Maliciously-Secure MrNISC in the Plain Model

Rex Fernando* Aayush Jain† Ilan Komargodski‡

November 7, 2022

Abstract

We study strong versions of round-optimal MPC. A recent work of Benhamouda and Lin (TCC '20) identified a version of secure multiparty computation (MPC), termed *Multiparty reusable Non-Interactive Secure Computation* (MrNISC), that combines at the same time several fundamental aspects of secure computation with standard simulation security into one primitive: round-optimality, succinctness, concurrency, and adaptivity. In more detail, MrNISC is essentially a two-round MPC protocol where the first round of messages serves as a reusable commitment to the private inputs of participating parties. Using these commitments, any subset of parties can later compute any function of their choice on their respective inputs by broadcasting one message each. Anyone who sees these parties' commitments and evaluation messages (even an outside observer) can learn the function output and nothing else. Importantly, the input commitments can be computed without knowing anything about other participating parties (neither their identities nor their number) and they are reusable across any number of computations.

By now, there are several known MrNISC protocols from either (bilinear) group-based assumptions or from LWE. They all satisfy semi-malicious security (in the plain model) and require trusted setup assumptions in order to get malicious security. We are interested in *maliciously* secure MrNISC protocols *in the plain model, without trusted setup*. Since the standard notion of polynomial simulation is un-achievable in less than four rounds, we focus on security with *super-polynomial*-time simulation (SPS).

Our main result is the first maliciously secure SPS MrNISC in the plain model. The result is obtained by generically compiling any semi-malicious MrNISC and the security of our compiler relies on several well-studied assumptions of an indistinguishability obfuscator, DDH over $\mathbb{Z}_p^*$ and asymmetric pairing groups, and a time-lock puzzle (all of which need to be sub-exponentially hard). As a special case, we obtain the first 2-round maliciously secure SPS MPC based on well-founded assumptions. This MPC is also concurrently self-composable and its first message is short (i.e., its size is independent of the number of the participating parties) and reusable throughout any number of computations. Prior to our work, for two round maliciously secure MPC, neither concurrent MPC nor reusable MPC nor MPC with first message independent in the number of parties was known from any set of assumptions. Of independent interest is one of our building blocks: the first construction of a one-round non-malleable commitment scheme from well-studied assumptions, avoiding keyless hash functions and non-standard hardness amplification assumptions.

*Carnegie Mellon University. Email: rex1fernando@gmail.com. Supported in part by a Simons Investigator Award, DARPA SIEVE award, NTT Research, NSF Frontier Award 1413955, BSF grant 2012378, a Xerox Faculty Research Award, a Google Faculty Research Award, and an Okawa Foundation Research Grant. This material is based upon work supported by the Defense Advanced Research Projects Agency through Award HR00112020024.

†Carnegie Mellon University. Email: aayushjain1728@gmail.com.

‡Hebrew University of Jerusalem and NTT Research. Email: ilank@cs.huji.ac.il. Incumbent of the Harry & Abe Sherman Senior Lectureship at the School of Computer Science and Engineering at the Hebrew University, supported in part by an Alon Young Faculty Fellowship, by a grant from the Israel Science Foundation (ISF Grant No. 1774/20), and by a grant from the US-Israel Binational Science Foundation and the US National Science Foundation (BSF-NSF Grant No. 2020643).

Contents

1 Introduction

In this work, we study the round complexity of cryptographic protocols, giving special attention to secure multi-party computation (MPC). MPC 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 [GMW87, BGW88, CCD88].

Round complexity is a fundamental measure of both the efficiency and power of cryptographic protocols. 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. 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 [BCCT13], and also to achieve new space and communication efficient secure multi-party computation protocols [FGKS22].

The round complexity of MPC protocols in particular 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 [BMR90a, KOS03, KO04, Wee10, GMPP16, ACJ17, BHP17, COSV17, CCG+20] has made dramatic improvements, with recent works finally achieving four rounds [COSV17, CCG+20, ACJ17, BHP17]. This was shown to be optimal by the works of [KO04, GMPP16], 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* (SPS). 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, GGJS12, KMO14], sidestepping the impossibility result of [BPS06]. In 2017, the work of [BGJ+17] constructed a concurrent MPC protocol in three rounds, thus bypassing both the lower bounds of [KO04, GMPP16] 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. A very recent work [ABG+21] partially advanced the state of the art in terms of round complexity, giving a two-round standalone-secure MPC protocol in the plain model. However, their security proof relies on ad-hoc (exponentially strong) assumptions that are novel to their work, and they do not achieve concurrent security.[2]

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, relying on well-studied assumptions.

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

[2] We discuss this work further in Section 1.3.

In this work, we study this question.

MrNISC. Going one step further, 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.[3] Benhamouda and Lin [BL20] constructed such a protocol for all efficiently computable functionalities relying on the DDH assumption in asymmetric bilinear groups. In two concurrent follow-up works, Ananth et al. [AJJM21] and Benhamouda et al. [BJKL21] 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.

1.1 Our Results

In this paper, we give the first affirmative answer to the above question. Specifically, relying on commonly-used, well-established assumptions, we obtain a maliciously secure SPS MrNISC in the plain model, without any trusted setup. In particular, this implies a concurrently secure SPS MPC in two rounds from the same assumptions. We state our (informal) theorem below.

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

Theorem 1 (Main Result, informal)**.** *Assume the existence of a subexponentially-secure indistinguishability obfuscation (iO) scheme, subexponential DDH (over both asymmetric pairing groups*[4] *and $\mathbb{Z}_p^*$), and subexponential time-lock puzzles. Then there exists a malicious-secure MrNISC in the plain model, with a super-polynomial simulator.*

Key ideas. Our result is obtained via a generic compiler which takes any subexponentially-secure semi-malicious secure MrNISC and upgrades it to malicious security. As mentioned above, the work of [BL20] showed that such a semi-malicious-secure MrNISC exists assuming subexponential DDH over asymmetric pairing groups. Our transformation 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.

Implications for (Classical) MPC. As mentioned, it is possible to view an MrNISC as a standard MPC. Specifically, we get the following:

- 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 [BGJ+17].
- 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 [BGMM20, BL20, AJJM21, BGSZ22, BJKL21].
- Our MrNISC implies the first 2-round maliciously secure SPS MPC based on **well-studied, falsifiable assumptions**.

1.1.1 Notable Building Blocks

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 that 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 2 (Informal)**.** *Assume the existence of a subexponentially-secure indistinguishability obfuscation (iO) scheme, subexponential DDH (over both $\mathbb{Z}_p^*$ and assymetric pairing groups), and*

[4] DDH assumption over asymmetric pairing groups is also referred to as the SXDH assumption. We will interchangeably use SXDH wherever we specifically require DDH over assymetric pairing groups.

subexponential time-lock puzzles. Then there exists a reusable statistical ZK argument with sometimes-statistical soundness as defined in Definition 14.

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:

Theorem 3 (Informal)**.** *Assume the existence of a subexponentially-secure indistinguishability obfuscation (iO) scheme, subexponential SXDH, and subexponential time-lock puzzles. 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 assumptions. In particular, their assumption incorporates a strong form of non-malleability into it. The works of [BL18, GKLW21] were able to obtain constructions based on different assumptions, including (among other things) a rather new assumption called *keyless multi-collision-resistant hash functions* [BKP18a]. This assumption, which is described in more detail below, is still somewhat strong as we do not have any instantiation of it besides using cryptographic hash functions. In contrast, our commitment scheme relies solely on well-established assumptions which have a long history of study.

Our construction is based heavily on and improves upon 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 which work in one round, so the construction of [Khu21] will not suffice as-is. We elaborate on this in Section 2 and Section 7.

1.1.2 Putting Things Together

Our compiler makes use of these two new tools in order to upgrade security of a semi-malicious MrNISC scheme. Informally, we achieve the following:

Theorem 4 (The Compiler, Informal)**.** *Assume the existence of subexponential variants of the following:*

- *a reusable two-round statistical zero knowledge argument with sometimes-statistical soundness,*
- *a one-round non-malleable commitment,*
- *a non-interactive perfectly-binding commitment,*
- *a pseudorandom function,*
- *a witness encryption scheme for NP,*
- *and finally, a semi-malicious MrNISC scheme.*

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

1.2 On the Necessity of iO

We make use of an obfuscation scheme when constructing both our zero knowledge scheme as well as our non-malleable commitment scheme. Also, it is directly used to get the witness encryption scheme. We do not know if iO can be avoided in constructing MrNISC in the plain model.

As mentioned above, constructions of one-round non-malleable commitments exist from other assumptions than iO [PPV08, BL18], however these constructions rely on assumptions that are problematic for various reasons. The only known route to avoid these assumptions is via iO [Khu21] but even then previous work failed to achieve one-round protocols.

We now discuss the need in a witness encryption scheme. Intuitively, it seems that some sort of witness encryption for a specific language is required when upgrading security for a semi-malicious MrNISC scheme in the plain model, for the following reason. Since one-round zero knowledge is impossible without setup [GO94], honest parties are forced to send their second-round semi-malicious MrNISC messages without knowing whether the first round is honest. Sending these messages in the clear would violate security, so the parties must somehow send a "locked" version of their second-round such that they are only revealed conditioned on the first round being honest. Since these messages must be publicly unlockable, this means that the second round is some form of witness encryption. We explain this in more detail in Section 2. It is an interesting open question whether it is possible to build a witness encryption scheme for this specific type of statement without relying on iO.

1.3 Related Work

A recent work of Agarwal, Bartusek, Goyal, Khurana, and Malavolta [ABG+21] 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 requires a special type of non-interactive non-malleable commitment. Notably, neither the non-interactive commitments of [BL18, KK19] nor the weakly non-interactive commitments of [Khu21], nor our new one-round non-malleable commitment scheme can be used to instantiate this (because they strongly rely on *exponential* full security and non-interactivity). The authors of [ABG+21] propose two instantiations which work for their construction. One instantiation relies on factoring-based *adaptive* one-way functions [PPV08],[5] a strong assumption that incorporates a strong non-malleability flavor. Another instantiation relies on an exponential variant of the "hardness amplifiability" assumption of [BL18], along with keyless multi-collision resistant hash functions [BKP18b]. Both of these assumptions 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 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

[5] 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 $y' \neq y$.

concrete (contrived) counter examples for this type of assumptions showing that they are generically false [DJMW12], 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 [ABG+21] relying on any well-studied assumptions, or even on assumptions not specifically formulated in order to achieve non-malleable commitments. These drawbacks unfortunately seem inherent in the techniques used by [ABG+21]. Our work uses a completely different approach from their work, and is thus able to achieve a strictly stronger result, without using ad-hoc assumptions.

2 Technical Overview

In this section, we give an overview of our constructions and the main ideas needed to prove their security. We 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, AJJM21, BJKL21]), 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. [AJL+12], 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 4 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*.

2.1 The MrNISC Protocol

To obtain our main result, we will start with a semi-malicious-secure MrNISC protocol [BL20, BJKL21] 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, 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, ABG+21], and/or keyless hash functions [BKP18b, LPS17, 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.

2.1.1 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 [BFJ+20] 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 [BFJ+20] as a starting point, but we will need to make significant modifications. Namely, to force a well-defined perfect soundness mode, we will make

Figure 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.

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 Section 9.1.

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.

2.1.2 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 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, so it seems at first glance that avoiding them would require making substantial progress on the difficult and well-studied question of non-interactive NMCs.

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.
- Second, our goal is to have every party use this commitment to commit to their input and randomness for the protocol. Recall that in the scheme of [Khu21], a well-defined commitment (C_j, τ_i) consists of *both* the committer's message C_j and the receiver's random string τ_i. Although honest parties P_j will always provide commitments C_j which are consistent across all τ_i, it is perfectly plausible for a corrupted party to publish some C_j where different τ_i yield commitments (C_j, τ_i) to different values.

