This raises the question, how close can we get to a non-interactive protocol without running into this impossibility?

We study this question via a recent new strong version of MPC, identified by a recent work by Benhamouda and Lin [BL20] and termed *Multiparty reusable Non-Interactive Secure Computation* (MrNISC). MrNISC requires the following general structure:

1. *Input encoding*: at any time, a party can publish an encoding of its input noninteractively, independent of the number of parties.

2. *Computation encoding*: At any time, any subset I of parties can jointly compute a

function f on their inputs $x_I = \{x_i\}_{i \in I}$ by broadcasting a single public message. Each party's message is only dependent on the input encodings of the parties in I.

Parties are allowed to join the system at any time by publishing their input encoding, even after an arbitrary number of computation sessions have occurred.

In this way, MrNISC achieves essentially the best-possible form of non-interactivity for MPC protocols without running into the aforementioned impossibility: once parties have committed to their input, any subset of parties can compute an arbitrary function on their committed inputs via a single round. Note that MrNISC is a strict generalization of two-round concurrent-secure MPC.

Several MrNISC protocols have been constructed in the *semi-malicious* regime, where security only holds for adversaries who follow the protocol specification.[2] Benhamouda and Lin [BL20] constructed such a protocol for all efficiently computable functionalities relying on the SXDH assumption in asymmetric bilinear groups. In two concurrent follow-up works, Ananth et al. [AJJ21] and Benhamouda et al. [BJK21] obtained MrNISC protocols relying on Learning With Errors (LWE). However, it was unknown whether it is possible to construct MrNISC in the plain model which satisfies the full malicious version of security, where adversaries can deviate arbitrarily from the protocol specification.

1.0.1 Our Results

In this dissertation, we give the first affirmative answer to the above question. Specifically, relying on well-founded assumptions, we obtain a maliciously secure SPS MrNISC in the plain model, without any trusted setup. Our result is obtained via a generic transformation from any semi-malicious secure MrNISC. Security of the construction relies on the existence of a subexponentially-secure indistinguishability obfuscator

[2]Semi-malicious security allows the adversary to choose arbitrary randomness for the parties, but otherwise requires honest behavior.

Theorem 1 (Main Result)**.** *Assume there exists subexponentially-secure variants of the following:*

- *a semi-malicious-secure MrNISC,*
- *an indistinguishability obfuscation scheme,*
- *a non-interactive witness indistinguishable argument,*
- *a one-way permutation in NC^1,*
- *a time-lock puzzle,*
- *either the DDH assumption or hardness of factoring, and*
- *quantum hardness of the learning-with-errors (LWE) assumption*

Then there exists a malicious-secure MrNISC in the plain model, with a super-polynomial simulator.

On the assumptions. Our work relies heavily on the idea of multiple *axes of hardness* [LPS17], where there are multiple ways to measure the hardness of a problem, such as circuit size and circuit depth. This allows one to define pairs of problems (A, B) where A is simultaneously harder than B (with respect to one axis) and easier than B (with respect to the other). Time-lock puzzles are a well-known way to achieve such scenarios based on circuit size and depth. In the course of our construction, we require an additional axis of hardness, and for this we use quantum hardness. Note that both DDH and factoring are solvable in quantum polynomial time, whereas LWE is thought to be computationally hard for quantum computers. Thus by setting appropriate security parameters, we can have a quantum machine which can break an instance of DDH or factoring in polynomial time but cannot break a LWE instance, and at the same time a classical machine which can break the LWE instance but not the DDH or factoring instance.

Implications for (Classical) MPC. We note that it is possible to view our main result via several different lenses in terms of classical MPC:

- Our MrNISC implies the first 2-round maliciously secure SPS MPC based on **well-founded falsifiable assumptions**.
- Our MrNISC implies the first 2-round maliciously secure SPS MPC with a **short and reusable first message**, based on any assumption. Namely, the first round message is not only independent of the function to be computed (which is necessary for reusability), but it is actually generated independently of the number of participating parties. All prior MPC protocols with this property only satisfy semi-malicious security in the plain model [BGM20, BL20, AJJ21, BGS21, BJK21].
- Our MrNISC implies the first **concurrent** two-round maliciously secure SPS MPC. Indeed, at any point in time, parties can join the protocol by publishing their input encodings and even start evaluation phases. This could happen even after some of the other parties published their input encodings and participated in several evaluation phases. The only previously known *malicious* (SPS) concurrent MPC required three rounds [BGJ17].

Other results. In the course of obtaining our main result, we achieve two intermediate results, in the areas of zero-knowledge and non-malleable commitments.

First, we give a new definition of two-round zero knowledge, called *reusable statistical zero-knowledge with sometimes-statistical soundness*. This new type of argument satisfies both statistical zero knowledge and a weakened form of statistical soundness. (Note that it is well-known that achieving both statistical zero knowledge and full statistical soundness is impossible for all statements in NP unless the polynomial-time hierarchy collapses [SV97].) We also require a strong form of reusability. We show the following theorem in Chapter 4:

Theorem 2. *Assume that the following assumptions hold:*

- *A subexponentially secure indistinguishability obfuscator exists,*
- *A time lock puzzle as in Definition 4 exist,*
- *a subexponentially-secure NIWI exists,*
- *subexponential hardness of the LWE assumption, and*
- *a subexponentially-secure OWP computable in* NC^1 *exists,*

then there exists a reusable statistical ZK argument with sometimes statistical soundness as defined in Definition 20.

Second, we give a new one-round non-malleable commitment in the simultaneous-message model under better assumptions than were previously known. This commitment satisfies a strong definition of security called CCA-non-malleability. We prove the following theorem in Chapter 3:

Theorem 3. *Assume that the following assumptions hold:*

- *A subexponentially secure indistinguishability obfuscator exists,*
- *A subexponentially secure non-interactive witness-indistinguishable argument exists,*
- *Subexponential hardness of the DDH or factoring assumption,*
- *LWE is subexponentially secure against quantum adversaries of subexponential size.*

Then, there exists a subexponentially-secure one-round CCA commitment scheme supporting a super-polynomial number of tags.

Non-interactive non-malleable commitments were first constructed by the work of [PPV08], using very strong and non-standard, non-falsifiable assumptions. The works of [BL18, GKL21] were able to obtain constructions based on falsifiable assumptions, namely (among other things)

an assumption called *keyless multi-collision-resistant hash functions*, which are described in more detail below. This assumption was first introduced in the work of [BKP18a, BDR18], so it is less than five years old, and is still not well-studied. In contrast, our commitment scheme relies solely on assumptions which have a long history of study.

Our construction is based heavily on the work of [Khu21], which achieves a weakened version of one-round non-malleable commitments. In order to achieve our main result, we need full CCA-non-malleable commitments, so the construction of [Khu21] will not suffice as-is. We elaborate on this in Section 1.1 and Chapter 3.

1.0.2 Related work

A recent work of Agarwal, Bartusek, Goyal, Khurana, and Malavolta [ABG21] gave the first two-round standalone maliciously secure MPC in the plain model. Although an exciting first step, the result is nonstandard in several ways. First, they require the existence of several primitives (including semi-malicious MPC) which are *exponentially secure* in the number of parties. Their construction also a special type of *non-interactive* non-malleable commitment. Unfortunately, the only known instantiations of the latter rely on strong and non-standard assumptions. One instantiation relies on factoring-based *adaptive* one-way functions [PPV08],[3] a non-falsifiable assumption that incorporates a strong non-malleability flavor. Another instantiation relies on *keyless* multi-collision resistant hash functions [BKP18b] and an exponential variant of the "hardness amplifiability" assumption of [BL18]. While both of these assumptions are (sub-exponentially) falsifiable, they are still highly non-standard:

1. A keyless multi-collision resistant hash function is a single publicly known function for which (roughly) collisions are "incompressible", namely, it is impossible to encode significantly more than k collisions using only k bits of information. While keyless

[3] An adaptive one-way function is a non-falsifiable hardness assumption postulating the existence of a one-way function f that is hard to invert on a random point $y = f(x)$ even if you get access to an inversion oracle that inverts it on every other point $y' \neq y$.

hash functions are formally a plain-model assumption, there is no known plain-model instantiation based on standard assumptions. The only known instantiation is either in the random oracle model, or by heuristically assuming that some cryptographic hash function, like SHA-256, is such.

2. Hardness amplification assumptions postulate (roughly) that the XOR of independently committed random bits cannot be predicted with sufficiently large advantage. There are concrete (contrived) counter examples for this type of assumptions showing that they are generically false [DJM12], although they certainly might hold for specific constructions.

   The specific variant used by Agarwal et al. is novel to their work. It assumes *exponential* hardness amplification against PPT adversaries, i.e., that there exists a constant $\delta > 0$ such that for large enough ℓ, the XOR of ℓ independently committed random bits cannot be predicted by a PPT adversary with advantage better than $2^{-\ell\delta}$. This assumption (similarly to [PPV08]'s adaptive one-way functions) also incorporates a non-malleability flavor.

Because of this, there is no way to instantiate the protocol of [ABG21] relying on any well-studied assumptions, or even on assumptions not specifically formulated in order to achieve non-malleable commitments. Our work strictly generalizes their work, and does not use ad-hoc assumptions.

1.1 Technical Overview

In this section, we give an overview of our constructions and the main ideas needed to prove their security. Let us start by reviewing the syntax of MrNISC, as defined by Benhamouda and Lin [BL20].

Model and syntax. A MrNISC consists of an input encoding phase done without coordination with other parties in the system (i.e., without even knowing they exist), and an evaluation phase in which only relevant parties participate by publishing exactly one message each. In other words, MrNISC is a strict generalization of 2-round MPC with the following properties:

- there is no bound on the number of parties;
- multiple evaluation phases can take place with the same input encodings;
- parties can join at any point in time and publish their input encoding, even after multiple evaluation phases occurred.

We assume all parties have access to a broadcast channel that parties use to transmit messages to all other parties. The formal syntax of an MrNISC consists of three polynomial-time algorithms $(\mathsf{Encode}, \mathsf{Eval}, \mathsf{Output})$, where Encode and Eval are probabilistic, and Output is deterministic. The allowed operations for a party P_i are:

- **Input Encoding phase**: each party P_i computes $m_{i,1}, \sigma_{i,1} \leftarrow \mathsf{Encode}(1^\lambda, x_i)$, where x_i is P_i's private input, $m_{i,1}$ is P_i's round 1 message, and $\sigma_{i,1}$ is P_i's round 1 private state. It broadcasts $m_{i,1}$ to all other parties.
- **Function Evaluation phase**: any set of parties I can compute an arity-$|I|$ function f on their respective inputs as follows. Each party P_i for $i \in I$ computes $m_{i,2} \leftarrow \mathsf{Eval}(f, \sigma_{i,1}, I, \{m_{j,1}\}_{j\in I})$, where f is the function to compute, x_i is P_i's private input, $\sigma_{i,1}$ is the private state of P_i's input encoding, $\{m_{j,1}\}_{j\in I}$ are the input encodings of all parties in I, and the output $m_{i,2}$ is P_i's round 2 message. It broadcasts $m_{i,2}$ to all parties in I
- **Output phase**: upon completion of the evaluation phase by each of the participating parties, anyone can compute $y \leftarrow \mathsf{Output}(\{m_{i,1}, m_{i,2}\}_{i\in I})$ which should be equal to $f(\{x_j\}_{j\in I})$.

Security. For security, we require that an attacker does not learn any information beyond what is absolutely necessary, which is the outputs of the computations. Formally, for every "real-world" adversary that corrupts the evaluator and a subset of parties, we design an "ideal world" adversary (called a simulator) that can simulate the view of the real-world adversary using just the outputs of the computations. As in all previous works on MrNISC (including [BL20, AJJ21, BJK21]), we assume static corruptions, namely that the adversary commits on the corrupted set of parties at the very beginning of the game. However, all previous works only achieved semi-malicious security (unless trusted setup assumptions are introduced). This notion of security, introduced by Asharov et al. [AJL12], only considers corrupted parties that follow the protocol specification, except letting them choose their inputs and randomness arbitrarily. In contrast, we consider the much stronger and more standard notion of *malicious* security, which allows the attacker to deviate from the specification of the protocol arbitrarily.

More precisely, in malicious security, the adversary can behave arbitrarily in the name of the corrupted parties. Specifically, after the adversary commits on the corrupted set of parties, it can send an arbitrary round 1 message for a corrupted party, ask for a round 1 message of any honest party (with associated private input), ask an honest party to send the round 2 message corresponding to an evaluation of an arbitrary function on the round 1 message of an arbitrary set of parties, and send an arbitrary round 2 message of a malicious party corresponding to an evaluation of an arbitrary function on the round 1 message of an arbitrary set of parties. The simulator needs to simulate the adversary's view with the assistance of an ideal functionality that can provide only the outputs of the computations that are being performed throughout the adversary's interaction.

Typically, protocols are called *maliciously secure* if for every polynomial-time adversary, there is a polynomial-time simulator for which the real-world experiment and the ideal-world experiment from above are indistinguishable. However, as mentioned, it is impossible to achieve such a notion of malicious security for MPC (let alone MrNISC) in merely two rounds

unless trusted setup assumptions are introduced. Therefore, we settle for super-polynomial time simulation (SPS), which means that the simulator can run in super-polynomial time. In contrast, the adversary is still assumed to run in polynomial time.

We refer to Section 5.1 for the precise definition.

Terminology. For the sake of brevity, we will sometimes refer to the *input encoding phase* as *round 1*, and the *function evaluation phase* as *round 2*.

1.1.1 The MrNISC Protocol

To obtain our main result, we will start with a semi-malicious-secure MrNISC protocol [BL20, BJK21] and introduce modifications to achieve malicious security. Recall that semi-malicious security only guarantees security when the adversary follows the honest protocol specification exactly, except that it can arbitrarily choose corrupted parties' randomness. We would like to use the following high-level approach used by many classical MPC protocols. During the input encoding phase, we require each party to commit to its input and randomness in addition to publishing a semi-malicious input encoding, and then to prove using zero-knowledge that all of its semi-malicious MrNISC messages were generated by following the prescribed protocol using that committed input and randomness. However, a problem arises when using this strategy with 2-round protocols. (Note that MrNISC requires that evaluation can be carried out in two rounds; in this way, it is a strict generalization of 2-round MPC.) This problem comes from the fact that zero-knowledge in the plain model requires at least two rounds. Assuming we use such a 2-round ZK scheme, honest parties would need to send their second-round MrNISC messages before finding out whether the first-round MrNISC messages were honest. This completely breaks security—if any party publishes semi-malicious messages based on a non-honest transcript, the semi-malicious protocol can make no security guarantees about these messages.

We need some way of overcoming this problem. That is, we need a way to publish

second-round messages so that they are only revealed if the first round is honest. To this end, we are going to use *witness encryption* as a locking mechanism: we "lock" the round 2 message of the underlying (semi-malicious) MrNISC and make sure that it can be unlocked only if all involved parties' proofs verify. More precisely, party i does:

1. *Round 1 message*: Commit to its input and randomness and publish a round 1 message using the underlying MrNISC with the committed input/randomness pair. At the same time, generate a verifier's first-round ZK message for the other parties.

2. *Round 2 message*: Compute a round 2 message using the underlying MrNISC with randomness derived from the secret state. Generate a zero-knowledge proof that this was done correctly. Publish a witness encryption hiding the aforementioned round 2 message that could be recovered by supplying valid proofs that all other parties' first-round messages were created correctly.

With this template in mind, even before starting to think about what a security proof will look, it is already evident that there are significant challenges in realizing the building blocks. Here are the three main challenges.

Challenge 1: The ZK argument system. The first challenge arises from trying to use ZK arguments as witnesses for the witness encryption scheme. Recall that witness encryption allows an encryptor to encrypt a message with respect to some statement Φ, and only if Φ is false, then the message is hidden. Witness encryption (WE) crucially only can provide security when Φ is *false*; in particular, if Φ is true, even if it is computationally hard to find a witness for Φ, no guarantees are made about the encrypted message being hidden. Thus, it seems like we would need a *statistically-sound* ZK argument, i.e., a ZK proof: if the verifier's first-round message is honest, with high probability, there should not exist an accepting second-round ZK message.

It is well-known that to achieve ZK in two rounds, it is necessary to have a simulator

that runs in super-polynomial time (i.e., an SPS simulator). In every such known two-round ZK, the simulator works by brute-forcing some trapdoor provided in round 1, and giving proof that "either the statement is true or I found the trapdoor." Because of the existence of this trapdoor, it would be impossible to make any such ZK argument statistically sound: an unbounded-time machine can always find the trapdoor and prove false statements. So it seems like the ZK scheme needs to satisfy two contradictory requirements: be statistically sound, and be a two-round scheme (which appears to preclude statistical soundness).

Challenge 2: Non-malleability attacks. Since the security of the underlying semi-malicious MrNISC holds only if the adversary knows some randomness for its messages, we need all parties to prove that they know the input and randomness corresponding to their messages. We are aiming for a protocol that can be evaluated in two rounds, so this necessitates using a non-malleable commitment (to prevent an attacker from, say copying the round 1 message of some other party). Unfortunately, as mentioned before, non-interactive non-malleable commitments without setup are only known from very strong non-standard assumptions, such as adaptive one-way functions [PPV08], hardness amplifiability [BL18, ABG21], and/or keyless hash functions [BKP18b, LPS20, BL18]. These are very strong and non-standard assumptions, for some of which we have no plain-model instantiation, except heuristic ones. Thus, we want to achieve a secure MrNISC protocol (in the plain model) without such strong assumptions.

Challenge 3: Adaptive reusability of the primitives. We emphasize that we are building an MrNISC protocol, which significantly strengthens standalone two-round MPC. Because of this, our ZK argument and commitment schemes must satisfy strong forms of reusability. There are several challenges in ensuring both the ZK argument and non-malleable commitment scheme satisfy the types of reusability that we need, and we introduce several new ideas to solve these challenges. We will elaborate on this challenge below after we describe our ideas for solving challenges 1 and 2.

1.1.1.1 Solving Challenge 1: How do we get a "statistically-sound" SPS ZK?

We now discuss how to achieve the seemingly contradictory requirements of getting a 2-round SPS ZK argument which has a statistical soundness property that would allow it to be a witness for the WE scheme. Our key idea is to relax the notion of statistical soundness to one that is obtainable in two rounds but still sufficient to use with WE.

Imagine we have a WE scheme where the distinguishing advantage of an adversary is tiny (say, subexponential in λ). It would then suffice to have a ZK protocol that is statistically sound a negligible fraction of the time, as long as it is quite a bit larger than the distinguishing advantage of the WE. In more detail, consider a hypothetical zero-knowledge protocol with the following properties:

- The first round between a computationally-bounded verifier and a prover fully specifies one of the two possible "modes": a *statistical ZK mode* and a *perfectly sound mode*.
- The perfectly sound mode occurs with some negligible probability ϵ, and in this mode, no accepting round 2 message exists for any false statement
- In the statistical ZK mode (which occurs with overwhelming probability $1-\epsilon$), the second message is simulatable by an SPS machine and a simulated transcript is statistically indistinguishable from a normal transcript.
- Furthermore, it is computationally difficult for a malicious prover to distinguish between the two modes.

If we had such a ZK protocol, it would enable us to argue hiding of the witness encryption scheme whenever the first round of the protocol is not honest. The idea of this argument is as follows. Suppose an adversary could learn something about the second-round messages from their witness encryptions in some world where the first round was not honest. In that case, it should also be able to do so even in the perfectly-sound mode (otherwise, it would

distinguish the modes). But in this mode, proofs for false statements do not exist; thus, the witness encryption provides full security. Even though this mode happens with negligible probability, it is still enough to contradict witness encryption security, whose advantage is much smaller.

To construct this new ZK scheme, we use ideas that are inspired by the extractable commitment scheme of Kalai, Khurana, and Sahai [KKS18]. This commitment scheme has the property that it is extractable with some negligible tunable probability but is also statistically hiding. This commitment was used in the works of [BFJ20] to get a two-round statistical zero-knowledge argument with super-polynomial simulation. To instantiate our new "sometimes perfectly-sound" ZK argument, we use the protocol of [BFJ20] as a starting point, but we will need to make significant modifications. Namely, to force a well-defined perfect soundness mode, we will make the first round of this protocol a "simultaneous-message" round, where both the prover and the verifier send a message. We elaborate further on this and other key ideas used in our construction in Chapter 4.

We note an important subtlety in this new definition and our construction. Namely, the statistical ZK and perfect soundness properties only hold with respect to the *second* round. If the verifier is unbounded-time, then after seeing a first-round prover's message, it can send a first-round verifier's message that forces perfect soundness all the time and thus disallows any prover from giving a simulated proof. On the other hand, if the prover is unbounded-time, then after seeing a first-round verifier's message, it can send a first-round prover's message, which causes the probability ϵ of the perfect soundness mode to be 0. Thus the frequency of perfect soundness mode and the ability of the simulator to give a simulated proof depend on the first round being generated by computationally bounded machines.

1.1.1.2 Solving Challenge 2: How do we avoid non-interactive non-malleability?

To solve challenge two, we must somehow get a non-malleable commitment (NMC) scheme which can be executed in the first round without using strong assumptions such as keyless hash

Figure 1.1: The diagram on the left depicts the communication pattern of Khurana's [Khu21] commitment scheme, whereas the diagram on the right depicts ours. The key difference is that in our scheme, the receiver's message and the sender's messages can be sent simultaneously, while in [Khu21] the receiver's message must be sent after the sender's message.

functions, hardness amlifiability, or adaptive one-way functions. Recall that unfortunately, all known instantiations of non-interactive NMCs (for a super-polynomial number of tags) currently require the use of (some combination of) these strong assumptions.

Our approach to solving this problem is inspired by the exciting work of Khurana [Khu21], which builds a new type of commitment that works as follows. The commitment phase is similar to a non-interactive commitment in that the only communication from the committer is a first-round message C. The role of the receiver is slightly different: The receiver chooses a random string τ internally, and it is both C and τ together that truly defines the commitment (and, correspondingly, the underlying value being committed to). Consequently, to compute an opening, the committer must receive a τ from the receiver. Non-malleability (and binding) hinges upon the fact that the τ chosen by the receiver is chosen after seeing the commitment. (See the left diagram below for an illustration of this scheme.) Crucially, this commitment can be constructed from well-founded assumptions (indistinguishability obfuscation, time-lock puzzles, and OWPs), bypassing the need for the strong assumptions discussed earlier.

We would like to use this commitment scheme in our protocol. There are two main issues that arise.

- First, to use this scheme, we would need the commitment phase to happen entirely in the first round. Namely, the receiver must publish τ simultaneously while the committer is publishing C. (See the right-hand diagram above.) In particular, in the security proof, we need to handle the case of malicious committers who publish C after seeing the round-1 τ. More formally, we must turn the commitment of [Khu21] into a *one-round simultaneous-message CCA-non-malleable commitment*.

- Second, our goal is to have every party use this commitment to commit to their input and randomness for the protocol. However, for security to hold, party P_i's committing message C_i must be used in conjunction with each party P_j's τ_j. Although for an honest committer, the committed value will be consistent across all possible pairs in the set $\{(C_i, \tau_j)\}_{j \neq i}$, it is possible for a malicious committer to generate a C_i where this is not true.

Solving the first issue involves identifying some technical challenges in the security proof of [Khu21] and making changes to the protocol to avoid these issues. Roughly, we replace an encryption given in the first round with a quantum-extractable commitment scheme. This allows us to carefully set the complexity hierarchy and thereby get security even if the τ's are chosen before C. We describe this in detail in Chapter 3. For the second issue, by adding a standard (malleable) perfectly binding commitment (e.g., Blum's commitment) at the MrNISC protocol level, we can force every party P_i to act consistently across all pairs $\{(C_i, \tau_j)\}_{j \neq i}$.

1.1.1.3 Solving Challenge 3: How do we get reusability?

We now describe the challenges which arise when trying to get the type of reusability required by MrNISC. The main problem is to ensure that all of the building blocks we use (i.e., the ZK scheme and the NMC scheme) support the reuse of their first-round message. It turns out that the non-malleable commitment we described in the previous section can be adapted

to this reusable setting without much modification. However, several challenges arise when adapting the sometimes-statistically-sound ZK scheme, which we discussed earlier, to the reusable setting. We focus on these challenges here.

Recall that the ZK scheme is a simultaneous message protocol, so a transcript consists of three messages of the form $(\mathsf{zk}_{1,P}, \mathsf{zk}_{1,V}, \mathsf{zk}_{2,P})$, a round-1 message of the prover and the verifier, and a round-2 message of the prover. What we need is for any prover to be able to publish a single $\mathsf{zk}_{1,P}$ in round 1, which can be used in many different sessions with respect to many different $\mathsf{zk}_{1,V}$ messages. In addition, we require a very strong form of reusability: even if a malicious verifier sees an entire transcript $(\mathsf{zk}_{1,P}, \mathsf{zk}_{1,V}, \mathsf{zk}_{2,P})$, and then chooses a new verifier's first-round message $\mathsf{zk}'_{1,V}$, zero-knowledge should still hold when the prover publishes a proof with respect to $\mathsf{zk}'_{1,V}$ and the prover's *original* message $\mathsf{zk}_{1,P}$. Similarly, a verifier should be able to publish a single $\mathsf{zk}_{1,V}$ which can be used in many different sessions with respect to many different $\mathsf{zk}_{1,P}$ messages, and the soundness properties of the ZK scheme should still hold.

Note that it is not immediately clear whether this reusability for ZK arguments are implied by a corresponding non-reusable version of ZK arguments. This turns out not to be the case. To satisfy reusability, we end up having to make several changes to our (non-reusable) sometimes-perfectly-sound ZK scheme. We describe this in more detail in Section 4.4.

1.1.1.4 Putting things together

We now have the main pieces that we will use to construct a malicious-secure MrNISC: the two-round sometimes-statistically-sound ZK, one-round simultaneous-message CCA-secure commitment, and the underlying semi-malicious MrNISC. Significant challenges arise when attempting to combine these pieces in the way described earlier to get a malicious MrNISC protocol. To see this, it will be convenient to briefly mention the approach we take for the security proof.

A simplified version of the sequence of hybrids we use is as follows. First, we extract the value underlying the commitments and check if anyone acted dishonestly. If so, we switch the honest parties' witness encryptions to encrypt 0 rather than the actual round 2 messages (this is hybrid 1). Second, we simulate the ZK proof (this is hybrid 2). Third, we switch the underlying value in the commitment to 0 (this is hybrid 3). Once the commitments are independent of the true input, we can use the simulator of the underlying MrNISC (this is hybrid 4). The last hybrid is identical to our simulator.

To make the transitions between the hybrids possible, we need to set the hardness of every primitive carefully. Each hybrid indistinguishability induces some hardness inequality for the involved primitives. Unfortunately, the inequalities seem to be in contradiction to each other. Observe that for the first indistinguishability (between hybrid 0 and hybrid 1), we need our ZK argument's soundness properties to hold against adversaries who can run the CCA extractor. That is,

$$T_{\mathsf{extractor}} \ll T_{\mathsf{sound}}.$$

For the transition between hybrid 2 to 3, we need to guarantee that the security of the commitment scheme holds even against an adversary that can run the ZK simulator. That is,

$$T_{\mathsf{ZKSim}} \ll T_{\mathsf{extractor}}.$$

Together, the above two inequalities imply that it is necessary to have $T_{\mathsf{ZKSim}} \ll T_{\mathsf{sound}}$. But this is impossible, at least using the techniques we use in constructing the ZK argument. Our simulator works by brute-forcing the verifier's $\mathsf{zk}_{1,V}$ message to obtain some secret and produces proofs with this knowledge. In other words, whoever has the secret can produce accepting proofs without knowing a witness—this is essentially an upper bound on the soundness of the scheme, i.e., $T_{\mathsf{sound}} \ll T_{\mathsf{ZKSim}}$, which means that our inequalities cannot be satisfied at the same time.

To solve this problem, we introduce another axis of hardness, namely, *circuit depth*. In particular, assume that it is possible to run the ZK simulator in some super-polynomial

depth d. To do this, we would have to construct a ZK argument where the secret embedded in $\mathsf{zk}_{1,V}$ is extractable in depth d. Further, assume that in polynomial depth, it is extremely hard to extract the secret from $\mathsf{zk}_{1,V}$ (much harder than size d). We can use such a ZK argument to solve the problem above. Namely, we can restrict the reduction for hybrids 0 and 1 to run in *polynomial depth*, and in this complexity class, it holds that $T_{\mathsf{extractor}} \ll T_{\mathsf{sound}}$. For the reduction for hybrids 2 and 3, we will allow the depth to be d, in which case the inequality $T_{\mathsf{ZKSim}} \ll T_{\mathsf{extractor}}$ is satisfied.

So we have reduced this problem to constructing a ZK argument which is simulatable in some super-polynomial depth d and whose soundness holds against size much larger than d as long as the depth is restricted to be polynomial. It turns out that it is possible to modify our original ZK argument to satisfy this property; we describe this in Section 4.4, where we explain the ZK argument in detail.

Several more minor technical issues arise when putting things together. One such problem is that of "simulation soundness," that is, we need to guarantee that the adversary cannot give valid ZK arguments for false statements even if it sees simulated arguments from the honest parties. We solve this issue using techniques from the work of [BGJ17]. At a very high level, if we use a ZK argument where the simulated proofs are indistinguishable from normal proofs even to an adversary who is powerful enough to run the simulator itself, and if we commit to the witnesses using a non-malleable commitment, it is possible to design a sequence of hybrids that guarantees simulation soundness.

This and other minor technical details result in a construction and sequence of hybrids that are slightly more involved than the simplified version presented in this overview. We refer the reader to Chapter 5 for details.

CHAPTER 2

Preliminaries

In this chapter, we collect the standard cryptographic definitions and notation which will be used in the rest of the book.

2.1 Miscellaneous Notation

For any distribution $\mathcal{X}$, we denote by $x \leftarrow \mathcal{X}$ the process of sampling a value x from the distribution $\mathcal{X}$. For a set X we denote by $x \leftarrow X$ the process of sampling x from the uniform distribution over X. For an integer $n \in \mathbb{N}$ we denote by $[n]$ the set $\{1, .., n\}$. A function $\mathsf{negl} : \mathbb{N} \rightarrow \mathbb{R}$ is negligible if for every constant $c > 0$ there exists an integer N_c such that $\mathsf{negl}(\lambda) < \lambda^{-c}$ for all $\lambda > N_c$. Throughout, when we refer to polynomials in security parameter, we mean constant degree polynomials that take positive value on non-negative inputs. We denote by $\mathsf{poly}(\lambda)$ an arbitrary polynomial in λ satisfying the above requirements of non-negativity.

Throughout this dissertation, all machines are assumed to be non-uniform. We will use λ to denote the security. We will use PPT as an acronym for "probabilistic (non-uniform) polynomial-time". In addition, we use the notation $T_1 \ll T_2$ (or $T_2 \gg T_1$) if for all polynomials p, $p(T_1) < T_2$ asymptotically.

The statistical distance between two distributions X and Y over a discrete domain Ω is defined as $\Delta(X, Y) = (1/2) \cdot \sum_{\omega \in \Omega} |\Pr[X = \omega] - \Pr[Y = \omega]|$.

$(\mathcal{C}, \epsilon)$-indistinguishability. By $\mathcal{C}$ we denote an abstract class of adversaries, where each adversary $\mathcal{A} \in \mathcal{C}$ grows in some complexity measure (i.e. size, depth, etc) based on the security parameter λ. Security definitions will always hold with respect to some class of adversaries which we will specify.

Definition 1 ($(\mathcal{C}, \epsilon)$-Indistinguishability)**.** *Let $\epsilon : \mathbb{N} \to (0, 1)$ be a function. We say that two distribution ensembles $\mathcal{X} = \{\mathcal{X}_\lambda\}_{\lambda \in \mathbb{N}}$ and $\mathcal{Y} = \{\mathcal{Y}_\lambda\}_{\lambda \in \mathbb{N}}$ are $(\mathcal{C}, \epsilon)$-indistinguishable if for any adversary $\mathcal{A} \in \mathcal{C}$, for any polynomial* poly*, and any $\lambda \in \mathbb{N}$,*

$$\left| \Pr_{x \leftarrow \mathcal{X}_\lambda} \left[\mathcal{A}\left(1^\lambda, x\right) \right] - \Pr_{y \leftarrow \mathcal{Y}_\lambda} \left[\mathcal{A}\left(1^\lambda, y\right) \right] \right| \leq \epsilon(\lambda).$$

We use the shorthand $\mathcal{X} \approx_{(\mathcal{C}, \epsilon)} \mathcal{Y}$ to denote this. If $\mathcal{A}$ is unbounded time then we say that $\mathcal{Y}$ and $\mathcal{X}$ are statistically *indistinguishable and we write $\mathcal{X} \approx_{(\infty, \epsilon)} \mathcal{Y}$, or alternately $\Delta(\mathcal{X}, \mathcal{Y}) \leq \epsilon$. (This corresponds to the standard definition of statistical distance.)*

2.2 Witness Encryption

Here, we recall the definition of witness encryption, originally due to Garg et al. [GGS13].

Definition 2. *A witness encryption scheme for an* NP *language L (with corresponding relation R) consists of the following two polynomial-time algorithms:*

- $\mathsf{WE.Enc}(1^\lambda, x, M)$*: The encryption algorithm takes as input the security parameter λ, a string $x \in \{0,1\}^*$, and a message $M \in \{0,1\}^*$. It outputs a ciphertext* CT*. This procedure is probabilistic.*

- $\mathsf{WE.Dec}(\mathsf{CT}, w)$*: The decryption algorithm takes as input a ciphertext* CT *along with a witness $w \in \{0,1\}^*$. It outputs a string $M \in \{0,1\}^*$ or the symbol $\perp$. This procedure is deterministic.*

These algorithms satisfy the following properties:

Correctness: *For any security parameter λ, for any message $M \in \{0,1\}^*$, any $x \in \{0,1\}^*$ such that $R(x,w) = 1$ for $w \in \{0,1\}^*$, we have that:*

$$\Pr[\mathsf{WE.Dec}(\mathsf{WE.Enc}(1^\lambda, x, M), w) = M] = 1.$$

$(\mathcal{C}, \epsilon)$***-Security:*** *Fix any ensemble $\mathcal{X}_\lambda$ of polynomial length strings such that every $x \in \mathcal{X}_\lambda$ satisfies $x \notin L$, and any ensemble of messages $\mathcal{M}_\lambda$ of polynomial length. For every $\lambda \in \mathbb{N}, x \in \mathcal{X}_\lambda$, and $M \in \mathcal{M}_\lambda$, it holds that*

$$\mathsf{WE.Enc}(1^\lambda, x, M) \approx_{(\mathcal{C},\epsilon)} \mathsf{WE.Enc}(1^\lambda, x, 0^{|M|}).$$

It is well known that witness encryption can be obtained directly from indistinguishability obfuscation by ofuscating a circuit that has the instance x and the message M hardwired, gets as input a witness, and outputs M if the instance-witness pair verify.

Theorem 4. *Assuming a $(\mathcal{C}, \epsilon)$-indistinguishability obfuscator for all polynomial-size circuits, then there is a $(\mathcal{C}, \epsilon)$-witness encryption scheme for all* NP*.*

2.3 Indistinguishability Obfuscation

In this section, we define the notion of an indistinguishability Obfuscation.

Definition 3 (Indistinguishability Obfuscator (iO) for Circuits [BGI01, BGI12])**.** *A probabilistic polynomial-time algorithm* iO *is called a secure indistinguishability obfuscator for polynomial-sized circuits if the following holds:*

- ***Completeness****: For every $\lambda \in \mathbb{N}$, every circuit C with input length n, every input $x \in \{0,1\}^n$, we have that*

$$\Pr\left[\tilde{C}(x) = C(x) \ : \ \tilde{C} \leftarrow \mathsf{iO}(1^\lambda, C)\right] = 1 \ .$$

- $(\mathcal{C}, \epsilon)$***-Indistinguishability:*** *For every two ensembles* $\{C_{0,\lambda}\}_{\lambda\in\mathbb{Z}^+}$ *and* $\{C_{1,\lambda}\}_{\lambda\in\mathbb{Z}^+}$ *of polynomial-sized circuits that have the same size, input length, and output length, and are functionally equivalent, that is,* $\forall\lambda \in \mathbb{Z}^+$, $C_{0,\lambda}(x) = C_{1,\lambda}(x)$ *for every input* x*, the distributions* $\mathsf{iO}(1^\lambda, C_{0,\lambda})$ *and* $\mathsf{iO}(1^\lambda, C_{1,\lambda})$ *are* $(\mathcal{C}, \epsilon)$ *indistinguishable.*

In this work, we require that iO is actually subexponentially secure against adversaries of subexponential size. As shown in [JLS21b, JLS21a] this can be instantiated assuming subexponential security of well studied hardness assumptions.

2.4 Time Lock Puzzles

We recall the notion of a time-lock puzzle scheme, originally due to [RSW96]. We adapt the definition from [BGJ16].

Definition 4. *A D-secure time lock puzzle* TLP *is a tuple of two algorithms* $(\mathsf{PGen}, \mathsf{Solve})$ *that satisfies the following properties.*

Syntax:

- $\mathsf{PGen}(1^\lambda, 1^t, x)$: *The puzzle generation algorithm is a randomized polynomial time algorithm takes as input a security parameter* λ *and a hardness parameter* t*. It also takes as input a solution* $x \in \{0,1\}^\lambda$*. It outputs a puzzle* Z*.*
- $\mathsf{Solve}(Z)$ *The puzzle solving algorithm takes as input a puzzle* Z*. It outputs* $x \in \perp \cup \{0,1\}^*$*.*

Completeness: *For every* $\lambda, t \in \mathbb{N}$ *and every* $x \in \{0,1\}^\lambda$, $\Pr[\mathsf{Solve}(\mathsf{PGen}(1^\lambda, 1^t, x)) = x] = 1$.

Efficiency: PGen *is a polynomial time algorithm in its input length, and* $\mathsf{Solve}(Z)$ *runs in time* $\mathsf{poly}(2^t, \lambda)$ *for every* Z *in support of* $\mathsf{PGen}(1^\lambda, 1^t, \cdot)$.

D-security: *Let* $\lambda \in \mathbb{N}$, $t = t(\lambda) \in \lambda^{\Omega(1/\log\log\lambda)} \cap \lambda^{O(1)}$ *and* $x \in \{0,1\}^{\lambda^{\Theta(1)}}$. *Then, it holds that for every Boolean circuit* $\mathcal{A}$ *with depth* $D(t)$ *and total size bounded by any polynomial in* 2^λ *it holds that:*

$$\left|\Pr[\mathcal{A}(\mathsf{PGen}(1^\lambda, 1^t, x)) = 1] - \Pr[\mathcal{A}(\mathsf{PGen}(1^\lambda, 1^t, 0^{|x|})) = 1]\right| \leq 2^{-\lambda}.$$

Note that we require security against sub-exponential size attackers and with sub-exponential distinguishing advantage. Specifically, we require that sub-exponential-size attackers (that are in depth at most $D(t)$) will not have advantage better than inverse sub-exponential. Sub-exponential *size* assumptions on the repeated squaring assumption were already made before, e.g., in [LPS20, DKP21, EFK20]).

The first and most popular instantiation of time-lock puzzles was proposed by Rivest, Shamir, and Wagner [RSW96]. It is based on the "inherently sequential" nature of exponentiation modulo an RSA integer. That is, that t repeated squarings mod N, where $N = pq$ is a product of two secret primes, require "roughly" t depth. More than twenty years after their proposal, there still does not exist a (parallelizable) strategy that can solve such puzzles of difficulty parameter t in depth $D(t)$ which is significantly less than 2^t, with any non-trivial advantage. This is true even for the decision problem variant, rather than the search problem. (Note that the decision version is the one that is typically defined and assumed in constructions, e.g., [BN00, BGJ16, LPS20, DKP21, EFK20]).

Another construction of time-lock puzzles, due to Bitansky et al. [BGJ16], based on indistinguishability obfuscation and (worst-case) non-parallelizing languages, is also an instantiation of the above definition, as long as the underlying are assumed to be sub-exponentially hard.

2.5 Correlation Intractable Hash Functions

We adapt definitions of a correlation intractable hash function family from [PS19a, CCH19].

Definition 5. *For any polynomials $k, (\cdot), s(\cdot) = \omega(k(\cdot))$ and any $\lambda \in \mathbb{N}$, let $\mathcal{F}_{\lambda,s(\lambda)}$ denote the class of NC^1 circuits of size $s(\lambda)$ that on input $k(\lambda)$ bits output λ bits. Namely, $f : \{0,1\}^{k(\lambda)} \to \{0,1\}^{\lambda}$ is in $\mathcal{F}_{\lambda,s}$ if it has size $s(\lambda)$ and depth bounded by $O(\log \lambda)$.*

We require the following property from such a function.

Definition 6 ($(\mathcal{C}, \epsilon)$-Somewhere-Statistical Correlation Intractable Hash Function Family)**.** *A hash function family $\mathcal{H} = (\mathsf{FakeGen}, \mathsf{Eval})$ is $(\mathcal{C}, \epsilon)$-somewhere-statistically correlation intractable (CI) with respect to $\mathcal{F} = \{\mathcal{F}_{\lambda,s(\lambda)}\}_{\lambda\in\mathbb{N}}$ as defined in Definition 5, if the following two properties hold:*

- ***Perfect Correlation Intractability:*** *For every $f \in \mathcal{F}_{\lambda,s}$ and every polynomial s,*

$$\Pr_{K\leftarrow\mathcal{H}.\mathsf{FakeGen}(1^\lambda,f)}\Big[\exists x \text{ such that } (x, \mathcal{H}.\mathsf{Eval}(K,x)) = (x, f(x))\Big] = 0.$$

- ***Computational Indistinguishability of Hash Keys:*** *Moreover, for every $f \in \mathcal{F}_{\lambda,s}$, for every $\mathcal{A} \in \mathcal{C}$, and every large enough $\lambda \in \mathbb{N}$,*

$$\Big|\Pr_{K\leftarrow\mathcal{H}.\mathsf{FakeGen}(1^\lambda,f)}[\mathcal{A}(K) = 1] - \Pr_{K\leftarrow\{0,1\}^\ell}[\mathcal{A}(K) = 1]\Big| < \epsilon(\lambda),$$

where ℓ denotes the size of the output of $\mathcal{H}.\mathsf{Setup}(1^\lambda, f)$.

The work of [PS19a] gives a construction of correlation intractable hash functions with respect to $\mathcal{F} = \{\mathcal{F}_{\lambda,s(\lambda)}\}_{\lambda\in\mathbb{N}}$, based on polynomial LWE with polynomial approximation factors. We observe that their construction also satisfies Definition 6, assuming LWE with an explicit efficiently computable advantage upper bound.

2.6 Sender Equivocal Oblivious Transfer

Definition 7 (Oblivious Transfer)**.** *An Sender-Equivocal Oblivious Transfer (OT) protocol consists of three randomized polynomial time algorithms:*

- $\mathsf{OT}_1(1^\lambda, b; r_1) \to \mathsf{ot}_1$: *The* OT_1 *algorithm takes as input a bit* $b \in \{0,1\}$ *and randomness* r*, and outputs the "receiver" message* ot_1*.*

- $\mathsf{OT}_2(\mathsf{ot}_1, m_0, m_1; r_2) \to \mathsf{ot}_2$: *The* OT_2 *algorithm takes as input a receiver message* ot_1*, two messages* m_0, m_1*, and randomness* r_2*, and it outputs the sender message* ot_2*.*

- $\mathsf{OT}_3(\mathsf{ot}_2, b, r_1) \to z$: *The* OT_3 *algorithm takes as input the sender message along with a bit* $b \in \{0,1\}$ *and randomness* r_1*. It outputs* $z \in \perp \cup \{0,1\}^*$*.*

We require a number of basic properties.
Correctness: Let $\lambda \in \mathbb{N}$*,* $b \in \{0,1\}$ *and* $(m_0, m_1) \in \{0,1\}^*$ *with* $|m_0| = |m_1|$*. Then, it holds that:*

$$\Pr[\mathsf{OT}_3(\mathsf{ot}_2, b, r_1) = m_b] = 1,$$

where $\mathsf{ot}_2 = \mathsf{OT}_2(\mathsf{ot}_1, m_0, m_1; r_2)$*,* $\mathsf{ot}_1 = \mathsf{OT}_1(1^\lambda, b; r_1)$ *and probability is taken over the coins of* r_1, r_2*.*

$(\mathcal{C}, \epsilon)$-Receiver Security: Let $\lambda \in \mathbb{N}$ *be the security parameter. Then, it holds that:*

$$\mathsf{OT}_1(1^\lambda, 0) \approx_{(\mathcal{C},\epsilon)} \mathsf{OT}_1(1^\lambda, 1).$$

Equivocation: There exist a polynomial time algorithm Equiv *such that the following property is satisfied. For every* $\lambda \in \mathbb{N}$ $b \in \{0,1\}$*,* $m_0, m_1 \in \{0,1\}^*$ *with length* ℓ*, with probability* 1 *over the coins* r_1 *of* $\mathsf{ot}_1 \leftarrow \mathsf{OT}_1(1^\lambda, b; r_1)$*, the following two distributions are identically distributed. Let* $v = (v_0, v_1)$ *where* $v_b = m_b$ *and* $v_{1-b} = 0^\ell$*.*

- *Distribution 1: Compute* $\mathsf{ot}_2 \leftarrow \mathsf{OT}_2(\mathsf{ot}_1, m_0, m_1; r_2)$*. Output* $(b, r_1, \mathsf{ot}_2, m_0, m_1, r_2)$*.*

- *Distribution 2: Compute* $\mathsf{ot}_2 \leftarrow \mathsf{OT}_2(\mathsf{ot}_1, v_0, v_1; r_2')$ *and* $r_2 \leftarrow \mathsf{Equiv}(b, r_1, \mathsf{ot}_2, r_2', m_0, m_1)$*. Output* $(b, r_1, \mathsf{ot}_2, m_0, m_1, r_2)$*.*

2.7 Equivocal Garbled Circuits for NC^1

Another primitive that we use is a an information theoretic variant of Yao's Garbled Circuits [Yao86] for NC^1 circuits. This variant allows one to to efficiently "invert" the randomness used for garbling.

Definition 8 (Syntax)**.** *An information theoretic garbling scheme* $\mathsf{Gb} = (\mathsf{Garble}, \mathsf{Eval})$ *for circuit class* $\mathcal{F} = \{\mathcal{F}_\lambda\}_\lambda$ *(looking ahead, we will work with* $\mathsf{poly}(\lambda)$ *sized circuits with* λ *input bits, and depth* $O(\log \lambda)$*) consists of the following algorithm.*

- $\mathsf{Garble}(1^\lambda, C; r) \to (\Gamma, \{\mathsf{Lab}_{b,i}\}_{b\in\{0,1\}, i\in[\lambda]})$: *The garbling algorithm takes as input a circuit* $C \in \mathcal{F}$*, and it outputs a garbled circuit* Γ *and input labels* $\{\mathsf{Lab}_{b,i}\}_{b\in\{0,1\},i\in[\lambda]}$.

  For any input $\mathbf{x}$*, we denote by* $\mathsf{Lab}_{\mathbf{x}}$ *the shorthand for* $\{\mathsf{Lab}_{x_i,i}\}_{i\in[\lambda]}$ *and* Lab *as the shorthand for* $\{\mathsf{Lab}_{b,i}\}_{b\in\{0,1\}}$.

- $\mathsf{Eval}(\Gamma, \{\mathsf{Lab}_{x_i,i}\}_{i\in[\lambda]}) \to z$: *The evaluation algorithm takes as input a garbled circuit* Γ*, and labels* $\{\mathsf{Lab}_{x_i,i}\}_{i\in[\lambda]}$ *for some input* $\mathbf{x} \in \{0,1\}^\lambda$*. It outputs* $z \in \{0,1\}^* \cup \perp$.

We require that such a scheme satisfies the following properties:
Correctness: Let $\lambda \in \mathbb{N}$*,* $C \in \mathcal{F}$ *and* $\mathbf{x} \in \{0,1\}^\lambda$*, then it holds that:*

$$\Pr_{\mathsf{Garble}(1^\lambda, C)\to\Gamma, \{\mathsf{Lab}_{b,i}\}_{b\in\{0,1\},i\in[\lambda]}} [\mathsf{Eval}(\Gamma, \{\mathsf{Lab}_{x_i,i}\}_{i\in[\lambda]}) = C(\vec{x})] = 1$$

Equivocation: Let $\lambda \in \mathbb{N}$*,* $C_0, C_1 \in \mathcal{F}$ *and* $\mathbf{x} \in \{0,1\}^\lambda$ *such that* $C_0(\mathbf{x}) = C_1(\mathbf{x})$*, then the following two distributions are identical.*

- *Distribution* 1*: Compute* $(\Gamma, \mathsf{Lab}) \leftarrow \mathsf{Garble}(1^\lambda, C_1; r)$*. Output* $(C_1, \Gamma, \mathsf{Lab}, r)$.

- *Distribution* 2*: Compute* $(\Gamma, \mathsf{Lab}) \leftarrow \mathsf{Garble}(1^\lambda, C_0; r)$*. Compute*

  $$\mathsf{GbEquiv}(\Gamma, \mathsf{Lab}_{\mathbf{x}}, C_1, \mathbf{x}) \to \mathsf{Lab}', r'$$

  such that $\mathsf{Lab}'_{x_i,i} = \mathsf{Lab}_{x_i,i}$ *for* $i \in [\lambda]$*. Output* $(C_1, \Gamma, \mathsf{Lab}', r')$.

Instantiation: To instantiate this, one can rely on the folklore instantiation of information-theoretic version of Yao's garbling scheme [Yao86] for NC^1 circuits, and in particular the point-of-permute formulation of the scheme [Yao86, BMR90].

CHAPTER 3

Non-Malleable Commitments

The goal of this chapter is to give a self-contained treatment of the state-of-the art in round-optimal non-malleable commitments (NMCs). We begin by briefly discussing some motivation for such commitments, along with some history of study of the topic. We then give a formal definition of CCA-non-malleable commitments, and a high-level explanation of how recent works achieve such commitments, giving special focus to the work of [Khu21]. Finally, we describe and prove security for our construction of NMCs, thus proving the following theorem.

Theorem 5. *Assume that the following assumptions hold:*

- *A subexponentially secure indistinguishability obfuscator exists,*
- *A subexponentially secure non-interactive witness-indistinguishable argument exists,*
- *A subexponentially secure quantum polynomial-time breakable non-interactive perfectly-binding commitment exists, and*
- *LWE is subexponentially secure against quantum adversaries of subexponential size.*

Then, there exists a one-round CCA commitment scheme (as in Definition 13) supporting a super-polynomial number of tags. The scheme is secure against adversaries of size polynomial in $2^{\lambda^{c(\log\log\lambda)^{-1}}}$ *for some* $c > 0$.

Since subexponentially secure quantum polynomial-time breakable non-interactive perfect-binding commitments are known from subexponential DDH or factoring, we achieve Theorem 3

from Chapter 1 as a corollary.

As a review, a commitment scheme is a protocol between two parties, a committer and a receiver. After the protocol, the committer can "open" the commitment by specifying the randomness used along with the committed value x. Commitment schemes are classically required to have two properties. The first property, called *hiding*, states that x should be hidden from the receiver before the opening is revealed. The second property, called *binding*, states that no committer should be able to compute openings to two different values for x. It is possible to have a commitment scheme which satisfies statistical (or perfect) hiding, or statistical (or perfect) binding, but not both. In this chapter, we focus on commitments for which perfect binding holds, or in other words, for any commitment transcript τ, there should exist a unique value x_τ which can be opened with respect to τ. In addition, we focus on commitment schemes which are secure in the plain model, without any form of trusted setup.

