Four-round Statistical Non-malleable Zero-knowledge

Susumu Kiyoshima

NTT Social Informatics Laboratories
susumu.kiyoshima@ntt.com

June 19, 2026

Abstract

We present a 4-round statistical non-malleable zero-knowledge (statistical NMZK) argument for NP in the plain model under standard hardness assumptions. Our construction can be based on any collision-resistant hash function and injective one-way function, and it guarantees simulation extractability in the delayed-input, one-many setting. Before this work, 4-round constructions were known for computational NMZK but not for statistical NMZK.

This article is a full version of the following article: Four-round Statistical Non-malleable Zero-knowledge, TCC 2025, ©IACR 2025.

Contents

1 Introduction **3**
- 1.1 Our Result . . . 5
- 1.2 Additional Related Work . . . 5

2 Technical Overview **6**
- 2.1 Overall Approach . . . 6
- 2.2 Constant-round Construction . . . 7
- 2.3 4-round Construction . . . 13

3 Preliminaries **14**
- 3.1 Notations and Conventions . . . 14
- 3.2 Interactive Proofs/Arguments . . . 14
- 3.3 Adaptive-input Interactive Proofs/Arguments . . . 15
- 3.4 Statistical Non-malleable Zero-knowledge . . . 17
- 3.5 Synchronous Non-malleable Commitments . . . 19

4 Tool: Special-purpose WIPOK System **22**

5 Our 4-round SNMZK Argument **26**
- 5.1 Protocol Description . . . 27
- 5.2 Proofs of Completeness and Soundness . . . 28
- 5.3 Description of Simulator-Extractor . . . 30
- 5.4 Running Time of Simulator-Extractor . . . 33
- 5.5 Statistical Simulation Property of Simulator-Extractor . . . 34
- 5.6 Witness Extractability of Simulator-Extractor . . . 35

A Proof of Theorem 3 **48**
- A.1 Preliminaries . . . 48
- A.2 Protocol Description . . . 50
- A.3 Security proof . . . 50

B Full Description of Hybrid Simulator-extractor $\mathcal{SE}'$ **55**

C Proof of Claim 4 **57**

1 Introduction

Non-malleable zero-knowledge. *Non-malleable zero-knowledge* (NMZK) [DDN00] is a notion of zero-knowledge defined in man-in-the-middle settings. In the most basic setting, NMZK is defined against an adversary that interacts with an honest prover and an honest verifier, and the goal is to guarantee that the interaction with the prover does not help the adversary prove a related statement to the verifier. In a work by Pass and Rosen [PR08b], this goal is formalized by requiring the existence of a *simulator-extractor*, which simulates the adversary's view while also extracting a witness for the statement that the adversary proves.[1] The rationale behind this formalization is that given such a simulator-extractor, a stand-alone simulator can prove the same statement as the man-in-the-middle adversary by using the extracted witness.

A central topic in the study of NMZK is round complexity. Regarding the asymptotic round complexity, the initial construction by Dolev, Dwork, and Naor [DDN00] has logarithmic round complexity, and Barak [Bar02] achieved asymptotically optimal round complexity by providing a constant-round construction using sub-exponentially hard collision-resistant hash functions. Soon after, Pass and Rosen [PR08b] gave a constant-round construction using standard (polynomially hard) collision-resistant hash functions. Finally, Goyal [Goy11] and Lin and Pass [LP15] gave constant-round constructions using one-way functions. Regarding the exact round complexity, Goyal et al. [GRRV21] gave a 4-round construction using one-way functions. (Having 4 rounds is optimal even for non-NM zero-knowledge proofs/arguments in the case of black-box zero-knowledge [GK96].) Ciampi et al. [COSV17c] gave a 4-round construction in the *delayed-input setting* using collision-resistant hash functions. (In the delayed-input setting, the statement to be proven is determined in the last round of the protocol.) Finally, Ciampi et al. [COSV17a] gave a 4-round construction in the delayed-input, *one-many* setting using one-way functions. (In the one-many setting, the adversary interacts with a single prover and polynomially many verifiers.)

Statistical non-malleable zero-knowledge. *Statistical non-malleable zero-knowledge (SNMZK)* is a version of NMZK where the simulated view must be statistically close to the real view.[2] It naturally inherits strong security guarantees from statistical ZK in the stand-alone setting. For example, security is not compromised even if the underlying hardness assumptions are broken after the protocol executions [MU10].[3]

Statistical simulation offers an additional benefit in man-in-the-middle settings. As a motivating example, consider a commitment scheme that has the following

[1] Throughout this paper, all interactive proofs/arguments under discussion are for NP. (Here, "for NP" means "for an NP-complete language"; such a proof/argument can be used for any NP language via a polynomial-time reduction.)

[2] The man-in-the-middle adversary is restricted to a probabilistic polynomial-time machine, and only the distinguisher is computationally unbounded.

[3] To break computational soundness, cheating provers need to break the underlying hardness assumptions during protocol executions.

commit-and-prove structure: the sender commits to a message m by using a (possibly malleable) non-interactive commitment and then proves knowledge of its valid opening by using an NMZK argument.[4] A natural question to ask is whether NMZK guarantees non-malleability for this commitment scheme. That is, when a man-in-the-middle adversary receives a commitment to a message m, does NMZK prevent the adversary from committing to a related message such as $m + 1$? At first glance, one might expect the answer to be yes based on the following reasoning.

1. Suppose we remove the proof part from the "left commitment" but still require the adversary to complete the proof part in the "right commitment" (see Figure 1). Since the non-interactive commitment of the left commitment is not guaranteed to be non-malleable, the adversary might maul it so that the non-interactive commitment of the right commitment is a commitment to a related message. However, in this case, the adversary cannot complete the proof part of the right commitment since it does not know the message committed to in the non-interactive commitment. (This part can be formalized by relying on the non-interactivity of the commit part and the argument-of-knowledge property of the proof part.)

2. Then, since receiving an NMZK argument is not helpful in generating a new NMZK argument, the adversary should fail to complete the proof part of the right commitment even given the proof part of the left commitment.

Unfortunately, as observed by Pass and Rosen [PR08b, Section 6.1], the second part of the above reasoning does not work when NMZK only guarantees computational simulation. To see this, suppose the adversary adaptively selects the non-interactive commitment of the right commitment based on the proof part of the left commitment. Then, since the message committed to by the adversary cannot be efficiently recovered, computational simulation does not rule out the possibility that giving an NMZK argument to the adversary could noticeably change the message that the adversary commits to. As a result, in the case of computational NMZK, it is unclear whether the above commitment scheme is non-malleable. Fortunately, as also noted by Pass and Rosen [PR08b], this problem disappears in the case of SNMZK. Indeed, statistical simulation guarantees that giving an NMZK argument to the adversary cannot noticeably change the message that the adversary commits to.[5]

It is known that asymptotically optimal round complexity can be achieved even in the case of SNMZK. Indeed, the aforementioned constant-round construction by Pass and Rosen [PR08b] guarantees SNMZK, and their construction can be extended to satisfy SNMZK in the one-many setting [PR08a]. However, the exact round complexity of SNMZK arguments remains largely unexplored. In particular, whether 4-round

[4] Note that the above "simulation extractability"-type definition of NMZK implies the standard argument-of-knowledge property.

[5] This application to non-malleable commitment schemes is provided as an illustrative example of how statistical simulations offer an additional benefit. (It is now well-known that non-malleable commitment schemes can be constructed without using NMZK.) Essentially, what this example demonstrates is that statistical simulations are useful in applications where we need to argue about some statistical properties of the statements proven by NMZK arguments.

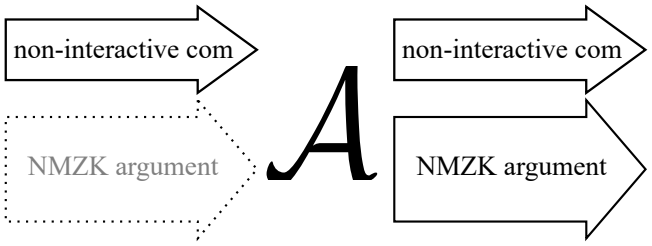

Figure 1: Commit-and-prove protocol with the proof part removed in the left commitment.

SNMZK arguments can be obtained under standard cryptographic assumptions was unknown.

1.1 Our Result

We present the following result regarding 4-round SNMZK arguments.

Theorem (informal). *Assume the existence of collision-resistant hash functions and injective one-way functions. Then, there exists a 4-round statistical non-malleable zero-knowledge argument for NP in the delayed-input, one-many setting.*

This result is obtained using black-box simulation techniques. Thus, like prior results on 4-round computational NMZK [GRRV21, COSV17c, COSV17a], this result matches the 4-round lower bound of black-box ZK. An interesting question left open by this result is whether a 4-round SNMZK argument can be constructed from collision-resistant hash functions (without additionally relying on injective one-way functions).

1.2 Additional Related Work

Let us review additional related work that studies NMZK in the plain model (i.e., the model where trusted setups such as common reference strings are unavailable).

Concurrent non-malleable zero-knowledge (concurrent NMZK) [BPS06] is a version of NMZK where man-in-the-middle adversaries can interact with polynomially many provers and verifiers. Concurrent NMZK has been extensively studied [BPS06, OPV10, LPTV10, LP11b, Ven14], and some works even achieve *statistical concurrent non-malleable zero-knowledge* [OOR+14, Kiy20]. However, the exact round complexity of concurrent NMZK remains largely unexplored. A major obstacle is that even obtaining constant-round concurrent (non-NM) zero-knowledge arguments currently requires strong hardness assumptions (e.g., those on which the existence of indistinguishability obfuscation can be based) [CLP13, CLP15, PPS15, FKP19].

Some recent works have investigated NMZK with practical efficiency [KLP22, BCO+24]. This paper focuses on establishing a feasibility result and does not aim to optimize practical efficiency.

2 Technical Overview

2.1 Overall Approach

We begin by outlining the main challenge in constructing 4-round SNMZK arguments. Recall that in the definition of NMZK, a simulator-extractor must simulate the man-in-the-middle adversary's view while also extracting a witness for the statement that the adversary proves.[6] We call the interaction between the honest prover and the adversary the *left interaction*, and the interaction between the adversary and the honest verifier the *right interaction*.

The main tool used in the existing 4-round computational NMZK arguments [GRRV21, COSV17c, COSV17a] is *non-malleable (NM) commitment schemes* [DDN00]. At a high level, NM commitment schemes guarantee that when a man-in-the-middle adversary interacts with an honest sender and an honest receiver, the value committed to by the adversary is independent of the value committed to by the honest sender. In this technical overview, we adopt the following (slightly stronger) CCA-style definition: no adversary can break the hiding property even given access to a computationally unbounded *committed-value oracle*, which interacts with the adversary as a receiver and then returns the committed value of the adversary's commitment.

What is common in the existing 4-round computational NMZK arguments is that NM commitments are made by the prover. In particular, in the existing 4-round computational NMZK arguments, NM commitments are made by the prover so that different values are committed in the real execution and the simulation. For example, in the constructions of [GRRV21, COSV17c], a witness for the statement is committed to in the real execution while a junk value (such as an all-0 string) is committed to in the simulation. At a very high level, when the prover makes NM commitments in this way, NMZK can be argued as follows. Suppose a man-in-the-middle adversary proves the validity of a statement in the right interaction by committing to a valid witness in an NM commitment. Then, the non-malleability of the NM commitment guarantees that when the left interaction is switched to simulation (and therefore the committed value of the NM commitment is switched to a junk value in the left interaction), the adversary continues to commit to a valid witness in the right interaction. Thus, a desired simulator-extractor can be obtained by considering a simulator that extracts the committed value of the NM commitment in the right interaction.

A challenge we face is that using NM commitment schemes in this way inherently prevents the simulation from being statistically indistinguishable. Specifically, the problem is that NM commitment schemes must be statistically binding by definition.[7] When a different value is committed in the real execution and the simulation, a computationally unbounded distinguisher can easily distinguish them by extracting the committed values of the NM commitments by brute force.

[6] In this overview, we focus on the one-one NMZK setting (where the adversary interacts with an honest prover and an honest verifier) rather than the one-many NMZK setting (where the adversary interacts with an honest prover and polynomially many honest verifiers).

[7] Non-malleability is not well-defined unless the value committed to by the adversary is uniquely determined.

Fortunately, this problem has already been addressed in the context of asymptotically round-efficient statistical concurrent NMZK arguments [Kiy20]. At a high level, the idea is to have the verifier make NM commitments (rather than having the prover do so).[8] When only the verifier makes NM commitments, the statistical binding of the NM commitments is unproblematic for statistical simulation. Furthermore, by having the verifier make NM commitments in a trapdoor generation phase (where a trapdoor is set up for simulation), we can guarantee that the left and right interactions have independent trapdoors. This property can be used to obtain a desired simulator-extractor by showing that any man-in-the-middle adversary cannot cheat in the right interaction even when the left interaction is simulated using the trapdoor.

Naturally, our approach to 4-round SNMZK involves combining this technique with those developed for 4-round computational NMZK arguments.

2.2 Constant-round Construction

First, we describe our construction without optimizing its round complexity. The starting point is the well-known approach by Feige and Shamir [FS90a] for round-efficient ZK arguments. Recall that, in the case of statistical ZK, the Feige–Shamir approach can be implemented using the following three components: a one-way permutation (OWP), a witness-indistinguishable proof of knowledge (WIPOK), and a statistically witness-indistinguishable argument of knowledge (SWIAOK).[9] Since our approach is to have the verifier make NM commitments, we consider combining the first two components with NM commitment schemes. Concretely, we consider the following construction for statement $x \in \mathbf{L}$.

1. **Trapdoor generation.** The verifier uses an NM commitment scheme twice to commit to two random strings, r_0 and r_1. Let i-τ_0 and i-τ_1 be the resulting commitments. Let dec_0 and dec_1 be the corresponding decommitments.

2. **Verifier proof.** The verifier uses the NM commitment scheme twice to commit to dec_0 and dec_1, resulting in two commitments, o-τ_0 and o-τ_1. Then, the verifier uses a WIPOK system to prove knowledge of $\beta \in \{0,1\}$ and $\mathsf{dec} \in \{0,1\}^*$ such that (i) dec is a valid decommitment of i-τ_β and (ii) o-τ_β is a correct commitment to dec. (The witness used by the verifier is chosen randomly from $(0, \mathsf{dec}_0)$ or $(1, \mathsf{dec}_1)$.)

3. **Prover proof.** The prover uses an SWIAOK system to prove either $x \in \mathbf{L}$ or knowledge of a valid decommitment of i-τ_b for some $b \in \{0,1\}$.

In the following, we call the NM commitments in the trapdoor generation phase the *inner NM commitments*, and the NM commitments in the verifier proof phase the *outer NM commitments*.

[8] The idea of having the verifier make NM commitments has also been used for computational NMZK and a related primitive, for different purposes from ours [JP14, CLP20, BCO+24].

[9] For simplicity, we focus on the case where the latter two satisfy *special soundness*. That is, a valid witness can be extracted from any two accepting transcripts if their commit messages are identical and their challenge messages are different.