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. Because of this issue, the non-malleable commitment of [Khu21] is really a two round commitment scheme. In this paper, relying on the axis of hardness given by a time-lock puzzle that we additionally use as an assumption, we construct a truly one round non-malleable commitment scheme, in the simultaneous message model. For the second issue, we use a surprisingly simple idea of adding a standard (potentially malleable) perfectly biding commitment scheme (e.g., Blum's commitment) at the MrNISC protocol level, we can use this NMC scheme even though it does not satisfy the standard notion of binding. The overview of the non-malleable commitment scheme can be found in Section 7.1.

2.1.3 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 9.

2.1.4 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, receiver-assisted one-round 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 9, 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 [BGJ+17]. 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 Section 6 for details.

3 Preliminaries

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 $\mathsf{negl} : \mathbb{N} \to \mathbb{R}$ 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 paper, 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.)*

3.1 Indistinguishability Obfuscation

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

Definition 2 (Indistinguishability Obfuscator (iO) for Circuits [BGI+01, BGI+12])**.** *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 [JLS21, JLS22] this can be instantiated assuming subexponential security of well studied hardness assumptions.

3.2 Witness Encryption

Here, we recall the definition of witness encryption, originally due to Garg et al. [GGSW13].

Definition 3. *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 5. *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*.*

3.3 Time Lock Puzzles

We recall the notion of a time-lock puzzle scheme, originally due to [RSW96]. We adapt the definition from [BGJ+16].

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 [LPS17, DKP21, FKPS21]).

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, BGJ+16, LPS17, DKP21, FKPS21]).

Another construction of time-lock puzzles, due to Bitansky et al. [BGJ+16], 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.

3.4 Correlation Intractable Hash Functions

We adapt definitions of a correlation intractable hash function family from [PS19, CCH+19].

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,*

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

- ***Computational Indistinguishability of Hash Keys:*** *Moreover, for every two functions $f_0, f_1 \in \mathcal{F}_{\lambda,s}$, for every $\mathcal{A} \in \mathcal{C}$, and every large enough $\lambda \in \mathbb{N}$,*

$$\left| \Pr_{K \leftarrow \mathcal{H}.\mathsf{FakeGen}(1^\lambda, f_0)} [\mathcal{A}(K) = 1] - \Pr_{K \leftarrow \mathcal{H}.\mathsf{FakeGen}(1^\lambda, f_1)} [\mathcal{A}(K) = 1] \right| < \epsilon(\lambda),$$

The work of [PS19] 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. In this work, we use the construction of [CCH+19] that builds this primitive using a circular-secure FHE. To instantiate a circular secure FHE, we use subexponentially secure iO and a circular secure perfectly rerandomizable encryption [CLTV15]. A circular secure perfectly rerandomizable encryption can be constructed using SXDH [BHHO08].

3.5 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 $(b, r_1, \mathsf{ot}_2, m_0, m_1, r_2)$.*

3.6 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, BMR90b].

4 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 9 ($(\mathcal{C}_{\mathsf{adv}}, \mathcal{C}_{\mathsf{sim}}, \epsilon)$-Maliciously Secure MrNISC)**.** *We say that an MrNISC 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 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 10 ($(\mathcal{C}_{\mathsf{adv}}, \mathcal{C}_{\mathsf{sim}}, \epsilon)$-Semi-Malicious Secure MrNISC)**.** *We say that an MrNISC 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 Main Building Blocks

In this section, we give formal definitions for our new notion of reusable sometimes-statistically-sound zero-knowledge arguments along with the receiver-assisted one-round CCA-secure commitments, both of which we make use of in our MrNISC protocol.

5.1 Reusable Statistical ZK Arguments with Sometimes-Statistical Soundness

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), i.e., the simulator can run longer than the soundness bound. 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 11 (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 12 to 14 below.*

Definition 12 (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 13 ($(\mathcal{C}_{\mathsf{sound}}, \epsilon_{\mathsf{sound},1}, \epsilon_{\mathsf{sound},2})$-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 14 ($(\mathcal{C}_{\mathcal{S}}, \mathcal{C}_{\mathsf{zk}}, \epsilon_{\mathcal{S}})$-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:*
   - *If $(x, w) \notin R_L$, 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 $\mathsf{expt}^0_{V^*,\mathsf{zk}} \approx_{(\infty,\epsilon_{\mathcal{S}})} \mathsf{expt}^1_{V^*,\mathsf{zk}}$.

An overview and complete details of our construction of the reusable SZK argument with sometimes-statistical soundness can be found in Section 9.

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

In the following, we define the syntax and required security properties of the commitment scheme which we construct in Section 7, and which we use in the main MrNISc construction in Section 6. This commitment is 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 the committer's message is some obfuscated program P. The committed value is only fixed when both P and τ are fixed. To reflect this, in the definition of syntax below, $\mathsf{ComputeOpening}$, $\mathsf{VerifyOpening}$, and CCAVal take both the committer's message P and the receiver's message τ as input.

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 15 (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* $\mathsf{VerifyOpening}$ *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 16 (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 17 (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 18 ($(\mathcal{C}, \epsilon)$-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* $\mathcal{A} \in \mathcal{C}$*:*

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

Our NMC construction is an extension of of [Khu21]. It 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 the full CCA security definition given above, relying on iO and other standard assumptions. We refer to Section 7 for details.

6 Malicious-Secure MrNISC

In this section, we give the formal construction and proof of security for our MrNISC protocol.

Required Primitives and Parameters. We make use of the following primitives in our construction.

- *Commitment:* A non-interactive perfectly binding commitment (NICommit).
- *Pseudo-Random Function* A pseudo-random function (PRF).
- *Witness Encryption:* We use witness encryption as in Definition 3. 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 11, 13 and 14 for circuit SAT constructed in Section 9.

- *One-round CCA commitments:* We use one-round (simultaneous-message) CCA commitments as in Definitions 15 to 18.
- *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 10.

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 14) 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 13 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.

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 page 27, 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.

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 defined in Section 4. 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 on page 6. 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 on page 28, 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 on page 27. 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.

6.1 Proof of Security

This section proves that the MrNISC protocol given above satisfies the definition of SPS malicious security from Section 4. Formally, we prove the following theorem:

Theorem 6. *Assume the existence of the following primitives, satisfying the complexity hierarchy above:*

- *A subexponentially-secure non-interactive perfectly binding commitment (*$\mathsf{NICommit}$*).*
- *A subexponentially-secure pseudo-random function (PRF).*

- *A subexponentially-secure witness encryption scheme for NP.*
- *A reusable SPS ZK argument for circuit SAT with sometimes-statistical soundness satisfying Definitions 11, 13 and 14*
- *One-round (simultaneous-message) subexponentially-secure CCA commitments as in Definitions 15 to 18.*
- *A subexponentially-secure semi-malicious MrNISC protocol satisfying the security notion given in Definition 10.*

Then the protocol above is malicious-secure MrNISC in the plain model as defined in Definition 9, with a super-polynomial simulator.

We can instantiate the primitives listed in the theorem in the following way, which satisfy the complexity hierarchy listed above:

- A subexponentially-secure non-interactive perfectly binding commitment ($\mathsf{NICommit}$) can be obtained from subexponential SXDH.
- A subexponentially-secure pseudo-random function (PRF) can be obtained from SXDH.
- A subexponentially-secure witness encryption scheme for NP can be obtained from subexponential indistinguishability obfuscation.
- Following Theorem 10, a SPS ZK argument for circuit SAT satisfying Definitions 11, 13 and 14 can be obtained from subexponential indistinguishability obfuscation, time-lock puzzles, subexponential DDH over $\mathbb{Z}_p^*$ and asymmetric pairing groups.
- Following Theorem 7, one-round (simultaneous-message) subexponentially-secure CCA commitments in Definitions 15 to 18 can be obtained from subexponential indistinguishability obfuscation, time-lock puzzles, and subexponential SXDH.
- The work of [BL20] showed that subexponentially-secure semi-malicious MrNISC protocol satisfying the security notion given in Definition 10 can be constructed from subexponential SXDH.

Using the above instantiations, we get Theorem 1 as a corollary to Theorem 6 above, which we restate now.

Corollary 1. *Assume the existence of a subexponentially-secure indistinguishability obfuscation (iO) scheme, subexponential DDH over both $\mathbb{Z}_p^*$ and asymmetric pairing groups, and time-lock puzzles. Then there exists a malicious-secure MrNISC in the plain model, with a super-polynomial simulator.*

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 4 (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 4, such that the outputs of the corresponding experiments $\mathsf{Expt}_{\mathcal{A}}^{\mathsf{Real}}(\lambda)$ and $\mathsf{Expt}_{\mathcal{A},\mathcal{S}}^{\mathsf{Ideal}}(\lambda)$ are indistinguishable.

6.1.1 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 it holds that $\hat{m}_{j,1} = \mathsf{SM.Encode}(1^\lambda, \tilde{x}_j; \tilde{r}_{j,\mathsf{SM},1})$. 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 6, 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 the 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.

6.1.2 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 4. 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.

6.1.3 Indistinguishability Between Hybrid_0 and Hybrid_1

Claim 1. *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 $i \leq k \in [n] \setminus \mathcal{C}$ 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 1, it is then sufficient to prove the following claim.

Claim 2. *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 2. 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 3. *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 2, 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 4. *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 2, 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 5. *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 2, 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 3 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

$$\left| \Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r}) = 1\right] - \Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r-1}) = 1\right] \right| \geq 1/\mathsf{poly}(\lambda). \tag{1}$$

We can lower-bound the left-hand side of (1) as

$$\Big|\Big(\Pr\big[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r})=1 \mid \mathsf{PS}_{k,j^*}\big]-\Pr\big[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r-1})=1 \mid \mathsf{PS}_{k,j^*}\big]\Big)\cdot\Pr\big[\mathsf{PS}_{k,j^*}\big]+$$
$$\Big(\Pr\big[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r})=1 \mid \overline{\mathsf{PS}}_{k,j^*}\big]-\Pr\big[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r-1})=1 \mid \overline{\mathsf{PS}}_{k,j^*}\big]\Big)\cdot\Pr\big[\overline{\mathsf{PS}}_{k,j^*}\big]\Big|,$$

which by claim Claim 4 is

$$\leq\Big|\Big(\Pr\big[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r})=1 \mid \mathsf{PS}_{k,j^*}\big]-\Pr\big[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r-1})=1 \mid \mathsf{PS}_{k,j^*}\big]\Big)\cdot\big(\Pr\big[\mathsf{PS}_{k,j^*}\big]+\Pr\big[\overline{PS}_{k,j^*}\big]\big)\Big|$$
$$+2\epsilon_{\mathsf{sound},2}\cdot\Pr\big[\overline{\mathsf{PS}}_{k,j^*}\big].$$

(i.e., substitute out $Pr\big[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r})=1 \mid \overline{\mathsf{PS}}_{k,j^*}\big]$ and $\Pr\big[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r-1})=1 \mid \overline{\mathsf{PS}}_{k,j^*}\big]$ for $\Pr\big[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r})=1 \mid \mathsf{PS}_{k,j^*}\big]+\epsilon_{\mathsf{sound},2}$ and $\Pr\big[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r-1})=1 \mid \mathsf{PS}_{k,j^*}\big]+\epsilon_{\mathsf{sound},2}$, respectively.)