Consider the problem of implementing a blind auction [DDN91], where different parties must bid for an item without knowing the other parties' bids. A natural idea would be to have all parties simultaneously broadcast a commitment to their bid, and then once all commitments have been received by all parties, have all parties broadcast openings. Say Alice and Bob are carrying out this protocol, and imagine the following scenario. Bob wants to win the auction, but wants to pay as little as possible: he would prefer not to make an inflated bid just in order to outbid Alice. To do this, Bob waits for Alice's commitment, which is to some unknown value $\$x$. (In other words, Bob is a *rushing adversary*, as defined in Chapter 1). Then, *without knowing* x, Bob "mauls" Alice's commitment to get a commitment to $\$x + 1$, and publishes this commitment. Since Bob has not learned x, this does not contradict hiding of the commitment scheme. Finally, once Alice reveals a decommitment, Bob somehow uses the decommitment to $\$x$ to compute a decommitment of his mauled commitment to $\$x + 1$. He then publishes the decommitment, winning the auction with minimal cost. Such a scenario should clearly be avoided by any secure blind auction protocol, however the standard properties of commitment schemes do not rule it

out.[1] In fact, many classical commitment schemes are easily malleable. Take for instance the non-interactive commitment scheme from one-way permutations [Blu81, GL89]. Given a one-way permutation f, such commitments have the form (y, r, c), where $y = f(x)$ for some random $x \xleftarrow{\$} \{0,1\}^\lambda$, $r \xleftarrow{\$} \{0,1\}^\lambda$ is randomly chosen, and $c = \langle r, x \rangle \oplus m$, where m is the bit committed to, and $\langle r, x \rangle$ is the dot product of r and x in $\mathbb{Z}_2^\lambda$. Perfect binding follows from the fact that that f is a permutation, and computational hiding follows from the Goldreich-Levin theorem [GL89]. Given such a commitment, choose r' by flipping the first bit of r. Then with probability $1/2$ over the choice of x, the commitment (y, r', c) commits to the opposite bit of (y, r, c).

Defining non-malleable commitments. Non-malleable commitments are formulated explicitly to prevent such scenarios as the one above. In general, given a NMC c to a value x, an adversary should not be able to produce a new commitment c' to any value related to x. Of course, we must specify what "related to" means, since an adversary could simply produce a new commitment c' which is a verbatim copy of c. We deal with this definitional issue in the following way. In every NMC we consider, each party must commit with respect to some particular identity, which in the literature is commonly called a *tag*, and which is inextricably linked to the commitment. Each party must use its own tag. With this in mind, the intuitive guarantee that a NMC should provide is the following: given a commitment c which commits to x with respect to some tag tag, an adversary should not be able to produce a new commitment c' with tag $\mathsf{tag}' \neq \mathsf{tag}$ to x or to any value related to x.

There are many definitions of non-malleable commitments in the literature which capture this intuition formally in various ways. One simple definition is as follows. We first fix some notation. A commitment scheme consists of a probabilistic polynomial-time (PPT) committer

[1] This problem is solvable via normal (malleable) commitment schemes, at the expense of rounds: let the parties commit to their bids in lexicographic order, one per round. Once all parties have committed, have the parties open their commitments one per round, *in reverse lexicographic order*. This strategy is used in the classical coin-flipping protocol of [Blu81] and takes $2n$ rounds, where n is the number of parties.

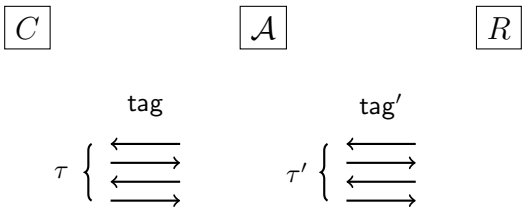

Figure 3.1: The non-malleability security experiment. Note that $\mathcal{A}$ may interleave messages in an arbitrary manner; the interleaving illustrated is one possible (trivial) interleaving.

C, a PPT receiver R, and a family of tag spaces $\{\mathcal{T}\}_\lambda$. At the beginning of the protocol, the committer takes as input the security parameter 1^λ, a message x, an identity $\mathsf{tag} \in \mathcal{T}_\lambda$, and randomness r. The receiver takes as input the security parameter 1^λ, the identity tag, and randomness r'. Denote by

$$\langle C(1^\lambda, \mathsf{tag}, x; r), R(1^\lambda, \mathsf{tag}; r')\rangle = \langle C, R\rangle(1^\lambda, \mathsf{tag}, x; r, r') = \tau$$

the transcript of the protocol with the inputs specified. To open the commitment, the committer publishes its randomness r. Recall that we are restricting ourselves to commitments which satisfy perfect binding; as such, there must exist some inefficient algorithm $\mathsf{Extract}$ which, on input a commitment transcript τ, identity tag, and receiver randomness r' outputs either the unique value x which can be opened with respect to τ, tag and r', or $\perp$ if no such value exists. In other words,

$$\mathsf{Extract}(\tau, \mathsf{tag}, r') = x \neq \perp \Leftrightarrow \exists r, \langle C, R\rangle(1^\lambda, \mathsf{tag}, x; r, r').$$

Security is defined with respect to a so-called *man-in-the-middle adversary* $\mathcal{A}$, which is a PPT interactive algorithm. $\mathcal{A}$ is initialized with input 1^λ and x. It interacts with an honest committer C, who either commits to x or $0^{|x|}$. $\mathcal{A}$ plays the role of the receiver in this interaction. It also interacts with an honest receiver R, playing the role of the committer. $\mathcal{A}$ is allowed to schedule the messages of both sessions in an arbitrary interleaved manner, and may deviate arbitrarily from the protocol in both sessions. Figure 3.1 illustrates the

interaction of $\mathcal{A}$ with C and R. The interaction with the committer is often referred to as the *left session*, and similarly the interaction with the receiver is referred to as the *right session*. At the end of the interaction, $\mathcal{A}$ outputs an arbitrary function of its view. Letting tag be the identity used in the left session and tag' the identity used in the right session, and assuming $\mathsf{tag} \neq \mathsf{tag}'$, define a random variable $\mathsf{expt}_{\mathsf{nmc}}(1^\lambda, x, \mathsf{tag}, \mathsf{tag}')$ to be the pair $(\mathsf{view}_{\mathcal{A}}, x')$, where $\mathsf{view}_{\mathcal{A}}$ is the output of $\mathcal{A}$ at the end of the interaction, and $x' = \mathsf{Extract}(\tau', \mathsf{tag}', r')$, where τ' is the transcript of the right session, tag' is the identity of the right session, and r' is the randomness used by R. For all x and for all $\{\mathsf{tag}_\lambda, \mathsf{tag}'_\lambda\}_\lambda$, where $\mathsf{tag}_\lambda \neq \mathsf{tag}'_\lambda$, it must be the case that $\{\mathsf{expt}_{\mathsf{nmc}}(1^\lambda, x, \mathsf{tag}_\lambda, \mathsf{tag}'_\lambda)\}_\lambda$ is computationally indistinguishable from $\{\mathsf{expt}_{\mathsf{nmc}}(1^\lambda, 0^{|x|}, \mathsf{tag}_\lambda, \mathsf{tag}'_\lambda)\}_\lambda$.

In the above security experiment, the adversary $\mathcal{A}$ participates in one left session as the receiver and one right session as the committer. This is called *one-one non-malleability*. A stronger definition, called *many-many non-malleability*, or *concurrent non-malleability*, has $\mathcal{A}$ interacting with an unspecified polynomial number of committers on the left and another unspecified polynomial number of receivers on the right, with a unique tag for each session. It is shown in [DDN91] that one-one NMCs also satisfy many-one non-malleability, and in [LPV08] that one-many NMCs also satisfy many-many non-malleability.

We consider one final definition, which is even stronger than many-many non-malleability, called *CCA-non-malleability* [CLP10]. In this definition, unlike the previous definitions, the adversary $\mathcal{A}$ interacts with a single challenger, with respect to some identity tag. The adversary chooses a message x, and plays the part of a receiver R in a session of $\{C, R\}(1^\lambda, \mathsf{tag}, \dots)$, either receiving a commitment to x or $0^{|x|}$. $\mathcal{A}$ also has access to a *committed value oracle* $\mathcal{O}$, which behaves as follows. At any time during the execution, $\mathcal{A}$ can initialize a session $\{C, R\}(1^\lambda, \mathsf{tag}', \dots)$ with $\mathcal{O}$, where $\mathsf{tag}' \neq \mathsf{tag}$, and where $\mathcal{A}$ plays the part of C. $\mathcal{O}$ behaves the same as the honest receiver R with some randomness r', and then at the end of the session returns the result of $\mathsf{Extract}(\tau, \mathsf{tag}', r')$ to $\mathcal{A}$. $\mathcal{A}$ may choose x adaptively based on the result of interactions with $\mathcal{O}$. Since this is the type of security we target for our NMC

construction, we will give a more formal definition in Section 3.1.

History of study. Non-malleable commitments were first introduced by Dolev, Dwork and Naor in their seminal work of [DDN91], which gave the first such construction, requiring $O(\log \lambda)$ rounds, where λ is the security parameter. Subsequently, [PR05a] gave the first construction which satisfies *concurrent* non-malleability, where the adversary can participate in many simultaneous sessions both as sender and receiver. Since then, much effort has been spent in understanding how many rounds are required for NMCs [Bar02, PR03, PR05b, LPV09, PW10, Wee10, LP11, Goy11, GLO12, Pas13, COS17a], resulting in four-round fully-concurrent protocols from one-way functions [COS17a] and three-round fully-concurrent protocols from subexponential one-way permutations [COS16], or from polynomial-secure zaps and various polynomial number-theoretic assumptions [Khu17]. In addition, the work of [PPV08] showed how to achieve *non-interactive* NMCs, however they relied on the existence of a very strong and unstudied primitive called an adaptive injective one-way function.

We briefly describe this assumption. The task is similar to that in normal one-way functions: given $y = f(x)$ for a randomly-chosen preimage x, an adversary must x. However, an adaptive one-way function additionally provides the adversary with an inversion oracle, which given any $y' \neq y$, inverts y'. Because of this oracle, this assumption is unfalsifiable: given an adversary $\mathcal{A}$ which claims to break the assumption for some given f, there is no way to efficiently test $\mathcal{A}$, because implementing the inversion oracle is by assumption computationally hard. In addition, the assumption itself already essentially incorporates non-malleability, so it is unsurprising that such a strong non-malleable hardness assumption yields a non-malleable commitment. For a long time, an important open question remained whether one or two-round NMCs were possible from more standard, falsifiable assumptions.

The work of [Pas13] seemed to answer this question, showing that it is impossible to construct two-round NMCs based on black-box reductions to polynomial-time falsifiable hardness assumptions. [Pas13] also seemingly ruled out NMCs even based on (black-box

reductions to) *sub-exponential* hardness assumptions.

In 2017, two exciting works [KS17, LPS17] bypassed this lower bound, achieving two-round NMCs from sub-exponential hardness assumptions. They did so by exploiting two different implicit assumptions. The work of [KS17] took advantage of the fact that the [Pas13] lower bound requires that the security reduction, which has black-box access to the NMC adversary, only queries the adversary polynomially many times. In contrast, the security proof in [KS17] invokes the adversary a sub-exponential number of times, and achieves security assuming sub-exponential Zaps, sub-exponential decisional diffie hellman, and sub-exponential one-way functions. The work of [LPS17] took advantage of the fact that [Pas13] assumes that regardless of what tags are being used in the left and right session, the reduction has one fixed complexity. In contrast, the [LPS17] construction uses two different axes of hardness, namely circuit size and depth, and the reduction runs in a different size and depth depending on the tag used in the right session. Security of [LPS17] is achieved assuming sub-exponential time-lock puzzles, sub-exponential zaps, sub-exponential collision-resistant hash functions, and sub-exponential non-interactive commitments.

Several subsequent results have built on the techniques of [KS17, LPS17] to improve the round complexity of NMCs even further [BL18, KK19, GKL21], albeit relying on stronger, less-standard assumptions. Notably, the work of [BL18] achieved the first construction of non-interactive NMCs based on falsifiable assumptions, relying on multi-collision-resistant keyless hash functions (among other more standard assumptions). As discussed in Chapter 1, this assumption, while falsifiable, is non-standard: it was first proposed within the last five years by the work of [BDR18], and has not had much history of study. Thus it is an important goal to achieve non-interactive NMCs using better assumptions.

The recent work of [Khu21] makes progress on this goal, constructing a special type of NMC relying on indistinguishability obfuscation (iO) along with other standard assumptions. The committer's message is an obfuscated program P, and the receiver obtains the "true" commitment by choosing a random string τ as input and then running the obfuscated program

with input τ. In this way, the committed value is determined by the pair (P, τ). Importantly, for non-malleability to hold, the receiver's τ must be chosen independently of the committer's message P; if the committer is allowed to see τ before choosing P then the proof fails.

Because of this, there are two possible ways to use this commitment. One option is that the receiver can publish its τ after receiving P. This achieves the full NMC security definition, however it takes two rounds. The other option is to have the receiver silently choose a τ without publishing it. When used in this way the NMC only takes one round, however it suffers from a problem known as "over-extraction" [LPS17], where the $\mathsf{Extract}$ algorithm sometimes outputs a non-$\perp$ value even though the commitment is invalid and has no opening. Because of this, it is only possible to achieve a weaker definition of security, which is called *non-malleability with respect to extraction*.

Our contribution. Our NMC construction is an improvement of [Khu21] which solves the problem above. Our construction takes the same form as that of [Khu21], namely, the committer publishes a message P, and the receiver publishes a random τ. We change the internals of the construction, though, to allow the receiver to publish τ during the first round, simultaneously while the committer is publishing P. We show that with our modifications, even a rushing committer who chooses P based on τ cannot break security. Thus we achieve a (simultaneous-message) one-round NMC which satisfies full CCA security, relying on iO and other standard assumptions.

The rest of the chapter is devoted to constructing and proving security of this NMC.

3.1 A Formal Definition of One-Round CCA-Non-Malleablility

We define the notion of one-round simultaneous-message CCA-non-malleable commitments which we aim to construct. For reference, we start with a formal statement of the standard definition of CCA security for non-interactive commitments. Afterwards, we state the

definition which we achieve, with some small differences in order to take advantage of the simultaneous-message model.

3.1.1 Non-Interactive CCA-Non-Malleable Commitments

Let $\mathcal{T} = \{\mathcal{T}_\lambda\}_{\lambda\in\mathbb{N}}$ be the tag space which is $[T(\lambda)]$, where $T = 2^{\mathsf{poly}(\lambda)}$.

Definition 9 (Syntax of Non-Interactive CCA-Non-Malleable Commitment)**.** *A noninteractive CCA-non-malleable commitment scheme for tag space $\mathcal{T}$ consists of the following algorithms.*

$\mathsf{CCACommit}(1^\lambda, \mathsf{tag}, m; r)$: *The probabilistic polynomial time commitment algorithm takes as input the security parameter λ, a tag $\mathsf{tag} \in \mathcal{T}_\lambda$, a message $m \in \{0,1\}^*$ and it outputs a commitment c.*

$\mathsf{ComputeOpening}(\tau, \mathsf{tag}, c, m, r)$: *The polynomial time deterministic algorithm* ComputeOpening *takes as input a string $\tau \in \{0,1\}^{\ell_t}$, a tag $\mathsf{tag} \in \mathcal{T}_\lambda$, a commitment c, a message $m \in \{0,1\}^*$, and the randomness r used to generate $c = \mathsf{CCACommit}(1^\lambda, \mathsf{tag}, m; r)$. It outputs the randomness r of the commit algorithm as the opening.*

$\mathsf{VerifyOpening}(\mathsf{tag}, c, m, r)$: *The polynomial time deterministic algorithm* VerifyOpening *takes a tag $\mathsf{tag} \in \mathcal{T}_\lambda$, a commitment c, a message $m \in \{0,1\}^*$, and an opening r. It outputs a value in $\{0,1\}$.*

Note that although this algorithm is trivial in the fully non-interactive setting, we will need a nontrivial ComputeOpening algorithm later in the one-round simultaneous-message setting, and thus it is useful to define it explicitly.

Such a scheme is said to be a noninteractive CCA-non-malleable commitment if it satisfies the following properties:

Definition 10 (Correctness of Opening)**.** *Let $\lambda \in \mathbb{N}$ be the security parameter, and consider any $\mathsf{tag} \in \mathcal{T}_\lambda$, any message $m \in \{0,1\}^*$, and any $c \leftarrow \mathsf{CCACommit}(1^\lambda, \mathsf{tag}, m; r)$. Then,*

$$\Pr[\mathsf{VerifyOpening}(\mathsf{tag}, c, m, r) = 1] = 1.$$

Definition 11 (Extraction)**.** *There exists an (inefficient) algorithm CCAVal with the following properties. For any $\lambda \in \mathbb{N}$ and any message $m \in \{0,1\}^*$, tag $\mathsf{tag} \in \mathcal{T}_\lambda$, and commitment c it holds that*

$$\Big(\exists r : \mathsf{VerifyOpening}(\mathsf{tag}, c, m, r) = 1\Big) \iff \mathsf{CCAVal}(\mathsf{tag}, c) = m.$$

In addition, CCAVal runs in time $2^{\mathsf{poly}(\lambda)}$ for some fixed polynomial poly.

We now specify the CCA security property. Note that as in the rest of this dissertation, security properties are parameterized by a class $\mathcal{C}$ of circuits against which this property holds, and an advantage ϵ.

Definition 12 ($(\mathcal{C}, \epsilon)$-CCA security)**.** *We define the following security game played between the adversary $\mathcal{A} \in \mathcal{C}$ and the challenger. We denote it by $\mathsf{expt}_{\mathcal{A},\mathsf{CCA}}(1^\lambda)$:*

1. *The adversary sends a challenge tag $\mathsf{tag}^* \in \mathcal{T}_\lambda$.*
2. *The adversary can submit arbitrary polynomially many queries (tag, c), for commitment c and $\mathsf{tag} \in \mathcal{T}_\lambda$. The challenger computes $\mathsf{CCAVal}(\mathsf{tag}, c)$ and sends the result to the adversary.*
3. *The adversary submits two messages $m_0, m_1 \in \{0,1\}^*$. The challenger samples $b \leftarrow \{0,1\}$, and computes $c^* \leftarrow \mathsf{CCACommit}(1^\lambda, \mathsf{tag}^*, m_b)$. The adversary gets c^* from the challenger.*
4. *The adversary repeats Step 2.*
5. *Finally, the adversary outputs a guess $b' \in \{0,1\}$. The experiment outputs 1 if $b' = b$ and 0 otherwise.*

A non-interactive CCA-NMC scheme satisfies $(\mathcal{C}, \epsilon)$-CCA *security if for all adversaries* $\mathcal{A} \in \mathcal{C}$*, it holds that*

$$\left| \Pr[\mathsf{expt}_{\mathcal{A},\mathsf{CCA}}(1^\lambda) = 1] - \frac{1}{2} \right| \leq \epsilon.$$

In the above security game, steps 2 and 4 implement the *committed value oracle* discussed in the introduction to this chapter.

3.1.2 One-Round Simultaneous-Message CCA-Non-Malleable Commitments

We now make some small modifications to the definition. First, we change the commitment to be a *simultaneous-message one-round commitment*, where both committer and receiver send a message during the single round. The receiver's message is a uniform random string τ, and, as mentioned in the introduction, the committer's message is some obfuscated program P. Second, $\mathsf{ComputeOpening}$, $\mathsf{VerifyOpening}$, and CCAVal now take both the committer's message P and the receiver's message τ as input. This reflects the fact that the committed value is only fixed when both P and τ are fixed. Finally, we change the CCA security game to take these changes into account. More specifically, we change the steps that implement the committed value oracle in order to handle the interactive nature of the protocol.

Let $\mathcal{T} = \{\mathcal{T}_\lambda\}_{\lambda \in \mathbb{N}}$ be the tag space which is $[T(\lambda)]$, where $T = 2^{\mathsf{poly}(\lambda)}$. The modified syntax is as follows.

Definition 13 (Syntax of one-round simultaneous-message CCA-non-malleable commitments)**.** *With respect to the tag space* $\mathcal{T}$*, the NMC consists of the following algorithms.*

$\mathsf{CCACommit}(1^\lambda, \mathsf{tag}, m; r)$: *The probabilistic polynomial time commitment algorithm takes as input the security parameter* λ*, a tag* $\mathsf{tag} \in \mathcal{T}_\lambda$*, and a message* $m \in \{0,1\}^*$*, and outputs a commitment* P*.*

$\mathsf{ComputeOpening}(\tau, \mathsf{tag}, \mathsf{P}, m, r)$: *The polynomial time deterministic algorithm* $\mathsf{ComputeOpening}$ *takes as input a string* $\tau \in \{0,1\}^{\ell_t}$*, a tag* $\mathsf{tag} \in \mathcal{T}_\lambda$*, a commitment* P*,*

a message $m \in \{0,1\}^*$, and the randomness r used to commit. It outputs an opening $\sigma \in \{0,1\}^*$. Above $\ell_t = \ell_t(\lambda, n)$ is a polynomial associated with the scheme.

$\mathsf{VerifyOpening}(\tau, \mathsf{tag}, \mathsf{P}, m, \sigma)$: *The polynomial-time deterministic algorithm* VerifyOpening *takes a string $\tau \in \{0,1\}^{\ell_t}$, a tag $\mathsf{tag} \in \mathcal{T}_\lambda$, a commitment P, a message $m \in \{0,1\}^*$, and an opening σ. It outputs a value in $\{0,1\}$.*

Such a scheme is said to be a one-round simultaneous-message CCA-non-malleable commitment if it satisfies the following properties:

Definition 14 (Correctness of Opening)**.** *Let $\lambda \in \mathbb{N}$ be the security parameter, and consider any $\mathsf{tag} \in \mathcal{T}_\lambda$, any message $m \in \{0,1\}^*$, any $\tau \in \{0,1\}^{\ell_t}$, any $\mathsf{P} \leftarrow \mathsf{CCACommit}(1^\lambda, \mathsf{tag}, m; r)$. Then,*

$$\Pr[\mathsf{VerifyOpening}(\tau, \mathsf{tag}, \mathsf{P}, m, \sigma) = 1] = 1,$$

where $\sigma = \mathsf{ComputeOpening}(\tau, \mathsf{tag}, \mathsf{P}, m, r)$.

Definition 15 (Extraction)**.** *There exists an (inefficient) algorithm* CCAVal *with the following properties. For any $\lambda \in \mathbb{N}$ and any message $m \in \{0,1\}^*$, tag $\mathsf{tag} \in \mathcal{T}_\lambda$, commitment P, and $\tau \in \{0,1\}^{\ell_t(\lambda)}$, it holds that*

$$\Big(\exists \sigma : \mathsf{VerifyOpening}(\tau, \mathsf{tag}, \mathsf{P}, m, \sigma) = 1\Big) \iff \mathsf{CCAVal}(\tau, \mathsf{tag}, \mathsf{P}) = m.$$

In addition, CCAVal *runs in time $2^{\mathsf{poly}(\lambda)}$ for some fixed polynomial* poly*.*

We now specify the CCA security property.

Definition 16 ($(\mathcal{C}, \epsilon)$-CCA security)**.** *We define the following security game played between the adversary $\mathcal{A} \in \mathcal{C}$ and the challenger. We denote it by $\mathsf{expt}_{\mathcal{A},\mathsf{CCA}}(1^\lambda)$:*

1. *The challenger manages a list L that is initially empty. The contents of the list are visible to the adversary at all stages.*

2. *The adversary sends a challenge tag* $\mathsf{tag}^* \in \mathcal{T}_\lambda$.

3. *The adversary submits queries of the following kind in an adaptive manner:*

   (a) *Adversary can ask for arbitrary polynomially many* τ*-queries. Challenger samples* $\tau' \leftarrow \{0,1\}^{\ell_t}$ *and appends* τ' *to* L.

   (b) *Adversary can ask for an arbitrary polynomially many* $(\tau, \mathsf{tag}, \mathsf{P})$*-queries for any* $\tau \in L$*, any* $\mathsf{tag} \neq \mathsf{tag}^*$*, and any commitment* P*. The challenger computes* $\mathsf{CCAVal}(\tau, \mathsf{tag}, \mathsf{P})$ *and sends the result to the adversary.*

4. *The adversary submits two messages* $m_0, m_1 \in \mathcal{M}_\lambda$*. The challenger samples* $b \leftarrow \{0,1\}$*, and computes* $\mathsf{P}^* \leftarrow \mathsf{CCACommit}(1^\lambda, \mathsf{tag}^*, m_b)$*. The adversary gets* P^* *from the challenger.*

5. *The adversary repeats Step 3.*

6. *Finally, the adversary outputs a guess* $b' \in \{0,1\}$*. The experiment outputs* 1 *if* $b' = b$ *and* 0 *otherwise.*

The one-round (simultaneous-message) CCA*-secure commitment scheme* CCA *scheme satisfies* $(\mathcal{C}, \epsilon)$-CCA *security if for all adversaries* $\mathcal{A} \in \mathcal{C}$*:*

$$\left| \Pr[\mathsf{expt}_{\mathcal{A},\mathsf{CCA}}(1^\lambda) = 1] - \frac{1}{2} \right| \leq \epsilon.$$

3.2 An Overview of the Construction

The NMCs of [LPS17, KS17, BL18, Khu21] all follow the same overall outline, first introduced by [LPS17] and [KS17]. We will follow the same general strategy, and so we introduce it here. There are three steps in this outline, which we describe below.

3.2.1 A small-tag commitment scheme

We first focus our attention on the number of tags which a commitment scheme supports. So far in our discussion of non-malleable commitments, we have ignored the size of the tag space $\mathcal{T}_\lambda$; however, this size is an important measure for how non-malleable a scheme is. Ideally, we want a commitment scheme to allow for a $\mathcal{T}_\lambda$ whose size is super-polynomial in λ, so that any unbounded polynomial number of tags are supported. For now, though, we focus on the weaker goal of constructing a scheme for which $|\mathcal{T}_\lambda|$ is much smaller than λ.

Let's assume we have a non-interactive commitment scheme $\mathsf{Com}(1^\lambda, m; r)$, such as the one mentioned in the introduction based on one-way permutations. If we assume that this commitment satisfies security against subexponentially-secure adversaries, then a very simple complexity-leveraging trick allows us to achieve "one-sided" non-malleable commitments. Consider the case of two tags. Choose λ_1 and λ_2 such that $\mathsf{Com}(1^\lambda_2, \cdot; \cdot)$ is hiding against 2^{λ_1}-sized adversaries, and define commitments under tag 1 to be $\mathsf{Com}(1^{\lambda_1}, \cdot; \cdot)$, and commitments under tag 2 to be $\mathsf{Com}(1^{\lambda_2}, \cdot; \cdot)$. Then given commitments c_1 and c_2, where c_1 is under tag 1 and c_2 is under tag 2, any circuit $\mathcal{C}$ of size $\mathsf{poly}(2^{\lambda_1})$ can completely break c_1 and extract the committed value, but c_2 still satisfies hiding against $\mathcal{C}$. Thus, given a polynomial-time adversary $\mathcal{A}$, we can run the CCA security game for challenge tag 2 in $\mathsf{poly}(2^{\lambda_1})$ time, where we brute-force extract the committed value for any oracle queries c_1 submitted by $\mathcal{A}$ with respect to tag 1. If $\mathcal{A}$ successfully wins the CCA game, then we have a $\mathsf{poly}(2^{\lambda_1})$-size circuit which breaks hiding of $\mathsf{Com}(1^{\lambda_2}, \cdot; \cdot)$. This extends to multiple tags as well, satisfying CCA security against tag i as long as the adversary only queries the committed value oracle on tags $i' < i$.

Before going on, it is useful to think about how many tags are possible with this scheme. If we have t tags, then there must be a complexity hierarchy $S_1 \ll B_1 = S_2 \ll B_2 \ll \ldots \ll S_t \ll B_t$, where S_i means the commitment for tag i is secure against circuits of this size, and B_i means that the commitment for tag i is broken in this size. Assume that $B_i = 2^\lambda$, or

in other words, the commitment scheme for tag i takes security parameter equal to λ, the security parameter for the overall NMC scheme.

Because we are assuming subexponential security, it must be the case that if $B_i = 2^{\lambda_i}$, then $S_i = 2^{\lambda_i^\epsilon}$ for some small constant $\epsilon < 1$. If we assume that this holds for a fixed ϵ for all i, then we have that

$$S_t = 2^{\lambda_1^{\left(\frac{1}{\epsilon}\right)^{t-1}}}$$

and thus, assuming $\mathsf{Com}(1^\lambda, \cdot; \cdot)$ is hiding against 2^{λ^ϵ}-sized adversaries, it must be the case that

$$\lambda_t \geq \lambda_1^{\left(\frac{1}{\epsilon}\right)^t}. \tag{3.1}$$

Since λ_t must be polynomial in the security parameter λ for the NMC, it is clear that t cannot even be anywhere close to linear in λ, let alone super-polynomial in λ. It is possible to set t to be super-constant, as follows. Given a security parameter λ for the NMC, set λ_1 to be $\lambda^{(1/\log\log\lambda)}$. Then by (3.1), we have that

$$\lambda_t \geq \lambda_1^{\left(\frac{1}{\epsilon}\right)^t} = \lambda^{\frac{\left(\frac{1}{\epsilon}\right)^t}{\log\log\lambda}}.$$

Setting $t = c' \log\log\log\lambda$ means that $\frac{\left(\frac{1}{\epsilon}\right)^t}{\log\log\lambda} \leq c$ for some constant c, and thus λ_t can be set to be polynomial in λ. Thus it is possible to have $t = O(\log\log\log\lambda)$.

Of course, our goal is to have a full non-malleable commitment, not just a one-directional one. The work of [LPS17] introduced a clever and elegant idea for obtaining two-sided non-malleability. Their main idea is to use two separate axes of hardness. To explain this, we must introduce the idea of a *time-lock puzzle* introduced by [RSW96].

The authors of [RSW96] introduced the following specific computational problem: given some large semiprime $N = pq$, a group element $g \in \mathbb{Z}_N^*$, and some number d, compute

$$y = g^{2^{2^d}} \pmod{N}.$$

[RSW96] conjectured that, if λ is the number of bits required to represent N,any circuit which can recover y given (N, g, d) must have depth approximately $2^{\gamma d}$, for some fixed $\gamma < 1$,

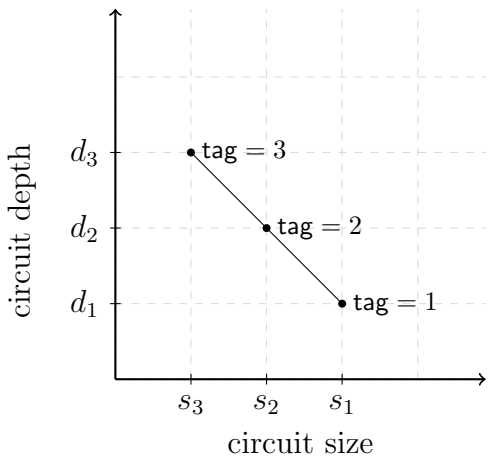

Figure 3.2: Two axes of hardness

or must be large enough to factor N. In other words, if we rule out trivial brute-force attacks, the problem is inherently sequential and non-parallelizeable. Time-lock puzzles are computational problems which follow this pattern: they are instantiated with two separate security parameters λ and d, with $d \ll \lambda$, where λ specifies the instance size (in other words, brute forcing the problem should take close to 2^λ) and d is the depth parameter. The assumption is that no circuit with depth less than $2^{\gamma d}$ can solve the problem, unless it is large enough to brute force the problem.

Assume that we have a commitment scheme $\mathsf{DCom}(1^\lambda, 1^d, m; r)$, which is based on time-lock-puzzles and which is "depth-robust," as described above. It is then possible to create two commitments c_1' and c_2' as before with Com, except now using depth: we can guarantee that c_1' is extractable in depth d_1, whereas c_2' hides the committed value for adversaries of depth $\mathsf{poly}(d_1)$, and is extractable in depth d_2. Using the two commitments Com and DCom for two different axes of hardness, size and depth, we can now specify a set of tags for which any two tags i and j, i is harder than j with respect to one axis, but j is harder than i with respect to another axis.

We do this as follows, as an example using three tags. Choose three depth parameters d_1,

d_2, and d_3, and three size parameters s_1, s_2, and s_3, such that $d_1 \ll d_2 \ll d_3 \ll s_3 \ll s_2 \ll s_1$ (see Figure 3.2). To compute a commitment for tag i and message m, compute random m_1, m_2 such that $m_1 \oplus m_2 = m$, and then use DCom to give a commitment c_i^d to m_1 which is extractable in depth d_i, and use Com to compute a commitment c_i^s to m_2 which is extractable in size s_i, and return (c_i^d, c_i^s). It is clear that to extract m, we must extract both c_i^d and c_i^s. Thus, for any tags $i < j$, it is possible to run the CCA game for an adversary who give challenge tag i and runs the oracle on commitments of tag j, by using a size $\mathsf{poly}(s_j)$ and depth $\mathsf{poly}(d_j)$ machine which can extract tag j but for which tag i commitments are hiding since $s_i \gg s_j$. On the other hand, if the adversary gives j as the challenge tag and queries tag i, a size s_i and depth d_j machine can extract tag i without breaking hiding of tag j, since $d_j \gg d_i$. Thus we have achieved two-sided non-malleability.

We now have a non-interactive non-malleable commitment with a small tag space, assuming the existence of time-lock puzzles and any perfectly-binding non-interactive commitment. This scheme has two problems, though. The first problem is that the argument above only works if the adversary only runs the committed value oracle for commitments with respect to one tag. This corresponds to a CCA version of one-one non-malleability, called *same-tag CCA non-malleability*. Note that if the adversary gives challenge tag 2 and queries the oracle for both tag 1 and 3, the extractor must run in size s_1 and d_3, and thus can also extract tag 2 as well.

The second problem is that for all known time-lock puzzles [RSW96, BGJ16] it is not possible to efficiently verify that a puzzle has been generated honestly. Because of this, the commitment DCom suffers from the problem of *over-extraction* which was discussed in the introduction. Namely, it can be possible for the extractor to extract a value even when the commitment is invalid and no opening to any value exists. In other words, the commitment above satisfies the requirements for a one-one CCA non-malleable commitment, except that in Definition 11, the implication is only in the forward direction. This is called *non-malleability with respect to extraction.*

So we need a transformation which achieves a larger tag space, which removes the same-tag restriction, and which solves the over-extraction problem. We dedicate the rest of the section to explaining how to do this.

3.2.2 Tag amplification

Our main contribution comes in the form of a *tag-amplification* construction, which compiles a commitment scheme with tag space $\mathcal{T}$, $|\mathcal{T}| = t$, into one with $\binom{t}{t/2}$ tags. At a very high level, we do the following. In the resulting commitment scheme, each tag T is of the form $\{s_1, \ldots, s_{t/2}\}$, where $s_i \in \mathcal{T}$ is a tag in the original scheme. A commitment $(\{c_{s_i}\}_{i\in[t/2]}, \pi)$ under tag T consists of commitments c_{s_i} under each tag s_i, along with a privacy-preserving proof π that all commitments are to the same underlying message. We call the commitments $\{c_{s_i}\}_{i\in[t/2]}$ the *inner-tag commitments*. Let us think about how we would reduce security of this scheme to security of the underlying inner-tag scheme. Intuitively, since we have a proof that all commitments are to the same value, during the CCA game we are not required to extract all the commitments, rather we only need to extract one c_{s_i} for each commitment $(\{c_{s_i}\}_{i\in[t/2]}, \pi)$. If the challenge tag is T^*, we can find an inner tag s_i for each query tag T such that $s_i \notin T^*$, and extract the corresponding commitment c_{s_i} when queried on tag T. Since we are not extracting any commitments with inner tags $s^* \in T^*$, we should be able to switch the challenge inner-tag commitments from m_0 to m_1 one by one, relying on CCA security of the inner-tag scheme. This strategy also happens to fix the problem of over-extraction: even if the inner scheme suffers from over-extraction, the resulting outer scheme does not, because soundness of the proofs guarantees that all inner commitments are well-formed.

This high-level approach was introduced in [KS17] and was additionally used in [BL18, Khu21]. The challenge with this strategy is to find a proof that has the privacy and round complexity requirements we need. Ideally, we want a zero-knowledge argument, so that we can simulate when switching each inner-tag commitment from m_0 to m_1. It is well-known

that non-interactive zero knowledge does not exist in the plain model; the main technical contributions of [BL18] and [Khu21] are in finding a way to get around this.

In the following, we describe the techniques of [Khu21], the issues in these techniques mentioned earlier, and how we solve them.

3.2.3 Khurana's Construction and Our Modifications

As stated before, the main technical contribution of [Khu21] is a tag-amplification procedure. Starting from a one-round CCA commitment scheme for small tags (say tags lie in $[T']$ where $T' = \log\log\lambda$), they build an "almost-one-round" scheme scheme with a much larger tag space (say supporting tags in $[T]$ where $T = T'^{\Omega(T')}$). This transformation can be applied once again on top of the resulting scheme to get a scheme supporting a super-polynomial number of tags. Thus, a constant number of applications suffice to construct a scheme for $2^{\Omega(\lambda)}$ tags. At the base level, we can use the scheme supporting $\log\log\log\lambda$ which was explained in Section 3.2.1, based on time-lock puzzles and perfectly-binding non-interactive commitments. (We explain how to address the problems of over-extraction and same-tag non-malleability in the inner commitment later.)

We first explain what is meant by "almost-one-round." In the scheme of [Khu21], the committer's message is an obfuscated program P. The receiver's message is a uniform random string τ, which is used as input to P in order to verify the commitment. Importantly, the committed value is only fixed once both P and τ are fixed; in particular, the committer's opening is computed based on τ as well as P. Also, for security to hold, P cannot be chosen after seeing τ. Because of this, the receiver's message must be sent in round 2. If one is willing to accept a weaker security guarantee, it is possible to have the receiver compute τ privately in her head, and to also carry out the verification of the commitment privately. The committer's opening can then be the randomness used to generate the obfuscation. However,

using the commitment in this way introduces the possibility of over-extraction,[2] because it is impossible to test for a well-formed obfuscation based on a polynomial number of queries to the obfuscated program. In this case, it is only possible to achieve the weaker notion of non-malleability with respect to extraction which was discussed above.

We now describe the scheme of [Khu21] in more detail. The tag-amplification procedure makes use of a base commitment scheme $\mathsf{nmc} = (\mathsf{CCACommit}, \mathsf{CCAVal})$ for small tags in $[T']$ where $T' = \log\log\lambda$, an indistinguishability obfuscator iO, a public-key encryption scheme PKE with dense public keys, a non-interactive witness indistinguishable proofs NIWI, a puncturable PRF PPRF, and a one-way permutation $\mathsf{OWP} : \{0,1\}^{\ell_{\mathsf{OWP}}} \to \{0,1\}^{\ell_{\mathsf{OWP}}}$ (actually a one-way function with verifiable range suffices, but we describe using a permutation for simplicity).

The scheme follows a variant of the general strategy for tag amplification given in Section 3.2.2. The tag space of the resulting scheme consists of subsets of $[T']$ of size exactly $T'/2$. Thus, $T = \binom{T'}{T'/2}$. The idea is the following: to commit to a message m with respect to $\mathsf{tag} \in [T]$, parse tag as $(t_1, \ldots, t_{T'/2})$ where each $t_i \in [T']$. Then, the commitment simply consists of an iO obfuscation of the program described in Figure 3.3, where pk and the PPRF key K_{PPRF} are freshly sampled by the committer and hardwired into the program. When evaluated on a random input ρ, the obfuscated program returns a set of inner-tag commitments along with a special "trapdoor commitment" c_0 along with a proof either that they are consistent or that c_0 commits to $\mathsf{OWP}^{-1}(\rho)$.

Recall that the strategy given in Section 3.2.2 seems to require a zero-knowledge argument in order to work. Since one-message zero-knowledge does not exist, the hope is that generating commitments this way can be useful to revive this approach. As explained above, a receiver can evaluate the program on a randomly chosen input ρ to compute $(c_0, c_1, \ldots, c_{T'/2}, \pi)$. If π verifies, then this means that unless c_0 is an encryption of $\mathsf{OWP}^{-1}(\rho)$, $c_1, \ldots, c_{T'/2}$ must be

[2]We note that over-extraction of this commitment used in this way is possible even if the inner commitment does not suffer from over-extraction.

The Circuit $G[t_1, \ldots, t_{T'/2}, m, K_{\mathsf{PPRF}}, \mathsf{pk}]$

Hardwired: Tags $(t_1, \ldots, t_{T'/2}) \in [T']^{T'/2}$, Message m and PPRF key K_{PPRF}, public key pk,

Input: $\rho \in \{0,1\}^{\ell_{\mathsf{OWP}}}$

Computation:

1. Compute $r \leftarrow \mathsf{PPRF.Eval}(K_{\mathsf{PPRF}}, \rho)$ and parse $r = (r_1, r_{2,1}, \ldots, r_{2,T'/2}, r_3)$. Compute:
   - $c_0 = \mathsf{PKE.Enc}(\mathsf{pk}, 0^{\ell_{\mathsf{OWP}}}; r_1)$,
   - For $i \in [T'/2]$, compute $c_i = \mathsf{CCA'.CCACommit}(t_i, m; r_{2,i})$,
   - Compute $\pi = \mathsf{NIWI.P}(X, W; r_3)$ where $X = (c_0, c_1, \ldots, c_{T'/2})$ and $W = (m, \{r_{2,i}\}_{i \in [T']})$ are so that $(X, W) \in L_G$ for the language L_G defined below.
2. Output $(c_0, c_1, \ldots, c_{T'/2}, \pi)$.

Language $L_G = L_{G_1} \vee L_{G,2}$:

$$L_{G,1} = \{(c_0, c_1, \ldots, c_{T'/2}) \mid \exists x \in \{0,1\}^{\ell_{\mathsf{OWP}}} \; s.t. \; c_0 = \mathsf{Enc}(\mathsf{pk}, x) \wedge \mathsf{OWP}(x) = \rho\}$$

$$L_{G,2} = \{(c_0, c_1, \ldots, c_{T'/2}) \mid \exists m \; s.t. \; \forall i \in [T'/2], \; c_i = \mathsf{CCA'.CCACommit}(t_i, m)\}$$

Figure 3.3: The Circuit $G[t_1, \ldots, t_{T'/2}, m, k_{\mathsf{PPRF}}]$

well-formed nmc commitments of the same message m. To switch the challenge commitment from m_0 to m_1, one can go "input-by-input". For $\alpha \in [0, 2^{\ell_{\mathsf{OWP}}} - 1]$, we switch the obfuscated circuit to commit to m_1 as opposed to m_0 when the input $\rho \leq \alpha$. To do so, we need to hardwire non-uniformly $\beta = \mathsf{OWP}^{-1}(\alpha)$ into the reduction at each hybrid so that the reduction can generate c_0 by encrypting β and using it to prove NIWI. For the security arguments to go through, it requires that the public-key encryption, NIWI and the base commitments are more secure with an advantage of at least $2^{-\ell_{\mathsf{OWP}}}$.

This yields the following contradiction. On the one hand, public-key encryption needs to be more secure than the OWP to argue security. On the other hand, we need OWP to be secure against the time it takes to break c_0 to extract a pre-image of ρ chosen by the challenger to show that the adversary does not query the CCAVal algorithm on non-well formed commitments.

Nevertheless, [Khu21] observed that if the receiver randomness ρ's are chosen *after* declaring the set of commitments that would be queried to the CCAVal oracle, this issue can be handled via the following clever idea: the reduction can guess the secret-keys $\{\mathsf{sk}_i\}$ associated with the public keys $\{\mathsf{pk}_i\}$ used in the commitment programs $\{P_i\}$ chosen by the adversary. If a program P_i produces "bad" outputs $(c_0, c_1, \ldots, c_{T'/2}, \pi)$ on a large fraction of points ρ, then one can recover inverses of $\mathsf{OWP}^{-1}(\rho)$ using a non-uniformly fixed secret key sk. This gives a non-uniform reduction to the security of OWP.

There is another reason why the construction of [Khu21] only provides security if the receiver randomness is chosen only after all commitments P_i for which $\mathsf{CCAVal}(\star)$ may be queried are displayed the adversary. The reason is that on the one hand, nmc needs to be more secure than OWP to argue indistinguishability; on the other hand, OWP needs to be secure against the circuit that can run $\mathsf{nmc.CCAVal}(\star)$ to handle CCAVal queries in the reduction which can be done in any order. This problem does not arise if adversary outputs commitments P_i for which CCAVal is queried up front: the reduction never really has to run CCAVal to recover inverses to OWP challenge as the commitments P_i are already revealed.

In summary, the above idea only works in the setting where receiver randomness is chosen after the adversary displays all commitments for which CCAVal may be queried. Still, unfortunately, it fails in our setting, where receiver randomness can be sampled uniformly, at any point, any number of times the adversary demands. Our main idea is to introduce a new axis of hardness. We use quantum-classical tradeoffs. We replace the public key encryption with a perfectly binding quantum polynomial-time breakable commitment scheme and rely on nmc schemes where $\mathsf{nmc.CCAVal}$ runs in quantum polynomial time.

How does this help? Consider that the OWP is quantum secure, then if the adversary submits a commitment P and a query $\mathsf{CCAVal}(\rho, \mathsf{tag}, P)$ such that $P[\rho] = (c_0, c_1, \ldots, c_{T'/2}, \pi)$ where π verifies, and $c_1, \ldots, c_{T'/2}$ are not consistent and well-formed, then one can form an efficient quantum adversary that runs in time polynomial in the time of the adversary that breaks the OWP security. The idea is that for this to happen, c_0 must be a commitment of $\mathsf{OWP}^{-1}(\rho)$ due to the perfect soundness of NIWI. Then, c_0 can be simply inverted by running a quantum polynomial-time extractor of the commitment. Note that the reduction also needs to respond to CCAVal queries while interacting with the adversary, but those can also be run in quantum polynomial time.

In the classical world, commitments to compute c_0 and nmc, NIWI and other primitives can be made to be more secure than OWP to go input by input and argue security of the commitment scheme. This brings us to one last issue. Except we are not aware of a quantum secure one-way permutation from well-studied assumptions. To deal with this issue, we observe that we could have also used a quantum secure collision-resistant hash function, where the keys are randomly chosen (such are known via the small-integer solution/LWE problems). In this case, c_0 will be used to commit to a collision in the hash function.

Technical Remarks. Before we describe our construction, we mention a technical issue. The underlying non-malleable commitments such as [BL18, KK19] have two issues that we have to deal with:

- As mentioned before, they satisfy security with one-tag restriction, and,
- To support $\Omega(\log\log\lambda)$, assuming subexponential security of the underlying assumptions, these schemes are only quasi-polynomially secure. Thus, our transformation should work with those parameters.

Our transformation below actually works with a quasi-polynomially secure base commitment. For the first problem, we follow as in [Khu21], and take a one-round nmc with one-tag restriction and convert it to a (simultaneous-message) one-round scheme for the same number of tags, but without this restriction. This transformation is extremely similar to our tag-amplification, and we sketch this in Section 3.3.1. We now describe our construction.

3.3 The Formal Construction and Security Proof

Our construction is nearly identical to the construction provided by [Khu21] except for a few important changes, which help us to address the shortcomings in the scheme of [Khu21], mentioned above.

As a starting point, we make use of a one-round CCA commitment CCA' with security parameter $\lambda_{\mathsf{CCA}'}$ for small tag space $T'(\lambda_{\mathsf{CCA}'})$. The tag-space $T'(\lambda_{\mathsf{CCA}'})$ is at least $\underbrace{\log\ldots\log(\lambda_{\mathsf{CCA}'})}_{O(1)\text{ times}}$ and at most $\lambda^{O(1)}$. We now describe the circuit class against which security holds and other properties involved:

- The scheme satisfies $(\mathcal{C}_{\mathsf{CCA}'}, \epsilon_{\mathsf{CCA}'})$-security where $\mathcal{C}_{\mathsf{CCA}'}$ consists of all circuits of size polynomial in $\frac{1}{\epsilon_{\mathsf{CCA}'}}$ where $\epsilon_{acc'} = 2^{\lambda^{c_1(\log\log\lambda_{\mathsf{CCA}'})^{-1}}}$ where $c_1 > 0$ is some constant.
- $\mathsf{CCA}'.\mathsf{CCAVal}(\star)$ runs in quantum polynomial time.
- Let $\ell_{\mathsf{CCA}'}(\lambda_{\mathsf{CCA}'})$ be the length of the string τ chosen by the receiver,

At the end of a single step of this transformation, we will build the scheme CCA scheme with security parameter λ. We set $\lambda_{\mathsf{CCA}'} = \lambda$. After a single step transformation, the resulting

scheme will be secure against adversaries of size polynomial in $s_{\mathsf{CCA}} = 2^{\lambda^{c_2}(\log\log\lambda)^{-1}}$ with advantage bounded by $\epsilon_{\mathsf{CCA}} = 2^{-\lambda^{c_2}(\log\log\lambda)^{-1}}$ for some other constant $c_2 > 0$. $\mathsf{CCA.CCAVal}(\star)$ will also be quantum polynomial-time implementable. The resulting tag space will be $T(\lambda)$ where $T = T'^{\Omega(T'/2)}$. As a result of applying the procedure a constant number of times, we get a super-polynomial number of tags. At the base level, this can be instantiated by taking the scheme of [LPS20, BL18], which satisfies CCA security with one-tag restriction, and then applying the transformation as given in Section 3.3.1 to give a one round scheme CCA' without this restriction. The scheme in [LPS20, BL18] can be instantiated using iterated squaring assumption and the DDH (or SXDH over bilinear maps) assumptions, both of which are polynomial-time quantum broken.

Required Primitives. We make use of the following primitives and instantiate them with the following parameters. These instantiated parameters for the primitives we use are loose for what we require.

- *One-round CCA commitments*: We require a one-round CCA commitment CCA' with security parameter $\lambda_{\mathsf{CCA}'}$ for small tag space $T'(\lambda_{\mathsf{CCA}'})$, satisfying the properties described above.

- *Perfectly-Binding Quantum Extractable Commitment*: We require a perfectly biding commitment scheme NICom, which is extractable in quantum polynomial time. Further, it takes as input λ_{NICom}, and guarantees $2^{-\lambda_{\mathsf{NICom}}}$ indistinguishability against adversaries of size polynomial in $2^{\lambda_{\mathsf{NICom}}}$. Such commitments can be built using e.g. subexponential hardness of DDH or subexponential hardness of factoring.

- *Quantum-Secure Collision Resistant Hash Functions with random keys*: We require a sub-exponentially secure family of hash functions $\{\mathcal{H}_{\lambda_h} : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}\}_{\lambda_h \in \mathbb{N}}$, where the key space is $\mathcal{K} = \{0,1\}^{\ell_{h_{key}}}$, input space is $\mathcal{X} = \{0,1\}^{\ell_{h_{inp}}}$, and the output space is

$\mathcal{Y} = \{0,1\}^{\ell_{h_{out}}}$. Above $\ell_{h_{inp}}, \ell_{h_{out}}, \ell_{h_{key}}$ are polynomials in λ_h. We require the following additional properties:

  - For every key $K \in \mathcal{K}$ there exists unequal $x, x' \in \mathcal{X}$ such that $\mathcal{H}(K,x) = \mathcal{H}(K,x')$.
  - When $K \leftarrow \mathcal{K}$, then for any quantum algorithm running in time polynomial in 2^{λ_h}, the advantage in finding the collisions is bounded by $2^{-\lambda_h}$

  We set λ_h as follows. Let λ_h be such that $\ell_{h_{key}} = \lambda^{c_3(\log\log\lambda)^{-1}}$ where $c_3 = c_1/100$. This means that $\lambda_h = \lambda^{c_3'(\log\log\lambda)^{-1}}$ for some $c_3' < c_3$. In the resulting scheme, c_2 can be arbitrary constant less than c_3'.