2.2.1 Case 1: NMZK in asynchronous message scheduling.

As a warm-up, let us analyze SNMZK in the setting where the messages in the left and right interactions are scheduled as shown in Figure 2 (left). Consider a natural simulator-extractor that simulates the left interaction and extracts a witness in the right interaction as follows. In the left interaction, it uses the extractor of the WIPOK system to obtain a valid decommitment of one of the inner NM commitments, and uses this decommitment as a witness for the SWIAOK system to simulate the prover's proof. In the right interaction, it uses the extractor of the SWIAOK system to obtain a witness for the statement. Statistical indistinguishability of the simulation can be shown in the standard manner by relying on the knowledge soundness of the WIPOK system and the statistical WI of the SWIAOK system.[10] Regarding witness extractability, assume for contradiction that the simulator-extractor fails to extract a witness for the statement in the right interaction with non-negligible probability. Then, the knowledge soundness of the SWIAOK system guarantees that a valid decommitment of one of the two inner NM commitments is extracted with non-negligible probability. Furthermore, by the witness indistinguishability of the WIPOK system, the simulator-extractor's extraction behavior cannot noticeably depend on which of the two decommitments is used as the witness in the WIPOK system in the right interaction. Specifically, there exists $\widetilde{\beta}^* \in \{0,1\}$ such that the following event, BAD, occurs with non-negligible probability.

- **Event BAD:** The WIPOK system in the right interaction is completed using the decommitment of the $\widetilde{\beta}^*$-th inner NM commitment as the witness, and the simulator-extractor extracts a valid decommitment of the $(1-\widetilde{\beta}^*)$-th inner NM commitment in the SWIAOK system.

In other words, with non-negligible probability, the simulator-extractor extracts a decommitment for the inner NM commitment that was not used as the witness in the WIPOK system in the right interaction. Furthermore, since the simulator-extractor simulates the left interaction by using a decommitment of one of the two inner NM commitments, it follows from an averaging argument that there exists $b^* \in \{0,1\}$ such that with non-negligible probability, BAD occurs while the simulator-extractor simulates the left interaction using a decommitment of the b^*-th inner NM commitment. Then, consider the following hybrid experiments.

- In the first hybrid, the witness used for SWIAOK in the left interaction is obtained by extracting the committed value of the b^*-th outer NM commitment by brute force (rather than using the extractor of the WIPOK system).

  Note that when the simulator-extractor obtains a decommitment of the b^*-th inner NM commitment from the extractor of the WIPOK system, the knowledge soundness of the WIPOK system guarantees that the same decommitment must

[10] More concretely, statistical indistinguishability is shown as follows. By the knowledge soundness of the WIPOK system, the simulator-extractor obtains a valid decommitment of one of the inner NM commitments except with negligible probability. Then, by the statistical WI of the SWIAOK system, the SWIAOK proof generated using this decommitment is statistically indistinguishable from the proof generated using the real witness.

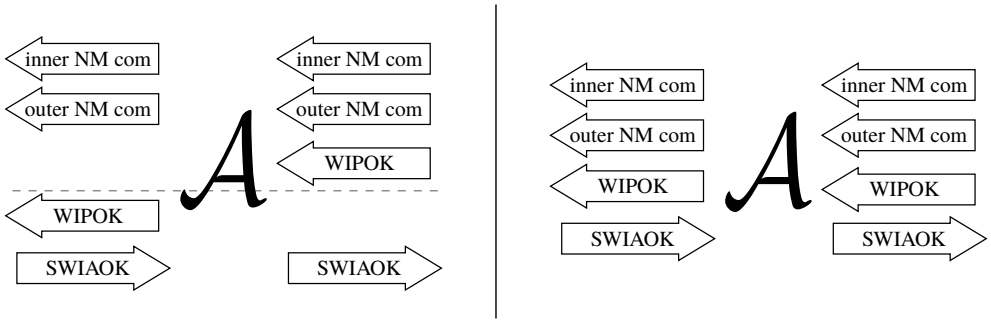

Figure 2: An asynchronous schedule (left) and the synchronous schedule (right). In the synchronous schedule, the adversary sends each message in the left interaction immediately after receiving the corresponding message in the right interaction, and vice versa. In the asynchronous schedule shown on the left, the left and right interactions are synchronized except for the WIPOK part, where the WIPOK part in the left interaction starts after the WIPOK part in the right interaction completes.

be committed in the b^*-th outer NM commitment (and thus can be extracted by brute force). Thus, even in this hybrid, BAD occurs with non-negligible probability.

- In the second hybrid, the value committed in the $(1 - \widetilde{\beta}^*)$-th outer NM commitment in the right interaction is switched to an all-0 string.

  The (CCA-style) non-malleability of the NM commitment scheme guarantees that even in this hybrid, BAD occurs with non-negligible probability.

Since the decommitment of the $(1 - \widetilde{\beta}^*)$-th inner NM commitment is no longer used in the right interaction in the second hybrid, we can break the non-malleability of the $(1 - \widetilde{\beta}^*)$-th inner NM commitment whenever BAD occurs.

2.2.2 Case 2: NMZK in synchronous message scheduling.

As a more interesting case, consider the setting where the messages in the left and right interactions are scheduled synchronously as shown in Figure 2 (right).

In this case, the above analysis fails when we attempt to use the witness indistinguishability of the WIPOK system in the right interaction. That is, it is hard to show that the simulator-extractor's extraction behavior cannot noticeably depend on which of the two decommitments is used as the witness in the WIPOK system in the right interaction. To see this, recall that simulating the left interaction involves rewinding the WIPOK system in the left interaction. In the synchronized scheduling, rewinding the left WIPOK system inherently involves rewinding the right WIPOK system. Thus, the existence of $\widetilde{\beta}^*$ as above does not follow readily from the witness indistinguishability of the WIPOK system.

To address this issue, we use a technique from prior work on 4-round computational NMZK arguments. More concretely, we adapt a technique from Ciampi et al. [COSV17b]. In the work of Ciampi et al., a problem that is closely related to the

above is addressed by designing a specific construction of WIPOK. Their WIPOK system is a careful combination of several low-level primitives (such as a non-interactive commitment scheme and an honest-verifier zero-knowledge system), and the security of their 4-round NMZK argument is analyzed by directly considering reductions to the security of these low-level primitives. We define a WIPOK system that satisfies key properties of their WIPOK system, and integrate it into our construction and analysis in a (somewhat) modular fashion.[11]

Concretely, we consider the following 3-round special-purpose WIPOK system. It is tailored for statements of the form $x_0 \in \mathbf{L} \vee x_1 \in \mathbf{L}$ (where both $x_0 \in \mathbf{L}$ and $x_1 \in \mathbf{L}$ have a unique witness), and it has the following "delayed-input" syntax that uses x_0 and x_1 only in the last round.

- For every $\beta \in \{0,1\}$, $(x_\beta, w_\beta) \in \mathbf{R_L}$, and $x_{1-\beta} \in \{0,1\}^{|x_\beta|}$,

$$\Pr\left[\mathsf{WI.V_D}(x_0, x_1, \tau) = 1 \;\middle|\; \begin{array}{l} (\mathsf{wi}_1, \mathsf{st}) \leftarrow \mathsf{WI.P}_1(1^{|x_\beta|}, \beta) \\ \mathsf{wi}_2 \leftarrow \mathsf{WI.V}(1^{|x_\beta|}) \\ \mathsf{wi}_3 \leftarrow \mathsf{WI.P}_2(\mathsf{st}, x_0, x_1, w_\beta, \mathsf{wi}_2) \\ \tau := (\mathsf{wi}_1, \mathsf{wi}_2, \mathsf{wi}_3) \end{array}\right] = 1.$$

Furthermore, its witness indistinguishability is established through a series of hybrid experiments, and some of the hybrid experiments are indistinguishable even against rewinding adversaries. Concretely, there exists a hybrid prover $(\mathsf{WI.}\widehat{\mathsf{P}}_1, \mathsf{WI.}\widehat{\mathsf{P}}_2)$ satisfying the following.

1. The hybrid prover $(\mathsf{WI.}\widehat{\mathsf{P}}_1, \mathsf{WI.}\widehat{\mathsf{P}}_2)$ follows the same syntax as the real prover $(\mathsf{WI.P}_1, \mathsf{WI.P}_2)$ except that $\mathsf{WI.}\widehat{\mathsf{P}}_2$ requires witnesses for both of the substatements x_0, x_1 regardless of the value β. That is, it is invoked as $\mathsf{wi}_3 \leftarrow \mathsf{WI.}\widehat{\mathsf{P}}_2(\mathsf{st}, x_0, x_1, \boxed{w_0, w_1}, \mathsf{wi}_2)$, where the difference from $\mathsf{WI.P}_2$ is highlighted by a box.

2. The indistinguishability between the hybrid prover $(\mathsf{WI.}\widehat{\mathsf{P}}_1, \mathsf{WI.}\widehat{\mathsf{P}}_2)$ and the real prover $(\mathsf{WI.P}_1, \mathsf{WI.P}_2)$ can be shown using an auxiliary prover algorithm $\mathsf{WI.}\widehat{\mathsf{P}}_{1.5}$ as follows.

$$\begin{aligned} &(\mathsf{WI.}\widehat{\mathsf{P}}_1, \mathsf{WI.}\widehat{\mathsf{P}}_2) \\ &\equiv ((\mathsf{WI.P}_1, \mathsf{WI.}\widehat{\mathsf{P}}_{1.5}), \mathsf{WI.}\widehat{\mathsf{P}}_2) && (1) \\ &\equiv (\mathsf{WI.P}_1, (\mathsf{WI.}\widehat{\mathsf{P}}_{1.5}, \mathsf{WI.}\widehat{\mathsf{P}}_2)) && (2) \\ &\approx (\mathsf{WI.P}_1, \mathsf{WI.P}_2) && (3) \end{aligned}$$

That is, (i) the execution of $\mathsf{WI.}\widehat{\mathsf{P}}_1$ can be decomposed into a sequential execution of $\mathsf{WI.P}_1$ and $\mathsf{WI.}\widehat{\mathsf{P}}_{1.5}$ as shown in (1), (ii) the execution of $\mathsf{WI.}\widehat{\mathsf{P}}_{1.5}$ can be deferred until the last round as shown in (2),[12] and (iii) a sequential execution of $\mathsf{WI.}\widehat{\mathsf{P}}_{1.5}$ and $\mathsf{WI.}\widehat{\mathsf{P}}_2$ is indistinguishable from $\mathsf{WI.P}_2$ as shown in (3).

[11] Ciampi et al. [COS22] make a similar attempt at modularization, resulting in a different formalization (called *non-malleable honest-verifier zero-knowledge with respect to commitment*).

[12] Specifically, $\mathsf{WI.}\widehat{\mathsf{P}}_{1.5}$ only generates auxiliary information to be used in $\mathsf{WI.}\widehat{\mathsf{P}}_2$ (without generating anything related to the message to be sent to the verifier in the first round). Thus, any (non-rewinding) verifier cannot detect whether $\mathsf{WI.}\widehat{\mathsf{P}}_{1.5}$ is executed in the first round or in the last round.

3. The hybrid prover $(\mathsf{WI}.\widehat{\mathsf{P}}_1, \mathsf{WI}.\widehat{\mathsf{P}}_2)$ satisfies witness indistinguishability even against rewinding adversaries. That is, any cheating verifier cannot tell whether $\mathsf{WI}.\widehat{\mathsf{P}}_1$ runs on input $\beta = 0$ or $\beta = 1$ even given multiple responses from $\mathsf{WI}.\widehat{\mathsf{P}}_2$.[13]

Note that given the hybrid prover as above, the witness indistinguishability indeed follows from a simple hybrid argument: first switch the real prover to the hybrid one, next flip the value of β, and finally switch back the hybrid prover to the real one. (The second step of this hybrid argument holds even against rewinding adversaries.)

Using the above special-purpose WIPOK system, we analyze the synchronous case as follows. Recall that in the asynchronous case, we considered a hybrid argument where (i) in the first hybrid, two witnesses are available for the WIPOK system in the right interaction, whereas (ii) in the last hybrid, only a single witness is available. Based on this observation, we modify our construction so that the verifier completes the WIPOK system using the hybrid prover algorithms $(\mathsf{WI}.\widehat{\mathsf{P}}_1, \mathsf{WI}.\widehat{\mathsf{P}}_2)$. Thus, an honest verifier uses both of the decommitments of the two inner NM commitments for the WIPOK system. On the positive side, this modification enables us to show the existence of $\widetilde{\beta}^*$ such that BAD occurs with non-negligible probability. The key point is that the hybrid prover of the special-purpose WIPOK system is witness indistinguishable even against rewinding adversaries. Therefore, despite the rewinding in the left interaction, we can show that the simulator-extractor's extraction behavior cannot noticeably depend on which of the two decommitments is used as the witness in the WIPOK system in the right interaction. On the negative side, this modification requires us to consider an additional hybrid experiment where we switch the hybrid prover $(\mathsf{WI}.\widehat{\mathsf{P}}_1, \mathsf{WI}.\widehat{\mathsf{P}}_2)$ to the real prover $(\mathsf{WI}.\mathsf{P}_1, \mathsf{WI}.\mathsf{P}_2)$. (Otherwise, the decommitment of the $(1 - \widetilde{\beta}^*)$-th inner NM commitment continues to be used in the right interaction, and therefore we cannot derive a contradiction in the last hybrid.) Given that the hybrid prover and the real prover are not guaranteed to be indistinguishable against rewinding adversaries, it might appear that we have not made any progress. To overcome this difficulty, we use the fact that indistinguishability against rewinding adversaries is not needed here. To see this, observe that, as evidenced in the analysis of the asynchronous case, we will later replace the rewinding extraction with brute-force extraction in the left interaction. That is, the WIPOK system in the left interaction is not rewound in later hybrids. This observation suggests that we only need to replace the hybrid prover with the real one in the "main thread" and do not need to perform this replacement in the "look-ahead threads" that are generated during the rewinding of the left WIPOK system (cf. Figure 3, first and fourth panels from the top).[14] In other words, we do not need to worry about using the hybrid prover in the look-ahead threads because the look-ahead threads will disappear in later hybrids when we switch the rewinding extraction to brute-force extraction. Concretely, first,

[13] This property does not contradict the proof-of-knowledge property since the hybrid prover has witnesses for both sub-statements x_0, x_1. In particular, rewinding the hybrid prover results in extracting witnesses for both sub-statements, and thus, is useless for determining the value of β.

[14] This technique was first introduced in the context of *security with super-polynomial-time simulation* by Garg et al. [GGJS12]. (Subsequently, it is also used in the context of NMZK by Ciampi et al. [COSV17b].)