Thus,

$$\Big|\Big(\Pr\big[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r})=1 \mid \mathsf{PS}_{k,j^*}\big]-\Pr\big[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r-1})=1 \mid \mathsf{PS}_{k,j^*}\big]\Big)\Big|\geq 1/\mathsf{poly}(\lambda)-2\epsilon_{\mathsf{sound},2},$$

which proves the claim. □

Claim 6. *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,*

$$\Big|\Pr\big[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r})=1\wedge\mathsf{PS}_{k,j^*}\big]-\Pr\big[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r-1})=1\wedge\mathsf{PS}_{k,j^*}\big]\Big|<\epsilon_{\mathsf{WE}}.$$

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,

$$\Big|\Pr\big[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r}(\mathsf{state}))=1\big]-\Pr\big[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r-1}(\mathsf{state}))=1\big]\Big|<\epsilon_{\mathsf{WE}}.$$

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

$$\Big|\Pr\big[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r}(\mathsf{state}_1))=1\big]-\Pr\big[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{0,k,r-1}(\mathsf{state}_1))=1\big]\Big|=0.$$

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 Section 6.

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 2 using the three claims proven above.

Proof of Claim 2. We directly achieve a contradiction by applying Claim 5 and Claim 6, along with the fact that $\epsilon_{\mathsf{sound},1} \gg \epsilon_{\mathsf{WE}}$. □

6.1.4 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 7. *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 7, it is then sufficient to prove the following claim.

Claim 8. *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 8. 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 9. *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 8, 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 10. *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 8, 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 11. *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 8, 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 9 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

$$\left| \Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r}) = 1\right] - \Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r-1}) = 1\right] \right| \geq 1/\mathsf{poly}(\lambda). \tag{2}$$

We can lower-bound the left-hand side of (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 10 is

$$\begin{aligned} &\leq \left| \left(\Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r}) = 1 \mid \mathsf{PS}_{k,j^*}\right] - \Pr\left[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r-1}) = 1 \mid \mathsf{PS}_{k,j^*}\right] \right) \cdot (\Pr\left[\mathsf{PS}_{k,j^*}\right] + \Pr\left[\overline{PS}_{k,j^*}\right]) \right| \\ &+ 2\epsilon_{\mathsf{sound},2} \cdot \Pr\left[\overline{\mathsf{PS}}_{k,j^*}\right]. \end{aligned}$$

(i.e., substitute out $Pr[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r}) = 1 \mid \overline{\mathsf{PS}}_{k,j^*}]$ and $\Pr[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r-1}) = 1 \mid \overline{\mathsf{PS}}_{k,j^*}]$ for $\Pr[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r}) = 1 \mid \mathsf{PS}_{k,j^*}] + \epsilon_{\mathsf{sound},2}$ and $\Pr[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r-1}) = 1 \mid \mathsf{PS}_{k,j^*}] + \epsilon_{\mathsf{sound},2}$, respectively.)

Thus,

$$\left| \left(\Pr[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r}) = 1 \mid \mathsf{PS}_{k,j^*}] - \Pr[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r-1}) = 1 \mid \mathsf{PS}_{k,j^*}] \right) \right| \geq 1/\mathsf{poly}(\lambda) - 2\epsilon_{\mathsf{sound},2},$$

which proves the claim. □

Claim 12. *Assuming the "perfect soundness holds during soundness mode" property of the zero knowledge argument, , it holds that for all k,*

$$\left| \Pr[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r}) = 1 \wedge \mathsf{PS}_{k,j^*}] - \Pr[\mathcal{D}(\widehat{\mathsf{expt}}_{\mathcal{A}}^{1,k,r-1}) = 1 \wedge \mathsf{PS}_{k,j^*}] \right| = 0.$$

Proof. This follows directly from the perfect soundness mode of the ZK argument scheme. □

We now finish the proof of Claim 8 using the three claims proven above.

Proof of Claim 8. We directly achieve a contradiction by applying Claim 11 and Claim 12. □

6.1.5 Proving Indistinguishability of the Remaining Hybrids

Claim 13. *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 14. *Assuming that the CCA non-malleable commitment scheme satisfies $(\mathcal{C},\epsilon)$-CCA security (Definition 18), 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 15. *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 16. *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 17. *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 18. *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 $\mathsf{poly}(T_3)$. □

7 Construction of One-Round CCA-Non-Malleable Commitments

This section is dedicated to constructing one-round simultaneous-message CCA-non-malleable commitments (CCA-NMCs) which satisfy the syntax and security requirements given in Section 5.2. We start with a high-level overview of the construction in Section 7.1, and then in Section 7.2 we give a formal description of the construction along with its security proof.

7.1 A High-Level Overview

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 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.

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 a two round 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 non-interactive scheme supporting say $\underbrace{\log\ldots\log}_{O(1)\text{ times}}\lambda$ tags from [LPS17], based on time-lock puzzles and perfectly-binding non-interactive commitments. (This scheme does not satisfy full CCA-security since it has the problems of over-extraction and same-tag non-malleability, but this notion is enough for their transformation. We explain these two problems and discuss how to address them in the inner commitment later.)

We first explain why Khurana's scheme is a two round scheme. 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,[6] where the $\mathsf{Extract}$ algorithm sometimes outputs a non-$\perp$ value even though the commitment is invalid and has no opening. Over-extraction is a problem in the construction of [Khu21] because it is impossible to test for a well-formed obfuscation based on a polynomial number of queries to the obfuscated program. Because of this, is only possible to achieve a weaker notion of security, called *non-malleability with respect to extraction*, which is insufficient for our purposes.

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 $T' = \underbrace{\log\ldots\log}_{O(1)\text{ times}}\lambda$, 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 discussed above. 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 2, where pk and the PPRF key K_{PPRF} are freshly sampled by the

[6] 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.

committer and hardwired into the program. When evaluated on a random input τ, 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 $\mathsf{OWP}^{-1}(\tau)$. Note that if we are able to make this strategy work, it will also 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.

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

Computation:

1. Compute $r \leftarrow \mathsf{PPRF.Eval}(K_{\mathsf{PPRF}}, \tau)$ 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 $L_G = L_{G_1} \vee L_{G,2}$:

$$L_{G,1} = \{(c_0, c_1, \ldots, c_{T'/2}) \mid \exists x \in \{0,1\}^{\ell_{\mathsf{OWP}}} \; s.t. \; c_0 = \mathsf{Enc}(\mathsf{pk}, x) \wedge \mathsf{OWP}(x) = \tau\}$$
$$L_{G,2} = \{(c_0, c_1, \ldots, c_{T'/2}) \mid \exists m \; s.t. \; \forall i \in [T'/2], \; c_i = \mathsf{CCA'.CCACommit}(t_i, m)\}$$

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

Recall that the strategy for tag amplification given in the beginning of this section 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 τ 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 $\mathsf{OWP}^{-1}(\tau)$, $c_1, \ldots, c_{T'/2}$ must be 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 $\tau \leq \alpha$. 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 τ chosen by the challenger to show that the adversary does not query the CCAVal algorithm on non-well formed commitments.

The security proof of [Khu21] is not able to handle this problem in the general case. Consequently, [Khu21] was forced to restrict the proof of security to only work in the case where all

the receivers' randomnesses $\{\tau\}$ are chosen *after* declaring the set of commitments that would be queried to the CCAVal oracle. In this case, the following clever idea given in [Khu21] offers a simple fix. Since the commitment programs $\{P_i\}$ are fixed by the adversary before receiving anything from the CCAVal oracle, a non-uniform reduction can take as advice the secret-keys $\{\mathsf{sk}_i\}$ associated with the public keys $\{\mathsf{pk}_i\}$ used in the $\{P_i\}$. If a program P_i produces "bad" outputs $(c_0, c_1, \ldots, c_{T'/2}, \pi)$ on a large fraction of points τ, then one can recover inverses of $\mathsf{OWP}^{-1}(\tau)$ using the secret key sk_i provided as advice. This gives a non-uniform reduction to the security of OWP.

There is another reason why the construction of [Khu21] only provides security in this restricted setting. On the one hand, nmc needs to be much more secure than OWP to argue indistinguishability, since the number of hybrids which rely on nmc security is proportional to the domain size of OWP, which is exponential in the security parameter of OWP. On the other hand, in order to prove that adversaries cannot use the trapdoor in the NIWI, we must construct a reduction to OWP which can run $\mathsf{nmc.CCAVal}(\star)$. Again, this problem does not arise if the adversary outputs commitments P_i for which CCAVal is queried before interacting with the CCAVal oracle, since in this case a cheating adversary will have already inverted OWP before the reduction needs to run $\mathsf{nmc.CCAVal}(\star)$. However, in the more general case, this is still an issue.

In order to fix this issue and to achieve full CCA-non-malleable commitments in one round, our main idea is to introduce a new axis of hardness. The tool we use to do this are time-lock puzzles with carefully chosen parameters. Recall that in a time-lock puzzle, one can generate a puzzle/commitment $c_0 = \mathsf{PGen}(t_{\mathsf{TLP}}, \lambda, m)$ for any $t_{\mathsf{TLP}} = \lambda^{\omega(1)}$ where the puzzle has following properties:

- The message m can be recovered by an algorithm Solve that runs in time/depth $d_{\mathsf{TLP},\mathsf{Ext}}$, which is more than $\mathsf{poly}(\lambda_{\mathsf{TLP}})$ but less than $2^{\lambda_t lp}$.
- There exists a constant $0 < \epsilon < 1$, such that any circuit of depth d^ϵ, but with size even as large as $2^{\lambda_{\mathsf{TLP}}}$, the advantage in distinguishing a puzzle/commitment of m_0 from m_1 is bounded by $2^{-\lambda}$.

We will change c_0 to be a time-lock puzzle with parameters $d_{\mathsf{TLP},\mathsf{Ext}}$ and λ_{TLP} chosen carefully. In particular, we set $d_{\mathsf{TLP},\mathsf{Ext}} \ll 2^\ell \ll 2^{\lambda_{\mathsf{TLP}}}$, where ℓ is the input size and security parameter for the one-way permutation. As described before, the program P on input τ, will produce $(c_0, c_1, \ldots, c_{T'/2}, \pi)$ where π will prove that either c_0 is a TLP commitment of $\mathsf{OWP}^{-1}(\tau)$ or $c_1, \ldots, c_{T'/2}$ are all well-formed commitments of m. In contrast to before, now it is possible to have one reduction, which runs in time and depth $\mathsf{poly}(d_{\mathsf{TLP}})$, in which c_0 is broken yet the one-way permutation is secure, and to have another reduction, which runs in depth $d^\epsilon_{\mathsf{TLP}}$, under which the advantage in distinguishing the value committed to by c_0 is $2^{-\lambda_{\mathsf{TLP}}}$, which is much less than $2^{-\ell}$.

We have fixed one part of the problem, however we still need to deal with the issue regarding the inner commitment nmc. Again, we need two different reductions with seemingly contradictory requirements. The first reduction should be able to run $\mathsf{nmc.CCAVal}$ to extract nmc commitments, and the one-way permutation should be secure for this reduction. The second reduction plays the CCA non-malleability game with respect to the nmc, and should have advantage much less than $2^{-\ell}$ in distinguishing.