- *Indistinguishability Obfuscation*: We require an indistinguishability Obfuscator iO. This scheme uses λ_{iO} as the security parameter and is secure against adversaries of size $2^{\lambda_{\mathsf{iO}}}$ with advantage $2^{-\lambda_{\mathsf{iO}}}$. Such a primitive can be built using well-studied assumptions as shown in [JLS21b, JLS21a].

- *Puncturable PRF*: We require a puncturable PRF, $\mathsf{PPRF} = (\mathsf{Puncture}, \mathsf{Eval})$. Assume the length of the key is randomly chosen of length $\ell_{\mathsf{PPRF}}(\lambda_{\mathsf{PPRF}})$ where λ_{PPRF} is its security parameter. The length of the output is some polynomial implicit in the scheme. We assume that the PPRF is secure against adversaries of size polynomial in $2^{\lambda_{\mathsf{PPRF}}}$ with a maximum advantage of $2^{-\lambda_{\mathsf{PPRF}}}$. This can be built from the subexponential hardness of LWE.

- *NIWI*: We require a non-interactive witness indistinguishable proof NIWI for NP, that is secure against adversaries of size polynomial in $2^{\lambda_{\mathsf{NIWI}}}$ with advantage bounded by $2^{-\lambda_{\mathsf{NIWI}}}$. This primitive can be built assuming subexponentially hard SXDH over Bilinear Maps.

We set $\lambda_{\mathsf{iO}} = \lambda_{\mathsf{NIWI}} = \lambda_{\mathsf{PPRF}} = \lambda_{\mathsf{NICom}}$ as a large enough polynomial. In particular, setting $2^{\lambda_{\mathsf{iO}}} \gg \mathsf{Time}(\mathsf{CCA}'.\mathsf{CCAVal}(\star)) \cdot 2^{\lambda}$ suffices.

Note that all the primitives described above exist from the primitives listed in Theorem 5.

The construction. We define the tag space T, as in [Khu21], to be the set $T = \binom{T'}{T'/2}$ which is precisely equal to the number of unique subsets of $[T']$ of size $[T'/2]$. Let ϕ be a polynomial time computable bijective map that takes as input $\mathsf{tag} \in [T]$, and outputs a unique subset $\{t_1, \ldots, t_{T'/2}\}$ of $[T']$ of size $T'/2$. These subsets are unique upto permutation. We assume that they are sorted in ascending order.

$\mathsf{CCA.CCACommit}(\mathsf{tag}, m; r)$: Compute the following steps.

- Compute $\phi(\mathsf{tag}) = (t_1, \ldots, t_{T'/2})$. Sample a PPRF key $K_{\mathsf{PPRF}} \leftarrow \{0,1\}^{\ell_{\mathsf{PPRF}}}$,
- Compute $\widetilde{G} \leftarrow \mathsf{iO}(G[t_1, \ldots, t_{T'/2}, m, K_{\mathsf{PPRF}}])$ by obfuscating the circuit described in Figure 3.4. Output $\widetilde{G}$.

$\mathsf{CCA.ComputeOpening}(\tau, \mathsf{tag}, \widetilde{G}, m, r)$: Compute the following steps.

- Parse $\tau = (\rho, \rho')$ where $\rho \in \{0,1\}^{\ell_{h_{key}}}$ and $\rho' \in \{0,1\}^{\ell_{\mathsf{CCA}'}}$,
- Compute $\phi(\mathsf{tag}) = (t_1, \ldots, t_{T'/2})$,
- Check if $\widetilde{G} = \mathsf{CCA.CCACommit}(\mathsf{tag}, m; r)$. Abort if its not the case. Derive the PPRF key K_{PPRF} used in code of G described in Figure 3.4.
- Compute $\widetilde{G}[\rho] = (c_0, c_1, \ldots, c_{T'/2}, \pi)$,
- From the code of Figure 3.4, use the PPRF key k_{PPRF} to derive r'_i as in the code such that $c_i = \mathsf{CCA}'.\mathsf{CCACommit}(t_i, m; r'_i)$. Compute and output $\sigma_i = \mathsf{CCA}'.\mathsf{ComputeOpening}(\rho', t_i, c_i, m, r'_i)$ for $i \in [T'/2]$.

$\mathsf{CCA.VerifyOpening}(\tau, \mathsf{tag}, \widetilde{G}, m, \sigma)$: Compute the following steps.

- Parse $\tau = (\rho, \rho')$ where $\rho \in \{0,1\}^{\ell_{h_{key}}}$ and $\rho' \in \{0,1\}^{\ell_{\mathsf{CCA}'}}$,
- Compute $\phi(\mathsf{tag}) = (t_1, \ldots, t_{T'/2})$ and $\sigma = (\sigma_1, \ldots, \sigma_{T'/2})$,
- Compute $\widetilde{G}[\rho] = (c_0, c_1, \ldots, c_{T'/2}, \pi)$ and verify π using $\mathsf{NIWI.Vf}$ for the language described in Figure 3.4. Abort if the proof does not verify,

- Output 1 if for every $i \in [T'/2]$, $\mathsf{CCA}'.\mathsf{VerifyOpening}(\rho', t_i, c_i, m, \sigma_i)$. Output $\perp$ otherwise.

The Circuit $G[t_1, \ldots, t_{T'/2}, m, K_{\mathsf{PPRF}}]$

Hardwired: Tags $(t_1, \ldots, t_{T'/2}) \in [T']^{T'/2}$, Message m and PPRF key K_{PPRF},

Input: $\rho \in \{0,1\}^{\ell_{h_{key}}}$

Computation:

1. Compute $r \leftarrow \mathsf{PPRF.Eval}(K_{\mathsf{PPRF}}, \rho)$ and parse $r = (r_1, r_{2,1}, \ldots, r_{2,T'/2}, r_3)$. Compute:
   - $c_0 = \mathsf{NICom}(0^{\ell_{h_{inp}}}; r_1)$,
   - For $i \in [T'/2]$, compute $c_i = \mathsf{CCA}'.\mathsf{CCACommit}(t_i, m; r_{2,i})$,
   - Compute $\pi = \mathsf{NIWI.P}(X, W; r_3)$ where $X = (c_0, c_1, \ldots, c_{T'/2})$ and $W = (m, \{r_{2,i}\}_{i \in [T']})$ are so that $(X, W) \in L_G$ for the language L_G defined below.
2. Output $(c_0, c_1, \ldots, c_{T'/2}, \pi)$.

Language $L_G = L_{G_1} \vee L_{G,2}$:

$$L_{G,1} = \{(c_0, c_1, \ldots, c_{T'/2}) \mid \exists x \neq x' \in \{0,1\}^{\ell_{h_{in}}} \ s.t. \ \mathcal{H}(\rho, x) = \mathcal{H}(\rho, x') \wedge c_0 = \mathsf{NICom}(x, x')\}$$

$$L_{G,2} = \{(c_0, c_1, \ldots, c_{T'/2}) \mid \exists m \ s.t. \ \forall i \in [T'/2], \ c_i = \mathsf{CCA}'.\mathsf{CCACommit}(t_i, m)\}$$

Figure 3.4: The Circuit $G[t_1, \ldots, t_{T'/2}, m, k_{\mathsf{PPRF}}]$

We now argue various properties involved. The correctness of opening is immediate due to the correctness of opening of the underlying commitment scheme CCA' and correctness and completeness of other primitives involved. To argue the extraction property, we now

describe the $\mathsf{CCA.CCAVal}$ algorithm.

$\mathsf{CCA.CCAVal}(\tau, \mathsf{tag}, \widetilde{G})$: Compute the following steps.

- Parse $\tau = (\rho, \rho')$ where $\rho \in \{0,1\}^{\ell_{h_{key}}}$ and $\rho' \in \{0,1\}^{\ell_{\mathsf{CCA}'}}$,
- Compute $\phi(\mathsf{tag}) = (t_1, \ldots, t_{T'/2})$,
- Compute $\widetilde{G}[\rho] = (c_0, c_1, \ldots, c_{T'/2}, \pi)$ and verify π using $\mathsf{NIWI.Vf}$ for the language described in Figure 3.4. Abort if the proof does not verify.
- Assuming the proof verifies, break open c_0 to recover x, x'. Check if $\mathcal{H}(\rho, x) = \mathcal{H}(\rho, x')$ and that $x \neq x'$. If this is true, abort.
- Assuming we have not yet aborted, output m if $m = \mathsf{CCA}'.\mathsf{CCAVal}(\rho', t_1, c_1) = \ldots = \mathsf{CCA}'.\mathsf{CCAVal}(\rho', t_{T'/2}, c_{T'/2})$.

The extraction property then follows immediately from the extraction property of the underlying CCA' scheme. The idea is that in the last step, if $m = \mathsf{CCA}'.\mathsf{CCAVal}(\rho', t_1, c_1) = \ldots = \mathsf{CCA}'.\mathsf{CCAVal}(\rho', t_{T'/2}, c_{T'/2})$, then there exists openings $\sigma_1, \ldots, \sigma_{T'/2}$, that opens $(c_1, \ldots, c_{T'/2})$ to m due to the extraction property of CCA'. Similarly, the reverse is also true. Note that even if CCA' suffers from over-extraction, CCA does not, because the NIWI proof guarantees that the inner commitments are well-formed. Note that assuming $\mathsf{CCA}'.\mathsf{CCAVal}$ is a quantum-polynomial-time algorithm, $\mathsf{CCA.CCAVal}$ is as well, since c_0 is quantum-polynomial-time-broken.

We now move on to the security proof.

3.3.0.1 Security Proof

The security proof can be structured by giving indistinguishable hybrids. The first one corresponds to the game where the challenger computes $\mathsf{CCA.CCACommit}(\mathsf{tag}^*, m_b)$ for a

random b, whereas the last hybrid is independent of b. We describe the first hybrid elaborately, and in later ones, we merely describe the change.

Hybrid_0 : In this hybrid,

1. The challenger manages a list L that is initially empty. The contents of the list are visible to the adversary at all stages.

2. The adversary sends a challenge tag $\mathsf{tag}^* \in \mathcal{T}_\lambda$.

3. The adversary submits queries of the following kind in an adaptive manner:

   (a) Adversary can ask for arbitrary polynomially many τ-query. Challenger samples $\tau' \leftarrow \{0,1\}^{\ell_{\mathsf{CCA}}}$ and appends τ' to L.

   (b) Adversary can ask for an abitrary polynomially many $(\tau, \mathsf{tag}, \mathsf{P})$-query for any $\tau \in L$, any $\mathsf{tag} \neq \mathsf{tag}^*$, and any commitment P. The challenger computes $\mathsf{CCAVal}(\tau, \mathsf{tag}, \mathsf{P})$ and sends the result to the adversary.

4. The adversary submits two messages m_0, m_1 of equal length. The challenger samples $b \leftarrow \{0,1\}$, and computes $\mathsf{P}^* \leftarrow \mathsf{CCA.CCACommit}(\mathsf{tag}^*, m_b)$. The adversary gets P^* from the challenger.

5. The adversary repeats Step 3.

6. Finally, the adversary outputs a guess $b' \in \{0,1\}$. The experiment outputs 1 if $b' = b$ and 0 otherwise.

$\mathsf{Hybrid}_{1,j\in[0,Q]}$: This hybrid is the same as the previous hybrid, except that we modify how the CCAVal queries corresponding to j^{th} τ query is responded. Recall that the challenger maintains a list L, and every time the adversary makes a τ query, a randomly sampled τ is added to this list. In this hybrid, let τ_j be the sampled τ the j^{th} $\tau-$query made by the adversary. In this hybrid we replace how CCAVal query is responded for $\mathsf{CCAVal}(\tau_j, \mathsf{tag}, \mathsf{P})$ for $\mathsf{tag} \neq \mathsf{tag}*$ and $\tau_j \in L$. The new code is defined as follows.

$\mathsf{CCA.CCAVal}^*(\tau_j, \mathsf{tag}, \mathsf{P})$: Compute the following steps.

- Parse $\tau_j = (\rho, \rho')$ where $\rho \in \{0,1\}^{\ell_{h_{key}}}$ and $\rho' \in \{0,1\}^{\ell_{\mathsf{CCA}'}}$,
- Compute $\phi(\mathsf{tag}) = (t_1, \ldots, t_{T'/2})$ and $\phi(\mathsf{tag}^*) = (t^*_1, \ldots, t^*_{T'/2})$,
- Since $\mathsf{tag} \neq \mathsf{tag}'$, there must exist a first index $i \in [T'/2]$ such that $t_i \neq \{t^*_1, \ldots, t^*_{T'/2}\}$.
- Compute $\mathsf{P}[\rho] = (c_0, c_1, \ldots, c_{T'/2}, \pi)$ and verify π using $\mathsf{NIWI.Vf}$ for the language described in Figure 3.4. Abort if the proof does not verify,
- Otherwise output m where $m = \mathsf{CCA}'.\mathsf{CCAVal}(\rho', t_i, c_i)$.

Note that $\mathsf{Hybrid}_{1,0}$ is the same as Hybrid_0. We now show that $\mathsf{Hybrid}_{1,j}$ is indistinguishable to $\mathsf{Hybrid}_{1,j+1}$. We show that if there is an adversary with size polynomial in 2^{λ_h} that distinguishes these hybrids with probability p, then there exists a (quantum) reduction that is running in time polynomial in $\mathsf{poly}(2^{\lambda_h})$, and wins in the collision-resistant hash function security game with probability p. Thus, showing that $p < 2^{-\lambda_h}$. We show our reduction.

- Reduction proceeds by maintaining a list L honestly,
- It generates all τ queries honestly, except that it for the $(j+1)^{th}$ query, it sets $\tau_{j+1} = (\rho, \rho')$ where ρ is received from the challenger of the hash function and ρ' is sampled randomly by the challenger.
- It answers CCAVal queries for every τ_i for $i \leq j$ using CCAVal^* (this is well defined because adversary does not use tag^*). It can be answered in quantum polynomial time as $\mathsf{CCA}'.\mathsf{CCAVal}$ can be implemented in quantum polynomial time.
- It answers CCAVal queries for every τ_i for $i > j+1$ using CCAVal. These queries can be answered in quantum polynomial time.
- For τ_{j+1} it does the following. Assume that the query is for $\mathsf{CCAVal}(\tau_{j+1}, \mathsf{tag}, \mathsf{P})$ for $\mathsf{tag} \neq \mathsf{tag}^*$. Then, do the following:

- Run $\mathsf{P}[\rho] = (c_0, c_1, \ldots, c_{T'/2}, \pi)$.
- Output $\perp$ if π does not verify. If it does, break open c_0 in quantum polynomial time to recover x, x'. Check if $\mathcal{H}(\rho, x) = \mathcal{H}(\rho, x')$ for $x \neq x'$. If this is true, output x, x' as the answer to the hash function challenger.
- Otherwise, output $\mathsf{CCA}'.\mathsf{CCAVal}(\rho', t_1, c_1)$ to the adversary.

Observe that because $\mathsf{CCA}'.\mathsf{CCAVal}$ runs in quantum polynomial time, the reduction runs in quantum time polynomial in s_{CCA}. Further observe, if $\mathcal{A}$ observes a difference between $\mathsf{Hybrid}_{1,j}$ and $\mathsf{Hybrid}_{1,j+1}$, then there must be a query of the form $\mathsf{CCAVal}(\tau_{j+1}, \mathsf{tag}, \mathsf{P})$ that produces different outputs. Parsing $\tau_{j+1} = (\rho, \rho')$, and $\mathsf{P}[\rho] = (c_0, c_1, \ldots, c_{T'/2}, \pi)$, this means that π verifies, but in the least there exists two indices i_1, i_2 such that $\mathsf{CCA}'.\mathsf{CCAVal}(\rho', t_{i_1}, c_{i_1}) \neq \mathsf{CCA}'.\mathsf{CCAVal}(\rho', t_{i_2}, c_{i_2})$. By soundness of NIWI, it means that c_0 must be a commitment of a collision for the hash key ρ. Thus, the reduction will succeed at that point.

Thus, we have the following claim.

Lemma 1. *Assuming that $\mathcal{H}$ is a secure against $\mathsf{poly}(2^{\lambda_h})$ sized quantum circuits, NIWI is sound, and CCA' satisfies perfect correctness/extraction properties, we have that for any adversary $\mathcal{A}$ of size $\mathsf{poly}(s_{\mathsf{CCA}})$ and for $j \in [0, Q-1]$, it holds that*

$$|\Pr[\mathcal{A}(\mathsf{Hybrid}_{1,j}) = 1] - \Pr[\mathcal{A}(\mathsf{Hybrid}_{1,j+1}) = 1]| \leq 2^{-\lambda_h}.$$

We now describe a series of hybrids. For $\alpha \in [0, 2^{\ell_{h_{key}}}]$.

$\mathsf{Hybrid}_{2,\alpha}$: This hybrid is the same as the previous hybrid, except that in order to generate P^*, we obfuscate the circuit in Figure 3.5. Namely, compute $\phi(\mathsf{tag}^*) = (t_1^*, \ldots, t_{T'/2}^*)$. Output $\mathsf{P}^* = \mathsf{iO}(G_1)$ where $G_1 = G_1[t_1^*, \ldots, t_{T'/2}^*, m_b, m_0, \alpha, K_{\mathsf{PPRF}}]$.

Note that the only difference between $\mathsf{Hybrid}_{1,Q}$ and $\mathsf{Hybrid}_{2,0}$ is how P^* is generated. In $\mathsf{Hybrid}_{1,Q}$, it is generated by obfuscating program $G[t_1^*, \ldots, t_{T'/2}^*, m_b, K_{\mathsf{PPRF}}]$, where is $\mathsf{Hybrid}_{2,0}$ it it generated by obfuscating program $G_1[t_1^*, \ldots, t_{T'/2}^*, m_b, m_0, 0, K_{\mathsf{PPRF}}]$. These programs

The Circuit $G_1[t_1, \ldots, t_{T'/2}, m_b, m_0, \alpha, K_{\mathsf{PPRF}}]$

Hardwired: Tags $(t_1, \ldots, t_{T'/2}) \in [T']^{T'/2}$, Messages m_b and m_0, PPRF key K_{PPRF}, and $\alpha \in [0, 2^{\ell_{h_{key}}}]$.

Input: $\rho \in \{0,1\}^{\ell_{h_{key}}}$.

Computation: The computation can be divided into two cases.

Case: $\rho < \alpha$

1. Compute $r \leftarrow \mathsf{PPRF.Eval}(K_{\mathsf{PPRF}}, \rho)$ and parse $r = (r_1, r_{2,1}, \ldots, r_{2,T'/2}, r_3)$.
2. Compute $c_0 = \mathsf{NICom}(0^{\ell_{h_{inp}}}; r_1)$ and for $i \in [T'/2]$, compute $c_i = \mathsf{CCA'.CCACommit}(t_i, m_0; r_{2,i})$,
3. Compute $\pi = \mathsf{NIWI.P}(X, W; r_3)$ where $X = (c_0, c_1, \ldots, c_{T'/2})$ and $W = (m_0, \{r_{2,i}\}_{i \in [T']})$ are so that $(X, W) \in L_G$ for the language L_G defined below.
4. Output $(c_0, c_1, \ldots, c_{T'/2}, \pi)$.

Case: $\rho \geq \alpha$

1. Compute $r \leftarrow \mathsf{PPRF.Eval}(K_{\mathsf{PPRF}}, \rho)$ and parse $r = (r_1, r_{2,1}, \ldots, r_{2,T'/2}, r_3)$.
2. Compute $c_0 = \mathsf{NICom}(0^{\ell_{h_{inp}}}; r_1)$ and for $i \in [T'/2]$, compute $c_i = \mathsf{CCA'.CCACommit}(t_i, m_b; r_{2,i})$,
3. Compute $\pi = \mathsf{NIWI.P}(X, W; r_3)$ where $X = (c_0, c_1, \ldots, c_{T'/2})$ and $W = (m_b, \{r_{2,i}\}_{i \in [T']})$ are so that $(X, W) \in L_G$ for the language L_G defined below.
4. Output $(c_0, c_1, \ldots, c_{T'/2}, \pi)$.

Language $L_G = L_{G_1} \vee L_{G,2}$:

$$L_{G,1} = \big\{(c_0, c_1, \ldots, c_{T'/2}) \mid \exists x \neq x' \in \{0,1\}^{\ell_{h_{in}}} \; s.t. \; \mathcal{H}(\rho, x) = \mathcal{H}(\rho, x') \wedge c_0 = \mathsf{NICom}(x, x')\big\}$$

$$L_{G,2} = \big\{(c_0, c_1, \ldots, c_{T'/2}) \mid \exists m \; s.t. \; \forall i \in [T'/2], \; c_i = \mathsf{CCA'.CCACommit}(t_i, m)\big\}$$

Figure 3.5: The Circuit $G_1[t_1, \ldots, t_{T'/2}, m_b, m_0, \alpha, k_{\mathsf{PPRF}}]$

have identical input-output behavior. Thus, if an adversary $\mathcal{A}$ distinguishes these hybrids with probability p, we can build a reduction that distinguishes iO with probability p. The reduction needs to invoke the code of $\mathcal{A}$ and answer polynomially many CCAVal queries. Therefore, its time is polynomial in the time of $\mathcal{A}$ and the time of CCAVal. We set λ_{iO} large enough to ensure the following claim.

Lemma 2. *Assuming that* iO *is secure against circuits that run in time* $\mathsf{poly}(2^{\lambda_{\mathsf{iO}}})$*, then for any adversary* $\mathcal{A}$ *of size polynomial in* $\mathsf{poly}(s_{\mathsf{CCA}})$*, it holds that:*

$$|\Pr[\mathcal{A}(\mathsf{Hybrid}_{1,Q}) = 1] - |\Pr[\mathcal{A}(\mathsf{Hybrid}_{2,\alpha}) = 1]| \leq 2^{-\lambda_{\mathsf{iO}}}.$$

Hybrid_3 : This hybrid is the same as the previous hybrid except to generate P^*, we obfuscate the circuit in Figure 3.4 by committing to m_0.

Note that the only difference between $\mathsf{Hybrid}_{2,2^{\ell_{h_{key}}}}$ and Hybrid_3 is how P^* is generated. In $\mathsf{Hybrid}_{2,2^{\ell_{h_{key}}}}$, it is generated by obfuscating program $G_1[t_1^*, \ldots, t_{T'/2}^*, m_b, m_0, \alpha = 2^{\ell_{h_{key}}}, K_{\mathsf{PPRF}}]$, where as in Hybrid_3 it it generated by obfuscating program $G[t_1^*, \ldots, t_{T'/2}^*, m_0, K_{\mathsf{PPRF}}]$. These programs have identical input output behavior. Thus if there exists an adversary $\mathcal{A}$ that distinguishes these hybrids with probability p, then we can build a reduction that distinguishes iO with probability p. The reduction needs to invoke the code of $\mathcal{A}$, and answer polynomially many CCAVal queries. Therefore, its time is polynomial in time of $\mathcal{A}$ and the time of CCAVal. We set λ_{iO} large enough to ensure the following claim.

Lemma 3. *Assuming that* iO *is secure against circuits that run in time* $\mathsf{poly}(2^{\lambda_{iO}})$*, then for any adversary* $\mathcal{A}$ *of size* $\mathsf{poly}(s_{\mathsf{CCA}})$*, it holds that:*

$$|\Pr[\mathcal{A}(\mathsf{Hybrid}_3) = 1] - |\Pr[\mathcal{A}(\mathsf{Hybrid}_{2,2^{\ell_{h_{key}}}}) = 1]| \leq 2^{-\lambda_{\mathsf{iO}}}.$$

Indistinguishability between $\mathsf{Hybrid}_{2,\alpha}$ and $\mathsf{Hybrid}_{2,\alpha+1}$ To prove indistinguishability between $\mathsf{Hybrid}_{2,\alpha}$ and $\mathsf{Hybrid}_{2,\alpha+1}$ we introduce indistinguishable intermediate hybrids.

Hybrid'_0 : This hybrid is the same as $\mathsf{Hybrid}_{2,\alpha}$.

Hybrid'_1 : This hybrid is the same as $\mathsf{Hybrid}_{2,\alpha}$ except that we generate P^* differently. We puncture the PPRF key K_{PPRF} at α, and hardwire the response at $\rho = \alpha$. Namely, compute the punctured key k^*_{PPRF} at α. We compute a value v as follows. Compute $(r_1, r_2, r_3) \leftarrow \mathsf{PPRF.Eval}(K_{\mathsf{PPRF}}, \alpha)$. Then:

- Compute $c_0 = \mathsf{NICom}(0^{\ell_{h_{inp}}}; r_1)$ and for $i \in [T'/2]$, compute $c_i = \mathsf{CCA}'.\mathsf{CCACommit}(t^*_i, m_b; r_{2,i})$,
- Compute $\pi = \mathsf{NIWI.P}(X, W; r_3)$ where $X = (c_0, c_1, \ldots, c_{T'/2})$ and $W = (m_b, \{r_{2,i}\}_{i\in[T']})$ are so that $(X, W) \in L_G$.
- Set $v = (c_0, c_1, \ldots, c_{T'/2}, \pi)$.

Output $\mathsf{P}^* = \mathsf{iO}(G_2)$ where $G_2 = G_2[t^*_1, \ldots, t^*_{T'/2}, m_b, m_0, \alpha, K^*_{\mathsf{PPRF}}, v]$ as described in Figure 3.6.

Note that the only difference between Hybrid'_0 and Hybrid'_1 is how P^* is generated. In Hybrid'_0, it is generated by obfuscating program $G_1[t^*_1, \ldots, t^*_{T'/2}, m_b, m_0, \alpha, K_{\mathsf{PPRF}}]$, where as in Hybrid'_1 it is generated by obfuscating $G_2[t^*_1, \ldots, t^*_{T'/2}, m_b, m_0, \alpha, K^*_{\mathsf{PPRF}}, v]$ where the key K^*_{PPRF} is punctured at α. Note that if PPRF key is correct at unpunctured points, these circuits have identical behavior on all inputs $\rho \neq \alpha$. On input $\rho = \alpha$, the outputs are made to be identical by setting $v = (c_0, c_1, \ldots, c_{T'/2}, \pi)$ which is computed as described in the Hybrid'_1 description. Thus, the security follows from the security of iO. We have that:

Lemma 4. *Assuming that* iO *is secure against circuits that run in time* $\mathsf{poly}(2^{\lambda_{iO}})$ *and* PPRF *is correct at unpunctured points, then for any adversary* $\mathcal{A}$ *of size* $\mathsf{poly}(s_{\mathsf{CCA}})$*, it holds that:*

$$|\Pr[\mathcal{A}(\mathsf{Hybrid}_3) = 1] - |\Pr[\mathcal{A}(\mathsf{Hybrid}_{2,2^{\ell_{h_{key}}}}) = 1]| \leq 2^{-\lambda_{\mathsf{iO}}}.$$

Hybrid'_2 : This hybrid is the same as the previous hybrid except while computing the hardwired value v, we replace $r_1, r_2, r_3 \leftarrow \mathsf{PPRF.Eval}(K_{\mathsf{PPRF}}, \alpha)$ to a truly random string.

The Circuit $G_2[t_1, \ldots, t_{T'/2}, m_b, m_0, \alpha, K_{\mathsf{PPRF}}, v]$

Hardwired: Tags $(t_1, \ldots, t_{T'/2}) \in [T']^{T'/2}$, Messages m_b and m_0, PPRF key K_{PPRF}, $\alpha \in [0, 2^{\ell_{h_{key}}}]$ and a value v.

Input: $\rho \in \{0,1\}^{\ell_{h_{key}}}$

Computation: The computation can be divided into two cases.

Case: $\rho < \alpha$

1. Compute $r \leftarrow \mathsf{PPRF.Eval}(K_{\mathsf{PPRF}}, \rho)$ and parse $r = (r_1, r_{2,1}, \ldots, r_{2,T'/2}, r_3)$.
2. Compute $c_0 = \mathsf{NICom}(0^{\ell_{h_{inp}}}; r_1)$ and for $i \in [T'/2]$, compute $c_i = \mathsf{CCA'.CCACommit}(t_i, m_0; r_{2,i})$,
3. Compute $\pi = \mathsf{NIWI.P}(X, W; r_3)$ where $X = (c_0, c_1, \ldots, c_{T'/2})$ and $W = (m_0, \{r_{2,i}\}_{i \in [T']})$ are so that $(X, W) \in L_G$ for the language L_G defined below.
4. Output $(c_0, c_1, \ldots, c_{T'/2}, \pi)$.

Case: $\rho > \alpha$

1. Compute $r \leftarrow \mathsf{PPRF.Eval}(K_{\mathsf{PPRF}}, \rho)$ and parse $r = (r_1, r_{2,1}, \ldots, r_{2,T'/2}, r_3)$.
2. Compute $c_0 = \mathsf{NICom}(0^{\ell_{h_{inp}}}; r_1)$ and for $i \in [T'/2]$, compute $c_i = \mathsf{CCA'.CCACommit}(t_i, m_b; r_{2,i})$,
3. Compute $\pi = \mathsf{NIWI.P}(X, W; r_3)$ where $X = (c_0, c_1, \ldots, c_{T'/2})$ and $W = (m_b, \{r_{2,i}\}_{i \in [T']})$ are so that $(X, W) \in L_G$ for the language L_G defined below.
4. Output $(c_0, c_1, \ldots, c_{T'/2}, \pi)$.

Case: $\rho = \alpha$ Output v.

Language $L_G = L_{G_1} \vee L_{G,2}$:

$$L_{G,1} = \{(c_0, c_1, \ldots, c_{T'/2}) \mid \exists x \neq x' \in \{0,1\}^{\ell_{h_{in}}} \ s.t. \ \mathcal{H}(\rho, x) = \mathcal{H}(\rho, x') \wedge c_0 = \mathsf{NICom}(x, x')\}$$

$$L_{G,2} = \{(c_0, c_1, \ldots, c_{T'/2}) \mid \exists m \ s.t. \ \forall i \in [T'/2], \ c_i = \mathsf{CCA'.CCACommit}(t_i, m)\}$$

Figure 3.6: The Circuit $G_2[t_1, \ldots, t_{T'/2}, m_b, m_0, \alpha, k_{\mathsf{PPRF}}, v]$

Thus, if an adversary $\mathcal{A}$ distinguishes these hybrids with probability p, we can build a reduction that breaks the pseudorandomness at the punctured points property of PPRF with probability p. The reduction needs to invoke the code of $\mathcal{A}$ and answer polynomially many CCAVal queries. Therefore, its time is polynomial in the time of $\mathcal{A}$ and the time of CCAVal. We set λ_{PPRF} large enough to ensure the following claim.

Note that the only difference between Hybrid'_1 and Hybrid'_2 is how $r = (r_1, r_2, r_3)$ is generated. In Hybrid'_1 it is generated by computing $\mathsf{PPRF}(K_{\mathsf{PPRF}}, \alpha)$ where as in Hybrid'_2 it is generated r is sampled randomly. Note the in both the hybrids, the key appears in a punctured form K^*_{PPRF}, punctured at α. Thus if there exists an adversary $\mathcal{A}$ that distinguishes these hybrids with probability p, then we can build a reduction that breaks the pseudorandomness at the punctured points property of PPRF with probability p. The reduction needs to invoke the code of $\mathcal{A}$ and answer polynomially many CCAVal queries. Therefore, its time is polynomial in the time of $\mathcal{A}$ and the time of CCAVal. We set λ_{PPRF} large enough to ensure the following claim.

Lemma 5. *Assuming that* PPRF *is secure against circuits that run in time* $\mathsf{poly}(2^{\lambda_{\mathsf{PPRF}}})$*, then for any adversary* $\mathcal{A}$ *of size* $\mathsf{poly}(s_{\mathsf{CCA}})$*, it holds that:*

$$|\Pr[\mathcal{A}(\mathsf{Hybrid}'_1) = 1] - |\Pr[\mathcal{A}(\mathsf{Hybrid}'_2) = 1]| \leq 2^{-\lambda_{\mathsf{PPRF}}}.$$

Hybrid'_3 : This hybrid is the same as the previous hybrid except that while we compute the hardwired value $v = (c_0, c_1, \ldots, c_{T'/2}, \pi)$, we switch the commitment as $c_0 = \mathsf{NICom}(x_\alpha, x'_\alpha)$ where $(x_\alpha, x'_\alpha) \in (\{0,1\}^{\ell_{h_{in}}})^2$, $x_\alpha \neq x'_\alpha$ and $\mathcal{H}(\alpha, x_\alpha) = \mathcal{H}(\alpha, x_\alpha)$.

Note that the only difference between Hybrid'_2 and Hybrid'_3 is how hardwiring $v = (c_0, c_1, \ldots, c_{T'/2}, \pi)$ is generated. In particular, it is about how c_0 is generated. In Hybrid'_2 it is generated by computing c_0 as an honest commitment of $0^{2\ell_{h_{in}}}$, whereas in Hybrid'_3 it is

generated by committing to a collision x_α, x'_α for the hash key α. The openings for these commitments are not used to compute π. Thus, if an adversary $\mathcal{A}$ distinguishes these hybrids with probability p, we can build a reduction that breaks the security of the commitment with probability p. The reduction is non-uniform and must know a collision for hash key α. The reduction also needs to answer polynomially many CCAVal queries. Therefore, its time is polynomial in time of $\mathcal{A}$ and the time of CCAVal. We set λ_{NICom} large enough to ensure the following claim.

Lemma 6. *Assuming that* NICom *is secure against circuits that run in time* $\mathsf{poly}(2^{\lambda_{\mathsf{NICom}}})$*, then for any adversary* $\mathcal{A}$ *of size* $\mathsf{poly}(s_{\mathsf{CCA}})$*, it holds that:*

$$|\Pr[\mathcal{A}(\mathsf{Hybrid}'_2) = 1] - |\Pr[\mathcal{A}(\mathsf{Hybrid}'_3) = 1]| \leq 2^{-\lambda_{\mathsf{NICom}}}.$$

Hybrid'_4 : This hybrid is the same as the previous hybrid except that while we compute the hardwired value $v = (c_0, c_1, \ldots, c_{T'/2}, \pi)$, we replace π as $\pi = \mathsf{NIWI.P}(X, W)$ where $X = (c_0, \ldots, c_{T'/2})$ and W is consists of opening of $c_0 = \mathsf{NICom}(x_\alpha, x'_\alpha)$.

Note that the only difference between Hybrid'_3 and Hybrid'_4 is how hardwiring $v = (c_0, c_1, \ldots, c_{T'/2}, \pi)$ is generated. In particular, it is about how π is generated. In Hybrid'_3 it is generated by using openings of $c_1, \ldots, c_{T'/2}$, whereas in Hybrid'_4 it is generated by using opening of c_0 as the witness. Thus, if an adversary $\mathcal{A}$ distinguishes between these hybrids with probability p, we can build a reduction that breaks the security of the NIWI with probability p. The reduction also needs to answer polynomially many CCAVal queries. Therefore, its time is polynomial in the time of $\mathcal{A}$ and the time of CCAVal. We set λ_{NIWI} large enough to ensure the following claim.

Lemma 7. *Assuming that* NIWI *is secure against circuits that run in time* $\mathsf{poly}(2^{\lambda_{\mathsf{NIWI}}})$*, then for any adversary* $\mathcal{A}$ *of size* $\mathsf{poly}(s_{\mathsf{CCA}})$*, it holds that:*

$$|\Pr[\mathcal{A}(\mathsf{Hybrid}'_3) = 1] - |\Pr[\mathcal{A}(\mathsf{Hybrid}'_4) = 1]| \leq 2^{-\lambda_{\mathsf{NIWI}}}.$$

Hybrid'_5 : This hybrid is the same as the previous hybrid except that while we compute the hardwired value $v = (c_0, c_1, \ldots, c_{T'/2}, \pi)$, we replace for $i \in [T'/2]$, $c_i = \mathsf{CCA}'.\mathsf{CCACommit}(t^*_i, m_0)$.

Note that the only difference between Hybrid'_4 and Hybrid'_5 is how hardwiring $v = (c_0, c_1, \ldots, c_{T'/2}, \pi)$ is generated. In particular, it is about how $c_1, \ldots, c_{T'/2}$ is generated. In Hybrid'_4 it is generated by computing each $c_i = \mathsf{CCA}'.\mathsf{CCACommit}(t^*_i, m_b)$ for $i \in [T^*]$, where as in Hybrid'_4 it is generated by computing each $c_i = \mathsf{CCA}'.\mathsf{CCACommit}(t^*_i, m_0)$ for $i \in [T^*]$. The openings of these commitments are not used in generating π. Note that in these hybrids, the adversary gets an oracle to $\mathsf{CCA}'.\mathsf{CCAVal}()$ oracle but it does not query it on $(t^*_1, \ldots, t^*_{T'/2})$. Thus if there exists an adversary $\mathcal{A}$ that distinguishes these hybrids with probability p, then we can build a reduction that breaks the security of the CCA' with probability p.

Lemma 8. *Assuming that* CCA' *is secure against circuits in* $\mathcal{C}_{\mathsf{CCA}'}$*, then for any adversary* $\mathcal{A}$ *of size* $\mathsf{poly}(s_{\mathsf{CCA}'})$*, it holds that:*

$$|\Pr[\mathcal{A}(\mathsf{Hybrid}'_4) = 1] - |\Pr[\mathcal{A}(\mathsf{Hybrid}'_5) = 1]| \leq T' \cdot \epsilon_{\mathsf{CCA}'}.$$

Hybrid'_6 : This hybrid is the same as the previous hybrid except that while we compute the hardwired value $v = (c_0, c_1, \ldots, c_{T'/2}, \pi)$, we replace π as $\pi = \mathsf{NIWI.P}(X, W)$ where $X = (c_0, \ldots, c_{T'/2})$ and W is consists of opening of $c_1, \ldots, c_{T'/2}$ committing to m_0.

Hybrid'_5 and Hybrid'_6 are indistinguishable due to the security of NIWI, and follow similarly as in the indistinguishability between Hybrid'_3 and Hybrid'_4. Thus we have:

Lemma 9. *Assuming that* NIWI *is secure against circuits that run in time* $\mathsf{poly}(2^{\lambda_{\mathsf{NIWI}}})$*, then for any adversary* $\mathcal{A}$ *of size* $\mathsf{poly}(s_{\mathsf{CCA}})$*, it holds that:*

$$|\Pr[\mathcal{A}(\mathsf{Hybrid}'_5) = 1] - |\Pr[\mathcal{A}(\mathsf{Hybrid}'_6) = 1]| \leq 2^{-\lambda_{\mathsf{NIWI}}}.$$

Hybrid'_7 : This hybrid is the same as the previous hybrid except that while we compute the hardwired value $v = (c_0, c_1, \ldots, c_{T'/2}, \pi)$, we switch the commitment c_0 as $c_0 = \mathsf{NICom}(0^{2\ell_{h_{in}}})$.

Hybrid'_6 and Hybrid'_7 are indistinguishable due to the security of NICom, and follow similarly as in the indistinguishability between Hybrid'_2 and Hybrid'_3. Thus we have:

Lemma 10. *Assuming that* NICom *is secure against circuits that run in time* $\mathsf{poly}(2^{\lambda_{\mathsf{NICom}}})$*, then for any adversary* $\mathcal{A}$ *of size* $\mathsf{poly}(s_{\mathsf{CCA}})$*, it holds that:*

$$|\Pr[\mathcal{A}(\mathsf{Hybrid}'_6) = 1] - |\Pr[\mathcal{A}(\mathsf{Hybrid}'_7) = 1]| \leq 2^{-\lambda_{\mathsf{NICom}}}.$$

Hybrid'_8 : This hybrid is the same as the previous hybrid except while computing the hardwired value v, we replace r_1, r_2, r_3 from being truly random to be generated by $(r_1, r_2, r_3) \leftarrow \mathsf{PPRF.Eval}(K_{\mathsf{PPRF}}, \alpha)$.

Hybrid'_7 and Hybrid'_8 are indistinguishable due to the security of PPRF and follow similarly as in the indistinguishability between Hybrid'_1 and Hybrid'_2. Thus we have:

Lemma 11. *Assuming that* PPRF *is secure against circuits that run in time* $\mathsf{poly}(2^{\lambda_{\mathsf{PPRF}}})$*, then for any adversary* $\mathcal{A}$ *of size* $\mathsf{poly}(s_{\mathsf{CCA}})$*, it holds that:*

$$|\Pr[\mathcal{A}(\mathsf{Hybrid}'_7) = 1] - |\Pr[\mathcal{A}(\mathsf{Hybrid}'_8) = 1]| \leq 2^{-\lambda_{\mathsf{PPRF}}}.$$

Hybrid'_9 : This hybrid is the same as $\mathsf{Hybrid}_{2,\alpha+1}$.

Hybrid'_8 and Hybrid'_9 are indistinguishable due to the security of iO and follow similarly as in the indistinguishability between Hybrid'_0 and Hybrid'_1. Thus we have:

Lemma 12. *Assuming that* iO *is secure against circuits that run in time* $\mathsf{poly}(2^{\lambda_{\mathsf{iO}}})$ *and* PPRF *is correct at unpunctured points, then for any adversary* $\mathcal{A}$ *of size* $\mathsf{poly}(s_{\mathsf{CCA}})$*, it holds that:*

$$|\Pr[\mathcal{A}(\mathsf{Hybrid}'_8) = 1] - |\Pr[\mathcal{A}(\mathsf{Hybrid}'_9) = 1]| \leq 2^{-\lambda_{\mathsf{PPRF}}}.$$

Final Advantage. Summing up the advantage, and plugging in our parameters, we have that the total advantage is bounded by:

$$2^{\ell_{h_{key}}} O(2^{-\lambda_{\mathsf{iO}}} + 2^{-\lambda_{\mathsf{NIWI}}} + 2^{-\lambda_{\mathsf{NICom}}} + 2^{-\lambda_{\mathsf{PPRF}}}) + 2^{\ell_{h_{key}}} O(T' \cdot \epsilon_{\mathsf{CCA}'}) + O(2^{-\lambda_h})$$
$$\leq \epsilon_{\mathsf{CCA}}.$$

3.3.1 Removing One-Tag Restriction

To remove one tag restriction, [Khu21] suggested the following approach. We explain the idea with the help of an ideal one-message zero-knowledge and standard one-round CCA commitments. Let nmc' be a commitment with tag space $T'(\lambda) = \underbrace{\log \ldots \log \lambda}_{O(1) \text{ times}}$ with one-tag restriction. We can build a CCA scheme without this restriction as follows. Suppose we want to commit to a message m with respect to a tag $t \in [T']$, then we can output the new commitment $\mathsf{nmc.CCACommit}(t, m)$ as: $(c_1, \ldots, c_{T'}, \pi)$ where:

- For $i \neq t$, $c_i = \mathsf{nmc.CCACommit}(i, m)$ is a commitment of m with tag i,
- $c_t = \perp$,
- π is proof that the commitment is generated in the way described above.

The reason this gets around the issue of one-tag restriction is because for any tag $t \neq t'$, we can run $\mathsf{nmc.CCAVal}(t', \star)$, by accessing just $\mathsf{nmc}'.\mathsf{CCAVal}(t, \star)$. This is because in the new commitment to the message m, $\mathsf{nmc.CCACommit}(t, m)$ does not invoke $\mathsf{nmc}'.\mathsf{CCACommit}()$ with respect to tag t (but uses every other tag), where as $\mathsf{nmc.CCACommit}(t', \star)$ will always have a component generated by using $\mathsf{nmc}'.\mathsf{CCACommit}(t, \star)$ as $t' \neq t$. Further, the soundness of π will ensure that all commitments that are queried are consistently generated as in the procedure so that extraction using $\mathsf{nmc}'.\mathsf{CCAVal}(t, \star)$ is correct. The security can then be proven by first simulating π and then switching the commitments one by one.

While this is the idea relying on a one-message zero-knowledge, the above idea can be formalized without such a zero-knowledge relying on the same techniques used in the

tag-amplification. Let CCA' be the underlying CCA scheme for tag space $[T']$ above. We build our scheme CCA without one-tag restriction for the same tag $[T']$ following the same approach as our tag amplification transformation, except that the obfuscation corresponds to a slightly different program. The only change is that now, on input the receiver string τ it will produce $c_0, c_1, \ldots, c_{T'}, \pi$ where $c_1, \ldots, c_{T'}$ are generated in the way described above.

We now describe this transformation below. We use the same primitives, notation, and parameters as our tag amplification transformation; the only change is that CCA' suffers from one-tag restriction and $T = T'$. The security proof is essentially the same as in our tag amplification construction.

$\mathsf{CCA.CCACommit}(\mathsf{tag}, m; r)$: Compute the following steps.

- Sample a PPRF key $K_{\mathsf{PPRF}} \leftarrow \{0,1\}^{\ell_{\mathsf{PPRF}}}$,
- Compute $\widetilde{F} \leftarrow \mathsf{iO}(F[\mathsf{tag}, m, K_{\mathsf{PPRF}}])$ by obfuscating the circuit described in Figure 3.7. Output $\widetilde{F}$.

$\mathsf{CCA.ComputeOpening}(\tau, \mathsf{tag}, \widetilde{G}, m, r)$: Compute the following steps.

- Parse $\tau = (\rho, \rho')$ where $\rho \in \{0,1\}^{\ell_{h_{key}}}$ and $\rho' \in \{0,1\}^{\ell_{\mathsf{CCA}'}}$,
- Check if $\widetilde{F} = \mathsf{CCA.CCACommit}(\mathsf{tag}, m; r)$. Abort if its not the case. Derive the PPRF key K_{PPRF} used in code of F described in Figure 3.7.
- Compute $\widetilde{F}[\rho] = (c_0, c_1, \ldots, c_{T'}, \pi)$,
- From the code of Figure 3.7, use the PPRF key k_{PPRF} to derive r'_t as in the code so that $c_t = \mathsf{CCA}'.\mathsf{CCACommit}(t, m; r'_i)$ for $t \in [T'] \setminus \mathsf{tag}$. Compute and output $\sigma_t = \mathsf{CCA}'.\mathsf{ComputeOpening}(\rho', t, c_t, m, r'_t)$ for $i \in [T'] \setminus t$.

$\mathsf{CCA.VerifyOpening}(\tau, \mathsf{tag}, \widetilde{F}, m, \sigma)$: Compute the following steps.

- Parse $\tau = (\rho, \rho')$ where $\rho \in \{0,1\}^{\ell_{h_{key}}}$ and $\rho' \in \{0,1\}^{\ell_{\mathsf{CCA}'}}$,

- Compute $\widetilde{F}[\rho] = (c_0, c_1, \ldots, c_{T'}, \pi)$ and verify π using $\mathsf{NIWI.Vf}$ for the language described in 3.7. Abort if the proof does not verify,
- Output 1 if for every $t \in [T'] \setminus \mathsf{tag}$, $\mathsf{CCA'.VerifyOpening}(\rho', t, c_t, m, \sigma_t)$. Output $\perp$ otherwise.

Remark 1 (Opening algrorithm for base scheme with T' tags)**.** *For base commitments as in [BL18, LPS20], the* $\mathsf{CCA'.ComputeOpening}$ *simply outputs the randomness to commit the message.*

The Circuit $F[\mathsf{tag}, m, K_{\mathsf{PPRF}}]$

Hardwired: Tag $\mathsf{tag} \in [T']$, Message m and PPRF key K_{PPRF},

Input: $\rho \in \{0,1\}^{\ell_{h_{key}}}$

Computation:

1. Compute $r \leftarrow \mathsf{PPRF.Eval}(K_{\mathsf{PPRF}}, \rho)$ and parse $r = (r_1, r_{2,1}, \ldots, r_{2,T'}, r_3)$. Compute:
   - $c_0 = \mathsf{NICom}(0^{\ell_{h_{inp}}}; r_1)$,
   - For $t \in [T']$, and $t \neq \mathsf{tag}$, compute $c_t = \mathsf{CCA'.CCACommit}(t, m; r_{2,i})$,
   - Set $c_{\mathsf{tag}} = \perp$,
   - Compute $\pi = \mathsf{NIWI.P}(X, W; r_3)$ where $X = (c_0, c_1, \ldots, c_{T'})$ and $W = (m, \{r_{2,i}\}_{i \in [T']})$ are so that $(X, W) \in L_F$ for the language L_F defined below.
2. Output $(c_0, c_1, \ldots, c_{T'}, \pi)$.

Language $L_G = L_{F_1} \vee L_{F,2}$:

$$L_{F,1} = \{(c_0, c_1, \ldots, c_{T'}) \mid \exists x \neq x' \in \{0,1\}^{\ell_{h_{in}}} \ s.t. \ \mathcal{H}(\rho, x) = \mathcal{H}(\rho, x') \wedge c_0 = \mathsf{NICom}(x, x')\}$$

$$L_{F,2} = \{(c_0, c_1, \ldots, c_{T'}) \mid \exists m \ s.t. \ \forall t \in [T'] \setminus \mathsf{tag}, \ c_t = \mathsf{CCA'.CCACommit}(t, m) \wedge c_{\mathsf{tag}} = \perp\}$$

Figure 3.7: The Circuit $F[\mathsf{tag}, m, k_{\mathsf{PPRF}}]$

CHAPTER 4

Zero Knowledge

This chapter is dedicated to defining and constructing our new notion of zero knowledge. Our definition and construction, besides being useful in achieving our main result, aim to answer two fundamental questions in this area.

Recall that zero-knowledge arguments are a form of privacy-preserving protocol between a verifier, who has a statement x, and a prover, who has (x, w), and wants to convince the verifier that $(x, w) \in R_L$ without revealing w. Here R_L is an NP relation for some language L. Such an argument must satisfy three properties. The first, called **completeness**, says that if $(x, w) \in R_L$, then the verifier should accept. The second, called **soundness**, says that if $x \notin L$, then even a malicious prover who can deviate arbitrarily from the protocol cannot convince a verifier to accept with non-negligible probability. The third property, called **zero knowledge**, says that there should exist a simulator which for any $x \in L$ can interact with a malicious verifier and produce an interaction which is indistinguishable from that of a real prover who knows x and w. Crucially, the simulator should be able to do this only knowing x and *without knowing* w. This guarantees that nothing is leaked about w to even a malicious verifier.

The first question we aim to answer is whether statistical soundness is possible with a two-round zero knowledge argument. In the context of two rounds, statistical soundness means that if $x \notin L$ then with high probability over the verifier's choice of the first-round message, there should not exist an accepting second-round message. For soundness to hold, the simulator should have some *advantage* which the prover does not have. Otherwise, the

prover will be able to simply run the simulator in order to break soundness. In the case of two-round zero knowledge, the only way known in the literature to achieve this is to give the simulator a *computational advantage*, or in other words to allow the simulator to run in super-polynomial time. (In fact, having a standard polynomial simulator in two-round zero knowledge was shown to be impossible by the work of [GO94].) More specifically, every known two-round zero knowledge works in the following way. The first-round verifier's message contains some sort of computational puzzle, which is guaranteed to have a solution, but for which the solution is hard to find in polynomial time. The simulator, since it runs in super-polynomial time, can brute force solve this puzzle, and by embedding a solution into its second-round message it is able to simulate; the protocol is designed so that any such solution is a "trapdoor" which allows construction of accepting second-round messages which are indistinguishable from honest ones. This does not break soundness, since a polynomial-time prover cannot find such solutions.