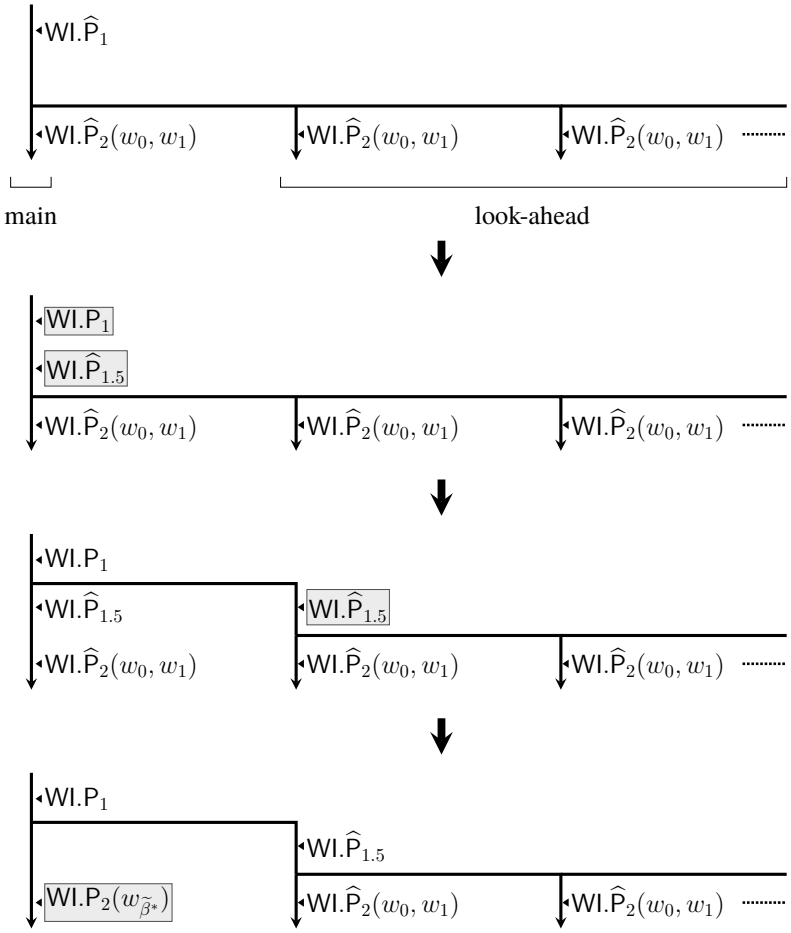

Figure 3: Sequence of hybrids where the hybrid prover of the WIPOK system is removed from the main thread.

we decompose $\mathsf{WI}.\widehat{\mathsf{P}}_1$ into $(\mathsf{WI}.\mathsf{P}_1, \mathsf{WI}.\widehat{\mathsf{P}}_{1.5})$, and next, use an averaging argument to show that BAD occurs with non-negligible probability even when the main thread and the look-ahead threads use different executions of $\mathsf{WI}.\widehat{\mathsf{P}}_{1.5}$ (cf. Figure 3, second and third panels from the top). Then, we rely on the indistinguishability between $(\mathsf{WI}.\widehat{\mathsf{P}}_{1.5}, \mathsf{WI}.\widehat{\mathsf{P}}_2)$ and $\mathsf{WI}.\mathsf{P}_2$ to show that BAD occurs with non-negligible probability even when $(\mathsf{WI}.\widehat{\mathsf{P}}_{1.5}, \mathsf{WI}.\widehat{\mathsf{P}}_2)$ is switched to $\mathsf{WI}.\mathsf{P}_2$ in the main thread (cf. Figure 3, fourth panel from the top). The rest of the analysis can proceed as in the asynchronous case since the decommitment of the $(1 - \widetilde{\beta}^*)$-th inner NM commitment is no longer used in the right interaction in the main thread.

Remark 1 (On the use of the special-purpose WIPOK). While the construction of the special-purpose WIPOK is taken from Ciampi et al. [COSV17b] without modifications, we use it differently from Ciampi et al.. Specifically, our SNMZK construction uses the hybrid prover algorithms $(\mathsf{WI}.\widehat{\mathsf{P}}_1, \mathsf{WI}.\widehat{\mathsf{P}}_2)$ in its honest execution, whereas their NMZK construction uses the real prover algorithms $(\mathsf{WI}.\mathsf{P}_1, \mathsf{WI}.\mathsf{P}_2)$ in its honest execution. Interestingly, our analysis relies on this particular use of the hybrid

prover algorithms, and we do not know whether our construction satisfies SNMZK if the real prover is used in its honest execution. (Thus, our analysis relies on the fact that, in the approach by Feige and Shamir [FS90a], the verifier has two witnesses for the WIPOK system.) To understand the difficulty, suppose the verifier of our construction completes the WIPOK system by using the real prover algorithms. If we follow the analysis above, we encounter a difficulty when we try to show the existence of $\widetilde{\beta}^*$ such that BAD occurs with non-negligible probability. To rely on the witness indistinguishability of the hybrid prover algorithms (which holds against rewinding adversaries), we need to consider a hybrid experiment where we switch the real prover algorithms to the hybrid prover algorithms. The problem is that this switch needs to be made in the main thread and all the look-ahead threads. Given that the hybrid prover and the real prover are not guaranteed to be indistinguishable against rewinding adversaries, we do not know how to establish indistinguishability. (This problem is avoided when the hybrid prover algorithms are used in the honest execution. The key is that the hybrid prover algorithms are used in the look-ahead threads from the beginning, and we do not need to consider any switch in the look-ahead threads.) $\Diamond$

2.3 4-round Construction

Finally, we explain how to reduce the round complexity to 4.

The idea is simple: we run all the components of our construction in parallel. As mentioned above, the special-purpose WIPOK system requires only 3 rounds. Since there exist a 3-round NM commitment scheme [GPR16, GR19] and a 4-round delayed-input SWIAOK system [LS91], we can reduce the round complexity to 4 by running all the components in parallel. (There are several minor issues that need to be addressed. For example, the decommitments of the inner NM commitments must be fixed in the first round so that they can be committed to in the outer NM commitments in parallel. Fortunately, the existing 3-round NM commitments [GPR16, GR19] have a commit-challenge-response structure and therefore have the required property.)

The analysis is largely the same as before. Indeed, the synchronous case can be analyzed in the same way as above. For the asynchronous case, when the round complexity is reduced to 4, there are only a limited number of ways to schedule the messages in the left and right interactions, and each of them can be handled without difficulty.

We conclude this technical overview by noting that our construction can be instantiated using any collision-resistant hash function and injective one-way function. The special-purpose WIPOK system can be instantiated using any injective one-way function [COSV17b], the 3-round NM commitment scheme with the required properties can be instantiated using any injective one-way function [GPR16, GR19], and the 4-round delayed-input SWIAOK system can be instantiated using any collision-resistant hash function [LS91, DPP97, HM96].

3 Preliminaries

3.1 Notations and Conventions

We use λ to denote the security parameter. We use ε to denote the empty string. For any $n \in \mathbb{N}$, we use $[n]$ to denote the set $\{1, \ldots, n\}$. We use poly to denote an unspecified polynomial, negl to denote an unspecified negligible function, and PPT as an abbreviation of "probabilistic polynomial-time." For any NP language $\mathbf{L}$, we use $\mathbf{R_L}$ to denote the corresponding witness relation. For any instance $x \in \mathbf{L}$, we use $\mathbf{R_L}(x)$ to denote the set of all witnesses for $x \in \mathbf{L}$. For any pair of probabilistic interactive Turing machines (P, V) and any $x, y, z \in \{0,1\}^*$, we use $\langle P(y), V(z)\rangle(x)$ to denote the random variable representing the output of V in an interaction between $P(x, y)$ and $V(x, z)$. For a set S, we denote by $s \leftarrow S$ the process of obtaining an element s by sampling uniformly from S. For any probabilistic algorithm Algo and an input x, we denote by $y \leftarrow \mathsf{Algo}(x)$ the process of obtaining an output y by running $\mathsf{Algo}(x)$ with uniform randomness. When we write $\mathsf{Algo}(x; \mathsf{rand})$, it means that Algo is run with input x and random tape rand. We also use $\mathsf{Algo}(x)$ to denote the range of $\mathsf{Algo}(x; \cdot)$, i.e., $\mathsf{Algo}(x) = \{y \mid \exists r \text{ s.t. } y = \mathsf{Algo}(x; r)\}$.

The reader is assumed to have basic knowledge of cryptography, such as the definitions of statistical/computational indistinguishability, one-way functions, pseudorandom generators, and interactive Turing machines. (For basic definitions, see standard textbooks such as [Gol01].) In this paper, every adversarial party is modeled as a non-uniform Turing machine, i.e., one that takes a non-uniform string (denoted by $z \in \{0,1\}^*$) as additional input. While we do not explicitly specify the length of non-uniform strings, the length is always assumed to be bounded by a polynomial in λ.[15]

3.2 Interactive Proofs/Arguments

We recall the definitions of *interactive proofs* and *arguments* [GMR89, BCC88], focusing on those for NP languages.

Definition 1 (Interactive proofs/arguments)**.** *Let $\mathbf{L}$ be an NP language. A pair of PPT interactive Turing machines (P, V) is called an* interactive proof (resp., argument) for $\mathbf{L}$ *if it satisfies the following.*

Completeness. *For every $(x, w) \in \boldsymbol{R_L}$,*

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

Soundness. *For every computationally unbounded (resp., PPT) interactive Turing machine $\mathcal{P}$, there exists a negligible function negl such that for every $x \in \{0,1\}^* \setminus \boldsymbol{L}$ and $z \in \{0,1\}^*$,*

$$\Pr\left[\langle \mathcal{P}(z), \mathsf{V}\rangle(x) = 1\right] \leq \mathsf{negl}(|x|) \ .$$

[15] That is, for any adversary $\mathcal{A}$, we implicitly fix a polynomial $\mathsf{poly}_{\mathcal{A}}$ such that on security parameter λ, the adversary $\mathcal{A}$ takes as auxiliary input a string z of length at most $\mathsf{poly}_{\mathcal{A}}(\lambda)$.

In the above, the machine P *is called the* prover*,* V *is called the* verifier*,* x *is called the* statement*, and the sequence of the messages exchanged between* P *and* V *is called the* transcript*.* $\diamondsuit$

We use the standard shorthand *interactive proof (resp., argument) for NP* to mean an interactive proof (resp., argument) for an NP-complete language; as usual, such a proof (resp., argument) can be used for any NP language via a polynomial-time reduction. Throughout this paper, unless otherwise stated, all interactive proofs/arguments under discussion are for NP in this sense.

Definition 2 (public-coin property)**.** *An interactive proof (resp., argument) is called a* public-coin interactive proof (resp., argument) *if (i) the messages sent by the verifier are random strings that are sampled uniformly and independently in each round and (ii) the verifier deterministically determines its final output based on the statement and the transcript.* $\diamondsuit$

Definition 3 (tag-based interactive proofs/arguments)**.** *An interactive proof (resp., argument) is called a* tag-based interactive proof (resp., argument) *if the prover and the verifier additionally take a string called a* tag *as common input.* $\diamondsuit$

In this paper, the notion of tag-based interactive proofs/arguments is used to define non-malleable zero-knowledge (where tags represent provers' identities). Security notions defined for ordinary interactive proofs/arguments are naturally extended to tag-based ones by requiring them to hold for all tags of appropriate length.

3.3 Adaptive-input Interactive Proofs/Arguments

We recall the definition of *adaptive-input interactive proofs/arguments* [LS91, CPS+16a, CPS+16b], i.e., interactive proofs/arguments such that (i) the statement being proven is used only in the last round and (ii) the security holds even against adversaries selecting the statement adaptively. For concreteness, we focus on 4-round constructions with public verifiability.

3.3.1 Basic Security

Definition 4 (adaptive-input interactive proofs/arguments)**.** *Let* $\boldsymbol{L}$ *be an NP language. A tuple of PPT Turing machines* $(\mathsf{P}_1, \mathsf{P}_2, \mathsf{V}_1, \mathsf{V}_2, \mathsf{V}_{\mathsf{D}})$ *is called a* (4-round) adaptive-input interactive proof (resp., argument) for $\mathbf{L}$ *if it satisfies the following.*

Delayed-input completeness. *For every* $(x, w) \in \boldsymbol{R_L}$*,*

$$\Pr\left[\mathsf{V}_{\mathsf{D}}(x,\tau)=1 \;\middle|\; \begin{array}{l}(\mathsf{m}_1,\mathsf{st}_V)\leftarrow \mathsf{V}_1(1^{|x|})\\ (\mathsf{m}_2,\mathsf{st}_P)\leftarrow \mathsf{P}_1(1^{|x|},\mathsf{m}_1)\\ \mathsf{m}_3\leftarrow \mathsf{V}_2(\mathsf{st}_V,\mathsf{m}_2)\\ \mathsf{m}_4\leftarrow \mathsf{P}_2(\mathsf{st}_P,x,w,\mathsf{m}_3)\\ \tau:=(\mathsf{m}_1,\ldots,\mathsf{m}_4)\end{array}\right]=1.$$

Adaptive-input soundness. *For every pair of computationally unbounded (resp., PPT) Turing machines* $(\mathcal{P}_1, \mathcal{P}_2)$*, there exists a negligible function* negl *such that for every* $\lambda \in \mathbb{N}$ *and* $z \in \{0,1\}^*$*,*

$$\Pr\left[\begin{array}{l|l} x \notin \boldsymbol{L} & (\mathsf{m}_1, \mathsf{st}_V) \leftarrow \mathsf{V}_1(1^\lambda) \\ \wedge\ \mathsf{V}_\mathsf{D}(x, \tau) = 1 & (\mathsf{m}_2, \mathsf{st}_\mathcal{P}) \leftarrow \mathcal{P}_1(1^\lambda, \mathsf{m}_1, z) \\ & \mathsf{m}_3 \leftarrow \mathsf{V}_2(\mathsf{st}_V, \mathsf{m}_2) \\ & (x, \mathsf{m}_4) \leftarrow \mathcal{P}_2(\mathsf{st}_\mathcal{P}, \mathsf{m}_3) \\ & \tau := (\mathsf{m}_1, \ldots, \mathsf{m}_4) \end{array}\right] \leq \mathsf{negl}(\lambda).$$

In the above, the messages m_2*,* m_3*, and* m_4 *are called the* commit*,* challenge*, and* response *messages.* ◊

Remark 2. In the case of public-coin adaptive-input interactive proofs/arguments, the syntax of the verifier is simplified by removing st_V from the output of V_1 and giving only $1^{|x|}$ to V_2 as input. ◊

3.3.2 Additional Security

First, we provide the definition of *adaptive-input computational special soundness*. The following definition is based on the definition of *computational special soundness* by Pass [Pas11] and the definition of *adaptive-input special soundness* by Ciampi et al. [CPS+16b].

Definition 5 (adaptive-input computational special soundness)**.** *Let* $\Pi = (\mathsf{P}_1, \mathsf{P}_2, \mathsf{V}_1, \mathsf{V}_2, \mathsf{V}_\mathsf{D})$ *be a public-coin adaptive-input interactive argument for an NP language* $\boldsymbol{L}$*. Then,* Π *is said to satisfy* adaptive-input computational special soundness *if there exists a deterministic polynomial-time Turing machine* Ext *(called the* extractor*) such that for every pair of PPT Turing machines* $(\mathcal{P}_1, \mathcal{P}_2)$*, there exists a negligible function* negl *such that for every* $\lambda \in \mathbb{N}$ *and* $z \in \{0,1\}^*$*,*

$$\Pr\left[\begin{array}{l|l} \forall b \in \{0,1\} : \mathsf{V}_\mathsf{D}(x_b, \tau_b) = 1 & \mathsf{m}_1 \leftarrow \mathsf{V}_1(1^\lambda) \\ \wedge\ (x_0, w_0) \notin \boldsymbol{R_L} & (\mathsf{m}_2, \mathsf{st}_\mathcal{P}) \leftarrow \mathcal{P}_1(1^\lambda, \mathsf{m}_1, z) \\ & \forall b \in \{0,1\} : \mathsf{m}_{b,3} \leftarrow \mathsf{V}_2(1^\lambda) \\ & (x_0, \mathsf{m}_{0,4}, x_1, \mathsf{m}_{1,4}) \leftarrow \mathcal{P}_2(\mathsf{st}_\mathcal{P}, \mathsf{m}_{0,3}, \mathsf{m}_{1,3}) \\ & \forall b \in \{0,1\} : \tau_b := (\mathsf{m}_1, \mathsf{m}_2, \mathsf{m}_{b,3}, \mathsf{m}_{b,4}) \\ & w_0 := \mathsf{Ext}(x_0, \tau_0, x_1, \tau_1) \end{array}\right] \leq \mathsf{negl}(\lambda).$$

◊

Next, we recall the definition of *adaptive-input statistical witness indistinguishability*.