Ideally, in order to achieve these two different reductions, we would like nmc to have properties similar to a time-lock puzzle. That is, we would like that $\mathsf{nmc.CCAVal}$ can be run in depth d_{TLP}, whereas for any machine playing the CCA game which runs in depth less than $d^\epsilon_{\mathsf{TLP}}$ and up to $2^{\lambda_{\mathsf{nmc}}}$ size, the distinguishing advantage is much less than $2^{-\ell}$. Unfortunately, we do not know of any base CCA-non-malleable nmc scheme which satisfies these properties. However, consider the

following simple black-box modification of a commitment scheme nmc. We take such a scheme nmc (which can be instantiated with, e.g., [LPS17]) and a time-lock puzzle and produce a new nmc' which works as follows. Let d_{CCAVal} and T_{CCAVal} be the depth and size required to run $\mathsf{nmc}.\mathsf{CCAVal}$. A commitment to a message m with respect to nmc' consists of a tlp-based commitment c' to m, along with a commitment P under nmc to m along with the randomness r used to generate the tlp-based commitment c'. The CCA commitment nmc is instantiated with security parameter λ, and the tlp-based commitment is instantiated so that it is extractable in depth $d_{\mathsf{TLP}} \gg d_{\mathsf{CCAVal}}$, but machines running in depth $\mathsf{poly}(d_{\mathsf{CCAVal}})$ and size $\mathsf{poly}(T_{\mathsf{CCAVal}})$ have advantage at most $2^{-\lambda}$ in distinguishing. It is not hard to show that nmc' is CCA-secure with advantage $2^{-\lambda}$. In addition, we now have a new "weak" extraction algorithm $\mathsf{nmc}'.\widetilde{\mathsf{CCAVal}}$, which runs in depth d_{TLP}; $\mathsf{nmc}'.\widetilde{\mathsf{CCAVal}}$ simply extracts the TLP-based commitment to m. For any commitment (c', P) which is honestly generated, the extracted values from $\mathsf{nmc}'.\mathsf{CCAVal}$ and $\mathsf{nmc}'.\widetilde{\mathsf{CCAVal}}$ are equal. Unfortunately, for commitments which are not honestly generated, no such guarantee exists.[7]

The bulk of our technical contributions in this section involves constructing (1) a complexity hierarchy and (2) a careful sequence of hybrids, which allows us to use an inner commitment with this weak version of extraction in order to achieve security. With these two pieces, we are able to use the (perfect) soundness of the NIWI in order to avoid using $\widetilde{\mathsf{CCAVal}}$ on dishonestly-generated inner commitments. Full details are in the next section.

Technical Remarks. Before we describe our construction, we mention a technical issue. The underlying non-malleable commitments such as [LPS17, 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 7.3.

7.2 The Construction and Security Proof

In this section, we prove the following theorem.

Theorem 7. *Assume the existence of a subexponentially-secure indistinguishability obfuscation (iO) scheme, subexponential SXDH, and subexponential time-lock puzzles. Then, there exists a subexponentially-secure one-round CCA commitment scheme supporting a super-polynomial number of tags.*

[7] Note that this is similar to but different than overextraction. In overextraction, there is a single extraction procedure CCAVal which might extract values when no opening exists, but CCA non-malleability still holds with respect to CCAVal. In our setting, there are two extraction procedures, but CCA non-malleability only holds with respect to one of them.

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 some polynomial $\lambda^{O(1)}$.

At the end of a single step of this transformation, we will get another scheme CCA supporting the larger tag space $T(\lambda)$ where $T = T'^{\Omega(T'/2)}$. Applying the transformation a constant number of times we will get the scheme with a superpolynomial number of tags.

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. The circuit classes and advantages of various primitives and their relation/complexity hierarchy is described below. Let T_{Adv} be the size of circuits against which we want security.

- *One-way Permutation*: We require a one-way permutation OWP.
  - The length of the output/input is ℓ.
  - It is secure against adversaries running in time $T_{\mathsf{OWP}} = 2^{\ell^c}$ with advantage at most $\epsilon_{\mathsf{OWP}} = 2^{-\ell^c}$ for some constant $c > 0$.

  This can be instantiated using subexponential SXDH.

- *One-round CCA commitments*: We require a one-round CCA commitment CCA' with small tag space $T'(\lambda_{\mathsf{CCA}'})$ that is at least $\underbrace{\log \ldots \log \lambda}_{O(1) \text{ times}}$ and at most a polynomial in λ.
  - The commitment satisfies CCA security against adversaries of size polynomial in T_{Adv} with advantage $\epsilon_{\mathsf{CCA}'} = 2^{-\lambda_{\mathsf{CCA}'}}$.
  - The extraction algorithm CCAVal' has size $T_{\mathsf{CCAVal}'}$ (which is larger than $1/\epsilon_{\mathsf{CCA}'}$) and depth $d_{\mathsf{CCAVal}'}$.
  - There is an alternate extraction algorithm $\widetilde{\mathsf{CCAVal}}'$, which has depth $d_{\widetilde{\mathsf{CCAVal}}'} \gg d_{\mathsf{CCAVal}'}$ and size polynomial in $d_{\widetilde{\mathsf{CCAVal}}'}$, and where for all λ, m, and tag, it holds that

$$\Pr\left[\widetilde{\mathsf{CCAVal}}'(\tau, \mathsf{tag}, \mathsf{P}) = \mathsf{CCAVal}'(\tau, \mathsf{tag}, \mathsf{P}) \;\middle|\; \begin{array}{l} \mathsf{P} \leftarrow \mathsf{CCACommit}(1^\lambda, \mathsf{tag}, m) \\ \tau \xleftarrow{\$} \{0,1\}^{\ell_{\mathsf{CCA}'}} \end{array}\right] = 1$$

  We explain how to instantiate this below. It can be instantiated using subexponential time-lock puzzles and subexponential SXDH.

- *Time Lock Puzzles*: We require a time lock puzzle as in Definition 4. The TLP satisfies the following parameters.
  - It is solvable by a circuit of depth $d_{\mathsf{TLP,Ext}}$ and size polynomial in $d_{\mathsf{TLP,Ext}}$.
  - It is secure against circuits of depth $\ll d_{\mathsf{TLP,Ext}}$ and size $2^{\lambda_{\mathsf{TLP}}}$, with an advantage of at most $2^{-\lambda_{\mathsf{TLP}}}$.

- *Indistinguishability Obfuscation*: We require an indistinguishability Obfuscator iO. We assume that iO is secure against adversaries of size $2^{\lambda_{other}}$ with advantage at most $2^{-\lambda_{other}}$.

- *Puncturable PRF*: We require a puncturable PRF, $\mathsf{PPRF} = (\mathsf{Puncture}, \mathsf{Eval})$. Assume the length of the key is randomly chosen of length $\ell_{\mathsf{PPRF}}(\lambda)$ where λ is its security parameter. We assume that the PPRF is secure against adversaries of size $2^{\lambda_{other}}$ with advantage at most $2^{-\lambda_{other}}$. This can be instantiated assuming subexponential SXDH.

- *NIWI*: We require a non-interactive witness indistinguishable proof NIWI for NP. We assume that the NIWI is perfectly sound, and satisfies witness indistinguishability against adversaries of size $2^{\lambda_{other}}$ with advantage at most $2^{-\lambda_{other}}$. This can be instantiated using subexponential SXDH.

Complexity hierarchy. We require the following:

$$T_{\mathsf{Adv}} \ll d_{\mathsf{CCAVal}'} \ll d_{\widetilde{\mathsf{CCAVal}}'} = d_{\mathsf{TLP},ext} \ll T_{\mathsf{OWP}} \ll 2^{\ell} \ll 2^{\lambda'_{\mathsf{CCA}}} \ll T_{\mathsf{CCAVal}'} \ll 2^{\lambda_{\mathsf{TLP}}}, 2^{\lambda_{other}}.$$

The above hierarchy implies the following, in terms of advantage:

$$2^{-\ell} \gg 2^{\lambda_{\mathsf{CCA}'}}, 2^{-\lambda_{\mathsf{TLP}}}, 2^{\lambda_{other}},$$

This means that the advantage against security of the timelock puzzle, the obfuscation scheme, the puncturable PRF, and the witness indistinguishability of the NIWI are much less than $2^{-\ell}$ for a machine that runs in time $\mathsf{poly}(T_{\mathsf{CCAVal}'})$ and depth $d_{\mathsf{CCAVal}'}$, and the advantage against security of CCA' is much less than $2^{-\ell}$ for a machine that runs in time $\mathsf{poly}(T_{\mathsf{Adv}})$ and depth d_{Adv}.

Instantiating CCA'. To instantiate a commitment CCA' which has the alternate extraction procedure $\widetilde{\mathsf{CCAVal}}'$ as described above, it suffices to combine the following two primitives:

- A CCA non-malleable commitment scheme CCA'' which satisfies CCA security against adversaries of size polynomial in T_{Adv} with advantage $\epsilon_{\mathsf{CCA}'} = 2^{-\lambda_{\mathsf{CCA}'}}$, and has an extraction algorithm CCAVal'' that runs in size $T_{\mathsf{CCAVal}'}$ and depth $d_{\mathsf{CCAVal}'}$
- a time-lock puzzle which is secure against circuits of depth $\ll d_{\widetilde{\mathsf{CCAVal}}}$ and size $\mathsf{poly}(T_{\mathsf{CCAVal}'}$ with advantage $2^{-\lambda_{\mathsf{CCA}'}}$, but is solvable by a circuit of depth $d_{\widetilde{\mathsf{CCAVal}}}$.

We combine them in the following simple way.

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

- Compute a commitment c' to m using the TLP-based non-interactive commitment and uniform randomness r.
- Compute a commitment P to (m, r) using $\mathsf{CCA}''.\mathsf{CCACommit}$ with respect to tag r.
- Output (P, c').

$\mathsf{CCA}'.\mathsf{ComputeOpening}(\tau, \mathsf{tag}, (\mathsf{P}, c'))$: Compute the following steps.

- use $\mathsf{CCA}''.\mathsf{ComputeOpening}(\tau, \mathsf{tag}, \mathsf{P})$ to obtain an opening σ to (m, r).
- Output (σ, r).

$\mathsf{CCA}'.\mathsf{VerifyOpening}(\tau, \mathsf{tag}, (\mathsf{P}, c'), m, (\sigma, r))$: Compute the following steps.

- Use $\mathsf{CCA}''.\mathsf{VerifyOpening}(\tau, \mathsf{tag}, \mathsf{P}, (m, r), \sigma)$ to verify that σ is an opening of P to (m, r). Otherwise, output 0.
- Use r to verify that c' is a tlp-based commitment to m, otherwise output 0.
- Output 1.

The extractor $\mathsf{CCA}'.\mathsf{CCAVal}$ is implemented by running $\mathsf{CCA}''.\mathsf{CCAVal}$ on the CCA''-based commitment to get (m, r), then verifying whether the TLP-based commitment is honest before outputting m. The alternate extractor $\widetilde{\mathsf{CCAVal}}$ is implemented by using the TLP solver to extract c' directly to obtain m. It is straightforward to prove that the commitment CCA' above satisfies all the required properties specified above.

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. Output $\widetilde{G}$.

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

- Parse $\rho = (\rho, \rho')$ where $\rho \in \{0,1\}^{\ell}$ 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.
- Compute $\widetilde{G}[\rho] = (c_0, c_1, \ldots, c_{T'/2}, \pi)$,
- From the code of Figure 3, 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 $\rho \in \{0,1\}^{\ell_f}$ 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. 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.

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.

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: $\rho \in \{0,1\}^{\ell_f}$

Computation:

1. If ρ is not in the image of f, output $\perp$, otherwise proceed as follows.
2. 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:
   - $c_0 = \mathsf{PGen}(0^\ell; r_1)$,
   - For $i \in [T'/2]$, compute $c_i = \mathsf{CCA'.CCACommit}(1^{\lambda_{\mathsf{CCA'}}}, t_i, m; r_{2,i})$,
   - 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.
3. Output $(c_0, c_1, \ldots, c_{T'/2}, \pi)$.