The structure described above means that all known two-round zero knowledge argument systems are inherently not statistically sound. This is because no matter what the statement x is, there always exist trapdoor accepting second-round messages which are obtained based on solutions to the puzzle embedded in the verifier's first-round message. Thus it is an important open question whether any sort of statistical soundness is achievable in two rounds.

The second question we address has to do with arguments that are both statistically sound and statistically zero-knowledge, in any number of rounds. There is a long line of well-known classical results [GMR85, Kan90, GK90, GG98, For87, AH87, BHZ87] which study the class SZK of languages which have zero-knowledge proofs that are both statistically sound and statistically zero knowledge. This line of work established that SZK cannot contain NP, otherwise the polynomial-time hierarchy collapses.

Thus, a natural question is, how close can one get to a statistically-sound, statistical zero knowledge protocol for all languages in NP, without running into this impossibility?

To answer both these two questions, we give a new definition of two-round zero knowledge,

which works in the following way. In the first round, both prover and verifier send a message. The first round then fixes one of two modes which the protocol will work in: *perfect soundness mode*, and *statistical zero knowledge mode*. In perfect soundness mode, there exist no second-round messages for false statements. In statistical zero knowledge mode, a super-polynomial machine can find simulated second-round messages for all statements, which are statistically indistinguishable from an honest prover's statements. Assuming both the first-round messages are generated by computationally bounded machines, the perfect soundness mode is guaranteed to happen with some fixed negligible, tunable probability $\approx \mu$, and a computationally bounded prover cannot tell which mode the protocol is in.

The intuition behind this definition is that, although it does not satisfy standard statistical soundness, it can provide an approximation of it which is useful when used in conjunction with other primitives which require statistical soundness, such as witness encryption. This is explained in detail in Chapter 5.

We call this definition *statistical zero knowledge with sometimes-statistical soundness*. To show that this definition is meaningful, we give a construction which satisfies it, specifically proving the following theorem.

Theorem 6. *Assume that the following assumptions hold:*

- *A subexponentially secure indistinguishability obfuscator exists,*
- *a time lock puzzle as in Definition 4 exist,*
- *a subexponentially-secure NIWI exists,*
- *a subexponentially-secure sender-equivocal oblivious transfer scheme exists,*
- *a subexponentially-secure OWP computable in* NC^1 *exists, and*
- *a subexponentially-secure somewhere-statistical correlation-intractable hash function exists,*

then there exists a reusable statistical ZK argument with sometimes statistical soundness as defined in Definition 20. For this scheme, the complexity parameters are defined in Definition 28.

Since both the oblivious transfer protocol and the correlation-intractable hash function are instantiable from subexponential hardness of LWE [BD18, PS19b], we achieve Theorem 2 in Chapter 1 as a corollary of Theorem 6.

Note that in order to use this scheme to achieve MrNISC, we also require a strong form of reusability. If this reusability is not required, it is possible to construct statistical zero knowledge with sometimes-statistical soundness from much simpler assumptions:

Theorem 7. *Assume subexponential hardness of the LWE assumption. Then there exists a (non-reusable) statistical ZK argument with sometimes statistical soundness.*

The rest of the chapter is organized as follows. We start with a formal definition of reusable statistical zero knowledge with sometimes-statistical soundness. We then give a high-level overview of the techniques used in constructing this new type of zero knowledge. Finally, we give a formal construction and security proof, thus finishing the proof of Theorem 6. Although we do not separately prove Theorem 7, it is implicit in the proof of Theorem 6: simply remove all the machinery used to achieve reusability, and instantiate the extractable commitment scheme with one that is closer to the one used in [BFJ20].

4.1 A Formal Definition

We define statistical zero-knowledge arguments with a specific communication pattern. The protocol that we need has a "simultaneous message" first round, where both the prover and verifier will simultaneously send a message. The syntax is the following:

1. The (honest) prover $P = (\mathsf{ZKProve}_1, \mathsf{ZKProve}_2)$ and verifier $V = (\mathsf{ZKVerify}_1, \mathsf{ZKVerify}_2)$ are each composed of two uniform PPT algorithms.

2. $\mathsf{ZKProve}_1$ and $\mathsf{ZKVerify}_1$ get as input only the security parameter λ. $\mathsf{ZKProve}_1$ outputs a message $\mathsf{zk}_{1,P}$ and a state σ_P. $\mathsf{ZKVerify}_1$ outputs a message $\mathsf{zk}_{1,V}$ and a state σ_V. The first round transcript is denoted $\tau_1 = (\mathsf{zk}_{1,P}, \mathsf{zk}_{1,V})$.

3. $\mathsf{ZKProve}_2$ gets σ_P, $\mathsf{zk}_{1,V}$, the instance x, and a witness w. It outputs a message $\mathsf{zk}_{2,P}$.

4. $\mathsf{ZKVerify}_2$ gets the instance x and $\tau = (\tau_1, \mathsf{zk}_{2,P})$, and outputs $0/1$.

Looking ahead, we shall consider two-round ZK protocols as above with super-polynomial simulation (SPS). Further, we will also require that for a given prover and a verifier, the first message is reusable for proving multiple statements. We denote $\langle P(w), V\rangle(1^\lambda, x)$ the output of the interaction between P and V, where P gets as input the witness w, and both P and V receive the instance x as a common input.

Definition 17 (Reusable Statistical Zero-Knowledge Arguments with Sometimes-Statistical Soundness)**.** *Let L be a language in NP with a polynomial-time computable relation R_L. A protocol between P and V is a $(\mathcal{C}_{\mathsf{sound}}, \mathcal{C}_{\mathcal{S}}, \mathcal{C}_{\mathsf{zk}}, \epsilon_{\mathsf{sound},1}, \epsilon_{\mathsf{sound},2}, \epsilon_{\mathcal{S}})$-reusable statistical zero-knowledge argument with sometimes-statistical soundness if it satisfies Definitions 18 to 20 below.*

Definition 18 (Perfect Completeness)**.** *Let L be a language in NP with a polynomial-time computable relation R_L. A protocol between P and V satisfies perfect completeness if for every security parameter 1^λ and $(x, w) \in R_L$, it holds that*

$$\Pr\left[\langle P(w), V\rangle(1^\lambda, x) = 1\right] = 1,$$

where the probability is over the random coins of P and V.

Additionally, we need a refined soundness property, defined next.

Definition 19 ($(\mathcal{C}_{\mathsf{sound}}, \epsilon_{\mathsf{sound},1}, \epsilon_{\mathsf{sound},2})$-statistical soundness)**.** *Consider any prover $P^* \in \mathcal{C}_{\mathsf{sound}}$ and a polynomial $p(\cdot)$, where on input the security parameter 1^λ, P^* outputs an*

instance $x \in \{0,1\}^p \setminus L$. *We require that there exists a "soundness mode indicator" machine* $\mathcal{E}$ *that on input* $(\tau_1, \mathsf{state}_V)$ *outputs either* 0 *or* 1 *such that the following properties hold.*

- ***Frequency of Soundness Mode.*** *For every prover* $P^* \in \mathcal{C}_{\mathsf{sound}}$,

$$\Pr\left[\mathcal{E}(\tau_1, \mathsf{state}_V) = 1\right] \geq \epsilon_{\mathsf{sound},1}(\lambda),$$

  where the probability is over the coins of the prover and the verifier in round 1.

- ***Perfect Soundness Holds During Soundness Mode.*** *For every prover* $P^* \in \mathcal{C}_{\mathsf{sound}}$ *and every round-1 state* $(\tau_1, \mathsf{state}_{P^*}, \mathsf{state}_V)$ *of the protocol, if* $\mathcal{E}(\tau_1, \mathsf{state}_V) = 1$ *then for all second-round messages* $\mathsf{zk}_{2,P}$ *sent by the prover corresponding to some false statement* $x \notin L$, *the verifier rejects on input* $(x, \tau_1, \mathsf{zk}_{2,P}, \mathsf{state}_V)$.

- ***Indistinguishability of Soundness Mode.*** *For every prover* $P^* \in \mathcal{C}_{\mathsf{sound}}$, *it holds that*

$$\{(\tau_1, \mathsf{state}_{P^*}) \mid \mathcal{E}(\tau_1, \mathsf{state}_V) = 1\} \approx_{(\mathcal{C}_{\mathsf{sound}}, \epsilon_{\mathsf{sound},2})} \{(\tau_1, \mathsf{state}_{P^*}) \mid \mathcal{E}(\tau_1, \mathsf{state}_V) = 0\}.$$

The full MrNISC protocol needs a powerful version of zero knowledge, as follows:

Definition 20 ($(\mathcal{C}_{\mathcal{S}}, \mathcal{C}_{\mathsf{zk}}, \epsilon_{\mathcal{S}})$-Adaptive Reusable Statistical Zero-Knowledge)**.** *We say a zero knowledge scheme satisfies* $(\mathcal{C}_{\mathcal{S}}, \mathcal{C}_{\mathsf{zk}}, \epsilon_{\mathcal{S},1}, \epsilon_{\mathcal{S},2})$*-adaptive reusable statistical zero-Knowledge if there exists a (uniform) simulator* $\mathsf{ZKSim} \in \mathcal{C}_{\mathcal{S}}$ *which takes as input the round-one transcript* τ_1, *the honest prover's state* σ_P, *and a statement* x *such that the following holds. Consider an adversary* $V^* \in \mathcal{C}_{\mathsf{zk}}$ *that takes as input* 1^λ *and an honestly generated prover's first round message* $\mathsf{zk}_{1,P}$, *and plays the following game* $\mathsf{expt}^b_{V^*,\mathsf{zk}}$*:*

1. V^* *may adaptively issue queries of the form* $(x, w, \mathsf{zk}^*_{1,V})$. *The challenger responds as follows:*

   - $f(x, w) \notin R_L$, *the challenger responds with* $\perp$.

- *If* $(x, w) \in R_L$ *and* $b = 0$*, the challenger responds with the honest prover's second message* $\mathsf{ZKProve}_2(\sigma_p, \mathsf{zk}^*_{1,V}, x, w)$.
- *If* $(x, w) \in R_L$ *and* $b = 1$*, the challenger responds with the simulated prover's message* $\mathsf{ZKSim}(\sigma_p, \mathsf{zk}^*_{1,V}, x)$.

2. *At the end of the game,* V^* *outputs an arbitrary function of its view, which is used as the output of the experiment.*

It must hold that

$$\mathsf{expt}^0_{V^*,\mathsf{zk}} \approx_{(\infty,\epsilon_S)} \mathsf{expt}^1_{V^*,\mathsf{zk}}.$$

4.2 An Overview of the Construction

We now give an overview of the zk construction. Recall that we want to construct a two-round (delayed instance) zk with SPS simulation. At the same time, the second message by the prover is still subject to perfect soundness with some probability over the first round messages. Further, the first round should be reusable across sessions.

Our starting point is the SPS ZK protocol/statistical ZAP arguments of [BFJ20, GJJ20]. The protocol relies on the following primitives:

- A correlation intractable hash function $\mathcal{H}(K, \star) \to \{0,1\}^\ell$ [CCH19, PS19a],
- A two-round statistically hiding sometimes extractable commitment $\mathsf{Com} = (\mathsf{Com}_{1,R}, \mathsf{Com}_{2,C})$ [KK19],

A (somewhere) statistically correlation intractable function is associated with an algorithm $\mathsf{FakeGen}$ that takes as input a polynomial time computable function $f : \{0,1\}^{\ell_{in}} \to \{0,1\}^\ell$, and outputs a key K_f, for which there does not exist in input $x \in \{0,1\}^{\ell_{in}}$ such that $\mathcal{H}(K_f, x) = f(x)$. These functions can be built from LWE. Further, $\mathsf{FakeGen}$ produces pseudorandom outputs, and thus the key K_f hides f computationally.

A two-round statistically hiding sometimes extractable commitment scheme on the other hand, has the following structure.

- In the first round, the receiver samples a $\mathbf{b}_R \in \{0,1\}^\mu$ and computes and outputs $\mathsf{com}_{1,R} = \mathsf{Com}_{1,R}(\mathbf{b}_R; r_R)$.
- In the second round, the committer samples $\mathbf{b}_C \in \{0,1\}^\mu$ randomly, and outputs any number of commitments $\mathbf{b}_C, \{\mathsf{com}_{2,C,i} = \mathsf{Com}_{2,C}(\mathbf{b}_C, \mathsf{com}_{1,R}, m_i)\}_{i\in[T]}$.

The protocol has the following property. If $\mathbf{b}_R \neq \mathbf{b}_C$ (or if $\mathsf{com}_{1,R}$ is not well-formed as per the protocol), then, the honestly generated commitments $\mathsf{com}_{2,C}$, statistically hide the messages $\{m_i\}_{i\in[T]}$. On the other hand, if, $\mathbf{b}_R = \mathbf{b}_C$, then there exists an efficient algorithm Dec such that: $\mathsf{Dec}(\mathbf{b}_R, r_R, \mathsf{com}_{2,C,i}) = m_i$ for $i \in [T]$ is binding for $\mathsf{com}_{2,C,i}$ (Dec always output a valid message; further this message is biding even when $\mathsf{com}_{2,C,i}$ is ill formed). Further, an honest receiver can ensure that $\mathbf{b}_C = \mathbf{b}_R$ with probability at least $\Omega(2^{-\mu})$. To an adversarial polynomial-time committer, the view is indistinguishable from the view when $\mathbf{b}_C \neq \mathbf{b}_R$. The works of [KKS18] showed that such commitments can be built from assumptions such as LWE or DDH.

Once we have these primitives, then the SPS ZK protocol of [BFJ20], follows the following template. Let (x, w) be the isntance witness pair.

- In the first round, the verifier chooses $\mathbf{b}_V \leftarrow \{0,1\}^\mu$, and outputs $\mathsf{zk}_{1,P} = (\mathsf{com}_{1,R}, K)$ where $\mathsf{com}_{1,R} = \mathsf{Com}_{1,R}(\mathbf{b}_V)$ and $K \leftarrow \mathcal{H}.\mathsf{FakeGen}(f)$ for some function f described later,
- In the second round, the prover samples $\mathbf{b}_P \leftarrow \{0,1\}^\mu$, and then computes $\mathsf{com}_{2,C,i} = \mathsf{Com}_{2,C}(\mathbf{b}_P, \mathsf{com}_{1,R}, a_i; r'_i)$ for $i \in [N]$ where $(a_1, \ldots, a_N)$ are the values committed to during a special Σ protocol[1] for proving x. Then,

[1]As an example, think of it as the Blum's Hamiltonicity Protocol. As in the construction of NIZK from LWE[CCH19, PS19a], it suffices to use a parallel repetion of a sigma protocol for NP with i) 1/2-special soundness, ii) efficient $\mathsf{BadChallenge}$ computation.

– The prover runs $\mathcal{H}(K, (x, \mathbf{b}_P, \mathsf{com}_{2,C})) = \mathbf{e}$,

– Outputs commitments $\mathsf{com}_{2,C} = \{\mathsf{com}_{2,C,i}\}_{i\in[N]}$ along with openings $\mathsf{com}_2, \{a_i, r'_i\}_{i\in\mathsf{Set}}$ where Set is the set dictated by the challenge $\mathbf{e}$ of the Σ protocol.

The statistical WI property follows from the fact that when $\mathbf{b}_P \neq \mathbf{b}_V$, then $\mathsf{com}_{2,C}$ are statistically hiding. One needs more work to prove that it is actually SPS ZK by using an inefficient equivocator of $\mathsf{com}_{2,C}$ with a simulator of the Σ protocol. For the soundness property, observe that when $\mathbf{b}_V = \mathbf{b}_P$, then the commitments are binding to the value computed by $\mathsf{Dec}(\mathbf{b}_V, r_R, \star)$ where r_R is used to compute $\mathsf{com}_{1,R}$. We exploit this to set f as follows. We set f to be the function that computes $\mathbf{e}^* = \mathsf{BadChallenge}(x, a_1, \ldots, a_N)$ where $(a_1, \ldots, a_N)$ is recovered by running $\mathsf{Dec}(\mathbf{b}_V, r_R, \star)$.

Perfect Soundness Mode. The protocol above does not have a perfect soundness mode. However, it turns out that in the simultaneous message model, there is a straightforward modification of the protocol above that gives us a perfect soundness mode. The modification is described as follows.

- In the first round, the verifier outputs $\mathsf{zk}_{1,V} = (\mathsf{com}_{1,R}, K)$ as before, but the prover outputs $\mathsf{zk}_{1,P} = \mathbf{b}_P$ in the clear.
- In the second round, the prover outputs as before, but using $\mathbf{b}_P$ displayed in the first round itself.

The reason why this protocol has a perfect soundness mode is that $\mathbf{b}_P$ is displayed in the first round itself, and so the first round already determines if the prover can cheat in the second round or not. Unfortunately, this naive approach fails in our setting where the same prover first message can be used repeatedly with multiple verifiers/receivers. In fact, it even fails when an honest prover interacts with a rushing malicious verifier. If such a verifier sees $\mathbf{b}_P$, then it can choose $\mathbf{b}_V = \mathbf{b}_P$, which will put the prover in the perfect soundness mode, and its proofs will no longer be simulatable.

Fixing Zero-Knowledge: A different criteria for soundness mode Imagine if we could modify the criteria for the soundness mode as follows. In this model, $\mathsf{zk}_{1,P}$ is $\alpha = \mathsf{OWP}(\mathbf{b}_P)$ for a one-way permutation as opposed to $\mathbf{b}_P$ in the clear, and $\mathsf{zk}_{1,V}$ is as before, $\mathsf{com}_{1,R} = \mathsf{Com}_{1,R}(\mathbf{b}_V)$. As before, perfect soundness must hold if $\mathbf{b}_V = \mathbf{b}_P$ and perfect zero-knowledge otherwise. This high-level approach appears to make sense, as intuitively a verifier must compute $\mathsf{com}_{1,R} = \mathsf{Com}_{1,R}(\mathsf{OWP}^{-1}(\alpha))$ to violate soundness.

To work this idea out in the reusable setting, we must tackle one more issue. We need to make sure that $\mathsf{zk}_{2,P}$ must not reveal information about $\mathbf{b}_P$ as in the reusable setting, one can choose $\mathsf{zk}'_{1,V}$ after seeing a second message $\mathsf{zk}_{2,P}$ used in some other session (which might contain information about $\mathbf{b}_P$). We make this intuition formal by this abstraction called "Sometimes Extractable Equivocal Commitments" or SEE. For the rest of the section, assume that the verifier's first message is "well-formed," and we expect the zero-knowledge property to hold only when this is the case. We will fix this issue later.

Sometimes Extractable Equivocal Commitments. A SEE scheme consists of three algrorithms $(\mathsf{Com}_{1,R}, \mathsf{Com}_{1,C}, \mathsf{Com}_{2,C})$ and is a commitment scheme that captures the issues pointed above in the simultaneous message model. In the first round,

- The receiver chooses $\mathbf{b}_R$ and computes and outputs $\mathsf{com}_{1,R} = \mathsf{Com}_{1,R}(\mathbf{b}_R; r)$,
- The committer chooses $\mathbf{b}_C$ and computes and outputs $\mathsf{com}_{1,C} = \mathsf{Com}_{1,C}(\mathbf{b}_R)$ deterministically. Further, the image of $\mathsf{com}_{1,C}$ is verifiable in that it is essentially a one-way permutation.

In the second round, the committer outputs $\mathsf{Com}_{2,C}(\mathsf{com}_{1,R}, \mathsf{com}_{1,C}, m; r')$. We want mostly similar properties as before (with a few additional properties): If $\mathbf{b}_R = \mathbf{b}_C$, then the commitment is fully extractable and perfectly binding (even when $\mathsf{com}_{2,C}$ is not well formed, and even to an outside observer who does not know the r used to generate $\mathsf{com}_{1,R}$), where as

when $\mathbf{b}_R \neq \mathbf{b}_C$, then $\mathsf{com}_{2,C}$ is statistically hiding. In fact, when $\mathsf{com}_{1,R}$ is well formed and $\mathbf{b}_R \neq \mathbf{b}_C$, then the commitment $\mathsf{com}_{2,C}$ should be efficiently equivocable.

To deal with the issues of reusability described above, it should be computationally hard for an adversarial receiver to create a well-formed commitment of $\mathsf{com}_{1,R} = \mathsf{Com}_{1,R}(\mathbf{b}_C)$ where $\mathbf{b}_C$ is chosen by the committer even, after seeing $\mathsf{com}_{1,C}$. Further, an honest receiver could always ensure that $\mathbf{b}_R = \mathbf{b}_C$ where $\mathbf{b}_C$ is used in $\mathsf{com}_{1,C}$ with a decent probability $\Omega(2^{-\mu})$.

Plugging in this commitment scheme with a correlation intractable hash function gives rise to the following zk protocol.

- In the first round, the verifier outputs $\mathsf{zk}_{1,V} = (\mathsf{Com}_{1,R}(\mathbf{b}_V; r), K)$ as before and the prover outputs $\mathsf{zk}_{1,P} = \mathsf{Com}_{1,C}(\mathbf{b}_P)$.
- In the second round, the prover computes $\mathsf{com}_{2,C,i} = \mathsf{Com}_{2,C}(\mathsf{com}_{1,R}, \mathsf{com}_{1,C}, a_i; r'_i)$ for $i \in [N]$ where $(a_1, \ldots, a_N)$ are the values committed to during a special Σ protocol. Then,
  - The prover runs $\mathcal{H}(K, (x, \mathsf{com}_{1,R}, \mathsf{com}_{1,C}, \mathsf{com}_{2,C})) = \mathbf{e}$,
  - Outputs commitments $\mathsf{com}_{2,C} = \{\mathsf{com}_{2,C,i}\}_{i\in[N]}$ along with openings $\mathsf{com}_{2,}\{a_i, r'_i\}_{i\in\mathsf{Set}}$ where Set is the set dictated by the challenge $\mathbf{e}$ of the Σ protocol.

Observe that now, the protocol has a perfect soundness mode, namely when $\mathbf{b}_P = \mathbf{b}_V$. Further, the verifier message is reusable across multiple prover sessions as the soundness holds with the same probability $\Omega(2^{-\mu})$ across multiple sessions. On the other hand, the prover's first message $\mathsf{zk}_{1,V} = \mathsf{Com}_{1,C}(\mathbf{b}_V)$ is also reusable with different verifiers, as it is computationally hard to produce $\mathsf{com}_{1,R} = \mathsf{Com}_{1,R}(\mathbf{b}_V)$ with the $\mathbf{b}_P = \mathbf{b}_V$. Assuming verifier's messages are well-formed, we can simulate the zk by equivocating the commitment. Two issues need discussion. The first concerns with the required complexity hierarchy for our MrNISC and the second, with the fact that in the arguments above, we did not show zero-knowledge against

adversaries that output non-well formed first messages (because commitment equivocation only works if $\mathsf{com}_{1,R}$ is well-formed).

Issue with Complexity Hierarchy wrt MrNISC. In the bigger scheme of things with other primitives in the MrNISC scheme, we are also using a one-round CCA commitment (CCA), and that protocol is intimately tied with the zk we are trying to build. As pointed out in Section 1.1.1.4, on the one hand, we need the zk to be sound against circuits that can perform CCAVal; on the other hand, CCA commitments need to be secure against circuits that are capable of running zk SPS simulator. This might feel like a deadlock, so we introduce a new axis of hardness.

We will use commitments CCA which are secure against circuits of some quasipolynomial size such that $\mathsf{CCA.CCAVal}$ runs in polynomial depth but size 2^{λ^c} for $c > 0$. In the zk we build:

- Soundness holds against adversaries of $\mathsf{poly}(\lambda)$ depth and size $2^{\lambda^{c_2}}$ for $c_2 > c$.
- The zk simulator can be implemented by a circuit of quasipolynomial size/and depth $T_{\mathsf{zk},\mathcal{S}}$ against which CCA security holds.

We incorporate time-lock puzzle-like properties in our commitment scheme and hence the zk protocol. To do this, within $\mathsf{zk}_{1,V}$, we add a time-lock puzzle encrypting secret information that allows one to equivocate $\mathsf{com}_{2,C}$ generated with respect to $\mathsf{com}_{1,R}$ in $\mathsf{zk}_{1,V}$.

Summing up. Summing up, as a first step we build an SEE scheme described above with. Below we list all the properties. The only new addition to what was described before is that $\mathsf{com}_{2,C}$ can be equivocated in polynomial time given the opening $\mathbf{b}_R, r$ of $\mathsf{com}_{1,R} = \mathsf{Com}_{1,R}(\mathbf{b}_R; r)$.

- Extractability: If $\mathsf{com}_{1,R} = \mathsf{Com}_{1,R}(\mathbf{b}_R; r)$ and $\mathsf{com}_{1,C} = \mathsf{Com}_{1,C}(\mathbf{b}_R; r_R)$, then $\mathsf{com}_{2,C} = \mathsf{Com}_{2,C}(\mathsf{com}_{1,R}, \mathsf{com}_{1,C}, m)$ is polynomial time extractable using Dec algorithm.

The result of $\mathsf{Dec}(\mathbf{b}_R, r_R, \mathsf{com}_{2,C})$ must be a valid message string and should be binding to an outside observer even when $\mathsf{com}_{2,C}$ is not well-formed. This property is identical to the one described before.

- EQUIVOCABILITY: If $\mathsf{com}_{1,R} = \mathsf{Com}_{1,R}(\mathbf{b}_R; r)$ and $\mathsf{com}_{1,C} = \mathsf{Com}_{1,C}(\mathbf{b}_C)$ where $\mathbf{b}_C \neq \mathbf{b}_R$, then there exists a polynomial time algorithm $\mathsf{SEE}.\mathcal{S}$ that takes as input $\mathbf{b}_R, r$, $\mathsf{com}_{2,C}, m, r'$ where $\mathsf{com}_{2,C} = \mathsf{Com}_{2,C}(\mathsf{com}_{1,R}, \mathsf{com}_{1,C}, 0, r')$ and outputs an opening s' such that $\mathsf{com}_{2,C} = \mathsf{Com}_{2,C}(\mathsf{com}_{1,R}, \mathsf{com}_{1,C}, m; s')$. Further, $\mathsf{com}_{2,C}, s', m$ generated this way is identical to the case when $\mathsf{com}_{2,C}$ was a commitment of m and s' was its opening. This is stronger than statistical indistinguishability. This property is useful because one can encrypt $(\mathbf{b}_R, r)$ as a part of $\mathsf{zk}_{1,P}$ using a time-lock puzzle, which will help the zk simulator.

- INDISTINGUISHABILITY OF $\mathbf{b}_R$: We require that an $\mathsf{com}_{1,R} = \mathsf{Com}_{1,R}(\mathbf{b}_R)$ hides $\mathbf{b}_R$. Further, for any computationally bounded committer $\mathbf{b}_C = \mathbf{b}_R$ with a probability of $2^{-\Omega(\mu)}$. We also require that the distribution of transcripts when this event happens are indistinguishable from when this event does not happen.

- HARD TO FORCE $\mathbf{b}_R = \mathbf{b}_C$: We require that a computationally bounded adversarial receiver given $\mathsf{com}_{1,C} = \mathsf{Com}_{1,C}(\mathbf{b}_C)$ for a randomly chosen $\mathbf{b}_C$ cannot come up with $\mathsf{com}_{1,R} = \mathsf{Com}_{1,R}(\mathbf{b}_C)$ with all but negligible probability.

We build such an SEE scheme relying on DDH assumption over $\mathbb{Z}_p^*$ in Section 4.3.2.

Once we have such a commitment scheme, we can solve all problems, except we need to control the circuit size that runs $\mathsf{zk}.\mathcal{S}$ by a quasi-polynomial sized circuit. Our main idea to get around this is to use a time-lock puzzle. We add $Z = \mathsf{TLP}(\mathbf{b}_V, r)$ to $\mathsf{zk}_{1,V}$. The TLP parameters are set so that a quasipolynomial sized circuit breaks it, but it is secure against all circuits of polynomial depth of size 2^{λ^c}.

Therefore in our modified protocol:

- In the first round, the verifier outputs $\mathsf{zk}_{1,V} = (Z = \mathsf{TLP}(\mathbf{b}_V, r), \mathsf{Com}_{1,R}(\mathbf{b}_V; r), K)$ and the prover outputs $\mathsf{zk}_{1,P} = \mathsf{Com}_{1,C}(\mathbf{b}_P)$.
- In the second round, the prover computes $\mathsf{com}_{2,C,i} = \mathsf{Com}_{2,C}(\mathsf{com}_{1,R}, \mathsf{com}_{1,C}, a_i; r'_i)$ for $i \in [N]$ where $(a_1, \ldots, a_N)$ are the values committed to during a special Σ protocol. Then,
  - The prover runs $\mathcal{H}(K, (x, \mathsf{com}_{1,R}, \mathsf{com}_{1,C}, \mathsf{com}_{2,C})) = \mathbf{e}$,
  - Outputs commitments $\mathsf{com}_{2,C} = \{\mathsf{com}_{2,C,i}\}_{i\in[N]}$ along with openings $\mathsf{com}_{2,}\{a_i, r'_i\}_{i\in\mathsf{Set}}$ where Set is the set dictated by the challenge $\mathbf{e}$ of the Σ protocol.

This is useful, and in particular, we can now simulate zk by first breaking Z to learn $\mathbf{b}_V, r$ and then using the equivocator of the commitments and the simulator of the Σ protocol to simulate the second message.

In our construction, to make the construction mode modular, we incorporate the TLP aspect in the SEE scheme (see Section 4.3.2) and not in our zk protocol. In our commitment scheme, the equivocation property is required to hold only against a receiver which generates $\mathsf{com}_{1,R}$ using the honest algorithm (although with adversarial randomness). This brings us to our last issue.

One Last Issue. This solves all the issues, except that the simulator fails if a verifier does not generate $\mathsf{com}_{1,R}$ as per the specification of the protocol. Indeed, Z may not be a time lock puzzle and give the randomness needed by the simulator to equivocate $\mathsf{com}_{2,C}$. To fix this issue, the verifier now supplies a simultaneous message non-interactive distributional indistinguishability proof NIDI (see Section 4.3.3 for details about NIDI), proving that the verifier messages are well-formed as in the protocol described above. This soundness property of this proof system guarantees that the verifier messages are well-formed, which is helpful for the simulator, and the distributional indistinguishability guarantees that $\mathsf{zk}_{1,V}$ generated

using $\mathbf{b}_V$ is computationally indistinguishable from $\mathsf{zk}_{1,V}$, generated using 0^μ. Analyzing the protocol and setting up parameters requires some care, and we describe it formally next.

4.3 Some Tools

This section defines and constructs two tools that we will use to build our reusable statistical ZK arguments with sometimes statistical soundness. We first give the definitions and then the constructions. The first notion is that of a *sometimes extractable equivocal commitments* (SEE), as explained informally in the previous section, which is new to this work. The second notion is that of Non-interactive Distributional Indistinguishability (NIDI), due to Khurana [Khu21]. Due to the reasons we explain below, we need to strengthen the definition and construction for our purposes.

4.3.1 Sometimes Extractable Equivocal Commitments

This section defines the notion of sometimes extractable equivocal commitments SEE that we use. These commitments are inspired by the ones used to build statistical ZAP arguments [BFJ20, GJJ20].

Definition 21. *An* SEE *is a tuple of three p.p.t. algorithms* $\mathsf{Com}_{1,R}, \mathsf{Com}_{1,C}, \mathsf{Com}_{2,C}$ *with the following syntax:*

- $\mathsf{Com}_{1,R}(1^\lambda, 1^t, 1^\mu, \mathbf{b_R}; r) \to \mathsf{com}_{1,R}$. *The* $\mathsf{Com}_{1,R}$ *denotes the first receiver message. It takes as input three security parameters* λ, t, μ *along with a string* $\mathbf{b_R} \in \{0,1\}^\ell$ *for some polynomial* $\ell = \ell(\mu)$. *It outputs* $\mathsf{com}_{1,R}$.
- $\mathsf{Com}_{1,C}(1^\lambda, 1^t, 1^\mu, \mathbf{b_C}) \to \mathsf{com}_{1,C}$. *The* $\mathsf{Com}_{1,C}$ *denotes the first committer message. It takes as input three security parameters* λ, t, μ *along with a string* $\mathbf{b_C} \in \{0,1\}^\ell$. *It deterministically outputs* $\mathsf{com}_{1,C}$.
- $\mathsf{Com}_{2,C}(\mathsf{com}_{1,R}, \mathsf{com}_{1,C}, m; r') \to \mathsf{com}_{2,C}$. *The* $\mathsf{Com}_{2,C}$ *denotes the second committer*

message. It takes as input first committer and receiver messages $\mathsf{com}_{1,R}, \mathsf{com}_{1,C}$ *along with a message* m *and outputs* $\mathsf{com}_{2,C}$ *which is referred to as the commitment.*

Such a scheme satisfies the following properties.

$(\mathcal{C}_D, \epsilon_D)$-Indistinguishability of $\mathsf{Com}_{1,R}$. Let $\lambda \in \mathbb{N}$ and $\mu \in \lambda^{O(1)}$, $t \in \lambda^{\Omega(1)(\log\log\lambda)^{-1}} \cap \lambda^{O(1)}$ and $\mathbf{b} \in \{0,1\}^\ell$. Then, it holds that:

$$\mathsf{Com}_{1,R}(1^\lambda, 1^\mu, 1^t, \mathbf{b}) \approx_{\mathcal{C}_D, \epsilon_D} \mathsf{Com}_{1,R}(1^\lambda, 1^\mu, 1^t, 0^\ell).$$

Verifiability of $\mathsf{Com}_{1,C}$. There exists a deterministic polynomial time algorithm Vf that takes as input $1^\lambda, 1^t, 1^\mu$ and $\mathsf{com}_{1,C}$ and outputs 1 if and only if $\mathsf{com}_{1,C} = \mathsf{Com}_{1,C}(1^\lambda, 1^t, 1^\mu, \mathbf{b})$ for some $\mathbf{b} \in \{0,1\}^\ell$.

Extraction when $\mathbf{b}_R = \mathbf{b}_C$ There exists a deterministic polynomial time algorithm Dec^* such that the following holds. Let $\lambda \in \mathbb{N}$, $\mu = \lambda^{O(1)}$, $t \in \lambda^{\Omega(1)(\log\log\lambda)^{-1}} \cap \lambda^{O(1)}$. Then, for any $\mathbf{b} \leftarrow \{0,1\}^\ell$ and any message $m \in \{0,1\}^*$

$$\Pr_{r,r'}[\mathsf{Dec}^*(\mathbf{b}, r, \mathsf{com}_{1,C}, \mathsf{com}_{1,R}, \mathsf{com}_{2,C}) = m] = 1,$$

where, $\mathsf{com}_{1,C} = \mathsf{Com}_{1,C}(1^\lambda, 1^\mu, 1^t, \mathbf{b})$, $\mathsf{com}_{1,R} = \mathsf{Com}_{1,R}(1^\lambda, 1^\mu, 1^t, \mathbf{b}; r)$ and $\mathsf{com}_{2,C} = \mathsf{Com}_{2,C}($ $\mathsf{com}_{1,R}, \mathsf{com}_{1,C}, m; r')$. We can define another deterministic algorithm Dec, that runs Dec^* to always compute the valid message v. It outputs a default string 0 if Dec^* fails to produce an output, otherwise it outputs the response of Dec^*. Observe that if $\mathsf{com}_{1,C}$ and $\mathsf{com}_{2,R}$ are generated semi-maliciously using same $\mathbf{b}_R = \mathbf{b}_C = \mathbf{b}$, then, this property means that the value given by Dec is perfectly binding to the commitment $\mathsf{com}_{2,C}$ even when this may not be well formed. In addition, we require that when $\mathbf{b}_R = \mathbf{b}_C = \mathbf{b}$, the commitment is perfectly binding *even to an outside observer who does not know* r.

Equivocation when $\mathbf{b}_R \neq \mathbf{b}_C$. We require that these exist an algorithm $\mathcal{S}$ such that the following holds. Let $\lambda \in \mathbb{N}$, $\mu = \lambda^{\Theta(1)}$ and $t = \lambda^{\Omega(1)(\log\log\lambda)^{-1}} \cap \lambda^{O(1)}$. Let $\mathbf{b}_1 \neq \mathbf{b}_2$ be both in $\{0,1\}^\ell$. Then, for any $m \in \{0,1\}^*$, with probability 1 over the coins of $\mathsf{Com}_{1,R} = \mathsf{Com}_1(1^\lambda, 1^\mu, 1^t, \mathbf{b}_1)$ and $\mathsf{Com}_{1,C}(1^\lambda, 1^\mu, 1^t, \mathbf{b}_2)$, the following distributions are identical:

- Distribution 1: $\mathsf{com}_{2,C} = \mathsf{Com}_{2,C}(\mathsf{com}_{1,R}, \mathsf{com}_{1,C}, m; r)$. Output $(\mathsf{com}_{1,R}, \mathsf{com}_{1,C}, \mathsf{com}_{2,C}, m, r)$.
- Distribution 2: $\mathsf{com}_{2,C} = \mathsf{Com}_{2,C}(\mathsf{com}_{1,R}, \mathsf{com}_{1,C}, 0^{|m|}; r')$. Compute $\mathcal{S}(\mathsf{com}_{1,R}, \mathsf{com}_{1,C}, r', m) \to r$. Output $(\mathsf{com}_{1,R}, \mathsf{com}_{1,C}, \mathsf{com}_{2,C}, m, r)$.

Additionally, $\mathcal{S}(\mathsf{com}_{1,R}, \mathsf{com}_{1,C}, r', m)$ runs in time $2^t \cdot \mathsf{poly}(\lambda, |m|)$.

Hard to force $\mathbf{b}_R = \mathbf{b}_C$ by adversaries in $\mathcal{C}_\mathcal{A}$. Let $\lambda \in \mathbb{N}$, $\mu = \lambda^{\Theta(1)}$ and $t = \lambda^{\Omega(1)(\log\log\lambda)^{-1}} \cap \lambda^{O(1)}$. Then, for any adversary $\mathcal{A}$ in class $\mathcal{C}_\mathcal{A}$, the advantage of any adversary in the following experiment is $2^{-\mu}$.

- The challenger samples $\mathbf{b}_C \leftarrow \{0,1\}^\ell$ and sends $\mathsf{com}_{1,C} = \mathsf{Com}_{1,C}(1^\lambda, 1^\mu, 1^t, \mathbf{b}_C)$.
- Adversary sends out $\mathsf{com}_{1,R}$. Adversary wins if it outputs

$$\mathsf{com}_{1,R} = \mathsf{Com}_{1,R}(1^\lambda, 1^\mu, 1^t, \mathbf{b}_C; r)$$

  for some $r \in \{0,1\}^*$.

4.3.2 Construction of Sometimes Extractable Equivocal Commitments

In this section, we present our construction of a sometimes extractable equivocal commitments. More formally, we prove the following theorem:

Theorem 8. *Assume that the following assumptions hold:*

- *A time lock puzzle as in Definition 4 exists,*

- *a subexponentially-secure sender-equivocal OT exists, and*
- *a subexponentially-secure one-way permutation computable in* NC^1 *exists,*

then there exists a SEE *with the properties listed in Definition 21 as per parameters described in Definition 22.*

First, we specify the various class of adversary that we will handle in this scheme. Refer to Definition 21 for these notations. Let λ, μ, t be three parameters involved where $\lambda \in \mathbb{N}$, $\mu = \lambda^{\Theta(1)}$ and $t \in \lambda^{\Omega(1)(\log\log\lambda)^{-1}}$.

Definition 22 (Complexity Parameters for SEE)**.** *Consider the following complexity classes as a function of* λ, μ, t*:*

- $\mathcal{C}_D$ *: consists of all circuits of any polynomial depth and size polynomial in* 2^λ*.*
- ϵ_D *: is set to* $2^{-\lambda}$*.*
- $\mathcal{C}_\mathcal{A}$ *will be set to all circuits of size* 2^μ*.*

Required Primitives. To build this primitive, we make use of the following primitives and instantiate them with the following parameters. These instantiated parameters for the primitives we use are loose for what we require.

- *One-Way Permutation:* We require a one way permutation OWP. We assume that OWP is secure against adversaries of size polynomial in $2^{\lambda_{\mathsf{OWP}}}$, with advantage bounded by $2^{-\lambda_{\mathsf{OWP}}}$, where λ_{OWP} is the security parameter of the one-way permutation. Let the function be described as $\mathsf{OWP} : \{0,1\}^\ell \to \{0,1\}^\ell$ where $\ell = \ell(\lambda_{\mathsf{OWP}})$ is some polynomial in λ_{OWP}. We set $\lambda_{\mathsf{OWP}} = 2\mu$ and $\ell = \ell(\mu)$. We additionally require that this function is computable in NC^1. Such a function can be constructed assuming the subexponential time hardness discrete log assumption over $\mathbb{Z}_p^*$.

- *Sender Equivocal Oblivious Transfer*: We require a sender equivocal oblivious transfer $\mathsf{OT} = (\mathsf{OT}_1, \mathsf{OT}_2, \mathsf{OT}_3)$ satisfying the properties in Definition 7. We will set $\lambda_{\mathsf{ot}} = 2\lambda$, and assume that the receiver security holds against adversaries of size polynomial in $2^{\lambda_{\mathsf{ot}}}$ and with maximum advantage of $2^{-\lambda_{\mathsf{ot}}}$. Such an OT can be built assuming subexponential time and advantage hardness of DDH.

- *Time Lock Puzzle*: We require a time lock puzzle as in Definition 4. The TLP satisfies the following parameters.

  - $\lambda_{\mathsf{TLP}} = 2\lambda$,
  - $t_{\mathsf{TLP}} = \min(t, \sqrt{\mu})$. Looking ahead, for our MrNISC, we use $t = \lambda^{\Theta(1)(\log\log\lambda)^{-1}}$, in which case $t_{\mathsf{TLP}} = t$.
  - The function $D(t_{\mathsf{TLP}}) = 2^{t^{\epsilon}_{\mathsf{TLP}}}$ for some constant $\epsilon > 0$.

  Therefore, TLP with these parameters ensures the security against adversary of size polynomial in $2^{\lambda_{\mathsf{TLP}}}$ and depth bounded by $2^{t^{\epsilon}_{\mathsf{TLP}}}$ with the advantage bounded by $2^{-\lambda_{\mathsf{TLP}}}$. Further, Solve can be run by a circuit of depth $\mathsf{poly}(2^{t_{\mathsf{TLP}}}, \lambda_{\mathsf{TLP}})$.

- *Equivocal Garbled Circuits*: We require a garbling scheme $\mathsf{Gb} = (\mathsf{Garble}, \mathsf{Eval}, \mathsf{GbEquiv})$ as described in Definition 8 for NC^1 satisfying the properties of correctness and equivocation. The security parameter will be set as ℓ defined above.

Construction. We describe the construction next. In the construction, we omit the security parameters. We also exhibit how by building a bit commitment. To commit to longer messages, $\mathsf{Com}_{2,C}$ described below is repeated in parallel.

$\mathsf{Com}_{1,R}(\mathbf{b}_R \in \{0,1\}^{\ell})$: Parse $\mathbf{b}_R = (b_1, \ldots, b_\ell)$. Compute the following:

- Compute $\mathsf{ot}_{1,i} \leftarrow \mathsf{OT}_1(b_i; r_i)$ for $i \in [\ell]$ using independent randomness r_i,
- Compute $Z \leftarrow \mathsf{TLP.PGen}(\mathbf{b}_R, \mathbf{r})$, where $\mathbf{r} = (r_1, \ldots, r_\ell)$ used for generating ot_1 messages above,

- Output $\mathsf{com}_{1,R} = (\mathsf{ot}_{1,1}, \ldots, \mathsf{ot}_{1,\ell}, Z)$.

$\mathsf{Com}_{1,C}(\mathbf{b}_C \in \{0,1\}^\ell)$: Compute and output $\mathsf{com}_{1,C} = \mathsf{OWP}(\mathbf{b}_C)$.

$\mathsf{Com}_{2,C}(\mathsf{com}_{1,R}, \mathsf{com}_{1,C}, m \in \{0,1\}; r', \{r'_i\}_{i\in[\ell]})$: Parse $\mathsf{com}_{1,R} = (\mathsf{ot}_{1,1}, \ldots, \mathsf{ot}_{1,\ell}, Z)$. Let $H = H[\mathsf{com}_{1,C}, m] : \{0,1\}^\ell \to \{0,1\}$ be the circuit that takes as input $\mathbf{b} \in \{0,1\}^\ell$. It checks that $\mathsf{OWP}(\mathbf{b}) = \mathsf{com}_{1,C}$ and if so, it outputs m and 0 otherwise. Run the following steps.

- Run $\mathsf{Garble}(H; r') \to \Gamma, \mathsf{Lab}$,
- Compute $\mathsf{ot}_{2,i} = \mathsf{OT}_2(\mathsf{ot}_{1,i}, \mathsf{Lab}_{0,i}, \mathsf{Lab}_{1,i}; r'_i)$ for $i \in [\ell]$.
- Output $\mathsf{com}_{2,C} = \Gamma, \{\mathsf{ot}_{2,i}\}_{i\in[\ell]}$.

Remark 2. *The opening of* $\mathsf{com}_{2,C}$ *consist of* $(m, r', r'_1, \ldots, r'_\ell)$.

We now argue the properties of the scheme.

Indistinguishability of $\mathsf{com}_{1,R}$: The indistinguishability property follows from the security of TLP and OT. We show this by indistinguishable hybrids. The first hybrid corresponds to the case when $\mathsf{com}_{1,R}$ is generated using $\mathbf{b}_R$, whereas the last hybrid corresponds to the case $\mathsf{com}_{1,R}$ is generated using 0^ℓ.

Hybrid_0 : In this hybrid, we compute $\mathsf{com}_{1,R} = (\mathsf{ot}_{1,1}, \ldots, \mathsf{ot}_{1,\ell}, Z)$ where: $\mathsf{ot}_{1,i} = \mathsf{OT}_1(b_i; r_i)$ for $i \in [\ell]$ and $Z = \mathsf{PGen}((\mathbf{b}_R, \mathbf{r}))$.

Hybrid_1 : This hybrid is the same as the previous one except that we compute $\mathsf{com}_{1,R} = (\mathsf{ot}_{1,1}, \ldots, \mathsf{ot}_{1,\ell}, Z)$ where: $\mathsf{ot}_{1,i} = \mathsf{OT}_1(b_i; r_i)$ for $i \in [\ell]$ and $Z = \mathsf{PGen}((0^\ell, \mathbf{r}'))$ where $\mathbf{r}'$ is independently sampled.

Claim 1. *For any adversary $\mathcal{A}$, of size polynomial in 2^λ and depth bounded by any polynomial $\mathsf{poly}(\lambda)$, it holds that:*

$$|\Pr[\mathcal{A}(\mathsf{Hybrid}_0) = 1] - \Pr[\mathcal{A}(\mathsf{Hybrid}_1) = 1]| \leq 2^{-(\lambda_{\mathsf{TLP}}=2\lambda)}$$

This claim follows from the security of TLP. TLP is secure against adversaries of size polynomial in $2^{\lambda_{\mathsf{TLP}}}$, and depth $D(t_{\mathsf{TLP}}) \geq 2^{t^{\epsilon}_{\mathsf{TLP}}} \in \lambda^{\omega(1)}$. Thus one can form a reduction, distinguishing these two hybrids to breaking the security of TLP. Since $\lambda_{\mathsf{TLP}} = 2\lambda$, the claim holds.

Hybrid_2 : This hybrid is the same as the previous one except that we compute $\mathsf{com}_{1,R} = (\mathsf{ot}_{1,1}, \ldots, \mathsf{ot}_{1,\ell}, Z)$ where: $\mathsf{ot}_{1,i} = \mathsf{OT}_1(0; r_i)$ for $i \in [\ell]$ and $Z = \mathsf{PGen}((0^\ell, \mathbf{r}'))$ where $\mathbf{r}'$ is independently sampled.

Claim 2. *For any adversary $\mathcal{A}$, of size polynomial in $2^{\lambda_{\mathsf{ot}}}$, it holds that:*

$$|\Pr[\mathcal{A}(\mathsf{Hybrid}_1) = 1] - \Pr[\mathcal{A}(\mathsf{Hybrid}_2) = 1]| \leq \ell \cdot 2^{-2\lambda}$$

This claim follows from the security of OT. OT is secure against adversaries of size polynomial in $2^{\lambda_{\mathsf{ot}}}$ with an advantage $2^{-\lambda_{\mathsf{ot}}}$. We make ℓ intermediate hybrids in which we switch one by one $\mathsf{ot}_{1,i}$ to be computed using 0 instead of b_i. Each intermediate hybrid is indistinguishable with an advantage $2^{-\lambda_{\mathsf{ot}}}$. Since $\lambda_{\mathsf{ot}} = 2\lambda$, the claim holds.

Hybrid_3 : This hybrid is the same as the previous one except that we compute $\mathsf{com}_{1,R} = (\mathsf{ot}_{1,1}, \ldots, \mathsf{ot}_{1,\ell}, Z)$ where: $\mathsf{ot}_{1,i} = \mathsf{OT}_1(0; r_i)$ for $i \in [\ell]$ and $Z = \mathsf{PGen}((0^\ell, \mathbf{r}))$ where $\mathbf{r}$ is the randomness to compute $\{\mathsf{ot}_{1,i}\}_{i\in[\ell]}$.

Claim 3. *For any adversary $\mathcal{A}$, of size polynomial in $2^{2\lambda}$ and depth bounded by any polynomial $\mathsf{poly}(\lambda)$, it holds that:*

$$|\Pr[\mathcal{A}(\mathsf{Hybrid}_2) = 1] - \Pr[\mathcal{A}(\mathsf{Hybrid}_3) = 1]| \leq 2^{-2\lambda}$$

This claim follows from the security of TLP. TLP is secure against adversaries of size $2^{\lambda_{\mathsf{TLP}}}$, and depth $D(t_{\mathsf{TLP}}) \in \lambda^{\omega(1)}$. Thus one can form a reduction, distinguishing these two hybrids to breaking the security of TLP. Since $\lambda_{\mathsf{TLP}} = 2\lambda$, the claim holds.

Summing up, these three hybrids prove the required claim.

Verifiability of $\mathsf{com}_{1,C}$: This property is straightforward to observe. Observe that $\mathsf{Com}_{1,C}(\mathbf{b}) = \mathsf{OWP}(\mathbf{b})$. Since OWP has verifiable range of $\{0,1\}^\ell$, therefore $\mathsf{com}_{1,C}$ is verifiable.

Extraction when $\mathbf{b}_R = \mathbf{b}_C$: This property is also straightforward to observe and follows from the perfect correctness of OT, and the garbling scheme Gb. We define the Dec^* function. $\mathsf{Dec}^*(\mathbf{b}_R, \mathbf{r}, \mathsf{com}_{1,C}, \mathsf{com}_{1,R}, \mathsf{com}_{2,C})$: This algorithm parses $\mathbf{b}_R = (b_1, \ldots, b_\ell)$, $\mathsf{com}_{1,R} = (\mathsf{ot}_{1,1}, \ldots, \mathsf{ot}_{1,\ell}, Z)$, $\mathbf{r} = (r_1, \ldots, r_\ell)$ and $\mathsf{com}_{2,C} = (\Gamma, \mathsf{ot}_{2,1}, \ldots, \mathsf{ot}_{2,\ell})$. It does the following:

- Run $\mathsf{Lab}'_{b_i,i} \leftarrow \mathsf{OT}_3(\mathsf{ot}_{2,i}, b_i, r_i)$ for $i \in [\ell]$,
- Output $\hat{m} \leftarrow \mathsf{Eval}(\Gamma, \{\mathsf{Lab}'_{\mathbf{b}_R}\})$.