Definition 6 (adaptive-input statistical WI)**.** *Let* $\Pi = (\mathsf{P}_1, \mathsf{P}_2, \mathsf{V}_1, \mathsf{V}_2, \mathsf{V}_\mathsf{D})$ *be an adaptive-input interactive argument for an NP language* $\boldsymbol{L}$*. Then,* Π *is called* adaptive-input statistical witness-indistinguishable (WI) *if for every triple of computationally unbounded Turing machines* $(\mathcal{V}_1, \mathcal{V}_2, \mathcal{V}_3)$*, there exists a negligible function* negl *such that for every* $\lambda \in \mathbb{N}$ *and* $z \in \{0,1\}^*$*, the random variable* out *satisfies* $\Pr[\mathsf{out} = 1] \leq 1/2 + \mathsf{negl}(\lambda)$ *in the following experiment.*

1. *Compute* $(\mathsf{m}_1, \mathsf{st}_{\mathcal{V},1}) \leftarrow \mathcal{V}_1(1^\lambda, z)$, $(\mathsf{m}_2, \mathsf{st}_P) \leftarrow \mathsf{P}_1(1^\lambda, \mathsf{m}_1)$, *and* $(\mathsf{m}_3, x, w_0, w_1, \mathsf{st}_{\mathcal{V},2}) \leftarrow \mathcal{V}_2(\mathsf{st}_{\mathcal{V},1}, \mathsf{m}_2)$. *If* $|x| \neq \lambda \vee \exists b \in \{0,1\}$ *s.t.* $(x, w_b) \notin \boldsymbol{R_L}$*, abort.*

2. *Sample a random bit* $\beta \in \{0,1\}$ *and compute* $\mathsf{m}_4 \leftarrow \mathsf{P}_2(\mathsf{st}_P, x, w_\beta, \mathsf{m}_3)$.

3. *Compute* $\widetilde{\beta} \leftarrow \mathcal{V}_3(\mathsf{st}_{\mathcal{V},2}, \mathsf{m}_4)$ *and let* $\mathsf{out} = 1$ *if and only if* $\widetilde{\beta} = \beta$. $\Diamond$

Finally, we provide an instantiation that satisfies the above two additional security notions.

Theorem 1. *Assume the existence of a collision-resistant hash function family. Then, there exists a 4-round public-coin adaptive-input interactive argument that satisfies both adaptive-input computational special soundness and adaptive-input statistical witness indistinguishability.*

Proof sketch. It suffices to instantiate the delayed-input argument of knowledge of Lapidot and Shamir [LS91] with a 2-round statistically hiding commitment scheme (which can be obtained from any collision-resistant hash function [DPP97, HM96]) and use a trick by Ciampi et al. [COSV17b, Appendix C.1] to obtain delayed-input computational special soundness. □

3.4 Statistical Non-malleable Zero-knowledge

We recall the definition of *non-malleable zero-knowledge* [DDN00]. Following recent work [GRRV21, COSV17c, COSV17a, BCO+24], we use the definition based on the notion of *simulation extractability* [PR08b] (which is known to imply the original definition of non-malleable zero-knowledge [DDN00]). Below, we provide the statistical zero-knowledge version of the definition in the *one-many* setting [PR08a] with a weak form of *adaptive input selection* [LP11a]. (See Remark 3 below for details about adaptive input selection.) The text below is adapted from [LP11a, Kiy20].

Let (P, V) be a tag-based, adaptive-input interactive argument for an NP language $\mathbf{L}$. For a man-in-the-middle adversary $\mathcal{A}$, consider a probabilistic experiment in which $\mathcal{A}$ participates in one left interaction and multiple right interactions.

- In the left interaction, $\mathcal{A}$ interacts with an honest prover of (P, V) as a verifier using a statement x and a tag tag.

- In the right interactions, $\mathcal{A}$ interacts with honest verifiers of (P, V) as a prover using statements $\widetilde{x}_1, \ldots, \widetilde{x}_m$ and tags $\widetilde{\mathsf{tag}}_1, \ldots, \widetilde{\mathsf{tag}}_m$.

More concretely, the experiment proceeds as follows. Prior to the interactions, all parties in the experiment receive as common input the security parameter λ in unary, and $\mathcal{A}$ also receives as auxiliary input $z \in \{0,1\}^*$. Then, $\mathcal{A}$ starts interacting with the prover and the verifiers in the left and right interactions while fully controlling the message schedule. (In particular, $\mathcal{A}$ can run the left and right interactions concurrently.) In the left (resp., each right) interaction, the tag $\mathsf{tag} \in \{0,1\}^\lambda$ (resp., $\widetilde{\mathsf{tag}}_i \in \{0,1\}^\lambda$) is selected by $\mathcal{A}$ at the beginning of the interaction, and the statement $x \in \{0,1\}^\lambda$

(resp., $\widetilde{x}_i \in \{0,1\}^\lambda$) is selected by $\mathcal{A}$ when sending the last verifier message (resp., prover message) to the honest party. The witness used by the honest prover in the left interaction is chosen by a *witness-selecting machine* M. Specifically, M is a PPT Turing machine such that when $\mathcal{A}$ selects the statement x in the left interaction, M takes as input the statement x, the current view of $\mathcal{A}$, and an auxiliary input z', and then picks a witness w as the private input to the honest prover. Let $\mathsf{view}_{\mathcal{A},M}(\lambda, z, z')$ be the random variable representing the joint view of $\mathcal{A}$ and the honest verifiers in the above experiment. We say that the pair (z, z') of auxiliary inputs is *good* if in the above experiment, (i) $\mathcal{A}$ always selects a true statement of length λ in the left interaction and (ii) M always picks a valid witness for the statement selected by $\mathcal{A}$. Let $Z_{\mathcal{A},M}(\lambda)$ be the set of good auxiliary-input pairs. Then, roughly speaking, (P, V) is *one-many statistical non-malleable zero-knowledge (SNMZK) with weak adaptive input selection* if for every PPT adversary $\mathcal{A}$, there exists a PPT machine called the *simulator-extractor* such that for every PPT witness-selecting machine M, every λ, and every good (z, z'), the simulator-extractor can statistically simulate the view of $\mathcal{A}$ in the above experiment while extracting witnesses for the statements proven by $\mathcal{A}$ in the accepted right interactions. The formal definition is given below.

Definition 7. *A tag-based adaptive-input interactive argument* (P, V) *for an NP language* $\boldsymbol{L}$ *is said to be* one-many statistical non-malleable zero-knowledge with weak adaptive input selection *if for every polynomial* $m(\cdot)$ *and every PPT man-in-the-middle adversary* $\mathcal{A}$ *that participates in at most* $m = m(\lambda)$ *right interactions, there exists an expected polynomial-time Turing machine* $\mathcal{SE}$ *(called the* simulator-extractor*) such that for every PPT witness-selecting machine* M*, the following hold.*

1. ***Syntax.*** *For any input* $(1^\lambda, z)$*, the output of* $\mathcal{SE}$ *is parsed as a pair consisting of a simulated view and a sequence of witness candidates* $\{\widetilde{w}_i\}_{i\in[m]}$*.*

2. ***Statistical simulation.*** *Let* $\mathsf{sim\text{-}view}(\lambda, z)$ *be a random variable representing the first component of the output of* $\mathcal{SE}(1^\lambda, z)$*. Then, for every sequence* $\{(z_\lambda, z'_\lambda)\}_{\lambda\in\mathbb{N}}$ *such that* $(z_\lambda, z'_\lambda) \in Z_{\mathcal{A},M}(\lambda)$*, the ensembles* $\{\mathsf{view}_{\mathcal{A},M}(\lambda, z_\lambda, z'_\lambda)\}_{\lambda\in\mathbb{N}}$ *and* $\{\mathsf{sim\text{-}view}(\lambda, z_\lambda)\}_{\lambda\in\mathbb{N}}$ *are statistically indistinguishable.*

3. ***Witness extraction.*** *For every* $\lambda \in \mathbb{N}$ *and* $(z, z') \in Z_{\mathcal{A},M}(\lambda)$*, the following holds. Let* $(\mathsf{view}, \{\widetilde{w}_i\}_{i\in[m]})$ *denote the output of* $\mathcal{SE}(1^\lambda, z)$*. Let* $\widetilde{x}_1, \ldots, \widetilde{x}_m$ *be the statements of the right interactions in* view*, and let* tag *and* $\widetilde{\mathsf{tag}}_1, \ldots, \widetilde{\mathsf{tag}}_m$ *be the tags of the left and the right interactions in* view*. Then, except with negligible probability, it holds* $(\widetilde{x}_i, \widetilde{w}_i) \in \boldsymbol{R_L}$ *for every* $i \in [m]$ *such that the* i*-th right interaction is accepting and* $\widetilde{\mathsf{tag}}_i \neq \mathsf{tag}$*.* $\Diamond$

Remark 3. The above formalization of adaptive input selection is weaker than the original one [LP11a] because the original formalization considers the case where the witness-selecting machine is computationally unbounded and takes the views of all the parties (including the honest ones) as input. In other words, we focus on the case where a witness for the left statement x can be computed efficiently using a "super witness" z'. (See [LP11a, Section 1] for more discussion.) $\Diamond$

3.5 Synchronous Non-malleable Commitments

We recall the definition of *non-malleable commitment schemes* [DDN00]. The following definition is based on the definition of Lin et al. [LPV08]. For concreteness, we specialize the definition in two respects: we consider 3-round constructions with public verifiability,[16] and we formulate non-malleability in the synchronous-message setting.

Definition 8. *A tuple of PPT Turing machines* $(\mathsf{NMC.S}_1, \mathsf{NMC.S}_2, \mathsf{NMC.R}, \mathsf{NMC.R}_\mathsf{D})$ *is called a* 3-round synchronous non-malleable commitment scheme *if it satisfies the following.*

Correctness. *For every* $\lambda \in \mathbb{N}$, $\mathsf{tag} \in \{0,1\}^\lambda$, *and* $m \in \{0,1\}^\lambda$,

$$\Pr\left[\mathsf{NMC.R}_\mathsf{D}(\tau, \mathsf{dec}) = m \;\middle|\; \begin{array}{l} (\mathsf{nm}_1, \mathsf{dec}, \mathsf{st}) \leftarrow \mathsf{NMC.S}_1(1^\lambda, \mathsf{tag}, m) \\ \mathsf{nm}_2 \leftarrow \mathsf{NMC.R}(1^\lambda, \mathsf{tag}, \mathsf{nm}_1) \\ \mathsf{nm}_3 \leftarrow \mathsf{NMC.S}_2(\mathsf{st}, \mathsf{nm}_2) \\ \tau := (\mathsf{nm}_1, \mathsf{nm}_2, \mathsf{nm}_3) \end{array}\right] = 1.$$

Statistical binding. *For every pair of computationally unbounded Turing machines* $(\mathcal{S}_1, \mathcal{S}_2)$*, there exists a negligible function* negl *such that for every* $\lambda \in \mathbb{N}$ *and* $z \in \{0,1\}^*$,

$$\Pr\left[\begin{array}{l} m \neq m' \\ \wedge\, m \neq \bot \wedge m' \neq \bot \\ \wedge\, \mathsf{NMC.R}_\mathsf{D}(\tau, \mathsf{dec}) = m \\ \wedge\, \mathsf{NMC.R}_\mathsf{D}(\tau, \mathsf{dec}') = m' \end{array} \;\middle|\; \begin{array}{l} (\mathsf{tag}, \mathsf{nm}_1, \mathsf{st}) \leftarrow \mathcal{S}_1(1^\lambda, z) \\ \mathsf{nm}_2 \leftarrow \mathsf{NMC.R}(1^\lambda, \mathsf{tag}, \mathsf{nm}_1) \\ (\mathsf{nm}_3, m, \mathsf{dec}, m', \mathsf{dec}') \leftarrow \mathcal{S}_2(\mathsf{st}, \mathsf{nm}_2) \\ \tau := (\mathsf{nm}_1, \mathsf{nm}_2, \mathsf{nm}_3) \end{array}\right] \leq \mathsf{negl}(\lambda).$$

Synchronous non-malleability. *For any tuple of PPT Turing machines* $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3, \mathcal{A}_D)$*, there exists a negligible function* negl *such that for any* $\lambda \in \mathbb{N}$ *and* $z \in \{0,1\}^*$*, the random variable* out *satisfies* $\Pr[\mathsf{out} = 1] \leq 1/2 + \mathsf{negl}(\lambda)$ *in the following experiment (see Figure 4).*

- **Initialization.** *Compute* $(m, \mathsf{tag}, \mathsf{st}_{\mathcal{A},0}) \leftarrow \mathcal{A}_0(1^\lambda, z)$*, where* $m \in \{0,1\}^\lambda$, $\mathsf{tag} \in \{0,1\}^\lambda$.[17] *Then, let* m^* *be defined as follows: sample a random bit* $b \in \{0,1\}$*; if* $b = 0$*, let* $m^* := m$*; otherwise, let* $m^* := m'$ *for a random string* $m' \in \{0,1\}^\lambda$.
- **Round 1.** *Compute* $(\mathsf{nm}_1, \mathsf{dec}, \mathsf{st}_S) \leftarrow \mathsf{NMC.S}_1(1^\lambda, \mathsf{tag}, m^*)$ *and* $(\widetilde{\mathsf{tag}}, \widetilde{\mathsf{nm}}_1, \mathsf{st}_{\mathcal{A},1}) \leftarrow \mathcal{A}_1(\mathsf{st}_{\mathcal{A},0}, \mathsf{nm}_1)$.
- **Round 2.** *Compute* $\widetilde{\mathsf{nm}}_2 \leftarrow \mathsf{NMC.R}(1^\lambda, \widetilde{\mathsf{tag}}, \widetilde{\mathsf{nm}}_1)$ *and* $(\mathsf{nm}_2, \mathsf{st}_{\mathcal{A},2}) \leftarrow \mathcal{A}_2(\mathsf{st}_{\mathcal{A},1}, \widetilde{\mathsf{nm}}_2)$.
- **Round 3.** *Compute* $\mathsf{nm}_3 \leftarrow \mathsf{NMC.S}_2(\mathsf{st}_S, \mathsf{nm}_2)$ *and* $(\mathsf{st}_{\mathcal{A},3}, \widetilde{\mathsf{nm}}_3) \leftarrow \mathcal{A}_3(\mathsf{st}_{\mathcal{A},2}, \mathsf{nm}_3)$.

[16] That is, decommitment verification does not require the receiver's internal state.

[17] If $m \notin \{0,1\}^\lambda$ or $\mathsf{tag} \notin \{0,1\}^\lambda$, then m and tag are redefined to be arbitrary default values of length λ.



Figure 4: Message flow in the synchronous non-malleability experiment (excluding the final output step).

Final output. *Let $\widetilde{m}$ be the committed value of $\widetilde{\tau} := (\widetilde{\mathsf{nm}}_1, \widetilde{\mathsf{nm}}_2, \widetilde{\mathsf{nm}}_3)$ defined as follows: let $\widetilde{M} := \{\widetilde{m} \mid \exists \widetilde{\mathsf{dec}}$ s.t. $\mathsf{NMC.R_D}(\widetilde{\tau}, \widetilde{\mathsf{dec}}) = \widetilde{m} \neq \bot\}$; if $\mathsf{tag} = \widetilde{\mathsf{tag}}$ or $|\widetilde{M}| \neq 1$, let $\widetilde{m} := \bot$; otherwise, let $\widetilde{m}$ be the unique value such that $\widetilde{m} \in \widetilde{M}$. Then, compute $b' \leftarrow \mathcal{A}_D(\mathsf{st}_{\mathcal{A},3}, \widetilde{m})$ and let $\mathsf{out} := 1$ if and only if $b' = b$.* $\lozenge$

We note that, for convenience, the above definition uses a somewhat nonstandard syntax and formulation. Specifically, (i) the decommitment verification algorithm $\mathsf{NMC.R_D}$ outputs either the committed value (when given a valid decommitment) or $\bot$ (when given an invalid decommitment), and (ii) the non-malleability experiment is formulated as a distinguishing game in which the adversary, after being given the committed value of the commitment it creates, tries to decide whether the honest sender committed to the string chosen by the adversary or to an independent uniformly random string. We also note that synchronous non-malleability trivially implies the following hiding property.