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

$$L_{G,1} = \{(c_0, c_1, \ldots, c_{T'/2}) \mid \exists x \in \{0,1\}^\ell \ s.t. \ \rho = f(x) \wedge c_0 = \mathsf{PGen}(x)\}$$
$$L_{G,2} = \{(c_0, c_1, \ldots, c_{T'/2}) \mid \exists m, \{r_{2,i}\} \ s.t. \ \forall i \in [T'/2], \ c_i = \mathsf{CCA'.CCACommit}(1^{\lambda_{\mathsf{CCA'}}}, t_i, m; r_{2,i})\}$$

Figure 3: The Circuit $G[t_1, \ldots, t_{T'/2}, m, k_{\mathsf{PPRF}}]$

- Parse $\tau = (\rho, \rho')$ where $\rho \in \{0,1\}^\ell$ 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. Abort if the proof does not verify.
- Assuming the proof verifies, break open c_0 to recover x. Check if $f(x) = \rho$. If this is true, halt the experiment and output $\perp_{\mathsf{special}}$.
- Assuming we have not yet aborted, output m if $m = \mathsf{CCA'}.\widetilde{\mathsf{CCAVal}}(\rho', t_1, c_1) = \ldots = \mathsf{CCA'}.\widetilde{\mathsf{CCAVal}}(\rho', t_{T'/2}, c_{T'/2})$.

The extraction property then follows immediately from the extraction property of the underlying $\mathsf{CCA'}$ scheme. The idea is that in the last step, if $m = \mathsf{CCA'.CCAVal}(\rho', t_1, c_1) = \ldots = \mathsf{CCA'.CCAVal}(\rho', t_{T'/2}, c_{T'/2})$, 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 $\mathsf{CCA'}$. Similarly, the reverse is also true. Note that even if $\mathsf{CCA'}$ suffers from over-extraction, CCA does not, because the NIWI proof guarantees that the inner commitments are well-formed. Note that since both $\mathsf{CCA'.CCAVal}$ and TLP is extractable by a circuit of depth 2^t_{TLP} and size polynomial in $2^{t_{\mathsf{TLP}}}$, $\mathsf{CCA.CCAVal}$ is also extractable by a circuit of depth 2^t_{TLP} and size polynomial in $2^{t_{\mathsf{TLP}}}$.

We now move on to the security proof.

7.2.1 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 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 arbitrary 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.

To start the proof, we first prove that the probability that the experiment outputs $\perp_{\mathsf{special}}$ is negligible, i.e,

$$\Pr\left[\mathsf{Hybrid}_0 \to \perp_{\mathsf{special}}\right] < \mathsf{negl}(\lambda).$$

We do this via a reduction $\mathcal{R}$ to security of the one-way permutation f.

Assume that there is an adversary $\mathcal{A}$ under which $\Pr\left[\mathsf{Hybrid}_0 \to \perp_{\mathsf{special}}\right] < \mathsf{poly}(\lambda)$. Let $Q = Q(\lambda)$ be a polynomial which upper bounds the maximum number of queries that $\mathcal{A}$ makes when instantiated with security parameter λ. $\mathcal{R}$ receives a challenger ρ from the one-way permutation challenger, and then runs the experiment Hybrid_0 with $\mathcal{A}$ honestly, except for these differences:

- At the beginning of the experiment, it chooses $j \xleftarrow{\$} [Q]$.
- It generates all τ queries honestly, except that it for the j^{th} query, it sets $\tau_j = (\rho, \rho')$ where ρ is received from the challenger and ρ' is sampled randomly by the challenger.
- It answers CCAVal queries for every τ_i for $i \neq j$ honestly.
- For CCAVal queries on τ_j, it does the following. Assume that the query is for $\mathsf{CCAVal}(\tau_j, \mathsf{tag}, \mathsf{P})$ for $\mathsf{tag} \neq \mathsf{tag}^*$. Then, do the following:
  - Run $\mathsf{P}[\rho] = (c_0, c_1, \ldots, c_{T'/2}, \pi)$.

– Return $\perp$ to $\mathcal{A}$ if π does not verify. If it does, break open c_0 to recover x. Check if $f(x) = \rho$. If this is true, output x as the answer to the one-way permutation challenger.

– Assuming we have not yet aborted, return m to $\mathcal{A}$ if $m = \mathsf{CCA}'.\widetilde{\mathsf{CCAVal}}(\rho', t_1, c_1) = \ldots = \mathsf{CCA}'.\widetilde{\mathsf{CCAVal}}(\rho', t_{T'/2}, c_{T'/2})$.

Observe that since answering the CCAVal queries takes time and depth $\mathsf{poly}(d_{\widetilde{\mathsf{CCAVal}}}, d_{\mathsf{TLP,Ext}})$ and since $\mathcal{A}$ runs in time $T_{\mathsf{Adv}} \ll d_{\widetilde{\mathsf{CCAVal}}}, d_{\mathsf{TLP,Ext}}$, the reduction R runs in time and depth $\mathsf{poly}(d_{\widetilde{\mathsf{CCAVal}}}, d_{\mathsf{TLP,Ext}})$, which is much less than T_{OWP}. Further, observe that if $\mathcal{A}$ causes $\Pr[\mathsf{Hybrid}_0 \to \perp_{\mathsf{special}}] > 1/\mathsf{poly}(\lambda)$ then with probability $1/\mathsf{poly}(\lambda)$ there is one query of the form $\mathsf{CCAVal}(\tau_i = (\rho, \rho'), \mathsf{tag}, \mathsf{P})$ where $i \in [Q]$ and where $P[\rho]$ outputs a commitment c_0 to the inverse of ρ under f. Since $j = i$ with probability $1/Q$, the probability that $\mathcal{R}$ wins the OWP game is $1/(Q \cdot \mathsf{poly}(\lambda)) = 1/\mathsf{poly}(\lambda)$. This contradicts the one-way permutation security against adversaries of time T_{OWP}. Thus we have the following claim.

Lemma 1. *Assuming that f is secure against T_{OWP}-sized circuits, for any $\mathcal{A}$ of size T_{Adv}, it holds that*

$$\Pr[\mathsf{Hybrid}_0 \to \perp_{\mathsf{special}}] < \mathsf{negl}(\lambda).$$

Hybrid Hybrid_1 : This hybrid is the same as Hybrid_0, except that we modify how $\mathcal{A}$'s extraction queries are handled, only extracting one of the inner commitments $\{c_i\}$ instead of extracting all of them. We implement this using a new procedure CCAVal_1. The new code for CCAVal_1 is as follows.

$\mathsf{CCA}.\mathsf{CCAVal}_1(\tau, \mathsf{tag}, \mathsf{P})$: Compute the following steps.

- Parse $\tau = (\rho, \rho')$ where $\rho \in \{0,1\}^\ell$ 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. Abort if the proof does not verify.
- Assuming the proof verifies, break open c_0 to recover x. Check if $f(x) = \rho$. If this is true, halt the experiment and output $\perp_{\mathsf{special}}$.
- Assuming we have not yet aborted, output m where $m = \mathsf{CCA}'.\widetilde{\mathsf{CCAVal}}(\rho', t_i, c_i)$.

It follows directly from perfect soundness of the NIWI that the view of $\mathcal{A}$ is identical from Hybrid_0 to Hybrid_1. Since the NIWI proves that either: (1) all inner commitments $\{c_i\}$ commit to the one consistent value value m, or (2) c_0 is an inverse of ρ, and neither hybrid ever runs the last step of $\mathsf{CCA.CCAVal}$ or $\mathsf{CCA.CCAVal}_1$ without verifying the NIWI and that c_0 is not an inverse of ρ, the behavior of the extraction oracle is the same across the two hybrids. We therefore have the following claim.

Lemma 2. *Assuming that* NIWI *is perfectly sound 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*

$$|\Pr[\mathcal{A}(\mathsf{Hybrid}_1) = 1] - \Pr[\mathcal{A}(\mathsf{Hybrid}_0) = 1]| = 0.$$

By Lemma 2 and the previous Lemma 1, we also have the following:

Lemma 3. *Assuming that f is secure against T_{OWP}-sized circuits and that* NIWI *is perfectly sound, for any $\mathcal{A}$ of size T_{Adv}, it holds that*

$$\Pr\left[\mathsf{Hybrid}_1 \to \bot_{\mathsf{special}}\right] < \mathsf{negl}(\lambda).$$

Hybrid Hybrid_2: This hybrid is the same as Hybrid_1, except that we modify how $\mathcal{A}$'s extraction queries are handled, using the algorithm $\mathsf{CCA}'.\mathsf{CCAVal}$ on the inner commitments instead of $\mathsf{CCA}'.\widetilde{\mathsf{CCAVal}}$. We implement this using a new procedure CCAVal_2. The code is below.

$\mathsf{CCA}.\mathsf{CCAVal}_2(\tau, \mathsf{tag}, \mathsf{P})$: Compute the following steps.

- Parse $\tau = (\rho, \rho')$ where $\rho \in \{0,1\}^{\ell}$ 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. Abort if the proof does not verify.
- Assuming the proof verifies, break open c_0 to recover x. Check if $f(x) = \rho$. If this is true, halt the experiment and output $\bot_{\mathsf{special}}$.
- Assuming we have not yet aborted, output m where $m = \mathsf{CCA}'.\mathsf{CCAVal}(\rho', t_i, c_i)$.

It follows from perfect soundness of the NIWI and the property of $\mathsf{CCA}'.\widetilde{\mathsf{CCAVal}}$ that the view of $\mathcal{A}$ is identical from Hybrid_1 to Hybrid_2. The NIWI proves that either: (1) every inner commitment c_i is generated honestly: $c_i \leftarrow \mathsf{CCA}'.\mathsf{CCACommit}(1^{\lambda_{\mathsf{CCA}'}}, t, m)$ for some valid inner tag t and message m, or (2) c_0 is an inverse of ρ. Neither hybrid ever runs the last step of $\mathsf{CCA.CCAVal}_1$ or $\mathsf{CCA.CCAVal}_2$ without verifying the NIWI and also that c_0 is not an inverse of ρ, so it always holds that $\mathsf{CCA}'.\widetilde{\mathsf{CCAVal}}(\rho', t_i, c_i) = \mathsf{CCA}'.\mathsf{CCAVal}(\rho', t_i, c_i)$. Thus behavior of the extraction oracle is the same across the two hybrids. We therefore have the following claim.

Lemma 4. *Assuming that* NIWI *is perfectly sound and that the extraction procedure* $\mathsf{CCA}'.\widetilde{\mathsf{CCAVal}}$ *has the property described in Section 7.2, 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*

$$\left|\Pr[\mathcal{A}(\mathsf{Hybrid}_2) = 1] - \Pr[\mathcal{A}(\mathsf{Hybrid}_1) = 1]\right| = 0.$$

By Lemma 4 and the previous Lemma 3, we also have the following:

Lemma 5. *Assuming that f is secure against T_{OWP}-sized circuits, that* NIWI *is perfectly sound, and that the extraction procedure* $\mathsf{CCA}'.\widetilde{\mathsf{CCAVal}}$ *has the property described in Section 7.2, for any $\mathcal{A}$ of size T_{Adv}, it holds that*

$$\Pr\left[\mathsf{Hybrid}_2 \to \bot_{\mathsf{special}}\right] < \mathsf{negl}(\lambda).$$

Hybrid Hybrid_3**:** This hybrid is the same as Hybrid_2, except that we modify how $\mathcal{A}$'s extraction queries are handled, removing the step that opens c_0 and checks whether it commits to a preimage of ρ under f. We implement this using a new procedure CCAVal_3. The code is below.

$\mathsf{CCA.CCAVal}_3(\tau, \mathsf{tag}, \mathsf{P})$: Compute the following steps.

- Parse $\tau = (\rho, \rho')$ where $\rho \in \{0,1\}^\ell$ 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. Abort if the proof does not verify.
- Assuming the proof verifies, output m where $m = \mathsf{CCA}'.\mathsf{CCAVal}(\rho', t_i, c_i)$.