The correctness is straightforward to observe. Parse $\mathbf{r}' = (r', r'_1, \ldots, r'_\ell)$. Let $\Gamma, \mathsf{Lab} = \mathsf{Garble}(H; r')$ where $H[\mathsf{Com}_{1,C}(\mathbf{b}_R), m]$ for some message m. Let $\mathsf{com}_{1,R} = (\mathsf{ot}_{1,1}, \ldots, \mathsf{ot}_{1,\ell}, Z)$ where $\mathsf{ot}_{1,i} = \mathsf{OT}_1(b_i, r_i)$ and $\mathsf{ot}_{2,i} = \mathsf{OT}_2(\mathsf{ot}_{1,i}, \mathsf{Lab}_{0,i}, \mathsf{Lab}_{1,i}; r'_i)$ for $i \in [\ell]$. Our first observation is that $\mathsf{Lab}'_{b_i,i} = \mathsf{Lab}_{b_i,i}$ for all $i \in [\ell]$ due to perfect correctness of OT. Therefore $\hat{m} = \mathsf{Eval}(\Gamma, \{\mathsf{Lab}_{\mathbf{b}_R}\})$. Due to perfect correctness of garbled circuit we have that $\mathsf{Eval}(\Gamma, \{\mathsf{Lab}_{\mathbf{b}_R}\}) = H[\mathsf{Com}_{1,C}(\mathbf{b}_R), m](\mathbf{b}_R)$. This is equal to m, by definition of H. We also note that when $\mathbf{b}_R = \mathbf{b}_C$, it is impossible to open the OT_2 messages in any way other than one that reveals $\hat{m}$, even to an outside observer that does not know the private OT receiver state. Thus we have that the commitment is binding even to an outside observer.

Hard to force $\mathbf{b}_R = \mathbf{b}_C$ by adversaries in $\mathcal{C}_\mathcal{A}$. This follows from the reduction to the security of OWP and the fact that Solve runs in time polynomial in $2^{t_{\mathsf{TLP}}}$. Let $\mathcal{A}$ be an adversary that wins in the security game for this property and is of the size polynomial in 2^μ

with an advantage more than $2^{-\mu}$. Then, we show how to build a reduction that runs in size polynomial in $2^{\lambda_{\mathsf{OWP}}}$ and wins in breaking the security of OWP with the same advantage.

- The reduction receives as input $\mathsf{com}_{1,C} = \mathsf{OWP}(\mathbf{b})$ for a randomly chosen $\mathbf{b} \leftarrow \{0,1\}^{\ell}$.
- The reduction sends to the adversary $\mathcal{A}$, $\mathsf{com}_{1,C}$ and receives $\mathsf{com}_{1,R}$ formatted as $\mathsf{ot}_{1,1}(b'_1, r'_1), \ldots, \mathsf{ot}_{1,\ell}(b'_\ell, r'_\ell), Z = \mathsf{PGen}(\mathbf{b}', r')$.
- The reduction solves Z using a circuit size polynomial in $\mathsf{poly}(2^{t_{\mathsf{TLP}}}) \leq 2^{\frac{\mu}{2}}$ and recovers $\mathbf{b}', \mathbf{r}'$.
- It outputs $\mathbf{b}'$ if $\mathsf{com}_{1,C} = \mathsf{OWP}(\mathbf{b}')$.

Note that the view of $\mathcal{A}$ is identical to the view in the required security property of Com. If $\mathcal{A}$ produces $\mathsf{com}_{1,R}$ using $\mathbf{b}'$ that equals to the random challenge $\mathbf{b}$, then the reduction successfully recovers it by breaking TLP in time $2^{\mu/2}$. If the size of the adversary $\mathcal{A}$ is polynomial in 2^{μ}, the size of the reduction is also polynomial in $2 \cdot 2^{\mu}$ which is a contradiction as $\lambda_{\mathsf{OWP}} = 2\mu$.

Equivocation with $\mathbf{b}_R \neq \mathbf{b}_C$. We describe our algorithm $\mathcal{S}$ and then prove that it runs in time polynomial $2^{t_{\mathsf{TLP}}}$ and satisfies the equivocation property.

$\mathcal{S}(\mathsf{com}_{1,R}, \mathsf{com}_{1,C}, \mathbf{r}', m)$: Parse $\mathsf{com}_{1,R} = (\mathsf{ot}_{1,1}, \ldots, \mathsf{ot}_{1,\ell}, Z)$, $\mathsf{com}_{1,C} = \mathsf{Com}_{1,C}(\mathbf{b}_C)$ and $\mathsf{com}_{2,C} = \Gamma, \{\mathsf{ot}_{2,i}\}_{i\in[\ell]}$. Recall, how are each of the strings generated in the equivocation game. $\mathsf{com}_{1,R}$ is generated by computing: For $i \in [\ell]$, $\mathsf{ot}_{1,i} = \mathsf{OT}_1(\mathbf{b}_{R,i}; r_i)$ using some randomness r_i and Z is generated by computing $\mathsf{PGen}(\mathbf{b}_R, \mathbf{r} = (r_1, \ldots, r_\ell))$. Receiver's randomness may be arbitrarily chosen. For the committer, $\mathsf{com}_{2,C}$ is generated honestly by committing to 0 using honestly generated randomness $\mathbf{r}'$. Parse $\mathbf{r}' = (r', r'_1, \ldots, r'_\ell)$. Γ, Lab is computed as $\mathsf{Garble}(H[\mathsf{com}_{1,C}, 0]; r')$. Then we compute $\mathsf{ot}_{2,i} = \mathsf{OT}_2(\mathsf{ot}_{1,i}, \mathsf{Lab}_{0,i}, \mathsf{Lab}_{1,i}; r'_i)$ for $i \in [\ell]$. Finally $\mathsf{com}_{2,C} = (\Gamma, \{\mathsf{ot}_{2,i}\}_{i\in[\ell]})$. Thus to equivocate, compute the following steps:

- Run $\mathsf{Solve}(Z) = (\mathbf{b}_R, \mathbf{r})$.
- Equivocate Garbled Circuit: Run

$$\mathsf{GbEquiv}(\Gamma, \mathsf{Lab}_{\mathbf{b}_R}, H[\mathsf{com}_{1,C}, m], \mathbf{b}_R) \rightarrow (\mathsf{Lab}', s)$$

  where Lab' is the new set of labels and s is the randomness that explains

$$\mathsf{Garble}(H[\mathsf{com}_{1,C}, m]; s) \rightarrow \Gamma, \mathsf{Lab}'$$

  . Further $\mathsf{Lab}'_{\mathbf{b}_R} = \mathsf{Lab}_{\mathbf{b}_R}$.
- Equivocate ot_2: For $i \in [\ell]$, compute $s_i = \mathsf{OT.Equiv}(\mathbf{b}_{R,i}, r_i, \mathsf{ot}_{2,i}, r'_i, \mathsf{Lab}'_{0,i}, \mathsf{Lab}'_{1,i})$.
- Output $(m, \mathsf{com}_{1,R}, \mathsf{com}_{1,C}, \mathsf{com}_{2,C}, \mathbf{s} = (s, s_1, \ldots, s_\ell))$.

The run time of the simulator above is polynomial in $2^{t_{\mathsf{TLP}}}$ which is polynomial in 2^t as per the setting of the parameters. The proof of security is immediate and follows from the equivocation property of the garbled circuit and OT. We show this by identical hybrids. The first hybrid corresponds to the case when m is committed, and the last hybrid corresponds to the simulator, where 0 is committed first and then equivocated to m.

Hybrid_0 : In this hybrid, compute $\mathsf{com}_{2,C}(\mathsf{com}_{1,R}, \mathsf{com}_{1,C}, m; \mathbf{r}')$. Output $(m, \mathsf{com}_{1,R}, \mathsf{com}_{1,C}, \mathsf{com}_{2,C}, \mathbf{r}')$.

Hybrid_1 : In this hybrid, we use the equivocation of the garbled circuit property. First generate $\Gamma, \mathsf{Lab} \leftarrow \mathsf{Garble}(H[\mathsf{com}_{1,C}, 0]; r')$. Observe that $H[\mathsf{com}_{1,C}, 0](\mathbf{b}_R) = H[\mathsf{com}_{1,C}, m](\mathbf{b}_R) = 0$. Therefore, due to the equivocation property of the garbled circuits, we can compute $\mathsf{GbEquiv}(\Gamma, \mathsf{Lab}_{\mathbf{b}_R}, H[\mathsf{com}_{1,C}, m], \mathbf{b}_R) \rightarrow (\mathsf{Lab}', s)$. We set $\mathsf{com}_{2,C} = \Gamma$ and $\{\mathsf{ot}_{2,i} = \mathsf{OT}_2(\mathsf{ot}_{1,i}, \mathsf{Lab}'_{0,i}, \mathsf{Lab}'_{1,i}; r'_i)\}_{i \in [\ell]}$. Output $(m, \mathsf{com}_{1,R}, \mathsf{com}_{1,C}, \mathsf{com}_{2,C}, (s, r'_1, \ldots, r'_\ell))$.

The two distributions above are identical due to the equivocation property of the garbled circuits.

Hybrid_2 : In this hybrid, we use the equivocation property of OT. We first generate $\mathsf{ot}_{2,i} = \mathsf{OT}_2(\mathsf{ot}_{1,i}, \mathsf{Lab}_{0,i}, \mathsf{Lab}_{1,i}; r'_i)$ for $i \in [\ell]$. Then, since $\mathsf{com}_{1,R}$ consists of OT_1 messages corresponding to $\mathbf{b}_R \neq \mathbf{b}_C$, we can equivocate $\mathsf{ot}_{2,i}$ as follows. We run

$$s_i = \mathsf{OT.Equiv}(\mathbf{b}_{R,i}, r_i, \mathsf{ot}_{2,i}, r'_i, \mathsf{Lab}'_{0,i}, \mathsf{Lab}'_{1,i}).$$

This can be done because $\mathsf{Lab}'_{b_{R,i},i} = \mathsf{Lab}_{b_{R,i},i}$. Thus at the end of this we have randomness s_i such that $\mathsf{ot}_{2,i} = \mathsf{OT}_2(\mathsf{ot}_{1,i}, \mathsf{Lab}'_{0,i}, \mathsf{Lab}'_{1,i}; s_i) = \mathsf{OT}_2(\mathsf{ot}_{1,i}, \mathsf{Lab}_{0,i}, \mathsf{Lab}_{1,i}; r'_i)$. Output of this hybrid is $(m, \mathsf{com}_{1,R}, \mathsf{com}_{1,C}, \mathsf{com}_{2,C}, \mathbf{s})$ where $\mathbf{s} = (s, s_1, \ldots, s_\ell)$.

This hybrid is identical to the previous hybrid due to the security of OT.

4.3.3 Non-Interactive Distributional Indistinguishability

This section defines the notion of Non-Interactive Distributional Indistinguishability arguments (NIDI for short). The definitions below are strengthenings of analogous definitions given by Khurana [Khu21], where the difference is that their definitions assume that the verifier's message comes after the prover's message; see Remark 4 for details.

Definition 23 (Syntax of NIDI)**.** *A* NIDI *for an* NP *language* L *and its relation* R_L *consists of the following algorithms.*

- $\mathsf{P}(1^{\lambda_S}, 1^{\lambda_D}, \mathcal{D})$: *The prove algorithm takes as input two security parameters* 1^{λ_S} *and* 1^{λ_D} *(one for the soundness property, and one for the distribution indistinguishability property), a polynomial time sampler* $\mathcal{D}(\cdot)$ *that on input* λ_D *samples from* $(\mathcal{X}, \mathcal{W})$ *consisting of tuples that are in* R_L*. It outputs a proof string* π*.*

- $\mathsf{V}(\tau, \pi)$: *The verification algorithm is a deterministic polynomial time algorithm that takes as input a string* $\tau \in \{0,1\}^{\ell_{\mathsf{NIDI}}(\lambda_S)}$ *for some polynomial* ℓ_{NIDI}*, a proof* π*, and it outputs a string in* $x \in \perp \cup \{0,1\}^*$*.*

A NIDI scheme satisfies a number of different properties: completeness, soundness and distributional indistinguishability.

Definition 24 (Completeness)**.** *We require that for any poly-time samplable distribution $\mathcal{D} = (\mathcal{X}, \mathcal{W})$ supported over instance-witness pairs in R_L, we have that for every $\lambda_S, \lambda_D \in \mathbb{N}$:*

$$\Pr_{\tau,\pi}[x \in L \mid \mathsf{V}(\tau, \pi) = x] = 1,$$

where $\pi \leftarrow \mathsf{P}(1^{\lambda_S}, 1^{\lambda_D}, \mathcal{D})$ and $\tau \leftarrow \{0,1\}^{\ell_{\mathsf{NIDI}}(\lambda_S)}$.

Definition 25 ($(\mathcal{C}_D, \mathcal{C}_{DI}, \epsilon_D, \epsilon_{DI})$- Distributional Indistinguishability)**.** *Let $\mathcal{D}_0 = (\mathcal{X}_0, \mathcal{W}_0)$ and $\mathcal{D}_1 = (\mathcal{X}_1, \mathcal{W}_1)$ be two polynomial-time distribution samplers supported over tuples in R_L. Further, assume that $\mathcal{X}_0$ and $\mathcal{X}_1$ are $(\mathcal{C}_D, \epsilon_D)$ indistinguishable. Then, we require that:*

$$\mathsf{P}(1^{\lambda_S}, 1^{\lambda_D}, \mathcal{D}_0) \approx_{\mathcal{C}_{DI}, \epsilon_{DI}} \mathsf{P}(1^{\lambda_S}, 1^{\lambda_D}, \mathcal{D}_1).$$

Definition 26 (Completeness, Extraction)**.** *There exist a (possibly inefficient) algorithm $E : \{0,1\}^* \to \{0,1\}$ with the following properties. Let $\lambda_S, \lambda_D \in \mathbb{N}$, $\tau \in \{0,1\}^{\ell_{\mathsf{NIDI}}(\lambda_S)}$ and π be any proof string such that $\mathsf{V}(\tau, \pi) \to x$ where $x \neq \perp$. Then:*

- $E(\tau, \pi) = 1 \implies x \in L$.
- *For any polynomial time samplable distribution $\mathcal{D} = (\mathcal{X}, \mathcal{W})$ supported over tuples in R_L, it holds that:*

$$\Pr\left[E(\tau,\pi) = 1 \;\middle|\; \begin{array}{l} \tau \xleftarrow{\$} \{0,1\}^{\ell_{\mathsf{NIDI}}(\lambda_S)} \\ \pi \leftarrow \mathsf{P}(1^{\lambda_S}, 1^{\lambda_D}, \mathcal{D}) \end{array}\right] = 1.$$

Definition 27 ($(\mathcal{C}_S, \epsilon_S)$-Soundness)**.** *We define the following security game played between the adversary $\mathcal{A} \in \mathcal{C}_S$ and the challenger. We denote it by $\mathsf{expt}_{\mathcal{A},\mathsf{NIDI},sound}(1^{\lambda_S}, 1^{\lambda_D})$:*

1. *$\mathcal{A}$ is given $1^{\lambda_S}, 1^{\lambda_D}$ as the input.*
2. *The challenger manages a list List that is initially empty. The contents of the list are visible to the adversary at all stages.*
3. *Adversary can ask adaptively a polynomial number of τ-query. If that happens, sample $\tau' \leftarrow \{0,1\}^{\ell_{\mathsf{NIDI}}(\lambda_S)}$ and append τ' to List.*

4. *Adversary outputs a proof string* π *and a* $\tau \in \mathsf{List}$. *The adversary wins if* $\mathsf{V}(\tau, \pi) = x$ *where* $x \neq \perp$ *and* $E(\tau, \pi) = 0$.

The NIDI *scheme satisfies* $(\mathcal{C}_S, \epsilon_S)$*-soundness if for all adversaries* $\mathcal{A} \in \mathcal{C}_S$*:*

$$\Pr[\mathsf{expt}_{\mathcal{A},\mathsf{NIDI},sound}(1^{\lambda_S}, 1^{\lambda_D}) = 1] \leq \epsilon_S$$

Remark 3. *Observe that the last two properties gives rise to a meaningful soundness property. The extraction property (Definition 26) ensures that whenever* $x \notin L$*, if* $\mathsf{V}(\tau, \pi) = x$ *then* $E(\tau, \pi) \neq 1$*. The Soundness property (Definition 27) then says that for a computationally bounded adversary it is hard to come up with a proof string* π *such that* $\mathsf{V}(\tau, \pi) = x$ *and* $E(\tau, \pi) \neq 1$*. This rules out a computationally bounded adversary producing false instances.*

Remark 4 (weaker soundness requirement)**.** *One could ask for a weaker soundness requirement where the proof string must be published before making any* τ *query. Such a* NIDI *will not be sufficient for us. The protocol in [Khu21] satisfies this weaker property.*

4.3.4 Construction of NIDI

In this section, we prove the following theorem:

Theorem 9. *Assume that the following assumptions hold:*

- *A subexponentially secure indistinguishability obfuscator exists,*
- *A time lock puzzle as in Definition 4 exists,*
- *A subexponentially-secure NIWI exists,*

then, there exist a NIDI *scheme that satisfies security definitions in Definitions 24, 27, 25, 26, and is secure against adversaries of subexponential size.*

The only difference from the primitives used in the construction by [Khu21] is the usage of a TLP as opposed to a public-key encryption scheme. This is the key component that helps us argue security in the presence of adaptive τ queries.

We start by giving a short overview of how we can construct a NIDI scheme satisfying properties specified in Definitions 23 to 25 and 27.

4.3.4.1 Overview of NIDI

We now describe the intuitive ideas behind the construction of NIDI given by Khurana [Khu21] and identify the reasons why the properties of the construction fall short of satisfying, and then we will describe our change to the construction.

Intuitively speaking, a NIDI scheme allows a prover to prove with respect to efficiently samplable distributions $\mathcal{D}$ supported over instance-witness pairs in some relation R_L corresponding to some NP language L. For example, the distribution can be the set of encryptions of 1 with respect to some public key PK, and the language could be the set of all ciphertexts with respect to public key PK.

The idea is that a prover can generate a proof Π using this distribution $\mathcal{D}$. A verifier can use this proof to sample from $\mathcal{D}$. It simply chooses a random string τ, and runs $x \leftarrow \mathsf{V}(\tau, \pi)$. The soundness guarantee of NIDI ensures that with high probability if τ is randomly chosen, x sampled by the verifier must be in L. Further, the distributional indistinguishability property guarantees that for computationally indistinguishable distributions $\mathcal{D}_0$ and $\mathcal{D}_1$ (such as encryptions of 0 vs. encryptions of 1) which are supported over R_L, NIDI generated using $\mathcal{D}_0$ is computationally indistinguishable to the proof generated using $\mathcal{D}_1$. Thus, NIDI can be useful in protocols where we need to sample well-formed messages per some specifications in one round (using a single message by a prover and a verifier) while maintaining indistinguishability guarantees. The prover displays Π and the verifier displays τ, and this let us sample $x \leftarrow \mathsf{V}(\tau, \pi)$.

The construction of Khurana [Khu21] uses iO, a public-key encryption PKE scheme with verifiable public keys[2] and perfect correctness, a non-interactive witness indistinguishable

[2]where given a string, it is efficiently checkable if it is a valid public key.

argument NIWI, and a one-way permutation OWP. The prover obfuscates a program Π that takes as input τ in the range of OWP. It derives using τ randomness r using a PRF key K hardwired inside the program. Using this randomness, the program computes $c = \mathsf{PKE.Enc}(\mathsf{pk}, 0)$ using the public-key pk, sampled by the prover and hardwired in the program. It also samples (x, w) by running sampler $\mathcal{D}$ and computes a NIWI proof π proving either $x \in L$ or c is an encryption of $\mathsf{OWP}^{-1}(\tau)$ (using w as its witness). The output of Π on input τ is (c, x, π). The verifier evaluates Π at τ to get (c, x, π), and then it verifies the NIWI, and if it succeeds, it outputs x.

To argue distributional indistinguishability, we go input by input as commonly done in many iO proofs. Using hybrid arguments, we can go from obfuscating a program that uses $\mathcal{D}_0$ to sample instance to a program that uses $\mathcal{D}_1$. We do this by changing the program's behavior undetectably one input at a time. For every input τ, we puncture the PRF key at τ, and hardwire the circuit output (c, x, π) at input τ. Then, we switch the encryption c to encrypt to $\mathsf{OWP}^{-1}(\tau)$ and then we generate NIWI π by using $\mathsf{OWP}^{-1}(\tau)$ as the witness for the statement. At this point, we start sampling x from $\mathcal{D}_1$ (instead of $\mathcal{D}_0$). We apply the same sequence of hybrids in reverse to reach a point where at input τ, the circuit's behavior is identical to the previous behavior except that it uses $\mathcal{D}_1$ to sample x when provided input τ. For this to work out, we need PKE, PRF, iO, the distributions $\mathcal{D}_0, \mathcal{D}_1$ and NIWI to be indistinguishable with an advantage lesser than $2^{-|\tau|}$.

Soundness, on the other hand, is a bit more involved. For soundness, we want that if an adversary outputs a program Π, that on input τ chosen by the verifier outputs (c, x, π) where π verifies but $x \notin L$, then it should somehow translate to recovering $\mathsf{OWP}^{-1}(\tau)$ efficiently. This means that PKE must be breakable by an algorithm against which OWP is still secure. Due to perfect soundness of NIWI, since $x \notin L$ it must mean that c encrypts $\mathsf{OWP}^{-1}(\tau)$. Because of this, a reduction could invert c to recover $\mathsf{OWP}^{-1}(\tau)$. This seems to be at odds with the requirement of PKE to be more secure than OWP as required in the distributional indistinguishability property.

To address this issue, Khurana observed that if Π is fixed first, and τ is chosen after, we can build a non-uniform polynomial-time reduction that breaks OWP security. The idea is that we can guess the secret key sk corresponding to the public key pk. This is independent of the τ chosen by the verifier. Because of this, if an adversary exists that wins in the soundness experiment, we can construct another circuit that wins in the OWP game.

Since we are working to construct round efficient MPC protocols, we cannot allow a verifier to choose τ after seeing the proof Π. It must be done simultaneously in the same round. As a result of this, the previous proof of Khurana breaks down for our security requirement. We fix this by introducing a new axis of hardness. We use a time lock puzzle to commit instead of a public key encryption. The commitment is secure against adversary of size $2^{\lambda^{c_1}}$ of depth polynomial in λ, but can be broken by a circuit of size $2^{\lambda^{c_2}}$ where $c_2 \ll c_1$. This ensures distributional indistinguishability against adversaries of polynomial depth and $2^{\lambda_{c_1}}$ size. For soundness, we choose OWP, so that it is secure against adversaries of size $2^{\lambda^{c_2}}$. Thus, we can show a reduction that breaks the commitment in size $2^{\lambda^{c_2}}$ to invert OWP.

We now describe our construction of the NIDI scheme (for any NP language L with its relation verifier R) satisfying all the properties described in Section 4.3.3. The scheme is almost identical to the construction of [Khu21] except for one change which we highlight below in red. Before we proceed, we describe the complexity classes involved.

Complexity Classes. We have the following:

- **Initial Distribution Properties.** We will consider distributions that are $\epsilon_D(\lambda_D) = 2^{-\lambda_D}$ indistinguishable against adversaries in the class $\mathcal{C}_D$ which consists of all circuits of depth $\mathsf{poly}(\lambda_D)$ and size 2^{λ_D}.

- **Properties of the resulting NIDI Proofs.** We will guarantee that the NIDI proofs for such distributions are indistinguishable for $\mathcal{C}_{DI} = \mathcal{C}_D$ described above (same circuit class). The advantage of adversaries in the security game will be bounded by $\epsilon_{DI} = O(\epsilon_D \cdot 2^{\ell(\lambda_S)})$

for some fixed polynomial ℓ described later.

- **Soundness properties.** We will ensure that the soundness holds against adversaries in $\mathcal{C}_S$ which consists of all adversaries of size 2^{λ_S}. The advantage will be bounded by $\epsilon_S = 2^{-\lambda_S}$.

Required Primitives. We make use of the following primitives and instantiate them with the following parameters. These instantiated parameters for the primitives we use are loose for what we require.

- *OWP*: We require a one way permutation OWP. We assume that OWP is secure against adversaries of size $2^{\lambda_{\mathsf{OWP}}}$, with advantage bounded by $2^{-\lambda_{\mathsf{OWP}}}$, where λ_{OWP} is the security parameter of the one-way permutation. Let the function be described as $\mathsf{OWP} : \{0,1\}^{\ell_{\mathsf{OWP}}} \to \{0,1\}^{\ell_{\mathsf{OWP}}}$ where $\ell_{\mathsf{OWP}} = \ell_{\mathsf{OWP}}(\lambda_{\mathsf{OWP}})$ is some polynomial in λ_{OWP}. We set $\lambda_{\mathsf{OWP}} = \lambda_S$ and $\ell = \ell_{\mathsf{OWP}}(\lambda_S)$. Such a function can be constructed assuming the subexponential time and advantage hardness of $\mathsf{DDH}/\mathsf{SXDH}$ assumption.

- *Indistinguishability Obfuscation*: We require an indistinguishability Obfuscator iO. This scheme uses λ_{iO} as the security parameter and is secure against adversaries of size $2^{\lambda_{\mathsf{iO}}}$ with advantage $2^{-\lambda_{\mathsf{iO}}}$. Such a primitive can be built using well-studied assumptions as shown in [JLS21b, JLS21a]. We set λ_{iO} as a large enough polynomial. In particular, setting $\lambda_{\mathsf{iO}} = \ell_{\mathsf{OWP}}\lambda_D$ suffices.

- *Time-Lock Puzzles*: We require a time lock puzzle as in Definition 4. The TLP satisfies the following parameters.

  - $\lambda_{\mathsf{TLP}} = \lambda_D \ell_{\mathsf{OWP}}$,
  - $t_{\mathsf{TLP}} = \lambda_S^{\rho}$ for a small constant $\rho > 0$,
  - The function $D(t) = 2^{t^{\epsilon}_{\mathsf{TLP}}}$ for some constant $\epsilon > 0$.

Therefore, TLP with these parameters ensures the security against adversary of size $2^{\lambda_{\mathsf{TLP}}}$ and depth bounded by $2^{t^{\epsilon}_{\mathsf{TLP}}}$ with the advantage bounded by $2^{\lambda_{\mathsf{TLP}}}$. Further, Solve can be run by a circuit of depth $\mathsf{poly}(2^{t_{\mathsf{TLP}}})$.

- *Puncturable PRF*: We require a puncturable PRF, $\mathsf{PPRF} = (\mathsf{Puncture}, \mathsf{Eval})$. Assume the length of the key is randomly chosen of length $\ell_{\mathsf{PPRF}}(\lambda_{\mathsf{PPRF}})$ where λ_{PPRF} is its security parameter. The length of the output is some polynomial implicit in the scheme. We assume that the PPRF is secure against adversaries of size $2^{\lambda_{\mathsf{PPRF}}}$ with a maximum advantage of $2^{-\lambda_{\mathsf{PPRF}}}$. We set $\lambda_{\mathsf{PPRF}} = \lambda_D \cdot \ell$.

- *NIWI*: We require a non-interactive witness indistinguishable proof NIWI for NP, that is secure against adversaries of size $2^{\lambda_{\mathsf{NIWI}}}$ with advantage bounded by $2^{-\lambda_{\mathsf{NIWI}}}$. We set $\lambda_{\mathsf{NIWI}} = \ell \cdot \lambda_D$. NIWIs can be instantiated assuming the subexponential time and advantage security of the SXDH assumption over bilinear maps.

Construction. We now describe the construction.

$\mathsf{NIDI.P}(1^{\lambda_S}, 1^{\lambda_D}, \mathcal{D})$: Sample a PPRF key $K \leftarrow \{0,1\}^{\ell_{\mathsf{PPRF}}}$.

The proving algorithm outputs $\widetilde{C} = \mathsf{iO}(C[\mathcal{D}, K])$ where the program $C[\mathcal{D}, K]$ is described in Figure 4.1.

$\mathsf{NIDI.V}(\tau, \widetilde{C})$: Run $\widetilde{C}(\tau)$. If this evaluation outputs $\perp$, output $\perp$. Otherwise, parse the output as (x, c, π). Run $\mathsf{NIWI.V}(x, c, \pi)$ for the language L'. If the verification fails output $\perp$. Otherwise, output x.

$E(\tau, \widetilde{C})$: Run $\widetilde{C}(\tau)$. Output 0 if this yields $\perp$. Otherwise parse the output as (x, c, π). Run $\mathsf{NIWI.V}(x, c, \pi)$. If the proof does not verify, output 0. Otherwise, check if $c = \mathsf{TLP.PGen}(\alpha)$ for some α. If this is not the case or $\mathsf{OWP}(\alpha) \neq \tau$, then output 1. Otherwise output 0.

The Circuit $C[\mathcal{D}, K]$

Hardwired: The PPRF key K, and the distribution sampled $\mathcal{D}$.

Input: $\tau \in \{0,1\}^\ell$

Computation:

1. Compute $r \leftarrow \mathsf{PPRF.Eval}(K, \tau)$.
2. Parse $r = (r_1, r_2, r_3)$. Compute:
   - $(x, w) = \mathcal{D}(r_1)$,
   - $c = \mathsf{TLP.PGen}(0^\ell; r_2)$,
3. For the statement $(x, c) \in L'$, compute $\pi = \mathsf{NIWI.P}((x, c), w; r_3)$. We define the language

$$L' = \left\{ (x', c') \;\middle|\; \exists w' : \; R\left(x', w'\right) = 1 \vee \exists \alpha : \; \left(c' = \mathsf{TLP.PGen}(\alpha) \wedge \mathsf{OWP}(\alpha) = \tau\right) \right\}$$

4. Output (x, c, π).

The code highlighted in red is the only difference from the construction proposed by [Khu21]. In their scheme, they generate $c = \mathsf{Enc}(\mathsf{pk}, 0^\ell)$ where pk is a public key for a dense cryptosystem, which is sampled and hardwired in the program. Any adversary breaking the soundness must commit/encrypt to an element in $\mathsf{OWP}^{-1}(\tau)$, and the reduction breaks open the encryption to win in the OWP game. This breaking is done by non-uniformly choosing the secret key for the public key pk. This only allows τ queries to come after the prover outputs a NIDI proof. A TLP helps us to bypass this issue.

Figure 4.1: The Circuit $C[\mathcal{D}, K]$

Observe that the completeness property is immediate. Similarly, the distributional indistinguishability property argument is also identical to the proof in [Khu21] because the public key encryption is replaced with a time-lock puzzle. All we need for the proof is the component c to satisfy the indistinguishability property.

Sketch of Indistinguishability: The idea for indistinguishability is to go input by input as common in applications of iO. Consider two distributions $\mathcal{D}_0$ and $\mathcal{D}_1$ which yields instances that are $(\mathcal{C}_D, \epsilon_D)$ indistinguishable. The proof will follow the following strategy. We will define 2^ℓ hybrids where a typical hybrid ($\mathsf{Hybrid}_{\tau'}$) is indexed by $\tau' \in [2^\ell]$. In $\mathsf{Hybrid}_{\tau'}$, we will generate an obfuscation $\widetilde{C}$ of program $C[\mathcal{D}_0, \mathcal{D}_1, K, \tau']$ described in Figure 4.2. Now to prove indistinguishability, we need to prove that $\mathsf{Hybrid}_{\tau'}$ and $\mathsf{Hybrid}_{\tau'}$ are $O(2^{-\lambda_D})$ indistinguishable for circuits in $\mathcal{C}_D$. This will yield a total advantage of $O(2^{-\lambda_D \ell})$. We can do this again by using standard tricks. Observe that the only change in the $C[\mathcal{D}_0, \mathcal{D}_1, K, \tau']$ and $C[\mathcal{D}_0, \mathcal{D}_1, K, \tau' + 1]$ is its behavior at the input $\tau' + 1$. In this case, we take the following hybrids. The indistinguishability between the hybrids are immediate and follow similarly to [Khu21].

- Hybrid'_0 : This is the same as $\mathsf{Hybrid}_{\tau'}$.
- Hybrid'_1 : In this hybrid the only is change is to puncture the PRF key K^* at $\tau' + 1$ and use it to generate the circuit we obfuscate. To do so, we hardwire the output (x, c, π) at input $\tau' + 1$ generated from $\mathcal{D}_0$ as before using $(r_1, r_2, r_3) = \mathsf{PPRF.Eval}(K, \tau' + 1)$. This hybrid is indistinguishable to the previous hybrid against adversaries in $\mathcal{C}_D$ with advantage $O(2^{-\lambda_{\mathsf{iO}}})$ due to the correctness property of the PPRF and the security of iO.
- Hybrid'_2 : In this hybrid the only change from the previous hybrid is that we generate (x, c, π) from $\mathcal{D}_0$ but now using true randomness (r_1, r_2, r_3). This hybrid is indistinguishable from the previous hybrid against adversaries in $\mathcal{C}_D$ with advantage $O(2^{-\lambda_{\mathsf{PPRF}}})$ due to the security property of the PPRF.
- Hybrid'_3 : In this hybrid the only change is to generate (x, c, π), where c is computed as

$\mathsf{TLP.PGen}(\alpha)$ where $\mathsf{OWP}(\alpha) = \tau' + 1$. This hybrid is indistinguishable to the previous hybrid against adversaries in $\mathcal{C}_D$ with advantage $O(2^{-\lambda_{\mathsf{TLP}}})$ due to the security property of the TLP.

- Hybrid'_4 : In this hybrid, the only change is to generate (x, c, π) by using the opening of c as a witness to generate π. This hybrid is indistinguishable from the previous hybrid against adversaries in $\mathcal{C}_D$ with advantage $O(2^{-\lambda_{\mathsf{NIWI}}})$ due to the security property of the NIWI.

- Hybrid'_5 : In this hybrid, the only change is to generate (x, c, π) by switching x to be sampled from $\mathcal{D}_1$. This hybrid is indistinguishable from the previous hybrid against adversaries in $\mathcal{C}_D$ with advantage $O(2^{-\lambda_D})$ due to the indistinguishability property of $\mathcal{D}_0$ and $\mathcal{D}_1$.

- Hybrid'_6 : In this hybrid, the only change is to generate (x, c, π) by using a witness of x to generate π. This hybrid is indistinguishable from the previous hybrid against adversaries in $\mathcal{C}_D$ with advantage $O(2^{-\lambda_{\mathsf{NIWI}}})$ due to the security property of NIWI.

- Hybrid'_7 : In this hybrid the only change is to generate (x, c, π) where c is computed as $\mathsf{TLP.PGen}(0^\ell)$. This hybrid is indistinguishable to the previous hybrid against adversaries in $\mathcal{C}_D$ with advantage $O(2^{-\lambda_{\mathsf{TLP}}})$ due to the security property of TLP.

- Hybrid'_8 : In this hybrid the only change is to generate (x, c, π) by using $(r_1, r_2, r_3) = \mathsf{PPRF.Eval}(K, \tau' + 1)$. This hybrid is indistinguishable to the previous hybrid against adversaries in $\mathcal{C}_D$ with advantage $O(2^{-\lambda_{\mathsf{PPRF}}})$ due to the security property of the PPRF.

- Hybrid'_9 : This hybrid is the same as $\mathsf{Hybrid}_{\tau'+1}$. This hybrid is indistinguishable from the previous hybrid against adversaries in $\mathcal{C}_D$ with advantage $O(2^{-\lambda_{\mathsf{iO}}})$ due to the correctness property of the PPRF and the security of iO.

Observe that the parameters $\lambda_{\mathsf{iO}}, \lambda_{\mathsf{NIWI}}, \lambda_{\mathsf{PPRF}}$ are set to be larger than $\lambda_D \ell$. Thus, the total advantage is bounded by $O(2^{-\lambda_D} + 2^{-\ell\lambda_D}) = O(2^{-\lambda_D})$. This finishes the overview.

The Circuit $C[\mathcal{D}_0, \mathcal{D}_1, K, \tau']$

Hardwired: The PPRF key K, and the distribution sampled $\mathcal{D}$.

Input: $\tau \in \{0,1\}^\ell$

Computation:

1. Compute $r \leftarrow \mathsf{PPRF.Eval}(K, \tau)$.
2. Parse $r = (r_1, r_2, r_3)$. Compute:
   - If $\tau \leq \tau'$, then $(x, w) = \mathcal{D}_1(r_1)$ otherwise $(x, w) = \mathcal{D}_0(r_1)$.
   - $c = \mathsf{TLP.PGen}(0^\ell; r_2)$,
3. For the statement $(x, c) \in L'$, compute $\pi = \mathsf{NIWI.P}((x, c), w; r_3)$. We define the language

$$L' = \left\{ \left(x', c'\right) \ | \exists w' : \ R\left(x', w'\right) = 1 \vee \exists \alpha : \ \left(c' = \mathsf{TLP.PGen}(\alpha) \wedge \mathsf{OWP}(\alpha) = \tau\right) \right\}$$

4. Output (x, c, π).

Figure 4.2: The Circuit $C[\mathcal{D}_0, \mathcal{D}_1, K, \tau']$

We now focus on the soundness argument:

Sketch of Soundness. Consider a circuit $\mathcal{A}$ of size 2^{λ_S} in the soundness security game. Assume that the adversary wins in the soundness experiment with probability ϵ. We will show that we can build a reduction of size $O(2^{\lambda_S})$ winning in the OWP inversion game with the ϵ/Q for some polynomial. Remember in the soundness game adversary is given a list $\tau_1, \ldots, \tau_Q$ of randomly chosen elements for some polynomial Q and it outputs $\widetilde{C}$ and an index $i \in [Q]$. For this $\widetilde{C}$, it holds that $\widetilde{C}[\tau_i] = (x_i, c_i, \pi_i)$ such that π_i verifies and $E(\tau_i, \widetilde{C}) = 0$. This means that c_i must be of the form $\mathsf{TLP.PGen}(\alpha_i)$ where $\mathsf{OWP}(\alpha_i) = \tau_i$. The reduction simply runs $\mathsf{TLP.Solve}(c_i)$ and outputs α_i as a preimage of τ_i. This means that the reduction succeeds with advantage at least ϵ/Q. Reduction needs to run $\mathcal{A}$ and then run $\mathsf{TLP.Solve}$, which runs in time polynomial in 2^{λ^ρ} for some small constant ρ. Thus, this takes $O(2^{\lambda_S})$ time as $\lambda_S = \lambda$.

4.4 The Formal Construction and Security Proof

In this section, we formally construct a reusable statistical ZK arguments with sometimes statistical soundness (henceforth denoted by $\mathsf{zk} = (\mathsf{ZKProve}_1, \mathsf{ZKVerify}_1, \mathsf{ZKProve}_2, \mathsf{ZKVerify}_2)$) as defined in Section 4.1. We now give the parameters associated with various adversary classes that we will guarantee security for. We will then follow it up with the parameters for the underlying primitives we use. Let λ be the security parameter for zk.

Definition 28 (Parameters of zk)**.** *We achieve* zk *for the following parameters.*

- *For the soundness property the parameters* $(\mathcal{C}_{sound}, \epsilon_{sound,1}, \epsilon_{sound,2})$ *we achieve will be as follows.* $\mathcal{C}_{sound}$ *consists of circuits with any* $\mathsf{poly}(\lambda)$ *depth Boolean circuits of size bounded by any polynomial in* 2^λ*. We will set* $\epsilon_{sound,1} = 2^{-\ell} = \Omega(2^{-\ell_\mu})$ *for some polynomial* $\ell(\lambda)$ *and* $\epsilon_{sound,2} = 2^{-\lambda}$*.* ℓ_μ *is defined when we define the parameters for* SEE *scheme.*

- *For the zero knowledge, $\mathcal{C}_{\mathcal{S}}, \mathcal{C}_{\mathsf{zk}}, \epsilon_{\mathcal{S}}$ are set as follows. $\mathcal{C}_{\mathcal{S}}$ is the complexity class of the simulator. $\mathcal{C}_{\mathcal{S}}$ consists of circuits of size 2^{λ^ρ} for some parameter $\rho \in \Theta(1) \log\log \lambda^{-1}$, which can be chosen as a parameter to the scheme. We will also set $\mathcal{C}_{\mathsf{zk}}$, which is the class of the zero-knowledge verifier to be the same as $\mathcal{C}_{sound}$ of $\mathsf{poly}(\lambda)$ depth circuits of size polynomial in 2^λ. $\epsilon_{\mathcal{S}}$ will be set as $2^{-\lambda}$.*

Used Primitives. We make use of the following primitives and instantiate them with the following parameters. These instantiated parameters for the primitives we use are loose for what we require.

- *NIDI Arguments*: We require a NIDI scheme (Definition 23) as per the following specifications. Such a NIDI uses two security parameters $\lambda_{\mathsf{NIDI},S}$ and $\lambda_{\mathsf{NIDI},D}$. We set $\lambda_{\mathsf{NIDI},S} = \lambda$. We set $\mathcal{C}_{\mathsf{NIDI},S}$ to consist of all adversaries of size polynomial in $2^{\lambda_{\mathsf{NIDI},S}}$. We set $\epsilon_{\mathsf{NIDI},S} = 2^{-\lambda_{\mathsf{NIDI},S}}$. For this choice of $\lambda_{\mathsf{NIDI},S} = \lambda$, let $\ell_{\mathsf{NIDI}}(\lambda)$ be the length of τ's used in the scheme. We set $\lambda_{\mathsf{NIDI},D}$ as a polynomial in λ. This polynomial will ensure that the distributions we use, on input $\lambda_{\mathsf{NIDI},D}$ satisfy the following parameters:

  - $\mathcal{C}_{\mathsf{NIDI},D}$ consists of all circuits of depth $\mathsf{poly}(\lambda)$ and size polynomial in 2^λ.
  - $\epsilon_{\mathsf{NIDI},D} = 2^{-\ell_{\mathsf{NIDI}} \cdot \ell_\mu \lambda}$. ($\ell_\mu$ is defined along with the instantiation for the sometimes extractable equivocal scheme).

  Further, this setting will ensure that:

  - $\mathcal{C}_{\mathsf{NIDI},DI} = \mathcal{C}_{\mathsf{NIDI},D}$.
  - $\epsilon_{\mathsf{NIDI},DI} = O(\epsilon_{\mathsf{NIDI},D} 2^{\ell_{\mathsf{NIDI}}})$.

  As shown in Theorem 9, this can be constructed assuming subexponential security of iO, a time lock puzzle scheme (Definition 4), and subexponential time and advantage security of the SXDH assumption.

- *Sometimes Extractable Equivocal Commitments*: We use three parameters λ_{com}, μ_{com}, and t_{com}, as follows.

  - We set $\mu_{\mathsf{com}} = \lambda$. This ensures that $\mathcal{C}_{\mathcal{A},\mathsf{com}}$ consists of all circuits of size polynomial in 2^{λ}. Let $\ell_{\mu}(\lambda)$ be the length of the challenges $\mathbf{b}_R$ to support this.
  - We set $t_{\mathsf{com}} = \lambda^{\rho}$. This ensures the commitments are extractable in size polynomial in $2^{t_{\mathsf{com}}}$.
  - We set $\lambda_{\mathsf{com}} = \ell_{\mathsf{NIDI}}(\lambda)\ell_{\mu}(\lambda)\lambda$. This choice ensures that $\mathcal{C}_{\mathsf{com},D}$ consists of all circuits of size polynomial in $2^{\lambda_{\mathsf{com}}}$ and depth polynomial in λ_{com}. This ensures $\epsilon_{\mathsf{com},D} = 2^{-\lambda_{\mathsf{com}}}$.

  As shown in Theorem 8, can be constructed assuming subexponential security of iO, a time lock puzzle scheme (Definition 4), and subexponential time and advantage security of the DDH over $\mathbb{Z}_p^*$.

- Σ-*protocol*: We use a statistically sound Σ protocol for NP, which is a parallel repetition of the following basic protocol. Assume that the length of the instance is a fixed polynomial in λ. We will build our zk protocol for the same length instances.

  - The first message $\Sigma_1(x, w)$ by the prover consists of non-interactive commitments of some messages $a_1, \ldots, a_N \in \{0,1\}^{N(\lambda)}$. We define $\Sigma.\mathsf{SampFirst}$ to mean the algorithm that outputs $a_1, \ldots, a_N$.
  - In the second round, the verifier outputs a bit $e \in \{0, 1\}$.
  - In the third round, the prover outputs z which consists of opening of some subset of the commitments based on the challenge bit e. Verifier accepts or rejects based on the transcript.

  The protocol satisfies several different properties. The first property is related to soundness, and the second property is related to zero-knowledge.

– Assuming we instantiate this protocol with a perfectly binding commitment, when x is unsatisfiable, then given any $a_1 \ldots, a_N$ an accepting proof of at most one out of two choices of $e \in \{0,1\}$ can exist. We call this the $\mathsf{BadChallenge}$. We assume that computing $\mathsf{BadChallenge}$ can be done by an NC^1 function Bad that takes x and $a_1, \ldots, a_N$ as the input.

– The protocol satisfies honest-verifier zero knowledge property. That is, given $e \in \{0,1\}$, for any x, one can efficiently sample $\Sigma.\mathcal{S}(e,x) \to z' = \mathsf{Set}, \{a'_i\}_{i\in\mathsf{Set}}$. The protocol ensures that the distribution of $\{a'_i\}_{i\in\mathsf{Set}}, e$ is identical to the case when $a_1, \ldots, a_N$ were committed to using an honest proof and then the prover gives out $(z = \mathsf{Set}, \{a_i\}_{i\in\mathsf{Set}}, e)$.

Looking ahead, we will compile such a protocol to a zk. The commitment we will use is sometimes extractable equivocal commitments.

- *Correlation Intractability Hash Function*: We require a CI hash function

$$\mathcal{H} = (\mathsf{FakeGen}, \mathsf{Eval})$$

(see Definition 6). We set $\lambda_{ci} = \ell_{\mathsf{NIDI}} \cdot \ell_\mu \lambda$. This ensures that the hash keys corresponding to two functions are distinguishable to circuits of size polynomial in $\mathcal{C}_{ci} = 2^{\lambda_{ci}}$ with advantage at most $\epsilon_{ci} = 2^{-\lambda_{ci}}$. Finally, for this choice of parameters, there exists a polynomial $\ell_{ci}(\lambda_{ci})$ such that the security holds for functions of bounded depth (say λ_{ci}) with $\ell_{ci}(\lambda_{ci})$ output bits. We use this as the parallel repetition parameter for the Σ protocol.

This can be constructed assuming subexponential time, and advantage hardness of LWE [PS19a].

- *Distribution* $\mathcal{D}_{\mathbf{b}_R}$: For $\mathbf{b}_R \in \{0,1\}^{\ell_\mu}$, we define the distribution $\mathcal{D}_{\mathbf{b}_R}$ as follows.

– Sample $\mathsf{com}_{1,R} \leftarrow \mathsf{Com}_{1,R}(\mathbf{b}_R; \mathbf{r})$.

– Sample $K \leftarrow \mathcal{H}.\mathsf{FakeGen}(f[\mathbf{b}_R, \mathbf{r}])$, where $f : \{0,1\}^* \to \{0,1\}^{\ell_{ci}}$ is a function described below.

Observe that for any two $\mathbf{b}_1$ and $\mathbf{b}_2$ in $\{0,1\}^{\ell_\mu}$, $\mathcal{D}_{\mathbf{b}_1}$ and $\mathcal{D}_{\mathbf{b}_2}$ are $O(\ell_\mu \cdot (2^{-\lambda_{ci}} + 2^{-\lambda_{\mathsf{com}}})) = O(2^{-\ell_\mu \cdot \ell_{\mathsf{NIDI}} \cdot \lambda})$ indistinguishable to circuits of depth polynomial in λ but size $2^{\lambda_{\mathsf{com}}}$. Let L_{NIDI} denote the language supporting these distributions $\mathcal{D}_{\mathbf{b}}$ for all $\mathbf{b}$.

- *Function* $f : \{0,1\}^* \to \{0,1\}^{\ell_{ci}}$: takes as input $(x, \mathsf{com}_{1,C}, \mathsf{com}_{1,R}, \mathsf{com}_{2,C})$, where $\mathsf{com}_{2,C} = (\mathsf{com}_{2,C,1}, \ldots, \mathsf{com}_{2,C,N \cdot \ell_{ci}})$.
  - It partitions $\mathsf{com}_{2,C}$ into ℓ_{ci} chunks. Each chunk is $(\mathsf{com}_{2,C,j \cdot N+1}, \ldots, \mathsf{com}_{2,C,(j+1)N})$ for $j \in [0, \ell_{ci} - 1]$.
  - It decrypts each chunk using $\mathsf{Com.Dec}$ using its private state $\mathbf{b}_R, \mathbf{r}$. Let us say that each chunk decrypts to $a_{jN+1}, \ldots a_{(j+1)N}$. If any of the decyption fails, we set it to be the 0 string of required length.
  - Then it computes $\mathsf{Bad}(x, a_{jN+1}, \ldots a_{(j+1)N}) = e_{j+1}$.
  - Finally it outputs $\mathbf{e} = (e_1, \ldots, e_{\ell_{ci}})$.

We now describe our construction.

Construction:

$\underline{\mathsf{ZKProve}_1(1^\lambda)}$: Compute the following steps.

- Sample $\tau \leftarrow \{0,1\}^{\ell_{\mathsf{NIDI}}}$ and $\mathbf{b}_P \leftarrow \{0,1\}^{\ell_\mu}$.
- Compute $\mathsf{com}_{1,C} = \mathsf{Com}_{1,C}(\mathbf{b}_P)$.
- Output $\mathsf{zk}_{1,P} = (\tau, \mathsf{com}_{1,C})$.

$\underline{\mathsf{ZKVerify}_1(1^\lambda)}$: Compute the following steps.

- Sample $\mathbf{b}_V \leftarrow \{0,1\}^{\ell_\mu}$.

- Compute $\Pi \leftarrow \mathsf{NIDI.P}(\mathcal{D}_{\mathbf{b}_V})$.
- Output $\mathsf{zk}_{1,V} = \Pi$.

$\underline{\mathsf{ZKProve}_2(\mathsf{zk}_{1,V}, \mathsf{zk}_{1,P}, x, w)}$: Compute the following steps.

- Parse $\mathsf{zk}_{1,V} = \Pi$ and $\mathsf{zk}_{1,P} = (\tau, \mathsf{com}_{1,C})$.
- Run $(\mathsf{com}_{1,R}, K) = \mathsf{NIDI.V}(\tau, \Pi)$. If the verification fails, output $\perp$ and stop proceeding. Otherwise, follow the next steps.
- Depending on x, w sample ℓ_{ci} repetitions of $\Sigma.\mathsf{SampFirst}$. Namely, for $j \in [\ell_{ci}]$, compute $(a_{(j-1)N+1}, \ldots, a_{jN}) \leftarrow \Sigma.\mathsf{SampFirst}$.
- For $k \in [N\ell_{ci}]$, compute $\mathsf{com}_{2,C,k} = \mathsf{Com}_{2,C}(\mathsf{com}_{1,R}, \mathsf{com}_{1,C}, a_k; s_k)$ for a freshly chosen s_k. Let $\mathsf{com}_{2,C} = \{\mathsf{com}_{2,C,k}\}_{k \in [N\ell_{ci}]}$.
- Run $\mathbf{e} = \mathcal{H}.\mathsf{Eval}(K, (x, \mathsf{com}_{1,C}, \mathsf{com}_{1,R}, \mathsf{com}_{2,C}))$.
- For $j \in [\ell_{ci}]$ determine Set_j, the set of commitments to be opened for j^{th} repetition, as per challenge bit e_j. Let Set be the union of these sets.
- Output $\mathsf{com}_{2,C}$ along with $\mathbf{e}$ and openings $z = \{a_k, s_k\}_{k \in \mathsf{Set}}$.