Computational hiding. For any tuple of PPT Turing machines $\mathcal{R} = (\mathcal{R}_0, \mathcal{R}_1, \mathcal{R}_\mathsf{D})$, there exists a negligible function negl such that for any $\lambda \in \mathbb{N}$ and $z \in \{0,1\}^*$, the random variable out satisfies $\Pr[\mathsf{out} = 1] \leq 1/2 + \mathsf{negl}(\lambda)$ in the following experiment.

Initialization. Compute $(m, \mathsf{tag}, \mathsf{st}_{\mathcal{R},0}) \leftarrow \mathcal{R}_0(1^\lambda, z)$.[18] Then, let m^* be defined as follows: sample a random bit $b \in \{0,1\}$; if $b = 0$, let $m^* := m$; otherwise, let $m^* := m'$ for a random string $m' \in \{0,1\}^\lambda$.

Rounds 1, 2, 3. Compute $(\mathsf{nm}_1, \mathsf{dec}, \mathsf{st}_S) \leftarrow \mathsf{NMC.S}_1(1^\lambda, \mathsf{tag}, m^*)$, $(\mathsf{nm}_2, \mathsf{st}_{\mathcal{R},1}) \leftarrow \mathcal{R}_1(\mathsf{st}_{\mathcal{R},0}, \mathsf{nm}_1)$, and $\mathsf{nm}_3 \leftarrow \mathsf{NMC.S}_2(\mathsf{st}_S, \mathsf{nm}_2)$.

Final output. Compute $b' \leftarrow \mathcal{R}_\mathsf{D}(\mathsf{st}_{\mathcal{R},1}, \mathsf{nm}_3)$ and let $\mathsf{out} := 1$ if and only if $b' = b$.

As discussed in Section 2.3, we require a non-malleable commitment scheme that satisfies an additional property. This property is summarized in the following theorem. At a high level, we require that any Round-1 message uniquely determines a "decommitment candidate" that can serve as a valid decommitment when the commit

[18] As in the non-malleability experiment, if $m \notin \{0,1\}^\lambda$ or $\mathsf{tag} \notin \{0,1\}^\lambda$, then m and tag are redefined to be arbitrary default values of length λ.

phase is completed honestly.[19] In particular, we require that (i) any valid decommitment is also a valid decommitment candidate and (ii) any valid decommitment candidate is also a valid decommitment when an honest committer completes the commit phase without aborting.

Theorem 2. *Assume the existence of injective one-way functions. Then, there exists a 3-round synchronous non-malleable commitment scheme satisfying the following* unique decommitment candidate *property: There exists a deterministic polynomial-time Turing machine* $\mathsf{NMC.R_{D,weak}}$ *such that the following hold.*

1. ***Uniqueness.*** *For every* $\mathsf{nm}_1 \in \{0,1\}^*$, $|\{\mathsf{dec} \mid \mathsf{NMC.R_{D,weak}}(\mathsf{nm}_1, \mathsf{dec}) = 1\}| \leq 1$.
2. ***Correctness.*** *For every* $\tau = (\mathsf{nm}_1, \mathsf{nm}_2, \mathsf{nm}_3) \in (\{0,1\}^*)^3$ *and* dec *such that* $\mathsf{NMC.R_D}(\tau, \mathsf{dec}) \neq \bot$, *it holds* $\mathsf{NMC.R_{D,weak}}(\mathsf{nm}_1, \mathsf{dec}) = 1$.
3. ***Soundness.*** *For every computationally unbounded Turing machine* $\mathcal{A}$ *and every* $\lambda \in \mathbb{N}$, $\mathsf{tag} \in \{0,1\}^\lambda$, $m \in \{0,1\}^\lambda$, *and* $z \in \{0,1\}^*$,

$$\Pr\left[\begin{array}{l|l} \exists \mathsf{dec}' \text{ s.t.} & (\mathsf{nm}_1, \mathsf{dec}, \mathsf{st}) \leftarrow \mathsf{NMC.S}_1(1^\lambda, \mathsf{tag}, m) \\ \mathsf{NMC.R_{D,weak}}(\mathsf{nm}_1, \mathsf{dec}') = 1 & \mathsf{nm}_2 \leftarrow \mathcal{A}(1^\lambda, \mathsf{tag}, \mathsf{nm}_1, z) \\ \wedge\ \mathsf{nm}_3 \neq \bot & \mathsf{nm}_3 \leftarrow \mathsf{NMC.S}_2(\mathsf{st}, \mathsf{nm}_2) \\ \wedge\ \mathsf{NMC.R_D}(\tau, \mathsf{dec}') \neq m & \tau := (\mathsf{nm}_1, \mathsf{nm}_2, \mathsf{nm}_3) \end{array}\right] = 0.$$

Furthermore, when a commitment is made with security parameter λ *and message length* λ, *the length of the corresponding decommitment is* λ.

Proof. The first part follows from [GPR15, Claim 5 and Theorem 1], where the unique decommitment candidate property can be verified by inspecting the protocol description in [GPR15, Figure 2]. Specifically, it suffices to see that (i) the commit phase has a commit-challenge-response structure such that the committer gives a non-interactive commitment in the first round, the receiver responds with a public-coin challenge, and the committer replies with a response message in the third round, and (ii) the decommitment consists of the decommitment of the non-interactive commitment. (The receiver checks that the decommitment is valid with respect to the non-interactive commitment and is also consistent with the entire transcript of the commit phase. If the check passes, the receiver recovers the committed value of the whole commitment from the decommitment and the transcript.) Let $\mathsf{NMC.R_{D,weak}}$ be an algorithm that checks whether the given decommitment is a valid decommitment of the non-interactive commitment, and suppose that the underlying non-interactive commitment scheme is instantiated with the perfectly binding non-interactive bit-commitment scheme based on the hard-core predicate of an injective one-way function [Blu82]. Then, the uniqueness holds because the non-interactive commitment has a unique decommitment. The correctness and soundness follow from the construction.

The furthermore part can also be verified by inspection. Specifically, it suffices to recall that the decommitment of the underlying non-interactive commitment consists

[19] We do not require that the Round-1 message uniquely determines the committed value.

of the committer randomness, whose length can be shortened to λ using a pseudorandom generator.[20] The unique decommitment candidate property is preserved as long as the pseudorandom generator is injective, and an injective pseudorandom generator can be constructed from any injective one-way function. □

4 Tool: Special-purpose WIPOK System

We present the main tool that we use in our 4-round SNMZK argument system. As described in Section 2, it is a 3-round public-coin proof of knowledge for statements of the form $x_0 \in \mathbf{L} \vee x_1 \in \mathbf{L}$, and we require it to satisfy a certain rewinding security notion. We start by defining the syntax of such 3-round proof-of-knowledge systems and their basic security.

Definition 9. *Let $\boldsymbol{L}$ be an NP language. A tuple of PPT Turing machines* $(\mathsf{WI.P}_1, \mathsf{WI.P}_2, \mathsf{WI.V}, \mathsf{WI.V}_\mathsf{D})$ *is called a* 3-round adaptive-input OR-indistinguishable special-sound proof *for $\boldsymbol{L}$ if it satisfies the following.*

Delayed-input correctness. *For every $\beta \in \{0,1\}$, $(x_\beta, w_\beta) \in \boldsymbol{R_L}$, and $x_{1-\beta} \in \{0,1\}^{|x_\beta|}$,*

$$\Pr\left[\mathsf{WI.V}_\mathsf{D}(x_0, x_1, \tau) = 1 \;\middle|\; \begin{array}{l} (\mathsf{wi}_1, \mathsf{st}) \leftarrow \mathsf{WI.P}_1(1^{|x_\beta|}, \beta) \\ \mathsf{wi}_2 \leftarrow \mathsf{WI.V}(1^{|x_\beta|}) \\ \mathsf{wi}_3 \leftarrow \mathsf{WI.P}_2(\mathsf{st}, x_0, x_1, w_\beta, \mathsf{wi}_2) \\ \tau \coloneqq (\mathsf{wi}_1, \mathsf{wi}_2, \mathsf{wi}_3) \end{array}\right] = 1.$$

Delayed-input special soundness. *There exists a deterministic polynomial-time Turing machine* Ext *(called the* extractor*) and a polynomial-time computable function* $\mathsf{EQ} : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}$ *such that the following hold.*

1. *The predicate* EQ *determines when two challenge messages are treated as equal and satisfies the following collision bound. For every $\lambda \in \mathbb{N}$,*

$$\Pr\left[\mathsf{EQ}(\mathsf{wi}_{0,2}, \mathsf{wi}_{1,2}) = 1 \;\middle|\; \begin{array}{l} \mathsf{wi}_{0,2} \leftarrow \mathsf{WI.V}(1^\lambda) \\ \mathsf{wi}_{1,2} \leftarrow \mathsf{WI.V}(1^\lambda) \end{array}\right] \leq \frac{1}{2^\lambda}.$$

2. *For each $b \in \{0,1\}$, fix any $x_b = (x_{b,0}, x_{b,1}) \in (\{0,1\}^*)^2$ and $\tau_b \coloneqq (\mathsf{wi}_{b,1}, \mathsf{wi}_{b,2}, \mathsf{wi}_{b,3}) \in (\{0,1\}^*)^3$ such that $|x_0| = |x_1|$. Then, if the two transcripts τ_0 and τ_1 are accepting, have the same commit message, and have challenge messages that are not treated as equal by* EQ*, then* Ext *outputs a valid witness for one of the two sub-statements in the first statement pair. More formally, if $\mathsf{wi}_{0,1} = \mathsf{wi}_{1,1}$, $\mathsf{EQ}(\mathsf{wi}_{0,2}, \mathsf{wi}_{1,2}) = 0$, and $\mathsf{WI.V}_\mathsf{D}(x_{b,0}, x_{b,1}, \tau_b) = 1$ for each $b \in \{0,1\}$, then, $\mathsf{Ext}(x_0, \tau_0, x_1, \tau_1)$ outputs a value w such that $\exists \beta \in \{0,1\} : (x_{0,\beta}, w) \in \boldsymbol{R_L}$.*

[20] Since the underlying non-interactive commitment scheme is a bit commitment scheme, the committed value can be efficiently recovered from the commitment and the committer randomness.

Adaptive-input OR indistinguishability. *For every pair of PPT Turing machines* $(\mathcal{V}_1, \mathcal{V}_2)$*, there exists a negligible function* negl *such that for every* $\lambda \in \mathbb{N}$ *and* $z \in \{0,1\}^*$*, the random variable* out *satisfies* $\Pr[\mathsf{out} = 1] \leq 1/2 + \mathsf{negl}(\lambda)$ *in the following experiment.*

1. *Sample a random bit* $\beta \in \{0,1\}$ *and compute* $(\mathsf{wi}_1, \mathsf{st}_P) \leftarrow \mathsf{WI.P}_1(1^\lambda, \beta)$.
2. *Compute* $(\mathsf{wi}_2, \{x_b, w_b\}_{b\in\{0,1\}}, \mathsf{st}_\mathcal{V}) \leftarrow \mathcal{V}_1(1^\lambda, \mathsf{wi}_1, z)$. *If* $\exists b \in \{0,1\}$ *s.t.* $(x_b, w_b) \notin \boldsymbol{R_L} \vee |x_b| \neq \lambda$*, abort.*
3. *Compute* $\mathsf{wi}_3 \leftarrow \mathsf{WI.P}_2(\mathsf{st}_P, x_0, x_1, w_\beta, \mathsf{wi}_2)$.
4. *Compute* $\widetilde{\beta} \leftarrow \mathcal{V}_2(\mathsf{st}_\mathcal{V}, \mathsf{wi}_3)$ *and let* $\mathsf{out} = 1$ *if and only if* $\widetilde{\beta} = \beta$.

Remark 4. This adaptive-input OR indistinguishability is weaker than the standard witness indistinguishability since it only requires indistinguishability between a proof generated with a witness for $x_0 \in \mathbf{L}$ and a proof generated with a witness for $x_1 \in \mathbf{L}$. (When $x_0 \in \mathbf{L}$ and $x_1 \in \mathbf{L}$ each have a unique witness, this notion implies standard witness indistinguishability.) $\lozenge$

Next, we summarize the required rewinding security notion in the following theorem.

Theorem 3. *Let* $\boldsymbol{L}$ *be an NP language, and assume the existence of injective one-way functions. Then, there exists a 3-round adaptive-input OR-indistinguishable special-sound proof,* $\mathsf{WI} = (\mathsf{WI.P}_1, \mathsf{WI.P}_2, \mathsf{WI.V}, \mathsf{WI.V_D})$*, for* $\boldsymbol{L}$*. Furthermore, there exists a pair of PPT Turing machines* $(\mathsf{WI.}\widehat{\mathsf{P}}_1, \mathsf{WI.}\widehat{\mathsf{P}}_2)$ *and an additional PPT Turing machine* $\mathsf{WI.}\widehat{\mathsf{P}}_{1.5}$ *satisfying the following.*

Delayed-input correctness of hybrid prover. *For every* $\beta \in \{0,1\}$*,* $(x_0, w_0) \in \boldsymbol{R_L}$*, and* $(x_1, w_1) \in \boldsymbol{R_L}$ *such that* $|x_0| = |x_1|$*,*

$$\Pr\left[\mathsf{WI.V_D}(x_0, x_1, \tau) = 1 \;\middle|\; \begin{array}{l} (\mathsf{wi}_1, \mathsf{st}, \mathsf{aux}) \leftarrow \mathsf{WI.}\widehat{\mathsf{P}}_1(1^{|x_\beta|}, \beta) \\ \mathsf{wi}_2 \leftarrow \mathsf{WI.V}(1^{|x_\beta|}) \\ \mathsf{wi}_3 \leftarrow \mathsf{WI.}\widehat{\mathsf{P}}_2(\mathsf{st}, \mathsf{aux}, x_0, x_1, w_0, w_1, \mathsf{wi}_2) \\ \tau := (\mathsf{wi}_1, \mathsf{wi}_2, \mathsf{wi}_3) \end{array}\right] = 1.$$

Indistinguishability between real and hybrid provers.