Since the only time $\mathcal{A}$ can distinguish between Hybrid_2 and Hybrid_3 is when the hybrid outputs $\perp_{\mathsf{special}}$, and Lemma 6 shows that this happens with negligible probability, it then follows that $\mathcal{A}$ has a negligible advantage in distinguishing the two hybrids. We thus have the following claim.

Lemma 6. *Assuming that f is secure against T_{OWP}-sized circuits, that* NIWI *is perfectly sound, and that the extraction procedure $\mathsf{CCA}'.\widetilde{\mathsf{CCAVal}}$ has the property described in Section 7.2, 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*

$$|\Pr[\mathcal{A}(\mathsf{Hybrid}_3) = 1] - \Pr[\mathcal{A}(\mathsf{Hybrid}_2) = 1]| = 0.$$

We now describe a series of hybrids, one for each $\gamma \in [0, 2^\ell]$.

Hybrid $\mathsf{Hybrid}_{4,\gamma}$: This hybrid is the same as the previous hybrid, except that in order to generate P^*, we obfuscate the circuit in Figure 4. Namely, compute $\phi(\mathsf{tag}^*) = (t^*_1, \ldots, t^*_{T'/2})$. Output $\mathsf{P}^* = \mathsf{iO}(G_1)$ where $G_1 = G_1[t^*_1, \ldots, t^*_{T'/2}, m_b, m_0, \gamma, K_{\mathsf{PPRF}}]$.

Note that the only difference between Hybrid_3 and $\mathsf{Hybrid}_{4,0}$ is how the challenge commitment P^* is generated. In Hybrid_3, it is generated by obfuscating program $G[t^*_1, \ldots, t^*_{T'/2}, m_b, K_{\mathsf{PPRF}}]$, where is $\mathsf{Hybrid}_{4,0}$ it it generated by obfuscating program $G_1[t^*_1, \ldots, t^*_{T'/2}, m_b, m_0, 0, K_{\mathsf{PPRF}}]$. These programs 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, or in other words, $\mathsf{poly}(T_{\mathsf{Adv}}, T_{\mathsf{CCA}'.\mathsf{CCAVal}})$. By the complexity hierarchy from Section 7.2, iO is secure against adversaries of size $2^{\lambda_{other}} \gg \mathsf{poly}(T_{\mathsf{Adv}}, T_{\mathsf{CCA}'.\mathsf{CCAVal}})$. We have the following claim:

Lemma 7. *Assume that iO is secure against circuits that run in time $2^{\lambda_{other}}$. Then, for any adversary $\mathcal{A}$ of size T_{Adv}, it holds that:*

$$|\Pr[\mathcal{A}(\mathsf{Hybrid}_3) = 1] - |\Pr[\mathcal{A}(\mathsf{Hybrid}_{4,0}) = 1]| \leq 2^{-\lambda_{other}}.$$

We now define the final hybrid.

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

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

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

Computation: The computation can be divided into two cases.
Case: $\rho < \gamma$

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 $c_0 = \mathsf{PGen}(0^\ell; r_1)$ 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: $\rho \geq \gamma$

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 $c_0 = \mathsf{PGen}(0^\ell; r_1)$ 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 $L_G = L_{G_1} \vee L_{G,2}$:

$$L_{G,1} = \big\{(c_0, c_1, \ldots, c_{T'/2}) \mid \exists x \in \{0,1\}^\ell \ s.t. \ f(x) = \rho \wedge c_0 = \mathsf{PGen}(x)\big\}$$
$$L_{G,2} = \big\{(c_0, c_1, \ldots, c_{T'/2}) \mid \exists m \ s.t. \ \forall i \in [T'/2], \ c_i = \mathsf{CCA'.CCACommit}(t_i, m)\big\}$$

Figure 4: The Circuit $G_1[t_1, \ldots, t_{T'/2}, m_b, m_0, \gamma, k_{\mathsf{PPRF}}]$

Hybrid Hybrid_5 : This hybrid is the same as the previous hybrid except to generate P^*, we obfuscate the circuit in Figure 3 by committing to m_0.

Note that the only difference between $\mathsf{Hybrid}_{4,2^\ell}$ and Hybrid_5 is how the challenge commitment P^* is generated. In $\mathsf{Hybrid}_{4,2^\ell}$, it is generated by obfuscating program $G_1[t_1^*, \ldots, t_{T'/2}^*, m_b, m_0, \gamma = 2^\ell, K_{\mathsf{PPRF}}]$, where as in Hybrid_5 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 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, or in other words, $\mathsf{poly}(T_{\mathsf{Adv}}, T_{\mathsf{CCA'.CCAVal}})$. By the complexity hierarchy from Section 7.2, iO is secure against adversaries of size $2^{\lambda_{other}} \gg \mathsf{poly}(T_{\mathsf{Adv}}, T_{\mathsf{CCA'.CCAVal}})$. We have the following claim:

Lemma 8. *Assume that* iO *is secure against circuits that run in time* $2^{\lambda_{other}}$. *Then, for any adversary* $\mathcal{A}$ *of size* T_{Adv}, *it holds that:*

$$|\Pr[\mathcal{A}(\mathsf{Hybrid}_5) = 1] - |\Pr[\mathcal{A}(\mathsf{Hybrid}_{4,\ell}) = 1]| \leq 2^{-\lambda_{other}}.$$

7.2.2 Indistinguishability between $\mathsf{Hybrid}_{4,\gamma}$ and $\mathsf{Hybrid}_{4,\gamma+1}$

The proof of security has now been reduced to showing that $\mathcal{A}$ has advantage $\leq \epsilon$ in distinguishing between $\mathsf{Hybrid}_{4,\gamma}$ and $\mathsf{Hybrid}_{4,\gamma+1}$, where $\epsilon \cdot 2^\ell < \mathsf{negl}(\lambda)$. We show this indistinguishability via a sequence of subhybrids, as follows.

Subhybrid Hybrid'_0 : This hybrid is the same as $\mathsf{Hybrid}_{4,\gamma}$.

Subhybrid Hybrid'_1 : This hybrid is the same as $\mathsf{Hybrid}_{4,\gamma}$ except that we generate P^* differently. We puncture the PPRF key K_{PPRF} at γ, and hardwire the response at $\rho = \gamma$. Namely, compute the punctured key k^*_{PPRF} at α. We compute a value v as follows. Compute $(r_1, r_2, r_3) \leftarrow \mathsf{PPRF.Eval}(K_{\mathsf{PPRF}}, \gamma)$. Then:

- Compute $c_0 = \mathsf{PGen}(0^\ell; r_1)$ and for $i \in [T'/2]$, compute $c_i = \mathsf{CCA'.CCACommit}(t_i^*, m_b; r_{2,i})$,
- 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 $G_2 = G_2[t_1^*, \ldots, t_{T'/2}^*, m_b, m_0, \gamma, K^*_{\mathsf{PPRF}}, v]$ as described in Figure 5.

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 $G_1[t_1^*, \ldots, t_{T'/2}^*, m_b, m_0, \gamma, K_{\mathsf{PPRF}}]$, where as in Hybrid'_1 it is generated by obfuscating $G_2[t_1^*, \ldots, t_{T'/2}^*, m_b, m_0, \gamma, K^*_{\mathsf{PPRF}}, v]$ where the key K^*_{PPRF} is punctured at γ. Note that if PPRF key is correct at un-punctured points, these circuits have identical behavior on all inputs $\rho \neq \gamma$. On input $\rho = \gamma$, 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. Since both hybrids run in $\mathsf{poly}(T_{\mathsf{Adv}}, T_{\mathsf{CCA'.CCAVal}}) \ll 2^{\lambda_{other}}$, we have that:

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

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

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

Computation: The computation can be divided into two cases.
Case: $\rho < \gamma$

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 $c_0 = \mathsf{PGen}(0^\ell; r_1)$ 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: $\rho > \gamma$

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 $c_0 = \mathsf{PGen}(0^\ell; r_1)$ 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: $\rho = \gamma$ Output v.
Language $L_G = L_{G_1} \vee L_{G,2}$:

$$L_{G,1} = \{(c_0, c_1, \ldots, c_{T'/2}) \mid \exists x \in \{0,1\}^\ell \text{ s.t. } f(x) = \rho \wedge c_0 = \mathsf{PGen}(x)\}$$
$$L_{G,2} = \{(c_0, c_1, \ldots, c_{T'/2}) \mid \exists m \text{ s.t. } \forall i \in [T'/2],\ c_i = \mathsf{CCA'.CCACommit}(t_i, m)\}$$

Figure 5: The Circuit $G_2[t_1, \ldots, t_{T'/2}, m_b, m_0, \gamma, k_{\mathsf{PPRF}}, v]$

Lemma 9. *Assuming that* iO *is secure against circuits that run in time* $2^{\lambda_{other}}$ *and* PPRF *is correct at unpunctured points, then for any adversary* $\mathcal{A}$ *of size* T_{Adv}*, it holds that:*

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

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

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.

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 $\mathsf{PPRF}(K_{\mathsf{PPRF}}, \gamma)$ 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 γ. 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, or in other words $\mathsf{poly}(T_{\mathsf{Adv}}, T_{\mathsf{CCA}'.\mathsf{CCAVal}})$. Since $\mathsf{poly}(T_{\mathsf{Adv}}, T_{\mathsf{CCA}'.\mathsf{CCAVal}}) \ll 2^{\lambda_{other}}$, we have the following claim.

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

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

Subhybrid 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 $c_0 = \mathsf{PGen}(x)$ where $x \in \{0,1\}^\ell$, and $f(x) = \gamma$.

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 time-lock puzzle commitment of 0^ℓ, whereas in Hybrid'_3 it is generated by committing to x such that $f(x) = \gamma$. 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 time lock puzzle with probability p. The reduction is non-uniform and must know a preimage for γ. The reduction also needs to answer polynomially many CCAVal queries. Therefore, its time is $\mathsf{poly}(T_{\mathsf{Adv}}, T_{\mathsf{CCA}'.\mathsf{CCAVal}})$, and its depth is $\mathsf{poly}(d_{\mathsf{CCAVal}'}, T_{\mathsf{Adv}})$. By the complexity hierarchy from Section 7.2, $\mathsf{poly}(d_{\mathsf{CCAVal}'}, T_{\mathsf{Adv}}) \ll d_{\mathsf{TLP},\mathsf{Ext}}$, and $\mathsf{poly}(T_{\mathsf{Adv}}, T_{\mathsf{CCA}'.\mathsf{CCAVal}}) \ll 2^{\lambda_{\mathsf{TLP}}}$, thus the time-lock puzzle provides security with advantage $2^{-\lambda_{\mathsf{TLP}}}$ against such a reduction. We thus have the following claim.

Lemma 11. *Assuming that* TLP *is secure against circuits of depth* $\ll d_{\mathsf{TLP},\mathsf{Ext}}$ *and size* $s^{\lambda_{\mathsf{TLP}}}$ *with advantage* $2^{-\lambda_{\mathsf{TLP}}}$*. Then, for any adversary* $\mathcal{A}$ *of size* T_{Adv}*, it holds that:*

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

Subhybrid 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.

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 $\mathsf{poly}(T_{\mathsf{Adv}},T_{\mathsf{CCA}'.\mathsf{CCAVal}})\ll 2^{\lambda_{other}}$ where λ_{other} is the security parameter of the NIWI (and where the $\ll$ is from the complexity hierarchy in Section 7.2).