$\underline{\mathsf{ZKVerify}_2(\mathsf{zk}_{1,V}, \mathsf{zk}_{1,P}, \mathsf{zk}_{2,P}, x)}$: Compute the following steps.

- Parse $\mathsf{zk}_{1,V} = \Pi$, $\mathsf{zk}_{1,P} = (\tau, \mathsf{com}_{1,C})$ and $\mathsf{zk}_{2,P} = (\mathsf{com}_{2,C}, \mathbf{e}, z = \{a_k, s_k\}_{k \in \mathsf{Set}})$.
- Compute $(\mathsf{com}_{1,R}, K) = \mathsf{NIDI.V}(\tau, \Pi)$ and check if

$$\mathbf{e} = \mathcal{H}.\mathsf{Eval}(K, (x, \mathsf{com}_{1,C}, \mathsf{com}_{1,R}, \mathsf{com}_{2,C})).$$

- Check if $z = \{a_k, s_k\}_{k \in \mathsf{Set}}$ are valid openings of $\{\mathsf{com}_{2,C,k}\}_{k \in \mathsf{Set}}$.
- Finally verify that $\{a_k\}_{k \in \mathsf{Set}}$ as a valid third message of ℓ_{ci} parallel repetition of Σ protocol according to $\mathbf{e}$ and instance x.
- Output 1 if every verification above succeeds, else output 0.

Remark 5. *We assume that the prover always outputs a valid first message* $\mathsf{zk}_{1,P}$*. This can be ensured as follows. If the first message is either not given out, or if one of* τ *and* $\mathsf{com}_{1,C}$ *is not valid, then we interpret* $\tau = 0^{\ell_{\mathsf{NIDI}}}$ *and* $\mathsf{com}_{1,C} = \mathsf{Com}_{1,C}(0^{\ell_\mu})$*.*

We now argue various properties involved.

Completeness. Completeness is straightforward to argue and follows from perfect completeness of NIDI, perfect correctness of the SEE, and perfect completeness of the Σ protocol.

4.4.1 Soundness

We now argue soundness. We first define the "soundness mode" and then argue all three properties.

Perfect Soundness Mode. In order for a proof to verify, $\mathsf{zk}_{1,P} = (\tau, \mathsf{com}_{1,C})$ needs to be verifiable. In particular, there must exist $\mathbf{b}_P$ such that $\mathsf{com}_{1,C} = \mathsf{Com}_{1,C}(\mathbf{b}_P)$ (where $\mathbf{b}_P$ is as chosen by the prover, or 0^{ℓ_μ} if the prover aborts, or outputs a non-well formed message). In the soundness game on the other hand, the verifier is honest and chooses $\mathbf{b}_V \leftarrow \{0,1\}^{\ell_\mu}$ and sets $\Pi = \mathsf{NIDI.P}(\mathcal{D}_{\mathbf{b}_V})$. We define the perfect soundness mode to be the mode when $\mathbf{b}_P = \mathbf{b}_V$.

Lemma 13. *When* $\mathbf{b}_P = \mathbf{b}_V$*, then there does not exist an accepting proof of any* $x \notin L$*.*

Proof. When $\mathbf{b}_P = \mathbf{b}_V$, then consider any accepting proof say $(\mathsf{com}_{2,C}, \mathbf{e}, \{a_k, s_k\}_{\mathsf{Set}})$. Observe that $\mathbf{e} = \mathcal{H}.\mathsf{Eval}(K, x, \mathsf{com}_{1,C}, \mathsf{com}_{1,R}, \mathsf{com}_{2,C})$. Note that K is generated by using $\mathsf{FakeGen}$ algorithm with input the function f, which uses $\mathbf{b}_V$ and randomness $\mathbf{r}$ to decrypt all the commitments $\{\mathsf{com}_{2,C,k}\}_{k\in[\ell_{ci}N]}$. Let us say that this decryption results in $\{a'_k\}_{k\in[\ell_{ci}N]}$. Due to the correctness of the decryption/extraction of SEE, $a_k = a'_k$ for every $k \in \mathsf{Set}$ as the adversary opens it in the proof. Now, when $x \notin L$, since the extractable commitment is perfectly binding because $\mathbf{b}_P = \mathbf{b}_V$, the Σ protocol ensures that there is atmost one $\mathbf{e}_{bad}$ such that $\{a'_k\}_{k\in[\ell_{ci}N]}$ can lead to a valid proof. This $\mathbf{e}_{bad}$ is computed by the function f. The

perfect CI property of $\mathcal{H}$ ensures that $\mathbf{e} \neq \mathbf{e}_{bad}$. On the other hand, if the adversary gives a valid proof for $\mathbf{e}$, then $\mathbf{e} = \mathbf{e}_{bad}$. This is a contradiction.

We now analyze the frequency of the perfect soundness mode.

Lemma 14. *For any honest polynomial time verifier V, and a cheating prover P^* in $\mathcal{C}_{sound}$, the soundness mode holds with probability at least $\Omega(2^{-\ell_\mu})$.*

Proof. We prove this by a simple reduction to the security of distributional indistinguishability of $\mathsf{NIDI.P}(\mathcal{D}_{\mathbf{b}_V})$. We show this using a hybrid argument.

Hybrid_0 : In this hybrid, the challenger samples randomly $\mathbf{b}_V$ and outputs $\mathsf{zk}_{1,V}$ as $\mathsf{NIDI.P}(\mathcal{D}_{\mathbf{b}_V})$. Then, the prover outputs $\mathsf{zk}_{1,P} = \tau, \mathsf{com}_{1,C}$. The challenger outputs 1 if $\mathsf{com}_{1,C} = \mathsf{Com}_{1,C}(\mathbf{b}_V)$.

Hybrid_1 : In this hybrid, the challenger samples randomly $\mathbf{b}_V$. It also samples randomly $\mathbf{b}'$ and outputs $\mathsf{zk}_{1,V}$ as $\mathsf{NIDI.P}(\mathcal{D}_{\mathbf{b}'})$. Then, the prover outputs $\mathsf{zk}_{1,P} = \tau, \mathsf{com}_{1,C}$. The challenger outputs 1 if $\mathsf{com}_{1,C} = \mathsf{Com}_{1,C}(\mathbf{b}_V)$.

To prove the claim, our first observation is that soundness mode holds when Hybrid_0 outputs 1. Second, observe that the probability that Hybrid_1 outputs 1 is exactly $2^{-\ell_\mu}$. Our claim follows from the fact that for any adversary $\mathcal{A} \in \mathcal{C}_{sound}$, it holds that these two hybrids are indistinguishable with advantage bounded by $\epsilon_{\mathsf{NIDI},DI}$. This is due to the security of NIDI and the indistinguishability property of the distribution $\mathcal{D}_{\mathbf{b}'}$ for a random $\mathbf{b}'$ from $\mathcal{D}_{\mathbf{b}_V}$. Thus, the claim holds. □

We now argue indistinguishability of the soundness mode property.

Lemma 15. *The construction of zk satisfies $(\mathcal{C}_{sound}, \epsilon_{sound,2})$ indistinguishability of soundness mode property with $\epsilon_{sound,2} = O(\epsilon_{\mathsf{NIDI},DI} 2^{\ell_\mu})$.*

Proof. Let P^* be a cheating prover in $\mathcal{C}_{sound}$, and V be an honest verifier in the soundness experiment. Let E be the distribution of the transcript. Let E_1 denote the distribution of

transcript when the soundness mode holds, and E_0 denote the distribution of transcript when the soundness mode does not hold. Let $\mathcal{A}$ be any adversary in $\mathcal{C}_{sound}$. Then, we want to bound the following probability.

$$p = \left| \Pr[\mathcal{A}(e) = 1 | e \leftarrow \mathsf{E}_0] - \Pr[\mathcal{A}(e) = 1 | e \leftarrow \mathsf{E}_1] \right|$$

Every instance of e consists of $\mathsf{zk}_{1,V}$ and $\mathsf{zk}_{1,P}$ output by the cheating prover. Let S denote the set of elements in the range of $\mathsf{Com}_{1,C}(\star)$. There are exactly 2^{ℓ_μ} elements in this set. For every $s \in S$, we define $\mathsf{E}_{0,s}$ to be the collection of transcripts in E_0 where the verifier submits s as $\mathsf{com}_{1,C}$. Likewise, we define $\mathsf{E}_{1,s}$ to be the collection of transcripts in E_1 where the verifier submits s as $\mathsf{com}_{1,C}$. Thus, due to triangle inequality we have that:

$$p < \sum_{s \in S} \left| \Pr[\mathcal{A}(e) = 1 \wedge \mathsf{zk}_{1,P} = s | e \leftarrow \mathsf{E}_{0,s}] - \Pr[\mathcal{A}(e) = 1 \wedge \mathsf{zk}_{1,P} = s | e \leftarrow \mathsf{E}_{1,s}] \right|.$$

To prove the claim it suffices to show that for every $s \in S$,

$$\left| \Pr[\mathcal{A}(e) = 1 \wedge \mathsf{zk}_{1,P} = s | e \leftarrow \mathsf{E}_{0,s}] - \Pr[\mathcal{A}(e) = 1 \wedge \mathsf{zk}_{1,P} = s | e \leftarrow \mathsf{E}_{1,s}] \right| < O(\epsilon_{\mathsf{NIDI},DI}).$$

To this end, assume towards contradiction that there exist s^* such that.

$$\left| \Pr[\mathcal{A}(e) = 1 \wedge \mathsf{zk}_{1,P} = s^* | e \leftarrow \mathsf{E}_{0,s^*}] - \Pr[\mathcal{A}(e) = 1 \wedge \mathsf{zk}_{1,P} = s^* | e \leftarrow \mathsf{E}_{1,s^*}] \right| = \epsilon_1.$$

We will use this to attack the indistinguishability of NIDI with the probability ϵ_1. We will show that if this happens, then we can build a reduction using $\mathcal{A}$ that is also in $\mathcal{C}_{sound}$, that distinguishes $\mathsf{NIDI.P}(\mathcal{D}_{\mathbf{b}_0})$ from $\mathsf{NIDI.P}(\mathcal{D}_{\mathbf{b}_1})$ with an advantage ϵ_1 where $\mathbf{b}_1 = \mathsf{Com}^{-1}_{1,C}(s^*)$ and $\mathbf{b}_0$ is uniformly sampled from $\{0,1\}^{\ell_\mu} \setminus \mathsf{Com}^{-1}_{1,C}(s^*)$. The reduction works as follows.

- Obtain the challenge NIDI proof Π.
- Send Π to the prover P^*. Prover outputs τ, s. If $s = s^*$, then output $\mathcal{A}(\Pi, \tau, s)$, otherwise output $\perp$.

If Π is generated using $\mathbf{b}_0$, then the transcript is not in the soundness mode when P^* outputs s^*, whereas if Π is generated using $\mathbf{b}_1$, then the transcript is in soundness mode when P^* outputs s^*. Observe that the advantage of the reduction is exactly equal to ϵ_1. Therefore, $\epsilon_1 \leq \epsilon_{\mathsf{NIDI},DI}$ as per the parameters set, which is a contradiction. $\square$

4.4.2 Zero-Knowledge

We now argue the zero-knowledge properties of the protocol. We begin by describing our simulator, $\mathsf{zk}.\mathcal{S}$ and then argue why the security holds. The simulator will run in time polynomial in 2^{λ^ρ}.

$\underline{\mathsf{zk}.\mathcal{S}(\mathsf{zk}_{1,V}, \mathsf{zk}_{1,P}, x)}$: Compute the following steps.

- Parse $\mathsf{zk}_{1,V} = \Pi$ and $\mathsf{zk}_{1,P} = (\tau, \mathsf{com}_{1,C})$.
- Run $(\mathsf{com}_{1,R}, K) \leftarrow \mathsf{NIDI.V}(\tau, \Pi)$. If the verification fails, output $\perp$. Else, continue.
- Compute $\mathsf{com}_{2,C}$ by setting $\mathsf{com}_{2,C,k} = \mathsf{Com}_{2,C}(\mathsf{com}_{1,R}, \mathsf{com}_{1,C}, 0; r'_k)$ for $k \in [N \cdot \ell_{ci}]$. Then set $\mathsf{com}_{2,C} = \{\mathsf{com}_{2,C,k}\}$.
- Run $\mathbf{e} \leftarrow \mathcal{H}.\mathsf{Eval}(K, (x, \mathsf{com}_{1,C}, \mathsf{com}_{1,R}, \mathsf{com}_{2,C}))$.
- Run the simulator of Σ protocol on input x and $\mathbf{e}$, and receive $\{a_k\}_{k \in \mathsf{Set}}$.
- Equivocate $\mathsf{com}_{2,C,k}$ for $k \in \mathsf{Set}$. That is, compute $\mathsf{SEE}.\mathcal{S}(\mathsf{com}_{1,R}, \mathsf{com}_{1,C}, r'_k, a_k) \rightarrow s_k$ for $k \in \mathsf{Set}$. Output $\perp$ if the equivocation fails.
- Set $z = \{a_k, s_k\}_{k \in \mathsf{Set}}$ and output $\mathsf{zk}_{2,P} = (\mathsf{com}_{2,C}, \mathbf{e}, z)$.

We now argue why the security property holds. Our first observation is that there is a simple criteria when the distribution $\mathsf{zk}.\mathcal{S}(\mathsf{zk}_{1,V}, \mathsf{zk}_{1,P}, x)$ is identical to $\mathsf{ZKProve}_2(\mathsf{zk}_{1,V}, \mathsf{zk}_{1,P}, x, w)$. In the zero-knowledge game $\mathsf{zk}_{1,P}$ is generated honestly containing $(\tau, \mathsf{com}_{1,C} = \mathsf{Com}_{1,C}(\mathbf{b}_P))$ for a randomly chosen $\mathbf{b}_P$. Now consider $\mathsf{zk}_{1,V}$ consisting of a NIDI proof Π. Let $(\mathsf{com}_{1,R}, K) \leftarrow$

$\mathsf{NIDI.V}(\tau, \Pi)$. Since the cheating verifier is of depth $\mathsf{poly}(\lambda)$ and size polynomial in 2^λ and NIDI is sound against such adversaries, it holds that one of the scenarios must happen:

- Either $\mathsf{NIDI.V}$ outputs $\perp$,
- or, if $\mathsf{NIDI.V}$ outputs $\mathsf{com}_{1,R}, K$, it must happen that $\mathsf{com}_{1,R} = \mathsf{Com}_{1,R}(\mathbf{b}_V)$ for some $\mathbf{b}_V$, or else the verifier violates soundness which is computationally hard.

We will show first that when $\mathsf{Com}_{1,C}(\mathbf{b}_V) \neq \mathsf{Com}_{1,C}(\mathbf{b}_P)$ of if V outputs $\perp$, then the simulator above produces an identical distribution to the honest proving algorithm. When, this does not happen, the verifier must either:

- Break soundness of NIDI, or,
- Force $\mathsf{zk}_{1,P} = \mathsf{Com}_{1,C}(\mathbf{b}_V)$ which is hard due to the security of SEE.

This will finish the analysis.

Lemma 16. *Let (x, w) be a valid instance-witness pair. Let $\mathsf{zk}_{1,P} = (\tau, \mathsf{com}_{1,C} = \mathsf{Com}_{1,C}(\mathbf{b}_P))$. Let $\mathsf{zk}_{1,V} = \Pi$ be such that either:*

- $\mathsf{NIDI.V}(\tau, \Pi) = \perp$, *or,*
- $\mathsf{NIDI.V}(\tau, \Pi) = \mathsf{com}_{1,R}, K$, *where $\mathsf{com}_{1,R} = \mathsf{Com}_{1,R}(\mathbf{b}_V)$. for $\mathbf{b}_V \neq \mathbf{b}_P$.*

Then, $(\mathsf{zk}_{1,V}, \mathsf{zk}_{1,P}, \mathsf{zk}_{2,P} = \mathsf{zk}.\mathcal{S}(\mathsf{zk}_{1,V}, \mathsf{zk}_{1,V}, x))$ is identically distributed to

$$(\mathsf{zk}_{1,V}, \mathsf{zk}_{1,P}, \mathsf{zk}_{2,P}) = \mathsf{ZKProve}_2(\mathsf{zk}_{1,V}, \mathsf{zk}_{1,V}, x, w)$$

where the randomness is only over the generation of proof $\mathsf{zk}_{2,P}$.

The proof of this is immediate. If $\mathsf{NIDI.V}(\tau, \Pi) = \perp$ then, both algorithms output $\perp$, which is identically distributed. In the case of the second criteria, the proof is also immediate. It follows from the equivocation property of the commitment scheme and honest verifier zero-knowledge of SEE. We show it by three hybrids where the first hybrid corresponds to

the actual proof, and the last hybrid corresponds to the simulator.

Hybrid_0: In this hybrid, run as in the honest algorithm to compute $\mathsf{zk}_{2,P}$: sample $\{a_k\}_{k\in[N\ell_{ci}]}$ as in the Σ protocol and then commit them to compute $\mathsf{com}_{2,C}$. Apply $\mathcal{H}$ on $\mathsf{com}_{2,C}$ to derive $\mathbf{e}$, and then open the commitments to $\{a_k\}_{k\in\mathsf{Set}}$ honestly.

Hybrid_1: In this hybrid, we make the following change to generate $\mathsf{zk}_{2,P}$: we sample $\{a_k\}_{k\in[N\ell_{ci}]}$ as in the Σ protocol but then commit $0's$ instead of $\{a_k\}$ to compute $\mathsf{com}_{2,C}$. Then, we apply $\mathcal{H}$ on $\mathsf{com}_{2,C}$ to derive $\mathbf{e}$. At the opening time, we open these commitments by using $\mathsf{SEE}.\mathcal{S}$ to equivocate these commitments to open to $\{a_k\}_{k\in\mathsf{Set}}$.

Note that since $\mathsf{com}_{1,C} \neq \mathsf{com}_{1,R}$, the distribution of these two hybrids are identical due to equivocation property of SEE.

Hybrid_2: In this hybrid, we make the following change to generate $\mathsf{zk}_{2,P}$: we generate $\mathsf{com}_{2,C}$ by committing to $0's$. Then, we apply $\mathcal{H}$ on $\mathsf{com}_{2,C}$ to derive $\mathbf{e}$. At the opening time, we first sample $\{a_k\}_{k\in\mathsf{Set}}$ using the honest verifier simulator of Σ protocol, and then open the commitments of 0 by using $\mathsf{SEE}.\mathcal{S}$ to $\{a_k\}_{k\in\mathsf{Set}}$.

This hybrid corresponds to $\mathsf{zk}.\mathcal{S}$. Note that due to the security of the Σ protocol, the last two hybrids are identical. □

The lemma above solves our problems completely, except that we must ensure that the conditions for when the distributions are not identical outlined above do not happen in the zero-knowledge security game.

We show this using a hybrid argument. The first hybrid corresponds to the case of the honest experiment. The last hybrid corresponds to the simulated experiment.

Hybrid_0 : This hybrid corresponds to the experiment where the responses are made using the honest $\mathsf{ZKProve}_2$ algorithm. Throughout, parse $\mathsf{zk}_{1,P} = (\tau, \mathsf{com}_{1,C})$.

Hybrid_1 : This hybrid is the same as before, except that we abort if the cheating verifier queries $(x_i, w_i, \mathsf{zk}_{1,V,i} = \Pi_i)$ such that $\mathsf{NIDI.V}(\tau, \Pi_i) = (\mathsf{com}_{1,R,i}, K_i)$ where $\mathsf{com}_{1,R,i} = \mathsf{Com}_{1,R}(\mathbf{b}_{V,i})$ such that $\mathsf{Com}_{1,C}(\mathbf{b}_{V,i}) = \mathsf{com}_{1,C}$.

Note that the above two hybrids are statistically close. This is because V^* is an adversary of polynomially bounded depth and size polynomial in 2^λ. The commitment scheme SEE ensures that any adversary of polynomial depth, and size bounded by $2^{\lambda_{\mathsf{com}}} \gg 2^\lambda$ cannot produce $\mathsf{com}_{1,R}$ with this property with advantage more than $2^{-\lambda_{\mathsf{com}}} \ll 2^{-\lambda}$. Thus, probability of abort is less than $2^{-\lambda_{\mathsf{com}}}$. We also make a note that the challenger for this hybrid can be run in time polynomial in $2^{t_{\mathsf{com}}} = 2^{\lambda^\rho}$. This is to break open $\mathsf{com}_{1,R}$.

Hybrid_2 : This hybrid is the same as before, except that we abort if the cheating verifier queries $(x_i, w_i, \mathsf{zk}_{1,V,i} = \Pi_i)$ such that $\mathsf{NIDI.V}(\tau, \Pi_i) = (\mathsf{com}_{1,R,i}, K_i)$ where $\mathsf{com}_{1,R,i} \neq \mathsf{Com}_{1,R}(\mathbf{b}_{V,i}; \mathbf{r}_i)$.

Note that the above two hybrids are statistically close. This is because if there is a cheating verifier V^* of depth $\mathsf{poly}(\lambda)$ and size polynomial in 2^λ that produces distinguishable hybrids, then we can build a reduction of size polynomial in 2^λ that violates soundness of NIDI. The reduction responds to the queries as in Hybrid_1. It also needs to run to respond to the queries to determine aborting conditions as in Hybrid_1. It can do so, by brute-force opening of $\mathsf{com}_{1,R,i}$, which can be done by a circuit of size $2^{t_{\mathsf{com}}=\lambda^\rho}$. Since NIDI is sound against adversaries of size polynomial $2^{\lambda_{\mathsf{NIDI}}} \gg 2^\lambda$ with advantage less than $2^{-\lambda}$, these two hybrids are statistically close (unless the reduction wins in the soundness game).

Hybrid_3 : This hybrid is the same as before, except that we simulate $\mathsf{zk}_{2,P}$ responses.

These hybrids are identical due to Lemma 16.

CHAPTER 5

Malicious-Secure MrNISC

In this chapter, we prove Theorem 1. We start by giving a formal definition of MrNISC.

5.1 MrNISC Syntax and Security

We define the syntax of MrNISC and formalize security notions for malicious adversaries as well as semi-malicious adversaries, following the general framework given by Benhamouda and Lin [BL20].

We assume all parties have access to a broadcast channel, which any party can transmit a message to all other parties. We consider protocols given in the form of three polynomial-time algorithms $(\mathsf{Encode}, \mathsf{Eval}, \mathsf{Output})$, where Encode and Eval are probabilistic, and Output is deterministic, for which we define the syntax as follows:

- **Input Encoding phase**: each party P_i computes $m_{i,1} \leftarrow \mathsf{Encode}(1^\lambda, x_i; r_{i,1})$, where x_i is P_i's private input, and the output $m_{i,1}$ is P_i's round 1 message.
- **Function Evaluation phase**: any set of parties I can compute an arity-$|I|$ function f on their respective inputs as follows. Each party P_i for $i \in I$ computes $m_{i,2} \leftarrow \mathsf{Eval}(f, x_i, r_{i,1}, I, \{m_{i,1}\}_{i\in I}; r_{i,2})$, where f is the function to compute, x_i is P_i's private input, $r_{i,1}$ is the randomness which P_i used to generate its input encoding, $\{m_{i,1}\}_{i\in I}$ are the input encodings of all parties in I, and the output $m_{i,2}$ is P_i's round 2 message.
- **Output phase**: Anyone can compute $y \leftarrow \mathsf{Output}(\{m_{i,1}, m_{i,2}\}_{i\in I})$.

Malicious security. We follow the standard real/ideal paradigm in the following definition. An MrNISC scheme is malicious-secure for every PPT adversary $\mathcal{A}$ in the real world there exists an ideal-world adversary $\mathcal{S}$ (the "simulator") such that the outputs of the following two experiments $\mathsf{Expt}^{\mathsf{Real}}_{\mathcal{A}}(\lambda)$ and $\mathsf{Expt}^{\mathsf{Ideal}}_{\mathcal{A},\mathcal{S}}(\lambda)$ are indistinguishable.

In the following, for ease of exposition, we assume that each party sends at most one computation encoding for any (f, I) pair, and that parties ignore any subsequent computation encodings.

Real experiment $\mathsf{Expt}^{\mathsf{Real}}_{\mathcal{A}}(\lambda, z)$. The experiment initializes the adversary $\mathcal{A}$ with security parameter 1^λ and auxiliary input z. In addition, the experiment initializes an empty list $\mathsf{honest\_outputs}$. $\mathcal{A}$ chooses the number of parties M and the set of honest parties $H \subseteq [M]$. $\mathcal{A}$ then submits queries to the experiment in an arbitrary number of iterations until it terminates. In every iteration k, it can submit one query of one of the following four types.

- Corrupt Input Encoding: The adversary $\mathcal{A}$ can corrupt a party $i \notin H$ and send an arbitrary first message $m^*_{i,1}$ on its behalf.
- Honest Input Encoding: The adversary $\mathcal{A}$ can choose an input x_i for honest party i and ask a party $i \in H$ to send its first message by running $m^*_{i,1} \leftarrow \mathsf{Encode}(1^\lambda, x_i; r_{i,1})$, where $r_{i,1}$ is freshly chosen randomness. This $m^*_{i,1}$ is sent to the adversary.
- Honest Computation Encoding: The adversary $\mathcal{A}$ can ask an honest party $i \in H$ to evaluate a function f on the inputs of parties I. If all first messages of parties in I are already published, party i computes and publishes $m^*_{i,2} \leftarrow \mathsf{Eval}(f, x_i, I, r_{i,1}, \{m^*_{i,1}\}_{i\in I}; r_{i,2})$. Otherwise, the party instead publishes $\perp$.
- Corrupt Computation Encoding: The adversary can send an arbitrary function evaluation encoding $m^*_{i,2}$ to the honest parties on behalf of some corrupted party $i \notin H$ with respect to some function f and set I. If all parties in I have sent their Eval messages

for (f, I), the experiment adds the honest parties' output $(f, I, \mathsf{Output}(\{m^*_{i,1}, m^*_{i,2}\}_{i \in I}))$ to the list $\mathsf{honest\_outputs}$.

The output of the real experiment is defined to be $(\mathsf{view}_{\mathcal{A}}, \tau, \mathsf{honest\_outputs})$, where $\mathsf{view}_{\mathcal{A}}$ is the output of $\mathcal{A}$ at the end of the computation, i.e. an arbitrary function of its view, τ is the transcript of queries sent by $\mathcal{A}$ along with the experiment's responses, and $\mathsf{honest\_outputs}$ is the list defined above.

Ideal experiment $\mathsf{Expt}^{\mathsf{Ideal}}_{\mathcal{A},\mathcal{S}}(\lambda, z)$. The ideal experiment initializes $\mathcal{A}$ with security parameter 1^λ and auxiliary input z. After $\mathcal{A}$ chooses the number of parties M and the set $H \subsetneq [M]$, the experiment initializes $\mathcal{S}$ with 1^λ, M, and H. In addition, the experiment initializes an empty list $\mathsf{honest\_outputs}$. Subsequently, the adversary can make the same queries as in the real world, which are handled as follows:

- Corrupt Input Encoding: When $\mathcal{A}$ sends a first message $m^*_{i,1}$ on behalf of some party $i \notin H$, the experiment forwards this encoding to $\mathcal{S}$, who responds with an extracted input x_i. $\mathcal{S}$ also has the option to declare that P_i's input is $\perp$, which means that $\mathcal{S}$ was not able to extract an input from $m^*_{i,1}$ (for example, if the adversary sends a bogus string as its message). The experiment then sends x_i (if it is not $\perp$) to the ideal functionality to be used as the input for party i.

- Honest Input Encoding: When the adversary $\mathcal{A}$ chooses honest input x_i and asks party $i \in H$ to send its first message, the experiment sends x_i to the ideal functionality to be used as the input for party i. The experiment then sends the index i (but not x_i) to the simulator $\mathcal{S}$, who generates a simulated honest input encoding $\tilde{m}_{i,1}$. This encoding is forwarded back to $\mathcal{A}$.

- Honest Computation Encoding: When the adversary $\mathcal{A}$ asks an honest party $i \in H$ for a function evaluation encoding with respect to function f and parties I,

assuming all parties in I have published input encodings, the experiment forwards this request to $\mathcal{S}$. If this is the last honest computation encoding generated with respect to f and I, and all corrupted parties in $j \in I \setminus H$ have sent first messages $m^*_{j,1}$ from which non-$\perp$ inputs have been extracted, then the experiment queries the ideal functionality on (f, I) to obtain the output y, which it forwards to the simulator as well. The simulator must then generate a simulated function evaluation encoding $\tilde{m}_{i,2}$ on behalf of party i, regardless of whether it receives y or not. This encoding is forwarded to $\mathcal{A}$.

- Corrupt Computation Encoding: When the adversary sends a function evaluation encoding $m^*_{i,2}$ on behalf of some corrupted party corresponding to some (f, I), the experiment forwards $(f, I, i, m^*_{i,2})$ to the simulator. If all parties have sent computation encodings, the simulator chooses whether to allow the honest parties to learn the output corresponding to (f, I). If so, the experiment adds (f, I, y) to the list $\mathsf{honest\_outputs}$; otherwise, the experiment adds $(f, I, \perp)$ to $\mathsf{honest\_outputs}$.

The output of the ideal experiment is defined to be $(\widehat{\mathsf{view}}, \tau, \mathsf{honest\_outputs})$, where $\widehat{\mathsf{view}}$ is the output of $\mathcal{A}$ at the end of the experiment, τ is the transcript of queries made by $\mathcal{A}$ along with the experiment's responses, and $\mathsf{honest\_outputs}$ is the list defined above. In addition, at any point in the experiment, $\mathcal{S}$ may choose to abort; in this case, the output of the experiment is whatever $\mathcal{S}$ outputs at that point.

Definition 29 ($(\mathcal{C}_{\mathsf{adv}}, \mathcal{C}_{\mathsf{sim}}, \epsilon)$-Maliciously Secure MPC)**.** *We say that an MPC protocol* Π *is* $(\mathcal{C}_{\mathsf{adv}}, \mathcal{C}_{\mathsf{sim}}, \epsilon)$-maliciously secure *if for every* $\mathcal{C}_{\mathsf{adv}}$ *adversary* $(\mathcal{A}, \mathcal{D})$ *there exists a* $\mathcal{C}_{\mathsf{sim}}$ *ideal-world adversary* $\mathcal{S}$ *(i.e., the simulator) such that for every string* z*,*

$$\left| \Pr\left[\mathcal{D}(\mathsf{Expt}^{\mathsf{Real}}_{\mathcal{A}}(\lambda, z)) = 1\right] - \Pr\left[\mathcal{D}(\mathsf{Expt}^{\mathsf{Ideal}}_{\mathcal{A},\mathcal{S}}(\lambda, z)) = 1\right] \right| < \epsilon(\lambda).$$

The standard notion of security requires for every polynomial $p(\cdot)$ the existence of a polynomial $q(\cdot)$ for which the protocol is (p, q, ϵ)-maliciously secure, where $\epsilon(\cdot)$ is a negligible

function. However, since we are interested in two-round MPC protocols, it is known that the standard polynomial notion of security is impossible. Therefore, we focus on the relaxed notion of super-polynomial security (SPS): there is a sub-exponential function $q(\cdot)$ such that for all polynomials $p(\cdot)$, the protocol is (p, q, ϵ)-maliciously secure.

The semi-malicious case. We define a variant of the above security definition, which closely mirrors the definition of semi-malicious secure multiparty computation [AJW11]. A *semi-malicious MrNISC adversary* is modeled as an algorithm which, whenever it sends a corrupted input or computation encoding on behalf of some party P_j, must also output some pair (x, r) which *explains its behavior*. More specifically, all of the protocol messages sent by the adversary on behalf of P_j up to that point, including the message just sent, must exactly match the honest protocol specification for P_j when executed with input x and randomness r. Note that the witnesses given in different rounds need not be consistent. We also allow the adversary to "abort" a function evaluation in two different scenarios. First, instead of sending a CORRUPT INPUT ENCODING message for P_j, the adversary can send $(j, \perp)$ to the experiment. In this case, the experiment will respond with $\perp$ for all HONEST COMPUTATION ENCODING requests for (f, I), and when all parties in I have been queried, it will add $(f, I, \perp)$ to $\mathsf{honest\_outputs}$. Second, instead of sending a CORRUPT COMPUTATION ENCODING message on behalf of P_j the adversary can again send $(j, f, I, \perp)$. Again, after receiving such a query, the experiment will respond with $\perp$ for all HONEST COMPUTATION ENCODING requests for (f, I), and when all parties in I have been queried, it will add $(f, I, \perp)$ to $\mathsf{honest\_outputs}$.

I have published computation encodings for (f, I). In this sense, the adversary may abort any individual function evaluation. Whenever an adversary aborts a CORRUPT INPUT ENCODING message on behalf of party P_j, it must abort any subsequent CORRUPT COMPUTATION ENCODING messages for P_j.

Definition 30 ($(\mathcal{C}_{\mathsf{adv}}, \mathcal{C}_{\mathsf{sim}}, \epsilon)$-Semi-Malicious Secure MPC)**.** *We say that an MPC protocol* Π

is $(\mathcal{C}_{\mathsf{adv}}, \mathcal{C}_{\mathsf{sim}}, \epsilon)$-semi-malicious secure *if for every* $\mathcal{C}_{\mathsf{adv}}$ *semi-malicious adversary* $(\mathcal{A}, \mathcal{D})$ *there exists* $\mathcal{C}_{\mathsf{sim}}$ *ideal-world adversary* $\mathcal{S}$ *(i.e., the simulator) such that for every string* z*,*

$$\left| \Pr\left[\mathcal{D}(\mathsf{Expt}^{\mathsf{Real}}_{\mathcal{A}}(\lambda, z)) = 1\right] - \Pr\left[\mathcal{D}(\mathsf{Expt}^{\mathsf{Ideal}}_{\mathcal{A},\mathcal{S}}(\lambda, z)) = 1\right] \right| < \epsilon(\lambda).$$

5.2 The Construction

Required Primitives and Parameters. We make use of the following primitives in our construction.

- *Commitment:* A non-interactive perfectly binding commitment ($\mathsf{NICommit}$).
- *Pseudo-Random Function* A pseudo-random function (PRF).
- *Witness Encryption:* We use witness encryption as in Definition 2. We use circuit SAT as our NP language.
- *Reusable Statistical ZK Arguments with Sometimes-Statistical Soundness:* We use the SPS ZK argument $(\mathsf{ZKProve}_1, \mathsf{ZKVerify}_1, \mathsf{ZKProve}_2, \mathsf{ZKVerify}_2)$ satisfying Definitions 17, 19 and 20 for circuit SAT constructed in Section 4.4.
- *One-round CCA commitments:* We use one-round (simultaneous-message) CCA commitments as in Definitions 13 to 16.
- *Semi-malicious MrNISC*: We use an underlying semi-malicious MrNISC protocol $(\mathsf{SM.Encode}, \mathsf{SM.Eval}, \mathsf{SM.Output})$, satisfying the security notion given in Definition 30.

Complexity hierarchy. In order to argue security, we require that the primitives we use are secure against adversaries of varying complexities. In particular, we require the following

complexity hierarchy to hold with respect to the primitives. Let T_1, T_2, T_3, T_4, T_5 be functions over λ, such that

$$T_1 \ll T_2 \ll T_3 \ll T_4 \ll T_5,$$

where $T \ll T'$ means that $p(T) < T'$ asymptotically for all polynomials p. We require the following:

- The ZK argument scheme satisfies $(\mathcal{C}_{\mathcal{S}}, \mathcal{C}_{\mathsf{zk}}, \epsilon_{\mathcal{S}})$-adaptive reusable statistical zero knowledge (Definition 20) where $\mathcal{C}_{\mathcal{S}}$ is the class of circuits of size $\mathsf{poly}(T_1)$ and depth T_1 (i.e. the simulator runs in size $\mathsf{poly}(T_1)$ and depth T_1), and $\mathcal{C}_{\mathsf{zk}}$ is the class of circuits of size $p(T_3)$ for all polynomials p, and $\epsilon_{\mathcal{S}}$ is any negligible function (i.e. statistical zero knowledge holds as long as the verifier's first-round message is generated by a machine in $\mathcal{C}_{\mathsf{zk}}$.

- The CCA non-malleable commitment scheme satisfies $(\mathcal{C}, \epsilon)$-CCA security, where $\mathcal{C}$ is the class of circuits of size $p(T_1)$ for all polynomials p, and ϵ is any negligible function.

- The CCA non-malleable commitment scheme's extractor CCAVal is a circuit of size T_2 and polynomial depth.

- The perfectly-binding commitment scheme is hiding against adversaries of size $p(T_2)$ for all polynomials p, and is extractable by a circuit of size T_3.

- The ZK argument scheme satisfies $(\mathcal{C}_{\mathsf{sound}}, \epsilon_{\mathsf{sound},1}, \epsilon_{\mathsf{sound},2})$-statistical soundness, where $\mathcal{C}_{\mathsf{sound}}$ is the class of circuits of size $p(T_5)$ and polynomial depth for all polynomials p (refer to Definition 19 for details on the meaning of $\mathcal{C}_{\mathsf{sound}}$), and $\epsilon_{\mathsf{sound},1} = 1/T_4$, and $\epsilon_{\mathsf{sound},2}$ is any negligible function.

- The witness encryption scheme satisfies $(\mathcal{C}, \epsilon)$-security, where $\mathcal{C}$ is the class of circuits of size $p(T_5)$ for all polynomials p, and $\epsilon = 1/T_5$.

- The pseudo-random function is secure against adversaries of size $p(T_5)$ for all polynomials p.

- The semi-malicious MrNISC protocol is secure against adversaries of size $p(T_5)$ for all polynomials p.

The Relation $\Phi_{\mathsf{zk},i,j}$

Hardwired: The function f and the set I, P_i's tag tag_i, P_i's CCA non-malleable commitment nmc_i, P_i's perfectly binding commitment com_i, P_i's first round semi-malicious MPC message $\hat{m}_{i,1}$, P_j's string τ_j, P_i's commitment $\mathsf{com}_{i,\hat{m}_{i,2}}$ to its semimalicious evaluation encoding $\hat{m}_{i,2}$, and the transcript $\rho_{\mathsf{sm},1}$ of the semi-malicious input encodings of all parties from I.

Input/Witness: $W_{\mathsf{zk},i} = (x_i, r_{i,\mathsf{SM},1}, K_i, r_{i,\mathsf{com}}, \sigma_{i,j,\mathsf{CCA}}, \hat{m}_{i,2})$.

Computation: Verify the following steps.

1. $\mathsf{VerifyOpening}(\tau_j, \mathsf{tag}_i, \mathsf{nmc}_i, (x_i, r_{i,\mathsf{SM},1}, K_i, r_{i,\mathsf{com}}), \sigma_{i,j,\mathsf{CCA}}) = 1$
2. $\mathsf{com}_i = \mathsf{NICommit}(1^\lambda, (x_i, r_{i,\mathsf{SM},1}, K_i); r_{i,\mathsf{com}})$
3. $\hat{m}_{i,1} = \mathsf{SM.Encode}(1^\lambda, x_i, r_{i,\mathsf{SM},1})$
4. $\hat{m}_{i,2} = \mathsf{SM.Eval}(f, x_i, r_{i,\mathsf{SM},1}, I, \rho_{\mathsf{sm},1}; PRF_{K_i}(f, I, 1))$
5. $\mathsf{com}_{i,\hat{m}_{i,2}} = \mathsf{NICommit}(1^\lambda, \hat{m}_{i,2}; PRF_{K_i}(f, I, 2))$

Output 1 if all the above checks succeed, otherwise output 0.

Figure 5.1: The Circuit $\Phi_{\mathsf{zk},i,j}$

Protocol. We give the protocol below, described in terms of the behavior of party P_i during the input encoding phase, the evaluation phase, and the output computation phase. In particular, we give this behavior by implementing the Encode, Eval and Output algorithms

The Relation $\Phi_{\mathsf{WE},i}$

Hardwired: The function f, the set I, the set of tags of all parties, P_i's first-round verifier zk messsage $\mathsf{zk}_{1,i,V}$, P_i's string τ_i, the first-round prover zk messages, commitments and semi-malicious encodings $\{\mathsf{zk}_{1,j,P}, \hat{m}_{j,1}, \mathsf{com}_j, \mathsf{nmc}_j\}_{j \in I \setminus \{i\}}$ included in the input encodings of all other parties in I.

Witness:

$$W_{\mathsf{WE},i} = (\{\mathsf{zk}_{2,j \to i,P}, \mathsf{com}_{j,\hat{m}_{j,2}}\}_{j \neq i}).$$

Computation: For every $j \in I \setminus \{i\}$,

1. Let

$$\Phi_{\mathsf{zk},j} = \Phi_{\mathsf{zk},j}[f, I, \mathsf{tag}_j, \mathsf{nmc}_j, \mathsf{com}_j, \hat{m}_{j,1}, \tau_i, \mathsf{com}_{j,\hat{m}_{j,2}}, \rho_{\mathsf{sm},1}]$$

   be the circuit described in Figure 5.1, with the values

$$[f, I, \mathsf{tag}_j, \mathsf{nmc}_j, \mathsf{com}_j, \hat{m}_{j,1}, \tau_i, \mathsf{com}_{j,\hat{m}_{j,2}}, \rho_{\mathsf{sm},1}]$$

   hardcoded.

2. Compute $\mathsf{ZKVerify}_2(\Phi_{\mathsf{zk},j}, \mathsf{zk}_{1,i,V}, \mathsf{zk}_{1,j,P}, \mathsf{zk}_{2,j \to i,P})$.

Output 1 if all the above checks succeed, otherwise output 0.

Figure 5.2: The Relation $\Phi_{\mathsf{WE},i}$

defined in Section 5.1. Assume that each party P_i has input x_i and a public identity denoted by $\mathsf{tag}_i \in \mathcal{T}_\lambda$. Note that the Output algorithm is public and can be performed without P_i's private input or state. Throughout the protocol description, we deal with PPT algorithms as follows. If a PPT algorithm P is invoked on some input x without any randomness explicitly given (i.e., we write $P(x)$), we implicitly assume that it is supplied with freshly chosen randomness. In some cases we will need to explicitly manipulate the randomness of algorithms, in which case we will write $P(x; r)$.

- **Input Encoding** $\mathsf{Encode}(1^\lambda, \mathsf{tag}_i, x_i)$**:** The input encoding algorithm takes as input 1^λ, where λ is the security parameter, along with P_i's tag tag_i and private input x_i, and does the following.

  1. Compute the semi-malicious input encoding $\hat{m}_{i,1} \leftarrow \mathsf{SM.Encode}(1^\lambda, x_i; r_{i,\mathsf{SM},1})$, where $r_{i,\mathsf{SM},1} \xleftarrow{\$} \{0,1\}^*$ is freshly chosen randomness.
  2. Choose a PRF key K_i.
  3. Compute a perfectly binding commitment

     $$\mathsf{com}_i \leftarrow \mathsf{NICommit}(1^\lambda, (x_i, r_{i,\mathsf{SM},1}, K_i); r_{i,\mathsf{com}})$$

     of the input and the semi-malicious encoding randomness, where $r_{i,\mathsf{com}} \xleftarrow{\$} \{0,1\}^*$ is freshly chosen randomness.
  4. Compute a CCA-non-malleable commitment

     $$\mathsf{nmc}_i \leftarrow \mathsf{CCACommit}(1^\lambda, \mathsf{tag}_i, (x_i, r_{i,\mathsf{SM}}, K_i, r_{i,\mathsf{com}}); r_{i,\mathsf{CCA}})$$

     of the same values committed to in the perfectly binding commitment, along with the randomness used for generating the perfectly binding commitment, where $r_{i,\mathsf{CCA}} \xleftarrow{\$} \{0,1\}^*$ is freshly chosen randomness.
  5. Compute a random string $\tau_i \xleftarrow{\$} \{0,1\}^\ell$.

6. Compute the first round verifier's message and state

$$(\sigma_{\mathsf{zk},1,i,V}, \mathsf{zk}_{1,i,V}) \leftarrow \mathsf{ZKVerify}_1(1^\lambda)$$

and the first round prover message and state

$$(\sigma_{\mathsf{zk},1,i,P}, \mathsf{zk}_{1,i,P}) \leftarrow \mathsf{ZKProve}_1(1^\lambda).$$

7. Output $m_{i,1} = (\hat{m}_{i,1}, \mathsf{com}_i, \mathsf{nmc}_i, \tau_i, \mathsf{zk}_{1,i,V}, \mathsf{zk}_{1,i,P})$.

- **Function Evaluation** $\mathsf{Eval}(f, \mathsf{tag}_i, x_i, r_{i,1}, I, \rho_1)$**:** The function evaluation algorithm takes as input the function f to be evaluated, the set I of participating parties, P_i's private input x_i, the randomness $r_{i,1}$ which P_i used to generate its input encoding, and the input encoding transcript ρ_1, and does the following:

1. Parse $\rho_1 = \{\hat{m}_{k,1}, \mathsf{com}_k, \mathsf{nmc}_k, \tau_k, \mathsf{zk}_{1,k,V}, \mathsf{zk}_{1,k,P}\}_{k\in[n]}$ to obtain

$$(r_{i,\mathsf{SM},1}, r_{i,\mathsf{com}}, r_{i,\mathsf{CCA}}, \sigma_{\mathsf{zk},1,i,V}, \sigma_{\mathsf{zk},1,i,P})$$

from $r_{i,1}$.

2. Compute the semi-malicious function evaluation encoding

$$\hat{m}_{i,2} \leftarrow \mathsf{SM.Eval}(f, x_i, r_{i,\mathsf{SM},1}, I, \rho_{\mathsf{sm},1}; PRF_{K_i}(f, I, 1))$$

of the underlying semi-malicious protocol, using the transcript $\rho_{\mathsf{sm},1} = \{\hat{m}_{k,1}\}_{k\in I}$ of the semi-malicious input encodings of all parties from I, where the randomness is chosen using the PRF key committed to during the input encoding phase.

3. Compute a commitment $\mathsf{com}_{i,\hat{m}_{i,2}} \leftarrow \mathsf{NICommit}(\hat{m}_{i,2}; PRF_{K_i}(f, I, 2))$ of the encoding $\hat{m}_{i,2}$ using randomness derived from the PRF key committed to during the input encoding phase.

4. For each P_j, $j \in I \setminus \{i\}$:

– Compute an opening

$$\sigma_{i,j,\mathsf{CCA}} \leftarrow \mathsf{ComputeOpening}(\tau_j, \mathsf{tag}_i, \mathsf{nmc}_i, (x_i, r_{i,\mathsf{SM},1}, K_i, r_{i,\mathsf{com}}), r_{i,\mathsf{CCA}})$$

for the non-malleable-commitment nmc_i with respect to τ_j.

– Compute a round two ZK prover's message $\mathsf{zk}_{2,i\to j,P} \leftarrow \mathsf{ZKProve}_2(\Phi_{\mathsf{zk},i,j}, W_{\mathsf{zk},i}, \sigma_{\mathsf{zk},1,i,P}, \mathsf{zk}_{1,j,V})$, where $\Phi_{\mathsf{zk},i,j}$ is the circuit SAT instance defined in Figure 5.1. Here $W_{\mathsf{zk},i} = (x_i, r_{i,\mathsf{SM},1}, K_i, r_{i,\mathsf{com}}, \sigma_{i,j,\mathsf{CCA}}, \hat{m}_{i,2})$ is the witness for generating this prover message.

5. Compute a witness encryption $\mathsf{WE}_i \leftarrow \mathsf{WE.Encrypt}(1^\lambda, \Phi_{\mathsf{WE},i}, r_{\mathsf{com},i,\hat{m}_{i,2}})$ where the circuit $\Phi_{\mathsf{WE},i}$ is described in Figure 5.2, and the plaintext $r_{\mathsf{com},i,\hat{m}_{i,2}} = PRF_{K_i}(f, I, 2)$ is the opening for $\mathsf{com}_{i,\hat{m}_{i,2}}$.

6. Return $m_{i,2} = (\mathsf{com}_{i,\hat{m}_{i,2}}, \{\mathsf{zk}_{2,i\to j,P}\}_{j\in I\setminus\{i\}}, \mathsf{WE}_i)$.

- **Output Computation** $\mathsf{Output}(\{m_{j,1}, m_{j,2}\}_{j\in I})$: The output computation algorithm takes as input the input encoding $m_{j,1}$ and the function evaluation encoding $m_{j,2}$ of every party P_j for $j \in I$ and does the following:

1. Parse

$$m_{j,1} = (\hat{m}_{j,1}, \mathsf{com}_j, \mathsf{nmc}_j, \tau_j, \mathsf{zk}_{1,j,v}, \mathsf{zk}_{1,j,p})$$

and

$$m_{j,2} = (\mathsf{com}_{j,\hat{m}_{j,2}}, \{\mathsf{zk}_{2,j\to k,P}\}_{k\in I\setminus\{j\}}, \mathsf{WE}_j)$$

for each $j \in I$.

2. For each $j, k \in I, j \neq k$:

– Run $\mathsf{ZKVerify}_2(\Phi_{\mathsf{zk},j,k}, \mathsf{zk}_{1,k,v}, \mathsf{zk}_{1,j,p}, \mathsf{zk}_{2,j\to k,p})$, where $\Phi_{\mathsf{zk},j,k}$ is described in Figure 5.1. If the verification fails, abort and output $\perp$.

3. For each $j \in I$:

– Compute the decryption $r_{\mathsf{com},j,\hat{m}_{j,2}} \leftarrow \mathsf{WE.Decrypt}(\mathsf{WE}_j, W_{\mathsf{WE},j})$ of the opening $r_{\mathsf{com},j,\hat{m}_{j,2}}$ to the commitment $\mathsf{com}_{j,\hat{m}_{j,2}}$, using the witness

$$W_{\mathsf{WE},j} = (\{\mathsf{zk}_{2,k\to j,P}, \mathsf{com}_{j,\hat{m}_{j,2}}\}_{k\neq j}).$$

If the decryption fails, abort and output $\perp$.

– Open $\mathsf{com}_{j,\hat{m}_{j,2}}$ to P_j's semi-malicious function evaluation encoding $\hat{m}_{j,2}$ using $r_{\mathsf{com},j,\hat{m}_{j,2}}$.

4. Compute the output $y \leftarrow \mathsf{Output}(\{\hat{m}_{j,1}, \hat{m}_{j,2}\}_{j\in I})$ using the values $\hat{m}_{j,2}$ obtained from decrypting the witness encryptions along with the semi-malicious input encodings $\hat{m}_{j,2}$.

5. Output y.

Correctness. Correctness of the protocol follows directly from correctness of the underlying primitives.

5.3 Proof of Security

This section proves that the MrNISC protocol given above satisfies the definition of SPS malicious security from Section 5.1.

Assume that there exists a real-world PPT adversary $\mathcal{A}$ for the MrNISC security game. That is, $\mathcal{A}$ takes as input 1^λ and some auxiliary input z, chooses the number of parties M and the set of honest parties $H \subseteq [M]$, and then interacts with the experiment in an execution of the protocol by submitting queries of the four types described in Section 5.1 (i.e., Corrupt Input Encoding, Honest Input Encoding, Honest Computation Encoding, and Corrupt Computation Encoding). We prove security by exhibiting an ideal world adversary $\mathcal{S}$ (referred to as the simulator) which runs in time $T_{\mathcal{S}} = 2^{\lambda^\epsilon}$, and interacts with the experiment as described in Section 5.1, such that the outputs of the

corresponding experiments $\mathsf{Expt}^{\mathsf{Real}}_{\mathcal{A}}(\lambda)$ and $\mathsf{Expt}^{\mathsf{Ideal}}_{\mathcal{A},\mathcal{S}}(\lambda)$ are indistinguishable.