1. *For every* $\lambda \in \mathbb{N}$ *and* $\beta \in \{0,1\}$*, the following two distributions are identically distributed.*

$$\left\{(\mathsf{wi}_1, \mathsf{st}, \mathsf{aux}) \mid (\mathsf{wi}_1, \mathsf{st}, \mathsf{aux}) \leftarrow \mathsf{WI.}\widehat{\mathsf{P}}_1(1^\lambda, \beta)\right\}.$$

$$\left\{(\mathsf{wi}_1, \mathsf{st}, \mathsf{aux}) \;\middle|\; \begin{array}{l} (\mathsf{wi}_1, \mathsf{st}) \leftarrow \mathsf{WI.P}_1(1^\lambda, \beta) \\ \mathsf{aux} \leftarrow \mathsf{WI.}\widehat{\mathsf{P}}_{1.5}(1^\lambda, \beta) \end{array}\right\}.$$

2. *For every PPT Turing machine* $\mathcal{D}$*, there exists a negligible function* negl *such that for every* $(x_0, w_0) \in \boldsymbol{R_L}$*,* $(x_1, w_1) \in \boldsymbol{R_L}$*,* $\beta \in \{0,1\}$*,* $(\mathsf{wi}_1, \mathsf{st}) \in \mathsf{WI.P}_1(1^{|x_\beta|}, \beta)$*,* $\mathsf{wi}_2 \in \{0,1\}^*$*, and* $z \in \{0,1\}^*$ *such that* $|x_0| = |x_1|$*,*

$$\left|\begin{array}{l} \Pr[\mathcal{D}(\mathsf{params}, \mathsf{wi}_3) = 1 \mid \mathsf{wi}_3 \leftarrow \mathsf{WI.P}_2(\mathsf{st}, x_0, x_1, w_\beta, \mathsf{wi}_2)] \\ -\Pr\left[\mathcal{D}(\mathsf{params}, \mathsf{wi}_3) = 1 \;\middle|\; \begin{array}{l} \mathsf{aux} \leftarrow \mathsf{WI.}\widehat{\mathsf{P}}_{1.5}(1^{|x_\beta|}, \beta) \\ \mathsf{wi}_3 \leftarrow \mathsf{WI.}\widehat{\mathsf{P}}_2(\mathsf{st}, \mathsf{aux}, x_0, x_1, w_0, w_1, \mathsf{wi}_2) \end{array}\right] \end{array}\right| \leq \mathsf{negl}(|x_\beta|),$$

where $\mathsf{params} := (x_0, x_1, w_0, w_1, \beta, \mathsf{wi}_1, \mathsf{wi}_2, \mathsf{st}, z)$.[21]

Adaptive-input (rewinding) OR-indistinguishability of hybrid prover. *For every PPT oracle Turing machine $\mathcal{V}$, there exists a negligible function* negl *such that for every $\lambda \in \mathbb{N}$ and $z \in \{0,1\}^*$,*

$$\Pr\left[\beta = \widetilde{\beta} \;\middle|\; \begin{array}{l} \beta \leftarrow \{0,1\} \\ (\mathsf{wi}_1, \mathsf{st}, \mathsf{aux}) \leftarrow \mathsf{WI}.\widehat{\mathsf{P}}_1(1^\lambda, \beta) \\ \widetilde{\beta} \leftarrow \mathcal{V}^{\mathcal{O}(\mathsf{st},\mathsf{aux},\cdot)}(1^\lambda, \mathsf{wi}_1, z) \end{array}\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda),$$

where the oracle $\mathcal{O}$ is defined as follows.

- $\mathcal{O}(\mathsf{st}, \mathsf{aux}, x_0, x_1, w_0, w_1, \mathsf{wi}_2)$*: If $|x_b| = \lambda$ and $(x_b, w_b) \in \boldsymbol{R_L}$ for every $b \in \{0,1\}$, return* $\mathsf{WI}.\widehat{\mathsf{P}}_2(\mathsf{st}, \mathsf{aux}, x_0, x_1, w_0, w_1, \mathsf{wi}_2)$*. Otherwise, return* $\bot$.

As stated in Section 2, Theorem 3 can be proved by using the WIPOK system given by Ciampi et al. [COSV17b]. (For concreteness, we provide a proof in Appendix A.) In the following, we give a lemma that we use in the analysis of our 4-round SNMZK argument. Intuitively, the lemma formalizes the argument sketched in Figure 3 in Section 2.2.

Lemma 1. *Fix any computationally unbounded Turing machine $\mathcal{A}_1$ and pair of PPT Turing machines $(\mathcal{A}_2, \widehat{\mathcal{A}}_2)$ such that for every $\lambda \in \mathbb{N}$, $\beta \in \{0,1\}$, and $z \in \{0,1\}^*$, $\mathcal{A}_1$ is well-behaving in the following sense.*

$$\Pr\left[\begin{array}{l} |\mathsf{st}_{\mathcal{A}}| \leq \mathsf{poly}(\lambda) \\ \wedge\, \forall b \in \{0,1\} : |x_b| = \lambda \\ \wedge\, \forall b \in \{0,1\} : (x_b, w_b) \in \boldsymbol{R_L} \end{array} \;\middle|\; \begin{array}{l} (\mathsf{wi}_1, \mathsf{st}, \mathsf{aux}) \leftarrow \mathsf{WI}.\widehat{\mathsf{P}}_1(1^\lambda, \beta) \\ (x_0, x_1, w_0, w_1, \mathsf{wi}_2, \mathsf{st}_{\mathcal{A}}) \leftarrow \mathcal{A}_1(1^\lambda, \mathsf{wi}_1, z) \end{array}\right] = 1.$$

Also, let $\mathcal{O}$ be the following oracle.

- $\mathcal{O}(\mathsf{st}, \mathsf{aux}, x_0, x_1, w_0, w_1, \mathsf{wi}_2)$*: If $|x_b| = \lambda$ and $(x_b, w_b) \in \boldsymbol{R_L}$ for every $b \in \{0,1\}$, return* $\mathsf{WI}.\widehat{\mathsf{P}}_2(\mathsf{st}, \mathsf{aux}, x_0, x_1, w_0, w_1, \mathsf{wi}_2)$*. Otherwise, return* $\bot$.

Then, there exists a negligible function negl *such that for any polynomial p, there exists a polynomial q such that for every $\lambda \in \mathbb{N}$, $\beta \in \{0,1\}$, and $z \in \{0,1\}^*$, if it holds*

$$\Pr\left[\begin{array}{l} \mathcal{A}_2(\mathsf{st}_{\mathcal{A}}, \mathsf{wi}_3) = 1 \\ \wedge\, \widehat{\mathcal{A}}_2^{\mathcal{O}(\mathsf{st},\mathsf{aux},\cdot)}(\mathsf{st}_{\mathcal{A}}) = 1 \end{array} \;\middle|\; \begin{array}{l} (\mathsf{wi}_1, \mathsf{st}, \mathsf{aux}) \leftarrow \mathsf{WI}.\widehat{\mathsf{P}}_1(1^\lambda, \beta) \\ (x_0, x_1, w_0, w_1, \mathsf{wi}_2, \mathsf{st}_{\mathcal{A}}) \leftarrow \mathcal{A}_1(1^\lambda, \mathsf{wi}_1, z) \\ \mathsf{wi}_3 \leftarrow \mathsf{WI}.\widehat{\mathsf{P}}_2(\mathsf{st}, \mathsf{aux}, x_0, x_1, w_0, w_1, \mathsf{wi}_2) \end{array}\right] \geq \frac{1}{p(\lambda)}, \tag{4}$$

then it also holds

$$\Pr\left[\begin{array}{l} \mathcal{A}_2(\mathsf{st}_{\mathcal{A}}, \mathsf{wi}_3) = 1 \\ \wedge\, \widehat{\mathcal{A}}_2^{\mathcal{O}(\mathsf{st},\mathsf{aux},\cdot)}(\mathsf{st}_{\mathcal{A}}) = 1 \end{array} \;\middle|\; \begin{array}{l} (\mathsf{wi}_1, \mathsf{st}, \mathsf{aux}) \leftarrow \mathsf{WI}.\widehat{\mathsf{P}}_1(1^\lambda, \beta) \\ (x_0, x_1, w_0, w_1, \mathsf{wi}_2, \mathsf{st}_{\mathcal{A}}) \leftarrow \mathcal{A}_1(1^\lambda, \mathsf{wi}_1, z) \\ \boxed{\mathsf{wi}_3 \leftarrow \mathsf{WI}.\mathsf{P}_2(\mathsf{st}, x_0, x_1, w_\beta, \mathsf{wi}_2)} \end{array}\right] \geq \frac{1}{q(\lambda)} - \mathsf{negl}(\lambda). \tag{5}$$

[21] Note that aux is not given to $\mathcal{D}$.

We emphasize that the oracle $\mathcal{O}$ continues to use the hybrid prover algorithm $\mathsf{WI}.\widehat{\mathsf{P}}_2$ in (5). (In the terminology used in Section 2.2, adversary $\mathcal{A}_2$ represents the "main thread" and $\widehat{\mathcal{A}}_2$ represents the "look-ahead threads.")

Proof of Lemma 1. Fix any p, λ, β, and $z \in \{0,1\}^*$ such that we have (4). Let $\varepsilon := 1/p(\lambda)$. From (4) and the indistinguishability between the real and hybrid provers, we have

$$\Pr\left[\begin{array}{l|l} \mathcal{A}_2(\mathsf{st}_{\mathcal{A}}, \mathsf{wi}_3) = 1 \\ \wedge\, \widehat{\mathcal{A}}_2^{\mathcal{O}(\mathsf{st},\mathsf{aux},\cdot)}(\mathsf{st}_{\mathcal{A}}) = 1 & \begin{array}{l} \boxed{(\mathsf{wi}_1, \mathsf{st}) \leftarrow \mathsf{WI}.\mathsf{P}_1(1^\lambda, \beta)} \\ (x_0, x_1, w_0, w_1, \mathsf{wi}_2, \mathsf{st}_{\mathcal{A}}) \leftarrow \mathcal{A}_1(1^\lambda, \mathsf{wi}_1, z) \\ \boxed{\mathsf{aux} \leftarrow \mathsf{WI}.\widehat{\mathsf{P}}_{1.5}(1^\lambda, \beta)} \\ \mathsf{wi}_3 \leftarrow \mathsf{WI}.\widehat{\mathsf{P}}_2(\mathsf{st}, \mathsf{aux}, x_0, x_1, w_0, w_1, \mathsf{wi}_2) \end{array} \end{array}\right] \geq \varepsilon. \tag{6}$$

Thus, it follows from an averaging argument that with probability at least $\varepsilon/2$ over the choice of $(\mathsf{wi}_1, \mathsf{st}) \leftarrow \mathsf{WI}.\mathsf{P}_1(1^\lambda, \beta)$ and $(x_0, x_1, w_0, w_1, \mathsf{wi}_2, \mathsf{st}_{\mathcal{A}}) \leftarrow \mathcal{A}_1(1^\lambda, \mathsf{wi}_1, z)$, we have

$$\Pr\left[\begin{array}{l|l} \mathcal{A}_2(\mathsf{st}_{\mathcal{A}}, \mathsf{wi}_3) = 1 \\ \wedge\, \widehat{\mathcal{A}}_2^{\mathcal{O}(\mathsf{st},\mathsf{aux},\cdot)}(\mathsf{st}_{\mathcal{A}}) = 1 & \begin{array}{l} \mathsf{aux} \leftarrow \mathsf{WI}.\widehat{\mathsf{P}}_{1.5}(1^\lambda, \beta) \\ \mathsf{wi}_3 \leftarrow \mathsf{WI}.\widehat{\mathsf{P}}_2(\mathsf{st}, \mathsf{aux}, x_0, x_1, w_0, w_1, \mathsf{wi}_2) \end{array} \end{array}\right] \geq \frac{\varepsilon}{2}. \tag{7}$$

Let G be the set that consists of every $(\mathsf{wi}_1, \mathsf{st}, x_0, x_1, w_0, w_1, \mathsf{wi}_2, \mathsf{st}_{\mathcal{A}})$ on which we have (7). For every $(\mathsf{wi}_1, \mathsf{st}, x_0, x_1, w_0, w_1, \mathsf{wi}_2, \mathsf{st}_{\mathcal{A}}) \in G$, Equation (7) trivially implies the following two lower bounds.

$$\Pr\left[\mathcal{A}_2(\mathsf{st}_{\mathcal{A}}, \mathsf{wi}_3) = 1 \;\middle|\; \begin{array}{l} \mathsf{aux} \leftarrow \mathsf{WI}.\widehat{\mathsf{P}}_{1.5}(1^\lambda, \beta) \\ \mathsf{wi}_3 \leftarrow \mathsf{WI}.\widehat{\mathsf{P}}_2(\mathsf{st}, \mathsf{aux}, x_0, x_1, w_0, w_1, \mathsf{wi}_2) \end{array}\right] \geq \frac{\varepsilon}{2}. \tag{8}$$

$$\Pr\left[\widehat{\mathcal{A}}_2^{\mathcal{O}(\mathsf{st},\mathsf{aux},\cdot)}(\mathsf{st}_{\mathcal{A}}) = 1 \;\middle|\; \mathsf{aux} \leftarrow \mathsf{WI}.\widehat{\mathsf{P}}_{1.5}(1^\lambda, \beta)\right] \geq \frac{\varepsilon}{2}. \tag{9}$$

Thus, when we add one more execution of $\mathsf{WI}.\widehat{\mathsf{P}}_{1.5}$ to the left-hand side of (6) so that the execution of $\widehat{\mathcal{A}}_2$ becomes independent of aux, we have

$$\begin{aligned} &\Pr\left[\begin{array}{l|l} \mathcal{A}_2(\mathsf{st}_{\mathcal{A}}, \mathsf{wi}_3) = 1 \\ \wedge\, \widehat{\mathcal{A}}_2^{\mathcal{O}(\mathsf{st},\boxed{\mathsf{aux}'},\cdot)}(\mathsf{st}_{\mathcal{A}}) = 1 & \begin{array}{l} (\mathsf{wi}_1, \mathsf{st}) \leftarrow \mathsf{WI}.\mathsf{P}_1(1^\lambda, \beta) \\ (x_0, x_1, w_0, w_1, \mathsf{wi}_2, \mathsf{st}_{\mathcal{A}}) \leftarrow \mathcal{A}_1(1^\lambda, \mathsf{wi}_1, z) \\ \mathsf{aux} \leftarrow \mathsf{WI}.\widehat{\mathsf{P}}_{1.5}(1^\lambda, \beta) \\ \mathsf{wi}_3 \leftarrow \mathsf{WI}.\widehat{\mathsf{P}}_2(\mathsf{st}, \mathsf{aux}, x_0, x_1, w_0, w_1, \mathsf{wi}_2) \\ \boxed{\mathsf{aux}' \leftarrow \mathsf{WI}.\widehat{\mathsf{P}}_{1.5}(1^\lambda, \beta)} \end{array} \end{array}\right] \\ &\geq \frac{\varepsilon}{2} \cdot \frac{\varepsilon}{2} \cdot \frac{\varepsilon}{2} = \frac{\varepsilon^3}{8}. \end{aligned} \tag{10}$$

We now analyze the left-hand side of (5). Using the indistinguishability between the

real and hybrid provers, we obtain

$$
\begin{aligned}
&\Pr\left[\begin{array}{l|l} \mathcal{A}_2(\mathsf{st}_{\mathcal{A}}, \mathsf{wi}_3) = 1 \\ \wedge\, \widehat{\mathcal{A}}_2^{\mathcal{O}(\mathsf{st},\mathsf{aux}',\cdot)}(\mathsf{st}_{\mathcal{A}}) = 1 & \begin{array}{l} (\mathsf{wi}_1, \mathsf{st}, \mathsf{aux}') \leftarrow \mathsf{WI}.\widehat{\mathsf{P}}_1(1^\lambda, \beta) \\ (x_0, x_1, w_0, w_1, \mathsf{wi}_2, \mathsf{st}_{\mathcal{A}}) \leftarrow \mathcal{A}_1(1^\lambda, \mathsf{wi}_1, z) \\ \mathsf{wi}_3 \leftarrow \mathsf{WI}.\mathsf{P}_2(\mathsf{st}, x_0, x_1, w_\beta, \mathsf{wi}_2) \end{array} \end{array}\right] \\
&\geq \Pr\left[\begin{array}{l|l} \mathcal{A}_2(\mathsf{st}_{\mathcal{A}}, \mathsf{wi}_3) = 1 \\ \wedge\, \widehat{\mathcal{A}}_2^{\mathcal{O}(\mathsf{st},\mathsf{aux}',\cdot)}(\mathsf{st}_{\mathcal{A}}) = 1 & \begin{array}{l} (\mathsf{wi}_1, \mathsf{st}, \mathsf{aux}') \leftarrow \mathsf{WI}.\widehat{\mathsf{P}}_1(1^\lambda, \beta) \\ (x_0, x_1, w_0, w_1, \mathsf{wi}_2, \mathsf{st}_{\mathcal{A}}) \leftarrow \mathcal{A}_1(1^\lambda, \mathsf{wi}_1, z) \\ \boxed{\mathsf{aux} \leftarrow \mathsf{WI}.\widehat{\mathsf{P}}_{1.5}(1^\lambda, \beta)} \\ \boxed{\mathsf{wi}_3 \leftarrow \mathsf{WI}.\widehat{\mathsf{P}}_2(\mathsf{st}, \mathsf{aux}, x_0, x_1, w_0, w_1, \mathsf{wi}_2)} \end{array} \end{array}\right] - \mathsf{negl}(\lambda) \\
&= \Pr\left[\begin{array}{l|l} \mathcal{A}_2(\mathsf{st}_{\mathcal{A}}, \mathsf{wi}_3) = 1 \\ \wedge\, \widehat{\mathcal{A}}_2^{\mathcal{O}(\mathsf{st},\mathsf{aux}',\cdot)}(\mathsf{st}_{\mathcal{A}}) = 1 & \begin{array}{l} \boxed{(\mathsf{wi}_1, \mathsf{st}) \leftarrow \mathsf{WI}.\mathsf{P}_1(1^\lambda, \beta)} \\ (x_0, x_1, w_0, w_1, \mathsf{wi}_2, \mathsf{st}_{\mathcal{A}}) \leftarrow \mathcal{A}_1(1^\lambda, \mathsf{wi}_1, z) \\ \mathsf{aux} \leftarrow \mathsf{WI}.\widehat{\mathsf{P}}_{1.5}(1^\lambda, \beta) \\ \mathsf{wi}_3 \leftarrow \mathsf{WI}.\widehat{\mathsf{P}}_2(\mathsf{st}, \mathsf{aux}, x_0, x_1, w_0, w_1, \mathsf{wi}_2) \\ \boxed{\mathsf{aux}' \leftarrow \mathsf{WI}.\widehat{\mathsf{P}}_{1.5}(1^\lambda, \beta)} \end{array} \end{array}\right] - \mathsf{negl}(\lambda).
\end{aligned}
\tag{11}
$$

Combining (10) with (11), we obtain (5). □

5 Our 4-round SNMZK Argument

In this section, we prove our main result.