Lemma 12. *Assuming that* NIWI *is secure against circuits that run in time* $2^{\lambda_{other}}$*, then for any adversary* $\mathcal{A}$ *of size* $\mathsf{poly}(T_{\mathsf{Adv}})$*, it holds that:*

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

Subhybrid 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. Thus if there exists an adversary $\mathcal{A}$ that distinguishes these hybrids with probability p, then we can build a reduction against the security of CCA'. Note that since the reduction gets oracle access to $\mathsf{CCA}'.\mathsf{CCAVal}(\cdot)$, and does not need to query on any of the challenge tags $(t_1^*,\ldots,t_{T'/2}^*)$, the reduction can be implemented in time $\mathsf{poly}(T_{\mathsf{Adv}})$. We thus have the following theorem.

Lemma 13. *Assuming that* CCA' *is secure against circuits of size* $\mathsf{poly}(T_{\mathsf{Adv}})$*. Then, for any adversary* $\mathcal{A}$ *of size* $\mathsf{poly}(T_{\mathsf{Adv}})$*, it holds that:*

$$|\Pr[\mathcal{A}(\mathsf{Hybrid}'_4)=1]-|\Pr[\mathcal{A}(\mathsf{Hybrid}'_5)=1]|\leq T'\cdot 2^{-\lambda_{\mathsf{CCA}'}}.$$

[Rex: ended here]

Subhybrid 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, as in both cases the reduction takes time $\mathsf{poly}(T_{\mathsf{Adv}}, T_{\mathsf{CCA}'.\mathsf{CCAVal}})$. Thus we have:

Lemma 14. *Assuming that* NIWI *is secure against circuits that run in time* $\mathsf{poly}(2^{\lambda_{other}})$*, then for any adversary* $\mathcal{A}$ *of size* $\mathsf{poly}(T_{\mathsf{Adv}})$*, it holds that:*

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

Subhybrid 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 TLP, and follow similarly as in the indistinguishability between Hybrid'_2 and Hybrid'_3. Thus we have:

Lemma 15. *Assuming that* TLP *is secure against circuits of depth* $\ll d_{\mathsf{TLP},\mathsf{Ext}}$ *and size* $s^{\lambda_{\mathsf{TLP}}}$ *with advantage* $2^{-\lambda_{\mathsf{TLP}}}$*. Then, for any adversary* $\mathcal{A}$ *of size* T_{Adv}*, it holds that:*

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

Subhybrid 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 $(r_1, r_2, r_3) \leftarrow \mathsf{PPRF.Eval}(K_{\mathsf{PPRF}}, \gamma)$.

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

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

Subhybrid Hybrid'_9 : This hybrid is the same as $\mathsf{Hybrid}_{4,\gamma+1}$.

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 17. *Assuming that* iO *is secure against circuits that run in time* $2^{\lambda_{other}}$ *and* PPRF *is correct at unpunctured points, then for any adversary* $\mathcal{A}$ *of size* T_{Adv}*, it holds that:*

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

Final Advantage. As we have shown, the advantage of $\mathcal{A}$ in distinguishing between each successive pair of hybrids Hybrid'_i and Hybrid'_{i+1} is at most some $\epsilon \ll 2^{-\ell}$. (Refer to the complexity hierarchy in Section 7.2 for why the particular advantage in each step is $\ll 2^{-\ell}$. It follows that $2^{\ell} \cdot \epsilon < \mathsf{negl}(\lambda)$ as we require.

7.3 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 6. Output $\widetilde{F}$.

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

- Parse $\tau = (\rho, \rho')$ where $\rho \in \{0,1\}^{\ell}$ 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 6.
- Compute $\widetilde{F}[\rho] = (c_0, c_1, \ldots, c_{T'}, \pi)$,
- From the code of Figure 6, 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 NIWI.Vf for the language described in 6. Abort if the proof does not verify,
- Output 1 if for every $t \in [T'] \setminus \mathsf{tag}$, $\mathsf{CCA}'.\mathsf{VerifyOpening}(\rho', t, c_t, m, \sigma_t)$. Output $\perp$ otherwise.

Remark 1 (Opening algrorithm for base scheme with T' tags)**.** *For base commitments as in [BL18, LPS17], the* $\mathsf{CCA}'.\mathsf{ComputeOpening}$ *simply outputs the randomness to commit the message.*

8 Primitives used for Constructing Our Zero-Knowledge Protocol

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 Non-interactive Distributional Indistinguishability (NIDI), due to Khurana [Khu21]. Unfortunately, due to the reasons we explain below, we need to strengthen the definition and construction for our purposes. The second notion is that of a *sometimes extractable equivocal commitments* (SEE), which is new to this work. A reader may skip this section and proceed onto Section 2.1.1 for an overview of our zero-knowledge argument construction, and then come back here for formal details about these ingredients.

8.1 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 3 for details.

Definition 19 (Syntax of NIDI)**.** *A* NIDI *for an* NP *language* L *and its relation* R_L *consists of the following algorithms.*

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:
   - $c_0 = \mathsf{PGen}(0^\ell; r_1)$,
   - For $t \in [T']$, and $t \neq \mathsf{tag}$, compute $c_t = \mathsf{CCA}'.\mathsf{CCACommit}(t, m; r_{2,i})$,
   - 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 $L_G = L_{F_1} \vee L_{F,2}$:

$$L_{F,1} = \{(c_0, c_1, \ldots, c_{T'}) \mid \exists x \in \{0,1\}^\ell \text{ s.t. } f(x) = \rho \wedge c_0 = \mathsf{PGen}(x)\}$$
$$L_{F,2} = \{(c_0, c_1, \ldots, c_{T'}) \mid \exists m \text{ s.t. } \forall t \in [T'] \setminus \mathsf{tag},\ c_t = \mathsf{CCA}'.\mathsf{CCACommit}(t, m) \wedge c_{\mathsf{tag}} = \perp\}$$

Figure 6: The Circuit $F[\mathsf{tag}, m, k_{\mathsf{PPRF}}]$

- $\mathsf{P}(1^{\lambda_S}, 1^{\lambda_D}, \mathcal{D})$: *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 20 (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 21 ($(\mathcal{C}_D, \mathcal{C}_{DI}, \epsilon_D, \epsilon_{DI})$- 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 22 (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 23 ($(\mathcal{C}_S, \epsilon_S)$-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 2. *Observe that the last two properties gives rise to a meaningful soundness property. The extraction property (Definition 22) ensures that whenever* $x \notin L$*, if* $\mathsf{V}(\tau,\pi) = x$ *then* $E(\tau,\pi) \neq 1$*. The Soundness property (Definition 23) 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 3 (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.*

8.2 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 [BFJ+20, GJJM20].

Definition 24. *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 exist 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.

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

8.3 Construction of NIDI

We start by giving a short overview of how we can construct a NIDI scheme satisfying properties specified in Definitions 19 to 21 and 23.

8.3.1 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[8] and perfect correctness, a non-interactive witness indistinguishable 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

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

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.

8.3.2 The Formal Construction

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 8.1. Formally, we prove the following theorem.

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

- *A subexponentially secure indistinguishability obfuscator exists,*
- *A time lock puzzle (as in Definition 4) exists,*
- *subexponential SXDH,*

then there exists a NIDI *scheme that satisfies security definitions in Definitions 20, 23, 21, 22, and is secure against adversaries of subexponential size.*

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 $\epsilon_S = 2^{-\lambda_S}$.

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.

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 [JLS21, JLS22]. 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.

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.

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 7.

$\mathsf{NIDI.V}(\tau, \widetilde{C})$: 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.

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

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

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

$E(\tau, \widetilde{C})$: 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.

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 8. 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
$$L' = \{(x', c') \ \ | \exists w' : \ R(x', w') = 1 \vee \exists \alpha : \ (c' = \mathsf{TLP.PGen}(\alpha) \wedge \mathsf{OWP}(\alpha) = \tau)\}$$
4. Output (x, c, π).

Figure 8: 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 $\lambda_S = \lambda$.

8.4 Construction of Sometimes Extractable Equivocal Commitments

In this section, we present our construction of a sometimes extractable equivocal commitments.

First, we specify the various class of adversary that we will handle in this scheme. Refer to Definition 24 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 25 (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.*

The rest of this section is devoted to proving the following theorem.

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

- *A time lock puzzle as in Definition 4 exist,*
- *subexponential SXDH,*

then, there exist a SEE *with the properties listed in Definition 24 as per parameters described in Definition 25.*

Used 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 in NC^1: 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 of DDH over $\mathbb{Z}_p^*$ as group exponentiation in $\mathbb{Z}_p^*$ is in NC^1 [BCH86].

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 SXDH [NP01].

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 $D(t_{\mathsf{TLP}}) = 2^{t_{\mathsf{TLP}}^\epsilon}$ 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 4. *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 19. *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 $D(t_{\mathsf{TLP}}) \geq 2^{t_{\mathsf{TLP}}^{\epsilon}} \in \lambda^{\omega(1)}$. 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 20. *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 21. *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.

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 $\mathsf{GbEquiv}(\Gamma, \mathsf{Lab}_{\mathbf{b}_R}$

$, H[\mathsf{com}_{1,C}, m], \mathbf{b}_R) \to (\mathsf{Lab}', s)$. 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 $s_i = \mathsf{OT}.\mathsf{Equiv}(\mathbf{b}_{R,i}, r_i, \mathsf{ot}_{2,i}, r'_i, \mathsf{Lab}'_{0,i}, \mathsf{Lab}'_{1,i})$. 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.

9 Construction of Reusable Statistical ZK arguments with Sometimes Statistical Soundness

In this section, we construct our zk protocol. Before we do that, we give an overview of this construction.

9.1 Overview

This section gives a brief overview of the zk construction. Recall that we want to construct a two-round (delayed instance) zk with SPS simulation. A 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 [BFJ+20, GJJM20]. The protocol relies on the following primitives:

- A correlation intractable hash function $\mathcal{H}(K, \star) \to \{0,1\}^\ell$ [CCH+19, PS19],
- 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 or circular secure FHE (we use the circular secure FHE construction, and instantiate the circular secure FHE using subepxonentially secure iO and SXDH). Further, the key generated by $\mathsf{FakeGen}$ 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 [BFJ+20], 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[9] for proving x. Then,
  - 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.

[9] As an example, think of it as the Blum's Hamiltonicity Protocol. As in the construction of NIZK from LWE[CCH+19, PS19], it suffices to use a parallel repetition of a sigma protocol for NP with i) 1/2-special soundness, ii) efficient $\mathsf{BadChallenge}$ computation.

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 algorithms $(\mathsf{Com}_{1,R}, \mathsf{Com}_{1,C}, \mathsf{Com}_{2,C})$ 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), 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 receiver-assisted one-round CCA commitment (CCA), and that protocol is intimately tied with the zk we are trying to build. As pointed out in Section 2.1.4, 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 two round statistically hiding extractable 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 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 8.4.

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 8.4) 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 8.1 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.

9.2 Construction

In this section, we 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 5.1. 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 26 (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 $2^{-\lambda}$.*

The rest of this section is devoted to proving the following theorem.

Theorem 10. *Assume that the following assumptions hold:*

- *A subexponentially secure indistinguishability obfuscator exists,*
- *A time lock puzzle as in Definition 4 exist,*
- *and subexponential DDH over both $\mathbb{Z}_p^*$ and asymmetric pairing groups.*

Then, there exists a reusable statistical ZK argument with sometimes statistical soundness as defined in Definition 14. For this scheme, the complexity parameters are defined in Definition 26.

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 19) 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 8, this can be constructed assuming subexponential security of iO, a time lock puzzle scheme (Definition 4), and subexponential SXDH.

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 9, can be constructed assuming subexponential security of iO, a time lock puzzle scheme (Definition 4), and subexponential DDH over $\mathbb{Z}_p^*$ and asymmetric pairing groups.