5.3.0.1 The Simulator

Upon being initialized with the number of parties M and the set $H \subsetneq [M]$, the simulator $\mathcal{S}$, initializes the semi-malicious simulator with the same M and H. It then responds to the environment's queries in the following manner:

- Corrupt Input Encoding: Upon receiving a corrupt input encoding

  $$m_{j,1} = (\hat{m}_{j,1}, \mathsf{com}_j, \mathsf{nmc}_j, \tau_j, \mathsf{zk}_{1,j,v}, \mathsf{zk}_{1,j,p})$$

  on behalf of P_j, $j \in \mathcal{C}$, the simulator $\mathcal{S}$ extracts com_j to obtain $(\tilde{x}_j, \tilde{r}_{j,\mathsf{SM},1}, \tilde{K}_j)$, and submits $\tilde{x}_j$ to the experiment to use as P_j's input if $\hat{m}_{j,1} = \mathsf{SM.Encode}(1^\lambda, \tilde{x}_j; \tilde{r}_{j,\mathsf{SM},1})$. Otherwise, it sends $\perp$.

- Honest Input Encoding: Upon receiving a query from the experiment asking for P_i's simulated input encoding, $\mathcal{S}$ does the following:

  1. Compute a perfectly binding commitment

     $$\mathsf{com}_i = \mathsf{NICommit}(1^\lambda, 0^{|x_i|+|r_{i,\mathsf{SM},1},|+|K_i|}).$$

  2. Compute a CCA-non-malleable commitment

     $$\mathsf{nmc}_i = \mathsf{CCACommit}(1^\lambda, \mathsf{tag}_i, 0^{|x_i|+|r_{i,\mathsf{SM},1},|+|K_i|+|r_{i,\mathsf{com}}|}).$$

  3. Compute a random string $\tau_i \xleftarrow{\$} \{0,1\}^\ell$.

  4. Compute the first round verifier's message and state

     $$(\sigma_{\mathsf{zk},1,i,V}, \mathsf{zk}_{1,i,V}) \leftarrow \mathsf{ZKVerify}_1(1^\lambda)$$

     and the first round prover message and state

     $$(\sigma_{\mathsf{zk},1,i,P}, \mathsf{zk}_{1,i,P}) \leftarrow \mathsf{ZKProve}_1(1^\lambda).$$

5. Ask the semi-malicious simulator to generate a semi-malicious input encoding $\hat{m}_{i,1}$ for party P_i.
6. Send $m_{i,1} = (\hat{m}_{i,1}, \mathsf{com}_i, \mathsf{nmc}_i, \tau_i, \mathsf{zk}_{1,i,v}, \mathsf{zk}_{1,i,p})$ to $\mathcal{A}$.

- Honest Computation Encoding: Upon receiving an honest computation encoding query asking for honest party P_i's encoding w.r.t f and I, the simulator does the following.

  1. Compute the extracted value $(\tilde{x}_j, \tilde{r}_{j,\mathsf{SM},1}, \tilde{K}_j, \tilde{r}_{j,\mathsf{com}}) \leftarrow \mathsf{CCAVal}(\tau_i, \mathsf{tag}_j, \mathsf{nmc}_j)$ of P_j's CCA-non-malleable commitment with respect to P_i's τ_i, for each $j \in I \cap \mathcal{C}$.
  2. For each $j \in I \cap \mathcal{C}$, check whether

  $$\hat{m}_{j,1} = \mathsf{SM.Encode}(1^\lambda, \tilde{x}_j; \tilde{r}_{j,\mathsf{SM},1})$$

  and

  $$\mathsf{com}_j = \mathsf{NICommit}(1^\lambda, (\tilde{x}_j, \tilde{r}_{j,\mathsf{SM},1}, \tilde{K}_j); \tilde{r}_{j,\mathsf{com}}),$$

  where $\hat{m}_{j,1}$ is the semi-malicious input encoding sent by P_j, com_j is the perfectly-binding commitment sent by P_j, and $\tilde{x}_j, \tilde{r}_{j,\mathsf{SM},1}, \tilde{K}_j, \tilde{r}_{j,\mathsf{com}}$ are the extracted values from before.

  - If both equalities hold for all $j \in I \cap \mathcal{C}$, then the simulator does the following.
    (a) Query the semimalicious simulator for P_i's semi-malicious computation encoding $\hat{m}_{i,2}$ with respect to (f, I). (If the experiment sent the function output y, forward this to the semi-malicious simulator to use when generating $\hat{m}_{i,2}$.)
    (b) Compute a commitment $\mathsf{com}_{i,\hat{m}_{i,2}} \leftarrow \mathsf{NICommit}(\hat{m}_{i,2}; r_{\mathsf{com},i,\hat{m}_{i,2}})$ obtained in the previous step, where $r_{\mathsf{com},i,\hat{m}_{i,2}}$ is freshly chosen randomness.
    (c) For each P_j, $j \in I \setminus \{i\}$:
    * Compute a simulated prover's second-round ZK message

    $$\mathsf{zk}_{2,i\to j,P} \leftarrow \mathsf{ZKSim}(\sigma_{\mathsf{zk},1,i,P}, \Phi_{\mathsf{zk},i,j}, \mathsf{zk}_{1,j,V}).$$

(d) Compute a witness encryption $\mathsf{WE}_i \leftarrow \mathsf{WE.Encrypt}(1^\lambda, \Phi_{\mathsf{WE},i}, r_{\mathsf{com},i,\hat{m}_{i,2}})$ where the circuit $\Phi_{\mathsf{WE},i}$ is described in Figure 5.2, and the plaintext $r_{\mathsf{com},i,\hat{m}_{i,2}}$ is the opening for $\mathsf{com}_{i,\hat{m}_{i,2}}$.

(e) Respond with $m_{i,2} = (\mathsf{com}_{i,\hat{m}_{i,2}}, \{\mathsf{zk}_{2,i\to j,P}\}_{j\in I\setminus\{i\}}, \mathsf{WE}_i)$.

– If the equalities do not hold for some $j \in I \cap \mathcal{C}$, then the simulator instead does the following:

(a) Compute a commitment $\mathsf{com}_{i,\hat{m}_{i,2}} \leftarrow \mathsf{NICommit}(0^{|\hat{m}_{i,2}|})$.

(b) For each P_j, $j \in I \setminus \{i\}$:

∗ Compute a simulated prover's second-round ZK message $\mathsf{zk}_{2,i\to j,P} \leftarrow \mathsf{ZKSim}(\sigma_{\mathsf{zk},1,i,P}, \Phi_{\mathsf{zk},i,j}, \mathsf{zk}_{1,j,V})$.

(c) Compute a witness encryption $\mathsf{WE.CT}_i \leftarrow \mathsf{WE.Encrypt}(1^\lambda, \Phi_{\mathsf{WE},i}, 0^{|r_{i,\mathsf{com}}|})$.

(d) Respond with $m_{i,2} = (\mathsf{com}_{i,\hat{m}_{i,2}}, \{\mathsf{zk}_{2,i\to j,P}\}_{j\in I\setminus\{i\}}, \mathsf{WE}_i)$.

- Corrupt Computation Encoding: On receiving a corrupt computation encoding $m_{j,2} = (\mathsf{com}_{j,\hat{m}_{j,2}}, \{\mathsf{zk}_{2,j\to i,P}\}_{i\in I\setminus\{j\}}, \mathsf{WE}_j)$ from the experiment on behalf of corrupted party P_j w.r.t. f and I, the simulator does the following:

1. Compute the extracted value $(\tilde{x}_j, \tilde{r}_{j,\mathsf{SM},1}, \tilde{K}_j, \tilde{r}_{j,\mathsf{com}}) \leftarrow \mathsf{CCAVal}(\tau_i, \mathsf{tag}_j, \mathsf{nmc}_j)$ of P_j's CCA-non-malleable commitment for each $i \in I \setminus \mathcal{C}$.
2. For each $i \in I \setminus \mathcal{C}$, check if there exists a $j \in I \cap \mathcal{C}$ such that:
   – $\mathsf{ZKVerify}_2(\phi_{\mathsf{zk},j,i}, \mathsf{zk}_{1,i,V}, \mathsf{zk}_{1,j,P}, \mathsf{zk}_{2,j\to i,P})$ verifies, and
   – Steps 2-5 of $\Phi_{\mathsf{zk},j,i}$ do not hold with respect to the extracted values $\tilde{x}_j, \tilde{r}_{j,\mathsf{SM},1}$, $\tilde{K}_j, \tilde{r}_{j,\mathsf{com}}$ and the input encoding phase of the protocol. Note that this is checkable in polynomial time given the values $\tilde{x}_j, \tilde{r}_{j,\mathsf{SM},1}, \tilde{K}_j, \tilde{r}_{j,\mathsf{com}}$.
3. If there does exist such a j, halt the experiment and output a special abort symbol $\perp^*$.

4. Otherwise, if all parties in I have submitted function evaluation encodings for f and if all parties' ZK messages have verified correctly and if all parties' WEs decrypt correctly, the simulator instructs the experiment to deliver the output y to the honest parties. If any ZK messages verify incorrectly or if any WE fails to decrypt, the simulator instructs the experiment to deliver the output $\perp$ to the honest parties.

5.3.0.2 The Hybrids

We prove the indistinguishability between the real and ideal worlds via a sequence of hybrids listed below. In each hybrid, we make changes to the behavior of the experiment, such that the first hybrid Hybrid_0 corresponds to the real world experiment, and the last hybrid Hybrid_8 corresponds to the ideal world experiment with simulator $\mathcal{S}$ described above.

- Hybrid_0**:** This hybrid performs the real-world experiment $\mathsf{Expt}^{\mathsf{Real}}_{\mathcal{A}}(\lambda)$ with $\mathcal{A}$. That is, the experiment responds to the queries of $\mathcal{A}$ as described in the real world defined in Section 5.1. At the end of the execution, the output of the hybrid is defined to be $(\mathsf{view}_{\mathcal{A}}, \tau, \mathsf{honest\_outputs})$.

- Hybrid_1**:** The behavior of this hybrid is identical to the previous hybrid, except for the following difference. Whenever $\mathcal{A}$ submits a Honest Computation Encoding query asking for honest party P_i's encoding w.r.t f and I, the experiment does the following:

  1. Compute the extracted value $(\tilde{x}_j, \tilde{r}_{j,\mathsf{SM},1}, \tilde{K}_j, \tilde{r}_{j,\mathsf{com}}) \leftarrow \mathsf{CCAVal}(\tau_i, \mathsf{tag}_j, \mathsf{nmc}_j)$ of P_j's CCA-non-malleable commitment for each $j \in I \cap \mathcal{C}$.
  2. For each $j \in I \cap \mathcal{C}$, check whether

  $$\hat{m}_{j,1} = \mathsf{SM.Encode}(1^\lambda, \tilde{x}_j; \tilde{r}_{j,\mathsf{SM},1})$$

  and

  $$\mathsf{com}_j = \mathsf{NICommit}(1^\lambda, (\tilde{x}_j, \tilde{r}_{j,\mathsf{SM},1}, \tilde{K}_j); \tilde{r}_{j,\mathsf{com}}),$$

where $\hat{m}_{j,1}$ is the semi-malicious input encoding sent by P_j, com_j is the perfectly-binding commitment sent by P_j, and $\tilde{x}_j, \tilde{r}_{j,\mathsf{SM},1}, \tilde{K}_j, \tilde{r}_{j,\mathsf{com}}$ are the extracted values from before.

  - If both equalities hold for all $j \in I \cap \mathcal{C}$, then the experiment generates $\mathsf{WE.CT}_i$ in the same way as in Hybrid_0.
  - If the equalities do not hold for some $j \in I \cap \mathcal{C}$, then the experiment instead computes $\mathsf{WE.CT}_i \leftarrow \mathsf{WE.Encrypt}(1^\lambda, \Phi_{\mathsf{WE},i}, 0^{|r_{i,\mathsf{com}}|})$.

Because of the use of CCAVal, this hybrid runs in size $O(T_2)$ and polynomial depth.

- Hybrid_2**:** This hybrid behaves identically to the previous hybrid, except for the following difference. Whenever $\mathcal{A}$ submits a Corrupt Computation Encoding

$$m_{j,2} = (\mathsf{com}_{j,\hat{m}_{j,2}}, \{\mathsf{zk}_{2,j\to i,P}\}_{i\in I\setminus\{j\}}, \mathsf{WE}_j)$$

on behalf of corrupted party P_j w.r.t. f and I, the experiment does the following:

1. Compute the extracted value $(\tilde{x}_j, \tilde{r}_{j,\mathsf{SM},1}, \tilde{K}_j, \tilde{r}_{j,\mathsf{com}}) \leftarrow \mathsf{CCAVal}(\tau_i, \mathsf{tag}_j, \mathsf{nmc}_j)$ of P_j's CCA-non-malleable commitment for each $i \in I \setminus \mathcal{C}$.
2. For each $i \in I \setminus \mathcal{C}$, check if there exists a $j \in I \cap \mathcal{C}$ such that:
   - $\mathsf{ZKVerify}_2(\phi_{\mathsf{zk},j,k}, \mathsf{zk}_{1,i,V}, \mathsf{zk}_{1,j,P}, \mathsf{zk}_{2,j\to i,P})$ verifies, and
   - Steps 2-5 of $\Phi_{\mathsf{zk},j,i}$ do not hold with respect to the extracted values $\tilde{x}_j, \tilde{r}_{j,\mathsf{SM},1}$, $\tilde{K}_j, \tilde{r}_{j,\mathsf{com}}$ and the input encoding phase of the protocol. Note that this is checkable in polynomial time given the values $\tilde{x}_j, \tilde{r}_{j,\mathsf{SM},1}, \tilde{K}_j, \tilde{r}_{j,\mathsf{com}}$.
3. If there does exist such a j, halt and output a special abort symbol $\perp^*$.

Because of the use of CCAVal, this hybrid runs in size $O(T_2)$ and polynomial depth.

- Hybrid_3**:** This hybrid behaves identically to the previous hybrid, except for the following difference. Whenever $\mathcal{A}$ submits a Honest Computation Encoding query asking

for honest party P_i's encoding w.r.t f and I, the experiment computes P_i's ZK prover's messages $\mathsf{zk}_{2,i\to j,P} \leftarrow \mathsf{ZKSim}(\Phi_{\mathsf{zk}}, \sigma_P, \mathsf{zk}_{1,j,V})$ using the zero-knowledge simulator instead of generating the message using the honest prover. This hybrid runs in size $\mathsf{poly}(T_1 + T_2) = \mathsf{poly}(T_2)$ and depth T_1 as we run the ZK Simulator and CCAVal.

- Hybrid_4**:** This hybrid behaves identically to the previous hybrid, except for the following difference. Whenever $\mathcal{A}$ submits a Honest Input Encoding query asking for honest party P_i's first message, the experiment generates $\mathsf{nmc}_i = \mathsf{CCACommit}(1^\lambda, \mathsf{tag}_i, 0^{|x_i|+|r_{i,\mathsf{SM},1},|+|K_i|+|r_{i,\mathsf{com}}|})$. This hybrid runs in the same size and depth as the previous hybrid.

- Hybrid_5**:** This hybrid behaves identically to the previous hybrid, except for the following difference. Whenever $\mathcal{A}$ submits a Honest Input Encoding query asking for honest party P_i's first message, the experiment generates $\mathsf{com}_i = \mathsf{NICommit}(1^\lambda, 0^{|x_i|+|r_{i,\mathsf{SM},1},|+|K_i|})$. This hybrid runs in the same size and depth as the previous hybrid.

- Hybrid_6**:** This hybrid behaves identically to the previous hybrid, except for the following difference. Whenever $\mathcal{A}$ submits a Honest Computation Encoding query asking for honest party P_i's encoding w.r.t f and I, the experiment uses true random strings when computing the semi-malicious function evaluation encoding and the perfectly binding commitment, instead of using PRF evaluations. In other words, the experiment computes $m_{i,2} \leftarrow \mathsf{SM.Eval}(f, x_i, r_{i,\mathsf{SM},1}, I, \rho_{\mathsf{sm},1}; r)$ and $\mathsf{com}_{i,\hat{m}_{i,2}} \leftarrow \mathsf{NICommit}(\hat{m}_{i,2}; r')$, where r and r' are freshly chosen randomness.

- Hybrid_7**:** This hybrid behaves identically to the previous hybrid, except for the following difference. Whenever $\mathcal{A}$ submits a Honest Computation Encoding query asking for honest party P_i's encoding w.r.t f and I, the experiment computes $\mathsf{com}_{i,\hat{m}_{i,2}} \leftarrow \mathsf{NICommit}(0^{|\hat{m}_{i,2}|})$ whenever the equalities checked in the steps for Hybrid_1 do not hold. This hybrid runs in the same size and depth as the previous hybrid.

- Hybrid_8**:** This hybrid behaves identically to the previous hybrid, except for the following differences. During the beginning of the protocol, the experiment initializes the semi-malicious simulator with M and H. It then responds to the adversary's queries in the following manner.

  - Whenever $\mathcal{A}$ submits an Honest Input Encoding query asking for honest party P_i's input encoding with respect to some input x_i, the experiment forwards x_i to the ideal functionality as P_i's input, and then queries the semi-malicious simulator for a simulated input encoding $\hat{m}_{i,1}$, which it uses when constructing the message $m_{i,1} = (\hat{m}_{i,1}, \mathsf{com}_i, \mathsf{nmc}_i, \tau_i, \mathsf{zk}_{1,i,v}, \mathsf{zk}_{1,i,p})$ to send to $\mathcal{A}$.
  - Whenever $\mathcal{A}$ submits a Corrupt Input Encoding query on behalf of P_j, $j \in \mathcal{C}$, the experiment extracts com_j to obtain $(\tilde{x}_j, \tilde{r}_{j,\mathsf{SM},1}, \tilde{K}_j)$. If P_j's $\hat{m}_{j,1}$ is honestly generated, the experiment submits $(j, \tilde{x}_j)$ to the ideal functionality. Otherwise it submits $(j, \perp)$.
  - Whenever $\mathcal{A}$ submits an Honest Computation Encoding query asking for honest party P_i's encoding w.r.t f and I, if the equalities checked in Hybrid_1 hold, do the following:
    - If $\mathcal{A}$ has already received honest computation encodings with respect to (f, I) for all other honest parties in I, and all corrupted parties in I have non-$\perp$ inputs, the experiment sends (f, I) to the ideal functionality, and receives back the output y. It sends (f, I, i, y) to the semi-malicious simulator, which replies with the semi-malicious computation encoding $\hat{m}_{i,2}$ for P_i.
    - If $\mathcal{A}$ has not already received all other honest computation encodings, or if some corrupted parties in I have $\perp$ as their extracted input, the experiment does not query the ideal functionality and sends (f, I, i) to the simulator, which replies with the semi-malicious computation encoding $\hat{m}_{i,2}$ for P_i.

The experiment then uses this $\hat{m}_{i,2}$ when constructing the message

$$m_{i,2} = (\mathsf{com}_{i,\hat{m}_{i,2}}, \{\mathsf{zk}_{2,i\to j,P}\}_{j\in I\setminus\{i\}}, \mathsf{WE}_i)$$

to send to $\mathcal{A}$. Note that if the equalities checked in Hybrid_1 do not hold, the experiment does not need to have a $\hat{m}_{i,2}$ message from P_i to respond to $\mathcal{A}$, since $\mathsf{com}_{i,\hat{m}_{i,2}}$ and WE_i are a commitment and WE of 0, respectively.

– Whenever $\mathcal{A}$ submits a Corrupt Computation Encoding on behalf of corrupted party P_j w.r.t. f and I, if all parties in I have submitted function evaluation encodings for f, and if all parties' ZK messages have verified correctly, their WEs have decrypted correctly, and the special abort condition has not occurred, the experiment instructs the ideal function to deliver the output y to the honest parties. If any ZK messages verify incorrectly or if any WE fails to decrypt, the experiment instructs the ideal functionality to deliver the output $\perp$ to the honest parties.

This hybrid is identical to the behavior of the ideal-world experiment. Here the simulator runs in size $\mathsf{poly}(T_3)$ and depth T_1.

We now describe indistinguishability between each pair of hybrids. The indistinguishability between Hybrid_0 and Hybrid_1 follows from the soundness properties of the SPS ZK protocol and the security of the Witness Encryption scheme. Because proving this indistinguishability is the most involved, we dedicate a separate section to the proof.

5.3.0.3 Indistinguishability Between Hybrid_0 and Hybrid_1

Claim 4. *Assuming:*

- $(\mathcal{C}_{\mathsf{WE}}, \epsilon)$*-security for the witness encryption scheme, where* $\mathcal{C}_{\mathsf{WE}}$ *is the class of circuits of size* $p(T_5)$ *for all polynomials* p *and* $\epsilon = 1/T_5$*,*

- *The zero knowledge protocol is* $(\mathcal{C}_{\mathsf{sound}}, \epsilon_{\mathsf{sound},1}, \epsilon_{\mathsf{sound},2})$*-statistically sound, where* $\mathcal{C}_{\mathsf{sound}}$ *is the class of circuits of size* $p(T_5)$ *and polynomial depth for all polynomials* p*, and* $\epsilon_{\mathsf{sound},1} = 1/T_4$*, and* $\epsilon_{\mathsf{sound},2}$ *is any negligible function,*
- *The* CCAVal *extraction procedure for the CCA-non-malleable commitment scheme is a circuit of size* T_2 *and polynomial depth, and*
- $T_2 \ll T_4 \ll T_5$*,*

Hybrid_0 *is computationally indistinguishable from* Hybrid_1*.*

We prove this claim via a sequence of subhybrids, which we describe here. Let $q = q(\lambda)$ be a polynomial upper bound on the number of Honest Computation Encoding queries made by $\mathcal{A}$.

- $\mathsf{Hybrid}_{0,0,0}$ is the same as Hybrid_0.
- $\mathsf{Hybrid}_{0,k,r}$ is the same as $\mathsf{Hybrid}_{0,k,r-1}$, except for the following differences. Whenever $\mathcal{A}$ submits its ℓ-th Honest Computation Encoding query asking for honest party P_i's encoding w.r.t f and I, the simulator does the following:

  1. Compute the extracted value $(\tilde{x}_j, \tilde{r}_{j,\mathsf{SM},1}, \tilde{K}_j, \tilde{r}_{j,\mathsf{com}}) \leftarrow \mathsf{CCAVal}(\tau_i, \mathsf{tag}_j, \mathsf{nmc}_j)$ of P_j's CCA-non-malleable commitment for each $j \in I \cap \mathcal{C}$.
  2. For each $j \in I \cap \mathcal{C}$, check whether

  $$\hat{m}_{j,1} = \mathsf{SM.Encode}(1^\lambda, \tilde{x}_j; \tilde{r}_{j,\mathsf{SM},1})$$

  and

  $$\mathsf{com}_j = \mathsf{NICommit}(1^\lambda, (\tilde{x}_j, \tilde{r}_{j,\mathsf{SM},1}, \tilde{K}_j); \tilde{r}_{j,\mathsf{com}}),$$

  where $\hat{m}_{j,1}$ is the semi-malicious input encoding sent by P_j, com_j is the perfectly-binding commitment sent by P_j, and $\tilde{x}_j, \tilde{r}_{j,\mathsf{SM},1}, \tilde{K}_j, \tilde{r}_{j,\mathsf{com}}$ are the extracted values from before.

- If both equalities hold for all $j \in I \cap \mathcal{C}$, then the simulator generates $\mathsf{WE.CT}_i$ in the same way as in Hybrid_0.
- If the equalities do not hold for some $j \in I \cap \mathcal{C}$, then if $i \leq k \in [n] \backslash \mathcal{C}$ and if $\ell \leq r$, the simulator instead computes $\mathsf{WE.CT}_i \leftarrow \mathsf{WE.Encrypt}(1^\lambda, \Phi_{\mathsf{WE},i}, 0^{|r_{i,\mathsf{com}}|})$.

- $\mathsf{Hybrid}_{0,n,q}$ is the same as Hybrid_1.

In the following, we denote with $\mathsf{expt}_{\mathcal{A}}^{0,k,r}$ the output of the simulator during $\mathsf{Hybrid}_{0,k,r}$. Note that for all $k \in [n]$, $\mathsf{Hybrid}_{0,k,q} = \mathsf{Hybrid}_{0,k+1,0}$. Thus, to prove Claim 4, it is then sufficient to prove the following claim.

Claim 5. *For all $k \in [n]$ and $r \in [q]$, assuming: Assuming:*

- *$(\mathcal{C}_{\mathsf{WE}}, \epsilon)$-security for the witness encryption scheme, where $\mathcal{C}_{\mathsf{WE}}$ is the class of circuits of size $p(T_5)$ for all polynomials p and $\epsilon = 1/T_5$,*
- *The zero knowledge protocol is $(\mathcal{C}_{\mathsf{sound}}, \epsilon_{\mathsf{sound},1}, \epsilon_{\mathsf{sound},2})$-statistically sound, where $\mathcal{C}_{\mathsf{sound}}$ is the class of circuits of size $p(T_5)$ and polynomial depth for all polynomials p, and $\epsilon_{\mathsf{sound},1} = 1/T_4$, and $\epsilon_{\mathsf{sound},2}$ is any negligible function,*
- *The CCAVal extraction procedure for the CCA-non-malleable commitment scheme is a circuit of size T_2 and polynomial depth, and*
- *$T_2 \ll T_4 \ll T_5$,*

$\mathsf{Hybrid}_{0,k,r}$ is computationally indistinguishable from $\mathsf{Hybrid}_{0,k,r-1}$.

We will rely on several subclaims in order to prove Claim 5. First we introduce some notation.

Assume for the sake of contradiction that there exists an adversary $(\mathcal{A}, \mathcal{D})$ and an index (k, r) such that $\mathcal{A}$ distinguishes between $\mathsf{Hybrid}_{0,k,r-1}$ and $\mathsf{Hybrid}_{0,k,r}$ with non-negligible

probability. That is, assume that

$$\left| \Pr\left[\mathcal{D}(\mathsf{expt}_{\mathcal{A}}^{0,k,r}) = 1\right] - \Pr\left[\mathcal{D}(\mathsf{expt}_{\mathcal{A}}^{0,k,r-1}) = 1\right] \right| \geq 1/p(\lambda),$$

for some polynomial p. Fix some $j^* \in \mathcal{C}$, and consider the event that during $\mathsf{Hybrid}_{0,k,r-1}$ or $\mathsf{Hybrid}_{0,k,r}$:

- $\mathcal{A}$ asks for P_k's honest input encoding,
- $\mathcal{A}$ sends corrupted party P_{j^*}'s input encoding to $\mathcal{S}$, where either

  $$\hat{m}_{j,1} \neq \mathsf{SM.Encode}(1^\lambda, \tilde{x}_j; \tilde{r}_{j,\mathsf{SM},1})$$

  or

  $$\mathsf{com}_j \neq \mathsf{NICommit}(1^\lambda, (\tilde{x}_j, \tilde{r}_{j,\mathsf{SM},1}, \tilde{K}_j); \tilde{r}_{j,\mathsf{com}}),$$

  and
- $\mathcal{A}$'s r-th Honest Computation Encoding query asks for P_k's encoding w.r.t. some (f, I) such that $j^* \in I$.

Define $\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r}$ and $\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r-1}$ to be the same as $\mathsf{expt}_{\mathcal{A}}^{0,k,r}$ and $\mathsf{expt}_{\mathcal{A}}^{0,k,r-1}$, except that whenever the event above does not occur, the simulator outputs a "dummy evaluation", where all parties behave according to the honest input specification, have input 0, and evaluate the constant $f(x_1, \ldots, x_n) = 0$ with $I = [n]$. Fixing the j^* that maximizes the probability of $\mathcal{A}$ distinguishing these two experiments, we then have that

$$\left| \Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r}) = 1\right] - \Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r-1}) = 1\right] \right| \geq 1/p'(\lambda),$$

for some polynomial $p'(\lambda)$.

Define PS_{k,j^*} to be the event that perfect soundness holds in the zero knowledge instance with prover P_{j^*} and verifier P_k which takes place during $\mathsf{Hybrid}_{0,k,\eta}$ for $\eta \in \{r-1, r\}$. Note that since both hybrids are identical up to the r-th Honest Computation Encoding query, this event is well-defined even if η is unspecified.

With this event defined, we can rewrite the probability

$$\Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r}) = 1\right]$$

as the following:

$$\Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r}) = 1 \mid \mathsf{PS}_{k,j^*}\right] \Pr\left[\mathsf{PS}_{k,j^*}\right] + \Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r}) = 1 \mid \overline{\mathsf{PS}}_{k,j^*}\right] \Pr\left[\overline{\mathsf{PS}}_{k,j^*}\right].$$

Claim 6. *Assuming the zero knowledge protocol is* $(\mathcal{C}_{\mathsf{sound}}, \epsilon_{\mathsf{sound},1}, \epsilon_{\mathsf{sound},2})$*-sound where* $\mathcal{C}_{\mathsf{sound}}$*,* $\epsilon_{\mathsf{sound},1}$*, and* $\epsilon_{\mathsf{sound},2}$ *are as in Claim 5, it holds that*

$$\Pr\left[\mathsf{PS}_{k,j^*}\right] \geq \epsilon_{\mathsf{sound},1}.$$

Proof. Assume this is not the case. Then we construct a reduction $\mathcal{R}$ to the soundness mode frequency property of the zero knowledge protocol. $\mathcal{R}$ is a circuit of size $\mathsf{poly}(T_2)$ which does the following:

1. Receive $\mathsf{zk}_{1,V}$ from the challenger.
2. Run $\mathsf{expt}_{\mathcal{A}}^{0,k}$, using $\mathsf{zk}_{1,V}$ as part of P_k's input encoding whenever this encoding is requested from $\mathcal{A}$.
3. Whenever $\mathcal{A}$ sends an input encoding on behalf of P_{j^*}, halt and output the $\mathsf{zk}_{1,j^*,P}$ message which is part of P_{j^*}'s input encoding.

By assumption, PS_{k,j^*} holds with probability $< \epsilon_{\mathsf{sound},1}$. This means that $\mathcal{E}(\tau_1, \sigma_{\mathsf{zk},V,k}) = 1$ with probability $< \epsilon_{\mathsf{sound},1}$. Thus $\mathcal{R}$ contradicts $(\mathcal{C}_{\mathsf{sound}}, \epsilon_{\mathsf{sound},1}, \epsilon_{\mathsf{sound},2})$-soundness of the zero knowledge protocol. □

Claim 7. *Assuming the zero knowledge protocol is* $(\mathcal{C}_{\mathsf{sound}}, \epsilon_{\mathsf{sound},1}, \epsilon_{\mathsf{sound},2})$*-sound where* $\mathcal{C}_{\mathsf{sound}}$*,* $\epsilon_{\mathsf{sound},1}$*, and* $\epsilon_{\mathsf{sound},2}$ *are as in Claim 5, and the extractor* CCAVal *for the CCA-non-malleable commitment scheme is a* T_2*-size circuit, it holds that for all* k *and* r*,*

$$\left| \Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r}) = 1 \mid \mathsf{PS}_{k,j^*}\right] - \Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r}) = 1 \mid \overline{\mathsf{PS}}_{k,j^*}\right] \right| \leq \epsilon_{\mathsf{sound},2}.$$

Proof. We prove the claim via a $\mathsf{poly}(T_2)$-size reduction to soundness of the zero knowledge protocol. Assume for the sake of contradiction that

$$\left| \Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r}) = 1 \mid \mathsf{PS}_{k,j^*}\right] - \Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r}) = 1 \mid \overline{\mathsf{PS}}_{k,j^*}\right] \right| > \epsilon_{\mathsf{sound},2}.$$

We construct the reduction $\mathcal{R}$, which behaves as follows:

1. Receive $\mathsf{zk}_{1,V}$ from the challenger.
2. Run $a \leftarrow \widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k}$ using $\mathsf{zk}_{1,k,V} = \mathsf{zk}_{1,V}$ whenever P_k's input encoding is queried, where a is the output of the experiment. Send $\mathsf{zk}_{1,j^*,P}$ to the challenger. Output $\mathcal{D}(a)$.

Note that the probability that $\mathcal{R}$ distinguishes between soundness modes is exactly

$$\left| \Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r}) = 1 \mid \mathsf{PS}_{k,j^*}\right] - \Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r}) = 1 \mid \overline{\mathsf{PS}}_{k,j^*}\right] \right|,$$

and thus $\mathcal{R}$ contradicts indistinguishability of soundness mode. □

Claim 8. *Assuming the existence of a distinguishing $\mathcal{A}$ as before, the zero knowledge protocol is $(\mathcal{C}_{\mathsf{sound}}, \epsilon_{\mathsf{sound},1}, \epsilon_{\mathsf{sound},2})$-sound where $\mathcal{C}_{\mathsf{sound}}$, $\epsilon_{\mathsf{sound},1}$, and $\epsilon_{\mathsf{sound},2}$ are as in Claim 5, and the extractor CCAVal for the CCA-non-malleable commitment scheme is a T_2-size circuit, it holds that for all k and r,*

$$\left| \Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r}) = 1 \wedge \mathsf{PS}_{k,j^*}\right] - \Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r-1}) = 1 \wedge \mathsf{PS}_{k,j^*}\right] \right| \geq \epsilon_{\mathsf{sound},1}/p(\lambda),$$

for some polynomial $p(\lambda)$.

Proof. By Claim 6 the left-hand side of the inequality is at least

$$\left| \left(\Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r}) = 1 \mid \mathsf{PS}_{k,j^*}\right] - \Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r-1}) = 1 \mid \mathsf{PS}_{k,j^*}\right] \right) \cdot \epsilon_{\mathsf{sound},1} \right|.$$

So it suffices to show that

$$\left| \Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r}) = 1 \mid \mathsf{PS}_{k,j^*}\right] - \Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r-1}) = 1 \mid \mathsf{PS}_{k,j^*}\right] \right| \geq 1/p(\lambda)$$

for some polynomial $p(\lambda)$.

Recall that by assumption we have

$$\left|\Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r})=1\right]-\Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r-1})=1\right]\right|\geq 1/\mathsf{poly}(\lambda). \tag{5.1}$$

We can lower-bound the left-hand side of (5.1) as

$$\left|\left(\Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r})=1\mid \mathsf{PS}_{k,j^*}\right]-\Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r-1})=1\mid \mathsf{PS}_{k,j^*}\right]\right)\cdot\Pr\left[\mathsf{PS}_{k,j^*}\right]+\right.$$
$$\left.\left(\Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r})=1\mid \overline{\mathsf{PS}}_{k,j^*}\right]-\Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r-1})=1\mid \overline{\mathsf{PS}}_{k,j^*}\right]\right)\cdot\Pr\left[\overline{\mathsf{PS}}_{k,j^*}\right]\right|,$$

which by claim Claim 7 is less than

$$\left|\left(\Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r})=1\mid \mathsf{PS}_{k,j^*}\right]-\Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r-1})=1\mid \mathsf{PS}_{k,j^*}\right]\right)\cdot\left(\Pr\left[\mathsf{PS}_{k,j^*}\right]+\Pr\left[\overline{PS}_{k,j^*}\right]\right)\right|$$
$$+2\epsilon_{\mathsf{sound},2}\cdot\Pr\left[\overline{\mathsf{PS}}_{k,j^*}\right].$$

(i.e., substitute out $Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r})=1\mid \overline{\mathsf{PS}}_{k,j^*}\right]$ and $\Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r-1})=1\mid \overline{\mathsf{PS}}_{k,j^*}\right]$ for $\Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r})=1\mid \mathsf{PS}_{k,j^*}\right]+\epsilon_{\mathsf{sound},2}$ and $\Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r-1})=1\mid \mathsf{PS}_{k,j^*}\right]+\epsilon_{\mathsf{sound},2}$, respectively.)

Thus,

$$\left|\left(\Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r})=1\mid \mathsf{PS}_{k,j^*}\right]-\Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r-1})=1\mid \mathsf{PS}_{k,j^*}\right]\right)\right|\geq 1/\mathsf{poly}(\lambda)-2\epsilon_{\mathsf{sound},2},$$

which proves the claim. □

Claim 9. *Assuming the "perfect soundness holds during soundness mode" property of the zero knowledge argument, and $(\mathcal{C}_{\mathsf{WE}},\epsilon)$-security for the witness encryption scheme, where $\mathcal{C}_{\mathsf{WE}}$ is the class of circuits of size $p(T_5)$ for all polynomials p and $\epsilon=1/T_5$, and $T_5\gg T_2$, the size of the extraction procedure* CCAVal *for the CCA commitment, it holds that for all k,*

$$\left|\Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r})=1\wedge \mathsf{PS}_{k,j^*}\right]-\Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r-1})=1\wedge \mathsf{PS}_{k,j^*}\right]\right|<\epsilon_{\mathsf{WE}}.$$

Proof. Fix a state state of the experiment just before the r-th Honest Computation Encoding. We show that given such a state where PS_{k,j^*} holds,

$$\left|\Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r}(\mathsf{state}))=1\right]-\Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r-1}(\mathsf{state}))=1\right]\right|<\epsilon_{\mathsf{WE}}.$$

We consider two cases. First is the case in which the "dummy evaluation" is triggered. In this case, the output of both $\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r}(\mathsf{state})$ and $\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r-1}(\mathsf{state})$ are both drawn from exactly the same distribution, and thus

$$\left| \Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r}(\mathsf{state}_1)) = 1\right] - \Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r-1}(\mathsf{state}_1)) = 1\right] \right| = 0.$$

The second case is where the "dummy evaluation" is not triggered, i.e. where the following three conditions are satisfied:

- $\mathcal{A}$ asks for P_k's honest input encoding,
- $\mathcal{A}$ sends corrupted party P_{j^*}'s input encoding to $\mathcal{S}$, where either

  $$\hat{m}_{j,1} \neq \mathsf{SM.Encode}(1^\lambda, \tilde{x}_j; \tilde{r}_{j,\mathsf{SM},1})$$

  or

  $$\mathsf{com}_j \neq \mathsf{NICommit}(1^\lambda, (\tilde{x}_j, \tilde{r}_{j,\mathsf{SM},1}, \tilde{K}_j); \tilde{r}_{j,\mathsf{com}}),$$

  and
- $\mathcal{A}$'s r-th Honest Computation Encoding query asks for P_k's encoding w.r.t. some (f, I) such that $j^* \in I$.

In this case, the difference between the two experiments is that when responding to the r-th Honest Computation Encoding in $\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r-1}(\mathsf{state})$, the simulator sends $\mathsf{WE.CT}_k \leftarrow \mathsf{WE.Encrypt}(1^\lambda, \Phi_{\mathsf{WE},k}, r_{k,\mathsf{com}})$ to $\mathcal{A}$ on behalf of P_k, whereas in $\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r}(\mathsf{state})$, the simulator sends $\mathsf{WE.CT}_k \leftarrow \mathsf{WE.Encrypt}(1^\lambda, \Phi_{\mathsf{WE},k}, 0^{|r_{k,\mathsf{com}}|})$. Here $\Phi_{\mathsf{WE},k}$ is the statement in Figure 5.2.

Assume for the sake of contradiction that

$$\left| \Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r}(\mathsf{state}_1)) = 1\right] - \Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r-1}(\mathsf{state}_1)) = 1\right] \right| \geq \epsilon_{\mathsf{WE}}.$$

WLOG fix the randomness of $\mathcal{A}$ which maximizes this probability. Note that if $\mathcal{A}$ is deterministic this means that state fully determines the statement $\Phi_{\mathsf{WE},k}$.

We build a reduction $\mathcal{R}$ which is of size T_2 and contradicts security of the witness encryption scheme. $\mathcal{R}$ has state hardcoded and does the following:

1. Receive $\mathsf{WE.CT}_k \leftarrow \mathsf{WE.Encrypt}(\Phi_{\mathsf{WE},k}, m)$ from the challenger, where m is either $r_{k,\mathsf{com}}$ or $0^{|r_{k,\mathsf{com}}|}$, and $\Phi_{\mathsf{WE},k}$ is the statement fixed by state and the randomness of $\mathcal{A}$.
2. Run $b \leftarrow \mathcal{D}(\widetilde{\mathsf{expt}}_{\mathcal{A}}^{0,k,r-1}(\mathsf{state}_1))$, where $\widetilde{\mathsf{expt}}_{\mathcal{A}}^{0,k,r-1}(\mathsf{state}_1)$ is computed in the same way as $\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r-1}(\mathsf{state}_1)$, except using $\mathsf{WE.CT}_k$ as P_k's witness encryption during the r-th Honest Computation Encoding.

If the challenger chooses $m = r_{k,\mathsf{com}}$ then the experiment run by $\mathcal{R}$ is exactly the same as $\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r-1}(\mathsf{state})$; if the challenger chooses $m = 0^{|r_{k,\mathsf{com}}|}$ then the experiment is exactly the same as $\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r}(\mathsf{state})$. Note that the statement $\Phi_{\mathsf{WE},k}$ is false because of perfect soundness of the zero knowledge scheme. Thus $\mathcal{R}$ is a size-T_2 machine which distinguishes between two different witness encryptions for the same false statement, thus contradicting security of the witness encryption scheme. □

We now finish the proof of Claim 5 using the three claims proven above.

Proof of Claim 5. We directly achieve a contradiction by applying Claim 8 and Claim 9, along with the fact that $\epsilon_{\mathsf{sound},1} \gg \epsilon_{\mathsf{WE}}$. □

5.3.0.4 Indistinguishability Between Hybrid_1 and Hybrid_2

The proof of indistinguishability between Hybrid_1 and Hybrid_2 is very similar to the previous proof. We include it for the sake of completeness.

Claim 10. *Assuming:*

- *The zero knowledge protocol is* $(\mathcal{C}_{\mathsf{sound}}, \epsilon_{\mathsf{sound},1}, \epsilon_{\mathsf{sound},2})$*-statistically sound, where* $\mathcal{C}_{\mathsf{sound}}$ *is the class of circuits of size* $p(T_5)$ *and polynomial depth for all polynomials* p*, and* $\epsilon_{\mathsf{sound},1} = 1/T_4$*, and* $\epsilon_{\mathsf{sound},2}$ *is any negligible function,*

- *The* CCAVal *extraction procedure for the CCA-non-malleable commitment scheme is a circuit of size* T_2 *and polynomial depth, and*
- $T_2 \ll T_4 \ll T_5$,

Hybrid_1 *is computationally indistinguishable from* Hybrid_2.

We prove this claim via a sequence of subhybrids, which we describe here. Let $q = q(\lambda)$ be a polynomial upper bound on the number of Corrupt Computation Encoding queries made by $\mathcal{A}$.

- $\mathsf{Hybrid}_{1,0,0}$ is the same as Hybrid_1.
- $\mathsf{Hybrid}_{1,k,r}$ is the same as $\mathsf{Hybrid}_{1,k,r-1}$, except for the following differences. Whenever $\mathcal{A}$ submits its ℓ-th Corrupt Computation Encoding, on behalf of some corrupted party P_j w.r.t. f and I, then the simulator does the following:
  1. Compute the extracted value $(\tilde{x}_j, \tilde{r}_{j,\mathsf{SM},1}, \tilde{K}_j, \tilde{r}_{j,\mathsf{com}}) \leftarrow \mathsf{CCAVal}(\tau_i, \mathsf{tag}_j, \mathsf{nmc}_j)$ of P_j's CCA-non-malleable commitment for each $i \in I \setminus \mathcal{C}$.
  2. For each $i \in I \setminus \mathcal{C}$, $i \leq k$, check if there exists a $j \in I \cap \mathcal{C}$ such that:
     - $\mathsf{ZKVerify}_2(\phi_{\mathsf{zk},j,k}, \mathsf{zk}_{1,i,V}, \mathsf{zk}_{1,j,P}, \mathsf{zk}_{2,j\to i,P})$ verifies, and
     - Steps 2-5 of $\Phi_{\mathsf{zk},j,i}$ do not hold with respect to the extracted values $\tilde{x}_j, \tilde{r}_{j,\mathsf{SM},1}$, $\tilde{K}_j, \tilde{r}_{j,\mathsf{com}}$ and the input encoding phase of the protocol. Note that this is checkable in polynomial time given the values $\tilde{x}_j, \tilde{r}_{j,\mathsf{SM},1}, \tilde{K}_j, \tilde{r}_{j,\mathsf{com}}$.
  3. If there does exist such a j, then if either $i < k$, or if $i = k$ and $\ell \leq r$, halt and output a special abort symbol $\perp^*$.
- $\mathsf{Hybrid}_{1,n,q}$ is the same as Hybrid_2.

Note that for all $k \in [n]$, $\mathsf{Hybrid}_{1,k,q} = \mathsf{Hybrid}_{1,k+1,0}$. Thus, to prove Claim 10, it is then sufficient to prove the following claim.

Claim 11. *For all $k \in [n]$ and $r \in [q]$, assuming: Assuming:*

- *The zero knowledge protocol is $(\mathcal{C}_{\mathsf{sound}}, \epsilon_{\mathsf{sound},1}, \epsilon_{\mathsf{sound},2})$-statistically sound, where $\mathcal{C}_{\mathsf{sound}}$ is the class of circuits of size $p(T_5)$ and polynomial depth for all polynomials p, and $\epsilon_{\mathsf{sound},1} = 1/T_4$, and $\epsilon_{\mathsf{sound},2}$ is any negligible function,*
- *The* CCAVal *extraction procedure for the CCA-non-malleable commitment scheme is a circuit of size T_2 and polynomial depth, and*
- $T_2 \ll T_4 \ll T_5$,

$\mathsf{Hybrid}_{1,k,r}$ *is computationally indistinguishable from* $\mathsf{Hybrid}_{1,k,r-1}$.

We will rely on several subclaims in order to prove Claim 11. First we introduce some notation. In the following, we denote with $\mathsf{expt}_{\mathcal{A}}^{1,k,r}$ the output of the simulator during $\mathsf{Hybrid}_{1,k,r}$.

Assume for the sake of contradiction that there exists an adversary $(\mathcal{A}, \mathcal{D})$ and an index (k, r) such that $\mathcal{A}$ distinguishes between $\mathsf{Hybrid}_{1,k,r-1}$ and $\mathsf{Hybrid}_{1,k,r}$ with non-negligible probability. That is, assume that

$$\left| \Pr\left[\mathcal{D}(\mathsf{expt}_{\mathcal{A}}^{1,k,r}) = 1\right] - \Pr\left[\mathcal{D}(\mathsf{expt}_{\mathcal{A}}^{1,k,r-1}) = 1\right] \right| \geq 1/p(\lambda),$$

for some polynomial p. Fix some $j^* \in \mathcal{C}$, and consider the event that during $\mathsf{Hybrid}_{1,k,r-1}$ or $\mathsf{Hybrid}_{1,k,r}$:

- $\mathcal{A}$ asks for P_k's honest input encoding,
- $\mathcal{A}$ sends corrupted party P_{j^*}'s input encoding to $\mathcal{S}$, and
- $\mathcal{A}$'s r-th Corrupt Computation Encoding query sends P_{j^*}'s computation encoding w.r.t. some (f, I) such that $k \in I$,and the conditions for special abort hold with respect to this encoding.

Define $\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r}$ and $\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r-1}$ to be the same as $\mathsf{expt}_{\mathcal{A}}^{1,k,r}$ and $\mathsf{expt}_{\mathcal{A}}^{1,k,r-1}$, except that whenever the event above does not occur, the simulator outputs a "dummy evaluation", where all parties behave according to the honest input specification, have input 0, and evaluate the constant $f(x_1, \ldots, x_n) = 0$ with $I = [n]$. Fixing the j^* that maximizes the probability of $\mathcal{A}$ distinguishing these two experiments, we then have that

$$\left| \Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r}) = 1\right] - \Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r-1}) = 1\right] \right| \geq 1/p'(\lambda),$$

for some polynomial $p'(\lambda)$.

Define PS_{k,j^*} to be the event that perfect soundness holds in the zero knowledge instance with prover P_{j^*} and verifier P_k which takes place during $\mathsf{Hybrid}_{1,k,\eta}$ for $\eta \in \{r-1, r\}$. Note that since both hybrids are identical up to the r-th Corrupt Computation Encoding query, this event is well-defined even if η is unspecified.

With this event defined, we can rewrite the probability

$$\Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r}) = 1\right]$$

as the following:

$$\Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r}) = 1 \mid \mathsf{PS}_{k,j^*}\right] \Pr\left[\mathsf{PS}_{k,j^*}\right] + \Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r}) = 1 \mid \overline{\mathsf{PS}}_{k,j^*}\right] \Pr\left[\overline{\mathsf{PS}}_{k,j^*}\right].$$

Claim 12. *Assuming the zero knowledge protocol is $(\mathcal{C}_{\mathsf{sound}}, \epsilon_{\mathsf{sound},1}, \epsilon_{\mathsf{sound},2})$-sound where $\mathcal{C}_{\mathsf{sound}}$, $\epsilon_{\mathsf{sound},1}$, and $\epsilon_{\mathsf{sound},2}$ are as in Claim 11, it holds that*

$$\Pr\left[\mathsf{PS}_{k,j^*}\right] \geq \epsilon_{\mathsf{sound},1}.$$

Proof. Assume this is not the case. Then we construct a reduction $\mathcal{R}$ to the soundness mode frequency property of the zero knowledge protocol. $\mathcal{R}$ is a circuit of size $\mathsf{poly}(T_2)$ which does the following:

1. Receive $\mathsf{zk}_{1,V}$ from the challenger.

2. Run $\mathsf{expt}_{\mathcal{A}}^{1,k}$, using $\mathsf{zk}_{1,V}$ as part of P_k's input encoding whenever this encoding is requested from $\mathcal{A}$.

3. Whenever $\mathcal{A}$ sends an input encoding on behalf of P_{j^*}, halt and output the $\mathsf{zk}_{1,j^*,P}$ message which is part of P_{j^*}'s input encoding.

By assumption, PS_{k,j^*} holds with probability $< \epsilon_{\mathsf{sound},1}$. This means that $\mathcal{E}(\tau_1, \sigma_{\mathsf{zk},V,k}) = 1$ with probability $< \epsilon_{\mathsf{sound},1}$. Thus $\mathcal{R}$ contradicts $(\mathcal{C}_{\mathsf{sound}}, \epsilon_{\mathsf{sound},1}, \epsilon_{\mathsf{sound},2})$-soundness of the zero knowledge protocol. □

Claim 13. *Assuming the zero knowledge protocol is $(\mathcal{C}_{\mathsf{sound}}, \epsilon_{\mathsf{sound},1}, \epsilon_{\mathsf{sound},2})$-sound where $\mathcal{C}_{\mathsf{sound}}$, $\epsilon_{\mathsf{sound},1}$, and $\epsilon_{\mathsf{sound},2}$ are as in Claim 11, and the extractor CCAVal for the CCA-non-malleable commitment scheme is a T_2-size circuit, it holds that for all k and r,*

$$\left| \Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r}) = 1 \mid \mathsf{PS}_{k,j^*}\right] - \Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r}) = 1 \mid \overline{\mathsf{PS}}_{k,j^*}\right] \right| \leq \epsilon_{\mathsf{sound},2}.$$

Proof. We prove the claim via a $\mathsf{poly}(T_2)$-size reduction to soundness of the zero knowledge protocol. Assume for the sake of contradiction that

$$\left| \Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r}) = 1 \mid \mathsf{PS}_{k,j^*}\right] - \Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r}) = 1 \mid \overline{\mathsf{PS}}_{k,j^*}\right] \right| > \epsilon_{\mathsf{sound},2}.$$

We construct the reduction $\mathcal{R}$, which behaves as follows:

1. Receive $\mathsf{zk}_{1,V}$ from the challenger.

2. Run $a \leftarrow \widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r}$ using $\mathsf{zk}_{1,k,V} = \mathsf{zk}_{1,V}$ whenever P_k's input encoding is queried, where a is the output of the experiment. Send $\mathsf{zk}_{1,j^*,P}$ to the challenger. Output $\mathcal{D}(a)$.