Theorem 4. *Assume the existence of collision-resistant hash functions and injective one-way functions. Then, for any NP language $\boldsymbol{L}$, there exists a 4-round adaptive-input interactive argument that satisfies one-many statistical non-malleable zero-knowledge with weak adaptive input selection.*

To prove Theorem 4, we prove the following lemma.

Lemma 2. *Assume the existence of the following building blocks.*

- *A 3-round synchronous non-malleable commitment scheme* $\mathsf{NMC} = (\mathsf{NMC.S}_1, \mathsf{NMC.S}_2, \mathsf{NMC.R}, \mathsf{NMC.R}_{\mathsf{D}})$ *that satisfies the unique decommitment candidate property described in Theorem 2. Let* $\mathsf{NMC.R}_{\mathsf{D},\mathsf{weak}}$ *be the predicate that is guaranteed to exist by the unique decommitment candidate property.*
- *A 4-round public-coin adaptive-input argument* $\mathsf{SWI} = (\mathsf{SWI.P}_1, \mathsf{SWI.P}_2, \mathsf{SWI.V}_1, \mathsf{SWI.V}_2, \mathsf{SWI.V}_{\mathsf{D}})$ *that satisfies both adaptive-input computational special soundness and adaptive-input statistical witness indistinguishability. Let* $\mathsf{SWI.Ext}$ *be the extractor that is guaranteed to exist by the adaptive-input computational special soundness.*
- *A 3-round adaptive-input OR-indistinguishable special-sound proof* $\mathsf{WI} = (\mathsf{WI.P}_1, \mathsf{WI.P}_2, \mathsf{WI.V}, \mathsf{WI.V}_{\mathsf{D}})$ *that satisfies the additional properties defined in Theorem 3. Let* $\mathsf{WI.Ext}$ *and* $\mathsf{WI.EQ}$ *be the extractor and predicate that are guaranteed to exist by the adaptive-input special soundness.*

Then, for any NP language $\mathbf{L}$, there exists a 4-round adaptive-input interactive argument that satisfies one-many statistical non-malleable zero-knowledge with weak adaptive input selection.

Since the building blocks listed in Lemma 2 can be constructed from any collision-resistant hash function and injective one-way function (cf. Theorem 2, Theorem 1, and Theorem 3), Lemma 2 implies Theorem 4. Thus, we focus on proving Lemma 2 below.

5.1 Protocol Description

The formal description of our 4-round SNMZK argument, $(\mathsf{P}_1, \mathsf{P}_2, \mathsf{V}_1, \mathsf{V}_2, \mathsf{V}_\mathsf{D})$, is given below.

4-round SNMZK argument $(\mathsf{P}_1, \mathsf{P}_2, \mathsf{V}_1, \mathsf{V}_2, \mathsf{V}_\mathsf{D})$

Let $(x, w) \in \mathbf{R}_\mathbf{L}$ be an instance of an NP language $\mathbf{L}$ and a corresponding witness. Let $\lambda := |x|$ denote the security parameter. Let $\mathsf{tag} \in \{0,1\}^\lambda$ be a string representing the tag.

Round 1. $\mathsf{V}_1(1^\lambda, \mathsf{tag})$:

1. $\forall b \in \{0,1\}$: Compute $(\text{i-nm}_{b,1}, \text{i-dec}_b, \text{i-st}_{\mathsf{nm},b}) \leftarrow \mathsf{NMC.S}_1(1^\lambda, \mathsf{tag}, r_b)$ for a random string $r_b \leftarrow \{0,1\}^\lambda$.
2. $\forall b \in \{0,1\}$: Compute $(\text{o-nm}_{b,1}, \text{o-dec}_b, \text{o-st}_{\mathsf{nm},b}) \leftarrow \mathsf{NMC.S}_1(1^\lambda, \mathsf{tag}, \text{i-dec}_b)$.
3. Compute $(\mathsf{wi}_1, \mathsf{st}_\mathsf{wi}, \mathsf{aux}_\mathsf{wi}) \leftarrow \mathsf{WI.}\widehat{\mathsf{P}}_1(1^{\lambda_V}, \beta)$ for a random bit $\beta \leftarrow \{0,1\}$, where λ_V is the length of the statements $x_{V,0}, x_{V,1} \in \mathbf{L}_V$ defined in Round 3.
4. Compute $\mathsf{swi}_1 \leftarrow \mathsf{SWI.V}_1(1^{\lambda_P})$, where λ_P is the length of the statement $x_P \in \mathbf{L}_P$ defined in Round 4.
5. Output $\mathsf{m}_1 := (\{\text{i-nm}_{b,1}, \text{o-nm}_{b,1}\}_{b\in\{0,1\}}, \mathsf{wi}_1, \mathsf{swi}_1)$ and $\mathsf{st}_V := (\{\text{i-st}_{\mathsf{nm},b}, \text{o-st}_{\mathsf{nm},b}\}_{b\in\{0,1\}}, \mathsf{st}_\mathsf{wi}, \mathsf{aux}_\mathsf{wi})$.

Round 2. $\mathsf{P}_1(1^\lambda, \mathsf{tag}, \mathsf{m}_1)$:

1. Compute the responses to $\{\text{i-nm}_{b,1}, \text{o-nm}_{b,1}\}_{b\in\{0,1\}}$, wi_1, and swi_1 as follows.
   - $\forall b \in \{0,1\}$: $\text{i-nm}_{b,2} \leftarrow \mathsf{NMC.R}(1^\lambda, \mathsf{tag}, \text{i-nm}_{b,1})$.
   - $\forall b \in \{0,1\}$: $\text{o-nm}_{b,2} \leftarrow \mathsf{NMC.R}(1^\lambda, \mathsf{tag}, \text{o-nm}_{b,1})$.
   - $\mathsf{wi}_2 \leftarrow \mathsf{WI.V}(1^{\lambda_V})$.
   - $(\mathsf{swi}_2, \mathsf{st}_\mathsf{swi}) \leftarrow \mathsf{SWI.P}_1(1^{\lambda_P}, \mathsf{swi}_1)$.
2. Output $\mathsf{m}_2 := (\{\text{i-nm}_{b,2}, \text{o-nm}_{b,2}\}_{b\in\{0,1\}}, \mathsf{wi}_2, \mathsf{swi}_2)$ and $\mathsf{st}_P := \mathsf{st}_\mathsf{swi}$.

Round 3. $\mathsf{V}_2(\mathsf{st}_V, \mathsf{m}_2)$:

1. Compute the responses to $\{\text{i-nm}_{b,2}, \text{o-nm}_{b,2}\}_{b\in\{0,1\}}$ as follows.
   - $\forall b \in \{0,1\}$: $\text{i-nm}_{b,3} \leftarrow \mathsf{NMC.S}_2(\text{i-st}_{\mathsf{nm},b}, \text{i-nm}_{b,2})$.
   - $\forall b \in \{0,1\}$: $\text{o-nm}_{b,3} \leftarrow \mathsf{NMC.S}_2(\text{o-st}_{\mathsf{nm},b}, \text{o-nm}_{b,2})$.
2. Compute the response to wi_2 by $\mathsf{wi}_3 \leftarrow \mathsf{WI}.\widehat{\mathsf{P}}_2(\mathsf{st}_{\mathsf{wi}}, \mathsf{aux}_{\mathsf{wi}}, x_{V,0}, x_{V,1}, w_{V,0}, w_{V,1}, \mathsf{wi}_2)$, where $x_{V,b} := (\text{i-}\tau_b, \text{o-}\tau_b)$, $\text{i-}\tau_b := (\text{i-nm}_{b,1}, \text{i-nm}_{b,2}, \text{i-nm}_{b,3})$, $\text{o-}\tau_b := (\text{o-nm}_{b,1}, \text{o-nm}_{b,2}, \text{o-nm}_{b,3})$, and $w_{V,b} := (\text{i-dec}_b, \text{o-dec}_b)$ for each $b \in \{0,1\}$. Here, $x_{V,b}$ is viewed as an instance of the following language $\mathbf{L}_V$.

$$\mathbf{L}_V := \left\{ (\text{i-}\tau, \text{o-}\tau) \;\middle|\; \begin{array}{l} \exists(\text{i-dec}, \text{o-dec}) \text{ s.t.} \\ \mathsf{NMC.R}_{\mathsf{D}}(\text{i-}\tau, \text{i-dec}) \neq \bot \\ \wedge\ \mathsf{NMC.R}_{\mathsf{D}}(\text{o-}\tau, \text{o-dec}) = \text{i-dec} \neq \bot \end{array} \right\}.$$

3. Compute the response to swi_2 by $\mathsf{swi}_3 \leftarrow \mathsf{SWI.V}_2(1^{\lambda_P})$.
4. Output $\mathsf{m}_3 := (\{\text{i-nm}_{b,3}, \text{o-nm}_{b,3}\}_{b\in\{0,1\}}, \mathsf{wi}_3, \mathsf{swi}_3)$.

Round 4. $\mathsf{P}_2(\mathsf{st}_P, x, w, \mathsf{m}_3)$:

1. Check whether $\mathsf{WI.V}_{\mathsf{D}}(x_{V,0}, x_{V,1}, \tau_{\mathsf{wi}}) \stackrel{?}{=} 1$, where $\tau_{\mathsf{wi}} := (\mathsf{wi}_1, \mathsf{wi}_2, \mathsf{wi}_3)$. If the check fails, abort.
2. Compute $\mathsf{swi}_4 \leftarrow \mathsf{SWI.P}_2(\mathsf{st}_{\mathsf{swi}}, x_P, w, \mathsf{swi}_3)$, where $x_P := (x, \text{i-nm}_{0,1}, \text{i-nm}_{1,1})$. Here, x_P is viewed as an instance of the language $\mathbf{L}_P := \{(x, \text{i-nm}_{0,1}, \text{i-nm}_{1,1}) \mid x \in \mathbf{L} \vee (\text{i-nm}_{0,1}, \text{i-nm}_{1,1}) \in \mathbf{L}_{\mathsf{td}}\}$, where $\mathbf{L}_{\mathsf{td}} := \{(\text{i-nm}_{0,1}, \text{i-nm}_{1,1}) \mid \exists(b, \mathsf{dec}) \text{ s.t. } \mathsf{NMC.R}_{\mathsf{D,weak}}(\text{i-nm}_{b,1}, \mathsf{dec}) = 1\}$.
3. Output $\mathsf{m}_4 := \mathsf{swi}_4$.

Verification. $\mathsf{V}_{\mathsf{D}}(x, (\mathsf{m}_1, \ldots, \mathsf{m}_4))$ outputs 1 if and only if $\mathsf{SWI.V}_{\mathsf{D}}(x_P, \tau_{\mathsf{swi}}) = 1$ for $\tau_{\mathsf{swi}} := (\mathsf{swi}_1, \mathsf{swi}_2, \mathsf{swi}_3, \mathsf{swi}_4)$.

For each $b \in \{0,1\}$, we call $\text{i-}\tau_b = (\text{i-nm}_{b,1}, \text{i-nm}_{b,2}, \text{i-nm}_{b,3})$ the *b-th inner* NMC *commitment*, and $\text{o-}\tau_b = (\text{o-nm}_{b,1}, \text{o-nm}_{b,2}, \text{o-nm}_{b,3})$ the *b-th outer* NMC *commitment*. The security of $(\mathsf{P}_1, \mathsf{P}_2, \mathsf{V}_1, \mathsf{V}_2, \mathsf{V}_{\mathsf{D}})$ is analyzed in the rest of this section.

5.2 Proofs of Completeness and Soundness

Since the delayed-input completeness can be verified by inspection, we focus on proving the adaptive-input soundness. Assume for contradiction that there exist a pair of PPT Turing machines $\mathcal{P} = (\mathcal{P}_1, \mathcal{P}_2)$ and a polynomial p such that for infinitely many

$\lambda \in \mathbb{N}$, there exist $\mathsf{tag} \in \{0,1\}^\lambda$ and $z \in \{0,1\}^*$ such that

$$\Pr\left[\begin{array}{l|l} x \notin \mathbf{L} & (\mathsf{m}_1, \mathsf{st}_V) \leftarrow \mathsf{V}_1(1^\lambda, \mathsf{tag}) \\ \wedge\ \mathsf{V}_\mathsf{D}(x,\tau) = 1 & (\mathsf{m}_2, \mathsf{st}_\mathcal{P}) \leftarrow \mathcal{P}_1(1^\lambda, \mathsf{tag}, \mathsf{m}_1, z) \\ & \mathsf{m}_3 \leftarrow \mathsf{V}_2(\mathsf{st}_V, \mathsf{m}_2) \\ & (x, \mathsf{m}_4) \leftarrow \mathcal{P}_2(\mathsf{st}_\mathcal{P}, \mathsf{m}_3) \\ & \tau := (\mathsf{m}_1, \ldots, \mathsf{m}_4) \end{array}\right] \geq \frac{1}{p(\lambda)}. \tag{12}$$

Fix any such $\mathcal{P}$, p, λ, tag, and z. Let $\varepsilon := 1/p(\lambda)$. Toward deriving a contradiction, we begin by considering the following experiment H_0.