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.

- 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 INTRACTABLE 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 circular-secure fully-homomorphic encryption [CCH+19], which in turn can be constructed from subexponentially-secure indistinguishability obfuscation and a circularly secure perfectly rerandomizable encryption [CLTV15]. A circularly secure perfectly rerandomizable encryption can be built using SXDH [BHHO08].

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 $(x, \mathsf{com}_{1,C}, \mathsf{com}_{1,R}, \mathsf{com}_{2,C} = (\mathsf{com}_{2,C,1}, \ldots, \mathsf{com}_{2,C,N\cdot\ell_{ci}}))$.

- 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:

$\mathsf{ZKProve}_1(1^\lambda)$: 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 $\mathsf{zk}_{1,P} = (\tau, \mathsf{com}_{1,C})$.

$\mathsf{ZKVerify}_1(1^\lambda)$: 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 $\mathsf{zk}_{1,V} = \Pi$.

$\mathsf{ZKProve}_2(\mathsf{zk}_{1,V}, \mathsf{zk}_{1,P}, x, w)$: 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 $z = \{a_k, s_k\}_{k\in\mathsf{Set}}$.

$\mathsf{ZKVerify}_2(\mathsf{zk}_{1,V}, \mathsf{zk}_{1,P}, \mathsf{zk}_{2,P}, x)$: 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.

9.3 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 18. *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 $x \notin L$, 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 19. *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 20. *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. □

9.4 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 21. *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 21.

References

[ABG+21] Amit Agarwal, James Bartusek, Vipul Goyal, Dakshita Khurana, and Giulio Malavolta. Two-round maliciously secure computation with super-polynomial simulation. In *TCC*, pages 654–685, 2021.

[ACJ17] Prabhanjan Ananth, Arka Rai Choudhuri, and Abhishek Jain. A new approach to round-optimal secure multiparty computation. In *CRYPTO*, pages 468–499, 2017.

[AJJM21] Prabhanjan Ananth, Abhishek Jain, Zhengzhong Jin, and Giulio Malavolta. Unbounded multi-party computation from learning with errors. In *EUROCRYPT*, pages 754–781, 2021.

[AJL+12] 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 *EUROCRYPT*, pages 483–501, 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.

[BCCT13] Nir Bitansky, Ran Canetti, Alessandro Chiesa, and Eran Tromer. Recursive composition and bootstrapping for SNARKS and proof-carrying data. In *STOC*, pages 111–120, 2013.

[BCH86] Paul Beame, Stephen A. Cook, and H. James Hoover. Log depth circuits for division and related problems. *SIAM J. Comput.*, 15(4):994–1003, 1986.

[BFJ+20] Saikrishna Badrinarayanan, Rex Fernando, Aayush Jain, Dakshita Khurana, and Amit Sahai. Statistical ZAP arguments. In *EUROCRYPT*, pages 642–667, 2020.

[BGI+01] Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil P. Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. In *CRYPTO*, pages 1–18, 2001.

[BGI+12] 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.

[BGJ+16] Nir Bitansky, Shafi Goldwasser, Abhishek Jain, Omer Paneth, Vinod Vaikuntanathan, and Brent Waters. Time-lock puzzles from randomized encodings. In *ITCS*, pages 345–356, 2016.

[BGJ+17] Saikrishna Badrinarayanan, Vipul Goyal, Abhishek Jain, Dakshita Khurana, and Amit Sahai. Round optimal concurrent MPC via strong simulation. In *TCC*, pages 743–775, 2017.

[BGMM20] James Bartusek, Sanjam Garg, Daniel Masny, and Pratyay Mukherjee. Reusable two-round MPC from DDH. In *TCC*, pages 320–348, 2020.

[BGSZ22] James Bartusek, Sanjam Garg, Akshayaram Srinivasan, and Yinuo Zhang. Reusable two-round MPC from LPN. In *PKC*, pages 165–193, 2022.

[BGW88] Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In *STOC*, pages 1–10, 1988.

[BHHO08] Dan Boneh, Shai Halevi, Michael Hamburg, and Rafail Ostrovsky. Circular-secure encryption from decision diffie-hellman. In *CRYPTO*, pages 108–125, 2008.

[BHP17] Zvika Brakerski, Shai Halevi, and Antigoni Polychroniadou. Four round secure computation without setup. In *TCC*, pages 645–677, 2017.

[BJKL21] Fabrice Benhamouda, Aayush Jain, Ilan Komargodski, and Huijia Lin. Multiparty reusable non-interactive secure computation from LWE. In *EUROCRYPT*, pages 724–753, 2021.

[BKP18a] Nir Bitansky, Yael Tauman Kalai, and Omer Paneth. Multi-collision resistance: a paradigm for keyless hash functions. In *STOC*, pages 671–684, June 2018.

[BKP18b] Nir Bitansky, Yael Tauman Kalai, and Omer Paneth. Multi-collision resistance: a paradigm for keyless hash functions. In *STOC*, pages 671–684, 2018.

[BL18] Nir Bitansky and Huijia Lin. One-message zero knowledge and non-malleable commitments. In *TCC*, pages 209–234, 2018.

[BL20] Fabrice Benhamouda and Huijia Lin. Mr NISC: Multiparty reusable non-interactive secure computation. In *TCC*, pages 349–378, 2020.

[BMR90a] Donald Beaver, Silvio Micali, and Phillip Rogaway. The round complexity of secure protocols (extended abstract). In *STOC*, pages 503–513. ACM Press, 1990.

[BMR90b] Donald Beaver, Silvio Micali, and Phillip Rogaway. The round complexity of secure protocols (extended abstract). In *STOC*, pages 503–513, 1990.

[BN00] Dan Boneh and Moni Naor. Timed commitments. In *Advances in Cryptology - CRYPTO*, pages 236–254, 2000.

[BPS06] Boaz Barak, Manoj Prabhakaran, and Amit Sahai. Concurrent non-malleable zero knowledge. In *FOCS*, pages 345–354, 2006.

[CCD88] David Chaum, Claude Crépeau, and Ivan Damgård. Multiparty unconditionally secure protocols (extended abstract). In *STOC*, pages 11–19, 1988.

[CCG+20] Arka Rai Choudhuri, Michele Ciampi, Vipul Goyal, Abhishek Jain, and Rafail Ostrovsky. Round optimal secure multiparty computation from minimal assumptions. In *TCC*, pages 291–319, 2020.

[CCH+19] Ran Canetti, Yilei Chen, Justin Holmgren, Alex Lombardi, Guy N. Rothblum, Ron D. Rothblum, and Daniel Wichs. Fiat-shamir: from practice to theory. In *STOC*, pages 1082–1090, 2019.

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

[CLTV15] Ran Canetti, Huijia Lin, Stefano Tessaro, and Vinod Vaikuntanathan. Obfuscation of probabilistic circuits and applications. In *TCC*, pages 468–497, 2015.

[COSV17] Michele Ciampi, Rafail Ostrovsky, Luisa Siniscalchi, and Ivan Visconti. Round-optimal secure two-party computation from trapdoor permutations. In *TCC*, pages 678–710, 2017.

[DJMW12] Yevgeniy Dodis, Abhishek Jain, Tal Moran, and Daniel Wichs. Counterexamples to hardness amplification beyond negligible. In *TCC*, pages 476–493, 2012.

[DKP21] Dana Dachman-Soled, Ilan Komargodski, and Rafael Pass. Non-malleable codes for bounded parallel-time tampering. In *CRYPTO*, pages 535–565, 2021.

[FGKS22] Rex Fernando, Yuval Gelles, Ilan Komargodski, and Elaine Shi. Maliciously secure massively parallel computation for all-but-one corruptions. In *CRYPTO*, 2022.

[FKPS21] Cody Freitag, Ilan Komargodski, Rafael Pass, and Naomi Sirkin. Non-malleable time-lock puzzles and applications. In *TCC*, pages 447–479, 2021.

[GGJS12] Sanjam Garg, Vipul Goyal, Abhishek Jain, and Amit Sahai. Concurrently secure computation in constant rounds. In *EUROCRYPT*, pages 99–116, 2012.

[GGSW13] Sanjam Garg, Craig Gentry, Amit Sahai, and Brent Waters. Witness encryption and its applications. In *STOC*, pages 467–476, 2013.

[GJJM20] Vipul Goyal, Abhishek Jain, Zhengzhong Jin, and Giulio Malavolta. Statistical zaps and new oblivious transfer protocols. In *EUROCRYPT*, pages 668–699, 2020.

[GKLW21] Rachit Garg, Dakshita Khurana, George Lu, and Brent Waters. Black-box non-interactive non-malleable commitments. In *EUROCRYPT*, pages 159–185, 2021.

[GMPP16] Sanjam Garg, Pratyay Mukherjee, Omkant Pandey, and Antigoni Polychroniadou. The exact round complexity of secure computation. In *EUROCRYPT*, pages 448–476, 2016.

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

[GO94] Oded Goldreich and Yair Oren. Definitions and properties of zero-knowledge proof systems. *Journal of Cryptology*, 7(1):1–32, 1994.

[JLS21] Aayush Jain, Huijia Lin, and Amit Sahai. Indistinguishability obfuscation from well-founded assumptions. In *STOC*, pages 60–73, 2021.

[JLS22] Aayush Jain, Huijia Lin, and Amit Sahai. Indistinguishability obfuscation from LPN over $\mathbb {F}_p$, dlin, and prgs in nc^0. In *EUROCRYPT*, pages 670–699, 2022.

[Khu21] Dakshita Khurana. Non-interactive distributional indistinguishability (NIDI) and non-malleable commitments. In *EUROCRYPT*, pages 186–215, 2021.

[KK19] Yael Tauman Kalai and Dakshita Khurana. Non-interactive non-malleability from quantum supremacy. In *CRYPTO*, pages 552–582, 2019.

[KKS18] Yael Tauman Kalai, Dakshita Khurana, and Amit Sahai. Statistical witness indistinguishability (and more) in two messages. In *EUROCRYPT*, pages 34–65, 2018.

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

[KO04] Jonathan Katz and Rafail Ostrovsky. Round-optimal secure two-party computation. In *CRYPTO*, pages 335–354, 2004.

[KOS03] Jonathan Katz, Rafail Ostrovsky, and Adam Smith. Round efficiency of multi-party computation with a dishonest majority. In *EUROCRYPT*, pages 578–595, 2003.

[KS17] Dakshita Khurana and Amit Sahai. How to achieve non-malleability in one or two rounds. In *FOCS*, pages 564–575, 2017.

[LPS17] Huijia Lin, Rafael Pass, and Pratik Soni. Two-round and non-interactive concurrent non-malleable commitments from time-lock puzzles. In *FOCS*, pages 576–587, 2017.

[NP01] Moni Naor and Benny Pinkas. Efficient oblivious transfer protocols. In *SODA*, pages 448–457, 2001.

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

[PPV08] Omkant Pandey, Rafael Pass, and Vinod Vaikuntanathan. Adaptive one-way functions and applications. In *CRYPTO*, pages 57–74, 2008.

[PS19] Chris Peikert and Sina Shiehian. Noninteractive zero knowledge for NP from (plain) learning with errors. In *CRYPTO*, 2019.

[RSW96] R. L. Rivest, A. Shamir, and D. A. Wagner. Time-lock puzzles and timed-release crypto, 1996. Technical report, MIT, Cambridge, MA, USA.

[SV97] Amit Sahai and Salil P. Vadhan. A complete promise problem for statistical zero-knowledge. In *FOCS*, pages 448–457, 1997.

[Wee10] Hoeteck Wee. Black-box, round-efficient secure computation via non-malleability amplification. In *FOCS*, pages 531–540, 2010.

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