Note that the probability that $\mathcal{R}$ distinguishes between soundness modes is exactly

$$\left| \Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r}) = 1 \mid \mathsf{PS}_{k,j^*}\right] - \Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r}) = 1 \mid \overline{\mathsf{PS}}_{k,j^*}\right] \right|,$$

and thus $\mathcal{R}$ contradicts indistinguishability of soundness mode. □

Claim 14. *Assuming the existence of a distinguishing $\mathcal{A}$ as before, the zero knowledge protocol is $(\mathcal{C}_{\mathsf{sound}}, \epsilon_{\mathsf{sound},1}, \epsilon_{\mathsf{sound},2})$-sound where $\mathcal{C}_{\mathsf{sound}}$, $\epsilon_{\mathsf{sound},1}$, and $\epsilon_{\mathsf{sound},2}$ are as in Claim 11, and the extractor CCAVal for the CCA-non-malleable commitment scheme is a T_2-size circuit, it holds that for all k and r,*

$$\left| \Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r}) = 1 \wedge \mathsf{PS}_{k,j^*}\right] - \Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r-1}) = 1 \wedge \mathsf{PS}_{k,j^*}\right] \right| \geq \epsilon_{\mathsf{sound},1}/p(\lambda),$$

for some polynomial $p(\lambda)$.

Proof. By Claim 12 the left-hand side of the inequality is at least

$$\left| \left(\Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r}) = 1 \mid \mathsf{PS}_{k,j^*}\right] - \Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r-1}) = 1 \mid \mathsf{PS}_{k,j^*}\right] \right) \cdot \epsilon_{\mathsf{sound},1} \right|.$$

So it suffices to show that

$$\left| \Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r}) = 1 \mid \mathsf{PS}_{k,j^*}\right] - \Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r-1}) = 1 \mid \mathsf{PS}_{k,j^*}\right] \right| \geq 1/p(\lambda)$$

for some polynomial $p(\lambda)$.

Recall that by assumption we have

$$\left| \Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r}) = 1\right] - \Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r-1}) = 1\right] \right| \geq 1/\mathsf{poly}(\lambda). \tag{5.2}$$

We can lower-bound the left-hand side of (5.2) as

$$\begin{aligned} &\left| \left(\Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r}) = 1 \mid \mathsf{PS}_{k,j^*}\right] - \Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r-1}) = 1 \mid \mathsf{PS}_{k,j^*}\right] \right) \cdot \Pr\left[\mathsf{PS}_{k,j^*}\right] + \right. \\ &\left. \left(\Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r}) = 1 \mid \overline{\mathsf{PS}}_{k,j^*}\right] - \Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r-1}) = 1 \mid \overline{\mathsf{PS}}_{k,j^*}\right] \right) \cdot \Pr\left[\overline{\mathsf{PS}}_{k,j^*}\right] \right|, \end{aligned}$$

which by claim Claim 13 is less than

$$\begin{aligned} &\left| \left(\Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r}) = 1 \mid \mathsf{PS}_{k,j^*}\right] - \Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r-1}) = 1 \mid \mathsf{PS}_{k,j^*}\right] \right) \cdot \left(\Pr\left[\mathsf{PS}_{k,j^*}\right] + \Pr\left[\overline{PS}_{k,j^*}\right]\right) \right| \\ &+ 2\epsilon_{\mathsf{sound},2} \cdot \Pr\left[\overline{\mathsf{PS}}_{k,j^*}\right]. \end{aligned}$$

(i.e., substitute out $Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r}) = 1 \mid \overline{\mathsf{PS}}_{k,j^*}\right]$ and $\Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r-1}) = 1 \mid \overline{\mathsf{PS}}_{k,j^*}\right]$ for $\Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r}) = 1 \mid \mathsf{PS}_{k,j^*}\right] + \epsilon_{\mathsf{sound},2}$ and $\Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r-1}) = 1 \mid \mathsf{PS}_{k,j^*}\right] + \epsilon_{\mathsf{sound},2}$, respectively.)

Thus,

$$\left| \left(\Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r}) = 1 \mid \mathsf{PS}_{k,j^*}\right] - \Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r-1}) = 1 \mid \mathsf{PS}_{k,j^*}\right] \right) \right| \geq 1/\mathsf{poly}(\lambda) - 2\epsilon_{\mathsf{sound},2},$$

which proves the claim. □

Claim 15. *Assuming the "perfect soundness holds during soundness mode" property of the zero knowledge argument, , it holds that for all k,*

$$\left| \Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r}) = 1 \wedge \mathsf{PS}_{k,j^*}\right] - \Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r-1}) = 1 \wedge \mathsf{PS}_{k,j^*}\right] \right| = 0.$$

Proof. This follows directly from the perfect soundness mode of the ZK argument scheme. □

We now finish the proof of Claim 11 using the three claims proven above.

Proof of Claim 11. We directly achieve a contradiction by applying Claim 14 and Claim 15. □

5.3.0.5 Proving Indistinguishability of the Remaining Hybrids

Claim 16. *Assuming the ZK argument scheme satisfies $(\mathcal{C}_{\mathcal{S}}, \mathcal{C}_{\mathsf{zk}}, \epsilon_{\mathcal{S}})$-adaptive reusable statistical zero knowledge, where $\mathcal{C}_{\mathcal{S}}$ is the class of circuits of size $\mathsf{poly}(T_1)$ and depth T_1 (i.e. the simulator runs in size $\mathsf{poly}(T_1)$ and depth T_1), and $\mathcal{C}_{\mathsf{zk}}$ is the class of circuits of size $p(T_3)$ for all polynomials p, and $\epsilon_{\mathcal{S}}$ is any negligible function, and the CCA extractor CCAVal is a circuit of size T_2, where $T_2 \ll T_3$, then for any polynomial time MPC adversary $\mathcal{A}$ and unbounded distinguisher $\mathcal{D}$, we have*

$$|\Pr[\mathcal{D}(\mathsf{Hybrid}_2) = 1] - \Pr[\mathcal{D}(\mathsf{Hybrid}_3) = 1]| < \mathsf{negl}(\lambda)$$

for some negligible negl.

Proof. This can be done by introducing $|[n] \setminus \mathcal{C}|$ intermediate hybrids. For simplicity, we use n hybrids, where $||\mathcal{C}|$ hybrids are non-functional. We index each hybrid as $\mathsf{Hybrid}_{2,i}$ for

$i \in [n]$. $\mathsf{Hybrid}_{2,i}$ is exactly the same as the same as $\mathsf{Hybrid}_{2,i-1}$ except that if $i \in [n] \setminus \mathcal{C}$, every $\mathsf{zk}_{2,i\to j,P}$ is now generated by running $\mathsf{ZKSim}(\sigma_{\mathsf{zk}_{1,i,P}}, \Phi_{\mathsf{zk},i,j}, \mathsf{zk}_{1,j,V})$. Note that the final hybrid in the series is exactly the same as Hybrid_3. To prove the claim, it suffices to show indistinguishability between each successive pair of subhybrids.

Assume for the sake of contradiction that $(\mathcal{A}, \mathcal{D})$ distinguishes between two successive subhybrids $\mathsf{Hybrid}_{2,i}$ and $\mathsf{Hybrid}_{2,i-1}$. We then construct a reduction $(\mathcal{R}, \mathcal{D})$ which breaks the statistical ZK property of the zero knowledge protocol. $\mathcal{R}$ is a circuit of size $\mathsf{poly}(T_2)$ and depth T_1 and does the following:

1. Receive $\mathsf{zk}_{1,P}$ from the challenger.
2. Run $\mathsf{Hybrid}_{2,i-1}$ with $\mathcal{A}$, using $\mathsf{zk}_{1,i,P} = \mathsf{zk}_{1,P}$ (i.e. use the challenger's round-one zk prover's message as the round-one prover's message for P_{i_2} as part of its input encoding.
3. When $\mathcal{A}$ asks for an honest computation encoding from P_i w.r.t. f and I, for each $j \in I \setminus \{i\}$, send the message $(\Phi_{\mathsf{zk},i,j}, W_{\mathsf{zk},i}, \mathsf{zk}_{1,j,V})$ to the challenger, and receive a response $\mathsf{zk}_{2,i\to j,P} = \mathsf{zk}_{2,P}$ from the challenger.
4. Generate P_{i_2}'s honest computation encoding in the same way as in $\mathsf{Hybrid}_{2,i-1}$ except using the challenger's responses $\{\mathsf{zk}_{2,i\to j,P}\}_{j\in I\setminus\{i\}}$ as the ZK2 messages instead of generating them honestly.
5. Output the result of the experiment.

If the challenger sends honestly generated proofs to $\mathcal{R}$, then the output of $\mathcal{R}$ is identical to $\mathsf{Hybrid}_{2,i-1}$; otherwise, if the challenger simulates the proofs, then the output of $\mathcal{R}$ is identical to $\mathsf{Hybrid}_{2,i}$. Note that $\mathcal{R}$ has size $\ll T_3$; thus by assumption $(\mathcal{R}, \mathcal{D})$ contradicts $(\mathcal{C}_{\mathcal{S}}, \mathcal{C}_{\mathsf{zk}}, \epsilon_{\mathcal{S}})$-statistical zero knowledge of the zero knowledge protocol. □

Claim 17. *Assuming that the CCA non-malleable commitment scheme satisfies $(\mathcal{C}, \epsilon)$-CCA security (Definition 16), where $\mathcal{C}$ contains all circuits of size $\mathsf{poly}(T_1)$ where T_1 is the size*

of the ZK simulator, and ϵ is any negligible function, we have that for any polynomial time MPC adversary $(\mathcal{A}, \mathcal{D})$:

$$|\Pr[\mathcal{D}[\mathsf{Hybrid}_3] = 1] - \Pr[\mathcal{D}[\mathsf{Hybrid}_4] = 1]| \leq \mathsf{negl}(\lambda),$$

for some negligible negl.

Proof. We show this by constructing intermediate hybrids $\mathsf{Hybrid}_{3,i}$ for $i \in [n]$. We define $\mathsf{Hybrid}_{3,i}$ to be identical to the previous hybrid except that if P_i is honest, nmc_i is generated as a non-malleable commitment of all zero string with tag tag_i during the Honest Input Encoding query. Note that $\mathsf{Hybrid}_{3,0}$ is identical to Hybrid_3 and $\mathsf{Hybrid}_{3,n}$ is identical to Hybrid_4. We show that for any two intermediate hybrids $\mathsf{Hybrid}_{3,i-1}$ and $\mathsf{Hybrid}_{3,i}$, it holds that for any polynomial time distingusher $\mathcal{D}$:

$$|\Pr[\mathcal{D}[\mathsf{Hybrid}_{3,i-1}] = 1] - \Pr[\mathcal{D}[\mathsf{Hybrid}_{3,i}] = 1]| \leq \mathsf{negl}(\lambda)$$

The only difference is how nmc_i is generated. If the advantage in distinguishing between the two is more than $\frac{1}{\mathsf{poly}(\lambda)}$ for some polynomial poly, then, we can create a reduction $\mathcal{R}$ that runs in time $\mathsf{poly}(T_1)$ and breaks the security of the one-round CCA commitment scheme with the same advantage. Here is how the reduction works:

- $\mathcal{R}$ submits $\mathsf{tag}^* = \mathsf{tag}_i$ to the CCA challenger.
- It runs the adversary $(\mathcal{A}, \mathcal{D})$ as in $\mathsf{Hybrid}_{3,i-1}$.
- $\mathcal{R}$ generates $\mathsf{nmc}_{i'}$ for all $i' \in [n] \setminus \mathcal{C}$ and $i' \neq i$ as in $\mathsf{Hybrid}_{3,i-1}$.
- For all $P_{i'}$, $i' \in [n] \setminus \mathcal{C}$, $\mathcal{R}$ sends a τ-query to the CCA challenger, and uses the response as the string $\tau_{i'}$ given the input encoding for $P_{i'}$.
- When $\mathcal{R}$ receives the Honest Input Encoding query from $\mathcal{A}$ for P_i with input x_i, it sends $\alpha_0 = (x_i, r_{i,\mathsf{SM}}, K_i, r_{i,\mathsf{com}})$ and $\alpha_1 = 0^{|x_i, r_{i,\mathsf{SM}}, K_i, r_{i,\mathsf{com}}|}$ to the challenger of the non-malleable commitment. It gets a response nmc^* which is a commitment with

respect to the tag tag_i. It it either a commitment of α_0 or α_1. The reduction uses this as nmc_i when constructing P_i's input encoding.

- Whenever $\mathsf{Hybrid}_{3,i-1}$ needs to extract a CCA commitment nmc_j w.r.t. tag_j and some honest $\tau_{i'}$, $\mathcal{R}$ sends a query $(\tau_{i'}, \mathsf{tag}_j, \mathsf{nmc}_j)$, and uses the response as the extracted value.

- Finally it outputs whatever $\mathcal{D}$ outputs.

Note that if nmc^* is a commitment of α_0, then the view is identical as in $\mathsf{Hybrid}_{3,i}$, otherwise it is as in $\mathsf{Hybrid}_{3,i-1}$. The reduction runs in time polynomial in T_1, since excluding the simulation for ZK rest of the steps are polynomial time. Further, the CCAVal algorithm is never invoked for the challenge tag tag_i. Thus if $\mathcal{D}$ distinguishes between the two cases with probability $\frac{1}{\mathsf{poly}(\lambda)}$, then, it must win in the CCA non-malleable commitment security game with the same advantage.

This proves the claim. □

Claim 18. *Assume that the perfectly binding commitment scheme is hiding against adversaries of size* $\mathsf{poly}(T_2)$*, where* T_2 *is the size of the* CCAVal *circuit. Then we have that*

$$|\Pr[\mathcal{D}[\mathsf{Hybrid}_4] = 1] - \Pr[\mathcal{D}[\mathsf{Hybrid}_5] = 1]| \leq \mathsf{negl}(\lambda)$$

for some negligible negl*.*

Proof. We show this by constructing intermediate hybrids $\mathsf{Hybrid}_{4,i}$ for $i \in [n]$. We define $\mathsf{Hybrid}_{4,i}$ to be identical to the previous hybrid except that if P_i is honest, com_i is generated as a non-malleable commitment of all zero string during the Honest Input Encoding query. Note that $\mathsf{Hybrid}_{4,0}$ is identical to Hybrid_4 and $\mathsf{Hybrid}_{4,n}$ is identical to Hybrid_5. We show that for any two intermediate hybrids $\mathsf{Hybrid}_{4,i-1}$ and $\mathsf{Hybrid}_{4,i}$, it holds that for any polynomial time distingusher $\mathcal{D}$:

$$|\Pr[\mathcal{D}[\mathsf{Hybrid}_{4,i-1}] = 1] - \Pr[\mathcal{D}[\mathsf{Hybrid}_{4,i}] = 1]| \leq \mathsf{negl}(\lambda)$$

The only difference is how com_i is generated. Assume there is an $(\mathcal{A}, \mathcal{D})$ where the distinguishing advantage between the two is more than $\frac{1}{\mathsf{poly}(\lambda)}$ for some polynomial poly. Then we can create a reduction $\mathcal{R}$ that runs in time $\mathsf{poly}(T_2)$, and contradicts hiding of the commitment scheme. First, fix the randomness of $\mathcal{A}$ and all randomness in $\mathsf{Hybrid}_{4,i-1}$ and $\mathsf{Hybrid}_{4,i}$ except for that used to generate P_i's perfectly-binding commitment com_i. There must be a way to fix this randomness so that $(\mathcal{A}, \mathcal{D})$ still has advantage $\frac{1}{\mathsf{poly}(\lambda)}$ in distinguishing the two hybrids. Note also that this fixes the input x_i which $\mathcal{A}$ chooses for P_i, and thus fixes the committed value in $\mathsf{Hybrid}_{4,i}$. The reduction then works as follows:

- It runs the adversary $(\mathcal{A}, \mathcal{D})$ as in $\mathsf{Hybrid}_{4,i-1}$.
- The reduction generates com_j for all $j \in [n] \setminus \mathcal{C}$ and $j \neq i$ as in $\mathsf{Hybrid}_{4,i-1}$.
- When the reduction receives an Honest Input Encoding request from $\mathcal{A}$ for P_i with input x_i, it sends $\alpha_0 = (x_i, r_{i,\mathsf{SM}}, K_i)$ and $\alpha_1 = 0^{|x_i, r_{i,\mathsf{SM}}, K_i|}$ to the challenger of the perfectly binding commitment. It gets a response com^*. It it either a commitment of α_0 or α_1. The reduction uses this in constructing P_i's input encoding
- The reduction runs the rest of the experiment exactly the same as $\mathsf{Hybrid}_{4,i-1}$.

Note that if com^* is a commitment of α_0, then the view is identical as in $\mathsf{Hybrid}_{4,i}$, otherwise it is as in $\mathsf{Hybrid}_{4,i-1}$. The reduction runs in time polynomial in T_2. Thus if $\mathcal{D}$ distinguishes between the two cases with probability $\frac{1}{\mathsf{poly}(\lambda)}$, then, it contradicts hiding of the perfectly binding commitment scheme against adversaries of size $\mathsf{poly}(T_2)$.

This proves the claim. □

Claim 19. *Assume that the PRF is secure against adversaries of size* $\mathsf{poly}(T_2)$*, where* T_2 *is the size of the* CCAVal *circuit. Then we have that*

$$|\Pr[\mathcal{D}[\mathsf{Hybrid}_5] = 1] - \Pr[\mathcal{D}[\mathsf{Hybrid}_6] = 1]| \leq \mathsf{negl}(\lambda)$$

for some negligible negl*.*

Proof. We show this by constructing intermediate hybrids $\mathsf{Hybrid}_{5,i}$ for $i \in [n]$. We define $\mathsf{Hybrid}_{5,i}$ to be identical to the previous hybrid except that if P_i is honest, then during any the HONEST COMPUTATION ENCODING query for P_i the hybrid generates $\hat{m}_{i,2}$ and $\mathsf{com}_{i,\hat{m}_{i,2}}$ using true randomness instead of the PRF evaluations. Note that $\mathsf{Hybrid}_{5,0}$ is identical to Hybrid_5 and $\mathsf{Hybrid}_{5,n}$ is identical to Hybrid_6. We show that for any two intermediate hybrids $\mathsf{Hybrid}_{5,i-1}$ and $\mathsf{Hybrid}_{5,i}$, it holds that for any polynomial time distinguisher $\mathcal{D}$:

$$|\Pr[\mathcal{D}[\mathsf{Hybrid}_{5,i-1}] = 1] - \Pr[\mathcal{D}[\mathsf{Hybrid}_{5,i}] = 1]| \leq \mathsf{negl}(\lambda)$$

Assume there is an $(\mathcal{A}, \mathcal{D})$ where the distinguishing advantage between the two is more than $\frac{1}{\mathsf{poly}(\lambda)}$ for some polynomial poly. Then we can create a reduction $\mathcal{R}$ that runs in time $\mathsf{poly}(T_2)$, and contradicts security of the PRF. The reduction works as follows:

- It runs the adversary $(\mathcal{A}, \mathcal{D})$ as in $\mathsf{Hybrid}_{5,i-1}$.
- The reduction generates computation encodings for all $j \in [n] \setminus \mathcal{C}$ and $j \neq i$ as in $\mathsf{Hybrid}_{5,i-1}$.
- Whenever a HONEST COMPUTATION ENCODING query is made requesting P_i's encoding, $\mathcal{R}$ makes two queries to the PRF oracle at indices $(f, I, 1)$ and $(f, I, 2)$, receiving strings r_1 and r_2. It then uses r_1 as the randomness when computing $\hat{m}_{i,2}$, and uses r_2 as the randomness when computing $\mathsf{com}_{i,\hat{m}_{i,2}}$.
- The reduction runs the rest of the experiment exactly the same as $\mathsf{Hybrid}_{5,i-1}$.

Note that if the oracle is supplying PRF values, then the view is identical as in $\mathsf{Hybrid}_{5,i}$. If the oracle is supplying true random values, the view is as in $\mathsf{Hybrid}_{5,i-1}$. The reduction runs in time polynomial in T_2. Thus if $\mathcal{D}$ distinguishes between the two cases with probability $\frac{1}{\mathsf{poly}(\lambda)}$, then, it contradicts security of the PRF against adversaries of size $\mathsf{poly}(T_2)$.

This proves the claim. □

Claim 20. *Assume that the perfectly binding commitment scheme is secure against adversaries of size* $\mathsf{poly}(T_2)$*, where* T_2 *is the size of the* CCAVal *circuit. Then we have that*

$$|\Pr[\mathcal{D}[\mathsf{Hybrid}_6] = 1] - \Pr[\mathcal{D}[\mathsf{Hybrid}_7] = 1]| \leq \mathsf{negl}(\lambda)$$

for some negligible negl*.*

Proof. We show this by constructing intermediate hybrids $\mathsf{Hybrid}_{6,i,r}$ for $i \in [n]$, $r \in [q]$, where q is a (polynomial) upper bound on the total number of Honest Computation Encoding queries that $\mathcal{A}$ makes. We define $\mathsf{Hybrid}_{6,i,r}$ to be identical to the previous hybrid except that if P_i is honest, then during the r-th Honest Computation Encoding query for P_i, the hybrid computes $\mathsf{com}_{i,\hat{m}_{i,2}} \leftarrow \mathsf{NICommit}(0^{|\hat{m}_{i,2}|})$ whenever the equalities checked in the steps for Hybrid_1 do not hold. Note that $\mathsf{Hybrid}_{6,i,q} = \mathsf{Hybrid}_{6,i+1,0}$, $\mathsf{Hybrid}_{6,1,0} = \mathsf{Hybrid}_6$, and $\mathsf{Hybrid}_{6,n,q} = \mathsf{Hybrid}_7$. We show that for any two intermediate hybrids $\mathsf{Hybrid}_{6,i,r-1}$ and $\mathsf{Hybrid}_{6,i,r}$, it holds that for any polynomial time distinguisher $\mathcal{D}$:

$$|\Pr[\mathcal{D}[\mathsf{Hybrid}_{6,i,r-1}] = 1] - \Pr[\mathcal{D}[\mathsf{Hybrid}_{6,i,r}] = 1]| \leq \mathsf{negl}(\lambda)$$

Assume there is an $(\mathcal{A}, \mathcal{D})$ where the distinguishing advantage between the two is more than $\frac{1}{\mathsf{poly}(\lambda)}$ for some polynomial poly. Then we can create a reduction $\mathcal{R}$ that runs in time $\mathsf{poly}(T_2)$, and contradicts security of the PRF. First, fix the randomness used in all rounds before the r-th Honest Computation Encoding made to P_i. Let (f, I) be this r-th query. In particular, this fixes whether or not the equalities check in the steps for Hybrid_1 hold w.r.t. P_i, f and I. It also fixes P_i's semi-malicious MrNISC message $\hat{m}_{i,2}$ which it computes when computing its r-th honest computation encoding. If we fix the randomness such that the distinguishing advantage is maximized, then the distinguishing advantage must still be polynomial in λ. This means that the equalities must not hold, otherwise the two hybrids are identical.

The reduction $\mathcal{R}$ then works as follows. It plays a game with a commitment challenger, which either gives a commitment to $\hat{m}_{i,2}$ or $0^{|\hat{m}_{i,2}|}$ $\mathcal{R}$ does the following:

- It runs the adversary $(\mathcal{A}, \mathcal{D})$ as in $\mathsf{Hybrid}_{6,i,r-1}$, with the randomness fixed as described above.
- When $\mathcal{A}$ submits the r-th *Honest Computation Encoding*, $\mathcal{R}$ queries the challenger to get com, which it then uses as $\mathsf{com}_{i,\hat{m}_{i,2}}$ when generating its response on behalf of P_i.
- $\mathcal{R}$ runs the rest of the experiment in the same way as $\mathsf{Hybrid}_{6,i,r-1}$.

Note that if the challenger sends $\mathcal{R}$ a commitment to $\hat{m}_{i,2}$, then the view is identical to that in $\mathsf{Hybrid}_{6,i,r-1}$. If the challenger sends a commitment to $0^{|\hat{m}_{i,2}|}$, the view is as in $\mathsf{Hybrid}_{6,i,r}$. The reduction runs in time polynomial in T_2. Thus if $\mathcal{D}$ distinguishes between the two cases with probability $\frac{1}{\mathsf{poly}(\lambda)}$, then, it contradicts the hiding of the perfectly binding commitment scheme against adversaries of size $\mathsf{poly}(T_2)$.

This proves the claim. □

Claim 21. *Assuming semi-malicious security of the underlying semi-malicious protocol holds against* $\mathsf{poly}(T_3)$*-time adversaries, where* T_3 *is the size of the* $\mathsf{NICommit}$ *commitment scheme extractor,*

$$|\Pr[\mathcal{D}[\mathsf{Hybrid}_7] = 1] - \Pr[\mathcal{D}[\mathsf{Hybrid}_8] = 1]| \leq \mathsf{negl}(\lambda).$$

Proof. Assume for the sake of contradiction that there exists an adversary $(\mathcal{A}, \mathcal{D})$ which distinguishes between the two hybrids with non-negligible probability. We build a reduction $(\mathcal{R}, \mathcal{D})$ to the semi-malicious security of the underlying semi-malicious MrNISC protocol. $\mathcal{R}$ runs in time $p(T_3)$, and behaves as follows. First, $\mathcal{R}$ is initialized with 1^λ and z; it then invokes $\mathcal{A}$ with the same 1^λ and z. When $\mathcal{A}$ chooses M and H, $\mathcal{R}$ forwards these to the challenger. $\mathcal{R}$ then interacts with $\mathcal{A}$ and the challenger as follows:

1. Whenever $\mathcal{A}$ submits an Honest Input Encoding query asking for honest party P_i's input encoding with respect to input x_i, $\mathcal{R}$ sends the same Honest Input Encoding

query for P_i to the semimalicious challenger. It then uses the response $\hat{m}_{i,1}$ when computing P_i's input encoding for $\mathcal{A}$.

2. Whenever $\mathcal{A}$ submits a Corrupt Input Encoding query on behalf of P_j, $j \in \mathcal{C}$, $\mathcal{R}$ extracts com_j to obtain $(\tilde{x}_j, \tilde{r}_{j,\mathsf{SM},1}, \tilde{K}_j)$. If P_j's $\hat{m}_{j,1}$ is honestly generated, then $\mathcal{R}$ submits $\hat{m}_{j,1}$ to the challenger as P_j's message, along with the explanation $(j, \tilde{x}_j, r_{j,\mathsf{SM},1})$. Otherwise, $\mathcal{R}$ submits $(j, \perp)$.

3. Whenever $\mathcal{A}$ submits an Honest Computation Encoding query asking for honest party P_i's encoding w.r.t f and I, if the equalities checked in Hybrid_1 hold, $\mathcal{R}$ sends the same Honest Computation Encoding query to the challenger. It uses the (semi-malicious) response $\hat{m}_{i,2}$ when constructing P_i's (malicious) response to $\mathcal{A}$'s query. If the checks do not hold, $\mathcal{R}$ responds to $\mathcal{A}$ without querying the challenger.

4. Whenever $\mathcal{A}$ submits a Corrupt Computation Encoding on behalf of corrupted party P_j w.r.t. f and I, if $\mathcal{R}$ already submitted $(j, \perp)$ as P_j's input encoding, then $\mathcal{R}$ submits the query $(j, f, I, \perp)$. Otherwise, $\mathcal{R}$ performs the "special abort" check (steps 1 to 3 in the simulator description) and outputs the special abort symbol $\perp^*$ if the check fails. If the check passes, $\mathcal{R}$ checks that

   (a) All ZK2 messages sent by P_j as part of its computation encoding verify correctly.

   (b) P_j's WE decrypts correctly. ($\mathcal{R}$ can do this by generating computation encodings "in the head" for any honest parties P_i who have not already sent their computation encodings.)

   If so, $\mathcal{R}$ forwards $\hat{m}_{j,2}$ along with the witness $(\tilde{x}_j, \tilde{r}_{j,\mathsf{SM},1}, \tilde{K}_j)$ to the challenger. Otherwise, $\mathcal{R}$ again submits the query $(j, f, I, \perp)$.

5. At the end of the experiment, $\mathcal{R}$ outputs the output of $\mathcal{A}$.

If the challenger enacts the real-world experiment for the semi-malicious protocol, then the output of $\mathcal{R}$, the transcript τ of queries made by $\mathcal{A}$ along with $\mathcal{R}$'s responses, and the

list $\mathsf{honest\_outputs}$ are identical to the view of $\mathcal{A}$ along with τ in the output of Hybrid_7. If the challenger enacts the ideal-world game, then the output of $\mathcal{R}$, τ, and $\mathsf{honest\_outputs}$ are identical to $\mathcal{A}$'s view and τ in the output of Hybrid_8. Thus by assumption we have a distinguisher $(\mathcal{R}, \mathcal{D})$ which contradicts security of the semi-malicious MrNISC against adversaries running in time $\mathsf{poly}(T_3)$. □

[ABG21] Amit Agarwal, James Bartusek, Vipul Goyal, Dakshita Khurana, and Giulio Malavolta. "Two-Round Maliciously Secure Computation with Super-Polynomial Simulation." In *Theory of Cryptography - 19th International Conference, TCC*, pp. 654–685, 2021.

[ACJ17] Prabhanjan Ananth, Arka Rai Choudhuri, and Abhishek Jain. "A New Approach to Round-Optimal Secure Multiparty Computation." In Jonathan Katz and Hovav Shacham, editors, *CRYPTO 2017, Part I*, volume 10401 of *LNCS*, pp. 468–499. Springer, Heidelberg, August 2017.

[AH87] William Aiello and Johan Håstad. "Perfect Zero-Knowledge Languages Can Be Recognized in Two Rounds." In *28th FOCS*, pp. 439–448. IEEE Computer Society Press, October 1987.

[AJJ21] Prabhanjan Ananth, Abhishek Jain, Zhengzhong Jin, and Giulio Malavolta. "Unbounded Multi-party Computation from Learning with Errors." In Anne Canteaut and François-Xavier Standaert, editors, *EUROCRYPT 2021, Part II*, volume 12697 of *LNCS*, pp. 754–781. Springer, Heidelberg, October 2021.

[AJL12] Gilad Asharov, Abhishek Jain, Adriana López-Alt, Eran Tromer, Vinod Vaikuntanathan, and Daniel Wichs. "Multiparty Computation with Low Communication, Computation and Interaction via Threshold FHE." In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pp. 483–501. Springer, Heidelberg, April 2012.

[AJW11] Gilad Asharov, Abhishek Jain, and Daniel Wichs. "Multiparty Computation with Low Communication, Computation and Interaction via Threshold FHE." Cryptology ePrint Archive, Report 2011/613, 2011. https://eprint.iacr.org/2011/613.

[Bar02] Boaz Barak. "Constant-Round Coin-Tossing with a Man in the Middle or Realizing the Shared Random String Model." In *43rd FOCS*, pp. 345–355. IEEE Computer Society Press, November 2002.

[BCC13] Nir Bitansky, Ran Canetti, Alessandro Chiesa, and Eran Tromer. "Recursive composition and bootstrapping for SNARKS and proof-carrying data." In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *45th ACM STOC*, pp. 111–120. ACM Press, June 2013.

[BCG14] Eli Ben-Sasson, Alessandro Chiesa, Christina Garman, Matthew Green, Ian Miers, Eran Tromer, and Madars Virza. "Zerocash: Decentralized Anonymous Payments from Bitcoin." In *2014 IEEE Symposium on Security and Privacy, SP 2014, Berkeley, CA, USA, May 18-21, 2014*, pp. 459–474. IEEE Computer Society, 2014.

[BD18] Zvika Brakerski and Nico Döttling. "Two-Message Statistically Sender-Private OT from LWE." In Amos Beimel and Stefan Dziembowski, editors, *TCC 2018, Part II*, volume 11240 of *LNCS*, pp. 370–390. Springer, Heidelberg, November 2018.

[BDR18] Itay Berman, Akshay Degwekar, Ron D. Rothblum, and Prashant Nalini Vasudevan. "Multi-Collision Resistant Hash Functions and Their Applications." In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part II*, volume 10821 of *LNCS*, pp. 133–161. Springer, Heidelberg, April / May 2018.

[BFJ20] Saikrishna Badrinarayanan, Rex Fernando, Aayush Jain, Dakshita Khurana, and Amit Sahai. "Statistical ZAP Arguments." In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part III*, volume 12107 of *LNCS*, pp. 642–667. Springer, Heidelberg, May 2020.

[BGI01] Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil P. Vadhan, and Ke Yang. "On the (Im)possibility of Obfuscating Programs." In Joe Kilian, editor, *Advances in Cryptology - CRYPTO 2001, 21st Annual International Cryptology Conference, Santa Barbara, California, USA, August 19-23, 2001, Proceedings*, volume 2139 of *Lecture Notes in Computer Science*, pp. 1–18. Springer, 2001.

[BGI12] Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil P. Vadhan, and Ke Yang. "On the (im)possibility of obfuscating programs." *J. ACM*, **59**(2):6:1–6:48, 2012.

[BGJ16] Nir Bitansky, Shafi Goldwasser, Abhishek Jain, Omer Paneth, Vinod Vaikuntanathan, and Brent Waters. "Time-Lock Puzzles from Randomized Encodings." In Madhu Sudan, editor, *ITCS 2016*, pp. 345–356. ACM, January 2016.

[BGJ17] Saikrishna Badrinarayanan, Vipul Goyal, Abhishek Jain, Dakshita Khurana, and Amit Sahai. "Round Optimal Concurrent MPC via Strong Simulation." In Yael Kalai and Leonid Reyzin, editors, *TCC 2017, Part I*, volume 10677 of *LNCS*, pp. 743–775. Springer, Heidelberg, November 2017.

[BGM20] James Bartusek, Sanjam Garg, Daniel Masny, and Pratyay Mukherjee. "Reusable Two-Round MPC from DDH." In Rafael Pass and Krzysztof Pietrzak, editors, *TCC 2020, Part II*, volume 12551 of *LNCS*, pp. 320–348. Springer, Heidelberg, November 2020.

[BGS21] James Bartusek, Sanjam Garg, Akshayaram Srinivasan, and Yinuo Zhang. "Reusable Two-Round MPC from LPN." *IACR Cryptol. ePrint Arch.*, p. 316, 2021.

[BHP17] Zvika Brakerski, Shai Halevi, and Antigoni Polychroniadou. "Four Round Secure Computation Without Setup." In Yael Kalai and Leonid Reyzin, editors, *TCC 2017, Part I*, volume 10677 of *LNCS*, pp. 645–677. Springer, Heidelberg, November 2017.

[BHZ87] R. B. Boppana, J. Hastad, and S. Zachos. "Does Co-NP Have Short Interactive Proofs?" *Inf. Process. Lett.*, **25**(2):127–132, may 1987.

[BJK21] Fabrice Benhamouda, Aayush Jain, Ilan Komargodski, and Huijia Lin. "Multiparty Reusable Non-interactive Secure Computation from LWE." In Anne Canteaut and François-Xavier Standaert, editors, *EUROCRYPT 2021, Part II*, volume 12697 of *LNCS*, pp. 724–753. Springer, Heidelberg, October 2021.

[BKP18a] Nir Bitansky, Yael Tauman Kalai, and Omer Paneth. "Multi-collision resistance: a paradigm for keyless hash functions." In Ilias Diakonikolas, David Kempe, and Monika Henzinger, editors, *50th ACM STOC*, pp. 671–684. ACM Press, June 2018.

[BKP18b] Nir Bitansky, Yael Tauman Kalai, and Omer Paneth. "Multi-collision resistance: a paradigm for keyless hash functions." In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC*, pp. 671–684. ACM, 2018.

[BL18] Nir Bitansky and Huijia Lin. "One-Message Zero Knowledge and Non-malleable Commitments." In *Theory of Cryptography - 16th International Conference, TCC*, pp. 209–234, 2018.

[BL20] Fabrice Benhamouda and Huijia Lin. "Mr NISC: Multiparty Reusable Non-Interactive Secure Computation." In Rafael Pass and Krzysztof Pietrzak, editors, *TCC 2020, Part II*, volume 12551 of *LNCS*, pp. 349–378. Springer, Heidelberg, November 2020.

[Blu81] Manuel Blum. "Coin Flipping by Telephone." In Allen Gersho, editor, *Advances in Cryptology: A Report on CRYPTO 81, CRYPTO 81, IEEE Workshop on Communications Security, Santa Barbara, California, USA, August 24-26, 1981*, pp. 11–15. U. C. Santa Barbara, Dept. of Elec. and Computer Eng., ECE Report No 82-04, 1981.

[BMR90] Donald Beaver, Silvio Micali, and Phillip Rogaway. "The Round Complexity of Secure Protocols (Extended Abstract)." In *22nd ACM STOC*, pp. 503–513. ACM Press, May 1990.

[BN00] Dan Boneh and Moni Naor. "Timed Commitments." In *Advances in Cryptology - CRYPTO*, pp. 236–254, 2000.

[BPS06] Boaz Barak, Manoj Prabhakaran, and Amit Sahai. "Concurrent Non-Malleable Zero Knowledge." In *47th FOCS*, pp. 345–354. IEEE Computer Society Press, October 2006.

[CCG20] Arka Rai Choudhuri, Michele Ciampi, Vipul Goyal, Abhishek Jain, and Rafail Ostrovsky. "Round Optimal Secure Multiparty Computation from Minimal Assumptions." In Rafael Pass and Krzysztof Pietrzak, editors, *TCC 2020, Part II*, volume 12551 of *LNCS*, pp. 291–319. Springer, Heidelberg, November 2020.

[CCH19] Ran Canetti, Yilei Chen, Justin Holmgren, Alex Lombardi, Guy N. Rothblum, Ron D. Rothblum, and Daniel Wichs. "Fiat-Shamir: from practice to theory." In Moses Charikar and Edith Cohen, editors, *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, Phoenix, AZ, USA, June 23-26, 2019*, pp. 1082–1090. ACM, 2019.

[CLP10] Ran Canetti, Huijia Lin, and Rafael Pass. "Adaptive Hardness and Composable Security in the Plain Model from Standard Assumptions." In *51st FOCS*, pp. 541–550. IEEE Computer Society Press, October 2010.

[COS16] Michele Ciampi, Rafail Ostrovsky, Luisa Siniscalchi, and Ivan Visconti. "Concurrent Non-Malleable Commitments (and More) in 3 Rounds." In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part III*, volume 9816 of *LNCS*, pp. 270–299. Springer, Heidelberg, August 2016.

[COS17a] Michele Ciampi, Rafail Ostrovsky, Luisa Siniscalchi, and Ivan Visconti. "Four-Round Concurrent Non-Malleable Commitments from One-Way Functions." In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology - CRYPTO 2017 - 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part II*, volume 10402 of *Lecture Notes in Computer Science*, pp. 127–157. Springer, 2017.

[COS17b] Michele Ciampi, Rafail Ostrovsky, Luisa Siniscalchi, and Ivan Visconti. "Round-Optimal Secure Two-Party Computation from Trapdoor Permutations." In Yael Kalai and Leonid Reyzin, editors, *TCC 2017, Part I*, volume 10677 of *LNCS*, pp. 678–710. Springer, Heidelberg, November 2017.

[DDN91] Danny Dolev, Cynthia Dwork, and Moni Naor. "Non-Malleable Cryptography (Extended Abstract)." In Cris Koutsougeras and Jeffrey Scott Vitter, editors, *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing, May 5-8, 1991, New Orleans, Louisiana, USA*, pp. 542–552. ACM, 1991.

[DJM12] Yevgeniy Dodis, Abhishek Jain, Tal Moran, and Daniel Wichs. "Counterexamples to Hardness Amplification beyond Negligible." In *Theory of Cryptography - 9th Theory of Cryptography Conference, TCC*, pp. 476–493. Springer, 2012.

[DKP21] Dana Dachman-Soled, Ilan Komargodski, and Rafael Pass. "Non-malleable Codes for Bounded Parallel-Time Tampering." In *Advances in Cryptology - CRYPTO*, pp. 535–565, 2021.

[EFK20] Naomi Ephraim, Cody Freitag, Ilan Komargodski, and Rafael Pass. "Non-Malleable Time-Lock Puzzles and Applications." *IACR Cryptol. ePrint Arch.*, p. 779, 2020.

[FGK22] Rex Fernando, Yuval Gelles, Ilan Komargodski, and Elaine Shi. "Maliciously Secure Massively Parallel Computation for All-but-One Corruptions." In *CRYPTO 2022*, Lecture Notes in Computer Science, 2022.

[For87] Lance Fortnow. "The Complexity of Perfect Zero-Knowledge (Extended Abstract)." In Alfred Aho, editor, *19th ACM STOC*, pp. 204–209. ACM Press, May 1987.

[GG98] Oded Goldreich and Shafi Goldwasser. "On the Limits of Non-Approximability of Lattice Problems." In *30th ACM STOC*, pp. 1–9. ACM Press, May 1998.

[GGJ12] Sanjam Garg, Vipul Goyal, Abhishek Jain, and Amit Sahai. "Concurrently Secure Computation in Constant Rounds." In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pp. 99–116. Springer, Heidelberg, April 2012.

[GGS13] Sanjam Garg, Craig Gentry, Amit Sahai, and Brent Waters. "Witness encryption and its applications." In *Symposium on Theory of Computing Conference, STOC*, pp. 467–476, 2013.

[GJJ20] Vipul Goyal, Abhishek Jain, Zhengzhong Jin, and Giulio Malavolta. "Statistical Zaps and New Oblivious Transfer Protocols." In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part III*, volume 12107 of *LNCS*, pp. 668–699. Springer, Heidelberg, May 2020.

[GK90] Oded Goldreich and Eyal Kushilevitz. "A Perfect Zero-Knowledge Proof for a Problem Equivalent to Discrete Logarithm." In Shafi Goldwasser, editor, *CRYPTO'88*, volume 403 of *LNCS*, pp. 57–70. Springer, Heidelberg, August 1990.

[GKL21] Rachit Garg, Dakshita Khurana, George Lu, and Brent Waters. "Black-Box Non-interactive Non-malleable Commitments." In Anne Canteaut and François-Xavier Standaert, editors, *Advances in Cryptology - EUROCRYPT 2021 - 40th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, October 17-21, 2021, Proceedings, Part III*, volume 12698 of *Lecture Notes in Computer Science*, pp. 159–185. Springer, 2021.

[GL89] Oded Goldreich and Leonid A. Levin. "A Hard-Core Predicate for all One-Way Functions." In David S. Johnson, editor, *Proceedings of the 21st Annual ACM*

Symposium on Theory of Computing, May 14-17, 1989, Seattle, Washington, USA, pp. 25–32. ACM, 1989.

[GLO12] Vipul Goyal, Chen-Kuei Lee, Rafail Ostrovsky, and Ivan Visconti. "Constructing Non-malleable Commitments: A Black-Box Approach." In *53rd FOCS*, pp. 51–60. IEEE Computer Society Press, October 2012.

[GMP16] Sanjam Garg, Pratyay Mukherjee, Omkant Pandey, and Antigoni Polychroniadou. "The Exact Round Complexity of Secure Computation." In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part II*, volume 9666 of *LNCS*, pp. 448–476. Springer, Heidelberg, May 2016.

[GMR85] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. "The Knowledge Complexity of Interactive Proof-Systems (Extended Abstract)." In *17th ACM STOC*, pp. 291–304. ACM Press, May 1985.

[GMW87] Oded Goldreich, Silvio Micali, and Avi Wigderson. "How to Play any Mental Game or A Completeness Theorem for Protocols with Honest Majority." In Alfred Aho, editor, *19th ACM STOC*, pp. 218–229. ACM Press, May 1987.

[GO94] Oded Goldreich and Yair Oren. "Definitions and Properties of Zero-Knowledge Proof Systems." *J. Cryptology*, **7**:1–32, 1994.

[Goy11] Vipul Goyal. "Constant round non-malleable protocols using one way functions." In Lance Fortnow and Salil P. Vadhan, editors, *43rd ACM STOC*, pp. 695–704. ACM Press, June 2011.

[JLS21a] Aayush Jain, Huijia Lin, and Amit Sahai. "Indistinguishability Obfuscation from LPN over F_p, DLIN, and PRGs in NCˆ0." *IACR Cryptol. ePrint Arch.*, p. 1334, 2021.

[JLS21b] Aayush Jain, Huijia Lin, and Amit Sahai. "Indistinguishability obfuscation from well-founded assumptions." In *STOC*, pp. 60–73, 2021.

[Kan90] Sampath Kumar Kannan. *Program Checkers for Algebraic Problems**. PhD thesis, 1990. AAI9103745.

[Khu17] Dakshita Khurana. "Round Optimal Concurrent Non-malleability from Polynomial Hardness." In Yael Kalai and Leonid Reyzin, editors, *TCC 2017, Part II*, volume 10678 of *LNCS*, pp. 139–171. Springer, Heidelberg, November 2017.

[Khu21] Dakshita Khurana. "Non-interactive Distributional Indistinguishability (NIDI) and Non-malleable Commitments." In Anne Canteaut and François-Xavier Standaert, editors, *EUROCRYPT 2021, Part III*, volume 12698 of *LNCS*, pp. 186–215. Springer, Heidelberg, October 2021.

[KK19] Yael Tauman Kalai and Dakshita Khurana. "Non-interactive Non-malleability from Quantum Supremacy." In *Advances in Cryptology - CRYPTO*, pp. 552–582, 2019.

[KKS18] Yael Tauman Kalai, Dakshita Khurana, and Amit Sahai. "Statistical Witness Indistinguishability (and more) in Two Messages." In *Advances in Cryptology - EUROCRYPT*, pp. 34–65, 2018.

[KMO14] Susumu Kiyoshima, Yoshifumi Manabe, and Tatsuaki Okamoto. "Constant-Round Black-Box Construction of Composable Multi-Party Computation Protocol." In Yehuda Lindell, editor, *TCC 2014*, volume 8349 of *LNCS*, pp. 343–367. Springer, Heidelberg, February 2014.

[KO04] Jonathan Katz and Rafail Ostrovsky. "Round-Optimal Secure Two-Party Computation." In Matthew Franklin, editor, *CRYPTO 2004*, volume 3152 of *LNCS*, pp. 335–354. Springer, Heidelberg, August 2004.

[KOS03] Jonathan Katz, Rafail Ostrovsky, and Adam Smith. "Round Efficiency of Multi-party Computation with a Dishonest Majority." In Eli Biham, editor, *EUROCRYPT 2003*, volume 2656 of *LNCS*, pp. 578–595. Springer, Heidelberg, May 2003.

[KS17] Dakshita Khurana and Amit Sahai. "How to Achieve Non-Malleability in One or Two Rounds." In Chris Umans, editor, *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pp. 564–575. IEEE Computer Society, 2017.

[LP11] Huijia Lin and Rafael Pass. "Constant-round non-malleable commitments from any one-way function." In Lance Fortnow and Salil P. Vadhan, editors, *43rd ACM STOC*, pp. 705–714. ACM Press, June 2011.

[LPS17] Huijia Lin, Rafael Pass, and Pratik Soni. "Two-Round and Non-Interactive Concurrent Non-Malleable Commitments from Time-Lock Puzzles." In Chris Umans, editor, *58th FOCS*, pp. 576–587. IEEE Computer Society Press, October 2017.

[LPS20] Huijia Lin, Rafael Pass, and Pratik Soni. "Two-Round and Non-Interactive Concurrent Non-Malleable Commitments from Time-Lock Puzzles." *SIAM J. Comput.*, **49**(4), 2020.

[LPV08] Huijia Lin, Rafael Pass, and Muthuramakrishnan Venkitasubramaniam. "Concurrent Non-malleable Commitments from Any One-Way Function." In Ran Canetti, editor, *Theory of Cryptography, Fifth Theory of Cryptography Conference, TCC 2008, New York, USA, March 19-21, 2008*, volume 4948 of *Lecture Notes in Computer Science*, pp. 571–588. Springer, 2008.

[LPV09] Huijia Lin, Rafael Pass, and Muthuramakrishnan Venkitasubramaniam. "A unified framework for concurrent security: universal composability from stand-alone non-malleability." In Michael Mitzenmacher, editor, *41st ACM STOC*, pp. 179–188. ACM Press, May / June 2009.

[Pas03] Rafael Pass. "Simulation in Quasi-Polynomial Time, and Its Application to Protocol Composition." In Eli Biham, editor, *EUROCRYPT 2003*, volume 2656 of *LNCS*, pp. 160–176. Springer, Heidelberg, May 2003.

[Pas13] Rafael Pass. "Unprovable Security of Perfect NIZK and Non-interactive Non-malleable Commitments." In Amit Sahai, editor, *Theory of Cryptography - 10th Theory of Cryptography Conference, TCC 2013, Tokyo, Japan, March 3-6, 2013. Proceedings*, volume 7785 of *Lecture Notes in Computer Science*, pp. 334–354. Springer, 2013.

[PPV08] Omkant Pandey, Rafael Pass, and Vinod Vaikuntanathan. "Adaptive One-Way Functions and Applications." In *Advances in Cryptology - CRYPTO*, pp. 57–74, 2008.

[PR03] Rafael Pass and Alon Rosen. "Bounded-Concurrent Secure Two-Party Computation in a Constant Number of Rounds." In *44th FOCS*, pp. 404–415. IEEE Computer Society Press, October 2003.

[PR05a] Rafael Pass and Alon Rosen. "Concurrent Non-Malleable Commitments." In *46th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2005), 23-25 October 2005, Pittsburgh, PA, USA, Proceedings*, pp. 563–572. IEEE Computer Society, 2005.

[PR05b] Rafael Pass and Alon Rosen. "New and improved constructions of non-malleable cryptographic protocols." In Harold N. Gabow and Ronald Fagin, editors, *37th ACM STOC*, pp. 533–542. ACM Press, May 2005.

[PS19a] Chris Peikert and Sina Shiehian. "Noninteractive Zero Knowledge for NP from (Plain) Learning with Errors." In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019*, 2019.

[PS19b] Chris Peikert and Sina Shiehian. "Noninteractive Zero Knowledge for NP from (Plain) Learning with Errors." In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part I*, volume 11692 of *LNCS*, pp. 89–114. Springer, Heidelberg, August 2019.

[PW10] Rafael Pass and Hoeteck Wee. "Constant-Round Non-malleable Commitments from Sub-exponential One-Way Functions." In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pp. 638–655. Springer, Heidelberg, May / June 2010.

[RSW96] R. L. Rivest, A. Shamir, and D. A. Wagner. "Time-lock puzzles and timed-release crypto.", 1996. Technical report, Massachusetts Institute of Technology, Cambridge, MA, USA.

[SV97] Amit Sahai and Salil P. Vadhan. "A Complete Promise Problem for Statistical Zero-Knowledge." In *38th FOCS*, pp. 448–457. IEEE Computer Society Press, October 1997.

[Wee10] Hoeteck Wee. "Black-Box, Round-Efficient Secure Computation via Non-malleability Amplification." In *51st FOCS*, pp. 531–540. IEEE Computer Society Press, October 2010.

[Yao86] Andrew Chi-Chih Yao. "How to Generate and Exchange Secrets (Extended Abstract)." In *27th FOCS*, pp. 162–167. IEEE Computer Society Press, October 1986.