1. **Main thread.** Run V_1, $\mathcal{P}_1$, V_2, and $\mathcal{P}_2$ as described in (12) to obtain a statement x and a transcript $\tau = (\mathsf{m}_1, \mathsf{m}_2, \mathsf{m}_3, \mathsf{m}_4)$. If $\mathsf{V}_\mathsf{D}(x,\tau) \neq 1$, abort. Let $(\mathsf{m}_2, \mathsf{st}_\mathcal{P})$ denote the output of $\mathcal{P}_1$.
2. **Look-ahead thread.** Define x', m'_3, and m'_4 as follows: parse m_3 as $\mathsf{m}_3 = (\{\text{i-nm}_{b,3}, \text{o-nm}_{b,3}\}_{b\in\{0,1\}}, \mathsf{wi}_3, \mathsf{swi}_3)$; then, compute $\mathsf{swi}'_3 \leftarrow \mathsf{SWI.V}_2(1^{\lambda_P})$, let $\mathsf{m}'_3 := (\{\text{i-nm}_{b,3}, \text{o-nm}_{b,3}\}_{b\in\{0,1\}}, \mathsf{wi}_3, \mathsf{swi}'_3)$, and compute $(x', \mathsf{m}'_4) \leftarrow \mathcal{P}_2(\mathsf{st}_\mathcal{P}, \mathsf{m}'_3)$. If $\mathsf{V}_\mathsf{D}(x', \tau') \neq 1$ for $\tau' := (\mathsf{m}_1, \mathsf{m}_2, \mathsf{m}'_3, \mathsf{m}'_4)$, abort.
3. Output $w := \mathsf{SWI.Ext}(x, \tau, x', \tau')$.

Claim 1. *Let β denote the bit used by V_1 for $\mathsf{WI}.\widehat{\mathsf{P}}_1$ in Round 1, and let $\{\text{i-nm}_{b,1}\}_{b\in\{0,1\}}$ denote the* NMC *messages sent by V_1 in Round 1. Let us say that event* BAD *occurs in H_0 if the output w is a valid witness for $(\text{i-nm}_{0,1}, \text{i-nm}_{1,1}) \in \mathbf{L}_\mathsf{td}$ and of the form $(1-\beta, *)$. Then, when λ is sufficiently large, event* BAD *occurs with probability at least $\varepsilon_0 := \varepsilon^3/32$ in H_0.*

Proof. It follows from (12) and an averaging argument that we have $x \notin \mathbf{L} \wedge \mathsf{V}_\mathsf{D}(x,\tau) = 1 \wedge x' \notin \mathbf{L} \wedge \mathsf{V}_\mathsf{D}(x',\tau') = 1$ with probability at least $(\varepsilon/2)^3$ in H_0. Thus, from the adaptive-input computational special soundness of SWI, there exists a negligible function negl such that

$$\Pr\left[w \text{ is a valid witness for } (\text{i-nm}_{0,1}, \text{i-nm}_{1,1}) \in \mathbf{L}_\mathsf{td}\right] \geq (\varepsilon/2)^3 - \mathsf{negl}(\lambda).$$

Thus, from the adaptive-input OR-indistinguishability of the hybrid prover of WI, there exists a negligible function negl' such that

$$\Pr\left[\begin{array}{l} w \text{ is a valid witness for } (\text{i-nm}_{0,1}, \text{i-nm}_{1,1}) \in \mathbf{L}_\mathsf{td} \\ \text{and of the form } (1-\beta, *) \end{array}\right] \geq \frac{(\varepsilon/2)^3 - \mathsf{negl}'(\lambda)}{2}.$$

Thus, when λ is sufficiently large, event BAD occurs with probability at least $\varepsilon^3/32$ in H_0. □

Next, we consider the following hybrid experiments.

- Hybrid H_1 is identical to H_0 except that in V_1 and V_2, the WI messages wi_1 and wi_3 are computed by $(\mathsf{wi}_1, \mathsf{st}_\mathsf{wi}) \leftarrow \mathsf{WI.P}_1(1^{\lambda_V}, \beta)$ and $\mathsf{wi}_3 \leftarrow \mathsf{WI.P}_2(\mathsf{st}_\mathsf{wi}, x_{V,0}, x_{V,1}, w_{V,\beta}, \mathsf{wi}_2)$ using the real prover algorithms (rather than by $(\mathsf{wi}_1, \mathsf{st}_\mathsf{wi}, \mathsf{aux}_\mathsf{wi}) \leftarrow \mathsf{WI.}\widehat{\mathsf{P}}_1(1^{\lambda_V}, \beta)$ and $\mathsf{wi}_3 \leftarrow \mathsf{WI.}\widehat{\mathsf{P}}_2(\mathsf{st}_\mathsf{wi}, \mathsf{aux}_\mathsf{wi}, x_{V,0}, x_{V,1}, w_{V,0}, w_{V,1}, \mathsf{wi}_2)$ using the hybrid prover algorithms).

- Hybrid H_2 is identical to H_1 except that in V_1, the $(1-\beta)$-th outer NMC commitment is computed by committing to a random string (i.e., by running $\mathsf{NMC.S}_1(1^\lambda, \mathsf{tag}, r')$ for random $r' \in \{0,1\}^\lambda$ rather than $\mathsf{NMC.S}_1(1^\lambda, \mathsf{tag}, \mathsf{i\text{-}dec}_{1-\beta})$).

Claim 2. *In H_1, event BAD occurs with probability at least $\varepsilon_1 := \varepsilon_0 - \mathsf{negl}(\lambda)$.*

Proof. This claim follows directly from the indistinguishability of the real and hybrid provers of WI. □

Claim 3. *In H_2, event BAD occurs with probability at least $\varepsilon_2 := \varepsilon_1 - \mathsf{negl}(\lambda)$.*

Proof. This claim follows directly from the hiding property of NMC. □

Now, we derive a contradiction by breaking the hiding property of NMC using H_2. Toward this end, it suffices to consider a cheating receiver that internally emulates H_2 while using the external committer to generate the $(1-\beta)$-th inner NMC commitment. (The committed value is either the random string $r_{1-\beta}$ that the cheating receiver selected or an independent random string.) Since this cheating receiver perfectly emulates the experiment H_2, event BAD occurs in the internally emulated H_2 with non-negligible probability. Furthermore, when event BAD occurs in the internally emulated H_2, the cheating receiver can obtain the value that the external committer committed to.[22] Thus, the cheating receiver can break the hiding property of NMC.

5.3 Description of Simulator-Extractor

We proceed to the proof of one-many statistical non-malleable zero-knowledge with weak adaptive input selection.

As explained in Section 2, our simulator-extractor simulates the left interaction using the extractor of WI and extracts a witness in each accepting right interaction using the extractor of SWI. Before describing how it works formally, let us mention two technical details omitted in Section 2. First, when the extractor of WI is used, our simulator-extractor keeps rewinding the adversary until it obtains two (rather than one) additional accepting transcripts, and the extractor is invoked on these two additional transcripts. At a high level, using the extractor in this way makes the rewinding process almost independent of the rest of the simulator-extractor, thereby making it easy to use Lemma 1 (the technical lemma shown for WI in Section 4).[23] Second, when the extractor of SWI is used for each right interaction, our simulator-extractor keeps rewinding the adversary until it obtains an additional accepting transcript in which the left interaction is completed in the same way as before. (For example, if the left interaction was completed using the 0-th inner NMC commitment for SWI before the rewinding, our simulator-extractor tries to obtain an accepting transcript in which the

[22] The soundness of the unique decommitment candidate property of NMC guarantees that when BAD occurs, any dec such that $\mathsf{NMC.R}_{\mathsf{D},\mathsf{weak}}(\mathsf{i\text{-}nm}_{1-\beta,1}, \mathsf{dec}) = 1$ is a valid decommitment of the $(1-\beta)$-th inner NMC commitment. Thus, it can be used to obtain the value the external committer committed to.

[23] Specifically, the rewinding process only determines which of the two inner NMC commitments are used by the simulator-extractor for the simulation of the SWI proof.

left interaction is again completed using the 0-th inner NMC commitment for SWI.) Using the extractor in this way simplifies the analysis of witness extractability.

Formally, for any PPT man-in-the-middle adversary $\mathcal{A}$ and a polynomial m such that $\mathcal{A}$ participates in at most $m(\lambda)$ right interactions, our simulator-extractor, $\mathcal{SE}$, works as follows.

Simulator-extractor $\mathcal{SE}$.

Simulation part. On input $(1^\lambda, z)$, the simulator-extractor $\mathcal{SE}$ internally invokes $\mathcal{A}(1^\lambda, z)$ and emulates the left and right interactions for $\mathcal{A}$ in the following way.

In each right interaction, $\mathcal{SE}$ interacts with $\mathcal{A}$ using the honest verifier algorithms. That is, $\mathcal{SE}$ interacts with $\mathcal{A}$ in the i-th right interaction as follows. (In what follows, we denote Round j of the i-th right interaction by Round i-j for each $j \in \{1, \ldots, 4\}$.)

Round i-1. Receiving $\widetilde{\mathsf{tag}}_i$, compute $(\widetilde{\mathsf{m}}_{i,1}, \widetilde{\mathsf{st}}_{V,i}) \leftarrow \mathsf{V}_1(1^\lambda, \widetilde{\mathsf{tag}}_i)$ and send $\widetilde{\mathsf{m}}_{i,1}$ to $\mathcal{A}$.

Round i-2. Receiving $\widetilde{\mathsf{m}}_{i,2}$, compute $\widetilde{\mathsf{m}}_{i,3} \leftarrow \mathsf{V}_2(\widetilde{\mathsf{st}}_{V,i}, \widetilde{\mathsf{m}}_{i,2})$ and send $\widetilde{\mathsf{m}}_{i,3}$ to $\mathcal{A}$.

In the left interaction, $\mathcal{SE}$ interacts with $\mathcal{A}$ as follows.

Round 2. Receiving $(\mathsf{tag}, \mathsf{m}_1)$, compute $(\mathsf{m}_2, \mathsf{st}_P) \leftarrow \mathsf{P}_1(1^\lambda, \mathsf{tag}, \mathsf{m}_1)$ and send m_2 to $\mathcal{A}$.

Round 4. Receiving (x, m_3), do the following.

4:1 Like the honest prover, check whether m_3 is accepting, i.e., whether $\mathsf{WI.V_D}(x_{V,0}, x_{V,1}, \tau_{\mathsf{wi}}) \stackrel{?}{=} 1$ for the WI statements $x_{V,0}, x_{V,1}$ and transcript τ_{wi} defined based on $(\mathsf{m}_1, \mathsf{m}_2, \mathsf{m}_3)$. If the check fails, abort the left interaction.

4:2 If the check passes, repeat the following until the condition described below is met.

Left rewinding. Rewind $\mathcal{A}$ to the point just before sending m_2 to $\mathcal{A}$ in Round 2. Then, sample a new Round-2 message m'_2 using $\mathsf{P}_1(1^\lambda, \mathsf{tag}, \mathsf{m}_1)$ with fresh randomness, send m'_2 to $\mathcal{A}$, and continue interacting with $\mathcal{A}$ until obtaining a new Round-3 message m'_3. (During the interaction with $\mathcal{A}$, the right interactions are simulated as before using fresh randomness.)

The condition for breaking the loop is to find *two* new accepting Round-3 messages $\mathsf{m}'_3, \mathsf{m}''_3$. Let m'_2 denote the Round-2 message that is sent to $\mathcal{A}$ when obtaining m'_3. Let $(x'_{V,0}, x'_{V,1})$ and τ'_{wi} denote the WI statements and transcript defined based on $(\mathsf{m}_1, \mathsf{m}'_2, \mathsf{m}'_3)$. Let m''_2, $(x''_{V,0}, x''_{V,1})$, and τ''_{wi} be defined similarly w.r.t. m''_3. (At the end of this step, the state of $\mathcal{A}$ is restored to the one that $\mathcal{A}$ had before the above rewinding.)

4:3 Compute $w_V \coloneqq \mathsf{WI.Ext}((x'_{V,0}, x'_{V,1}), \tau'_{\mathsf{wi}}, (x''_{V,0}, x''_{V,1}), \tau''_{\mathsf{wi}})$. If $\exists \beta \in \{0,1\}$ s.t. $(x'_{V,\beta}, w_V) \in \mathbf{R}_{\mathbf{L}_V}$, parse w_V as $(\text{i-dec}, \text{o-dec})$ and let $w_{\mathsf{td}} \coloneqq (\beta, \text{i-dec})$. If there is no such β, output fail and abort.

4:4 Compute $\mathsf{swi}_4 \leftarrow \mathsf{SWI.P}_2(\mathsf{st}_{\mathsf{swi}}, x_P, w_{\mathsf{td}}, \mathsf{swi}_3)$ by using w_{td} with the SWI prover state $\mathsf{st}_{\mathsf{swi}}$, statement x_P, and challenge swi_3 that are defined based on $(x, \mathsf{m}_1, \mathsf{m}_2, \mathsf{m}_3)$. Then, send $\mathsf{m}_4 \coloneqq \mathsf{swi}_4$ to $\mathcal{A}$.

Let sim-view be the joint view of $\mathcal{A}$ and the honest verifiers emulated in the above. We call the execution of $\mathcal{A}$ recorded in sim-view the *main thread*, and we call the executions of $\mathcal{A}$ generated during the left rewinding the *left look-ahead threads*. We call w_{td} (the witness used for SWI in Step 4:4) the *trapdoor witness*.

Extraction part. After completing the simulation part, $\mathcal{SE}$ defines witness candidates $\{\widetilde{w}_i\}_{i\in[m(\lambda)]}$ in the following way. For each $i \in [m(\lambda)]$ such that the i-th right interaction is rejecting or uses the same tag as the left interaction, $\mathcal{SE}$ lets $\widetilde{w}_i \coloneqq \bot$. For every other i, $\mathcal{SE}$ does the following. Let $\widetilde{x}_i$ and $\widetilde{\mathsf{m}}_{i,4}$ denote the statement and message sent by $\mathcal{A}$ in Round i-4.

i:1 Repeat the following until the condition described below is met.

Right rewinding. Rewind $\mathcal{A}$ to the point just before sending $\widetilde{\mathsf{m}}_{i,3}$ to $\mathcal{A}$ in Round i-3. Then, define a new Round-3 message $\widetilde{\mathsf{m}}'_{i,3}$ as follows: parse $\widetilde{\mathsf{m}}_{i,3}$ as $\widetilde{\mathsf{m}}_{i,3} = (\{\widetilde{\text{i-nm}}_{b,3}, \widetilde{\text{o-nm}}_{b,3}\}_{b\in\{0,1\}}, \widetilde{\mathsf{wi}}_3, \widetilde{\mathsf{swi}}_3)$; then, sample a new challenge message $\widetilde{\mathsf{swi}}'_3 \leftarrow \mathsf{SWI.V}_2(1^{\lambda_P})$ for SWI and let $\widetilde{\mathsf{m}}'_{i,3} \coloneqq (\{\widetilde{\text{i-nm}}_{b,3}, \widetilde{\text{o-nm}}_{b,3}\}_{b\in\{0,1\}}, \widetilde{\mathsf{wi}}_3, \widetilde{\mathsf{swi}}'_3)$. Then, send $\widetilde{\mathsf{m}}'_{i,3}$ to $\mathcal{A}$ and continue interacting with $\mathcal{A}$ until obtaining a new Round-4 message $\widetilde{\mathsf{m}}'_{i,4}$ along with a new statement $\widetilde{x}'_i$ in Round i-4. (During the interaction with $\mathcal{A}$, the left and other right interactions are simulated as before using fresh randomness, including the left rewinding in Step 4:2.)

Let us call the interactions with $\mathcal{A}$ during the right rewinding the *right look-ahead threads*. Then, roughly speaking, the condition for breaking the loop is to find a right look-ahead thread in which the i-th right interaction is accepting *and* the left interaction is completed in the same way as in the main thread. Formally, the condition for breaking the loop is defined as follows. In the main and right look-ahead threads, for each $b \in \{0,1\}$, let EVENT_b denote the event that (i) the i-th right interaction is accepting and (ii) Round 4 of the left interaction is scheduled after Round i-3 and is completed using a witness of the form $(b, *)$ for SWI. Similarly, let EVENT_2 denote the event that (i) the i-th right interaction is accepting and (ii) either Round 4 of the left interaction is scheduled before Round i-3 or the left interaction is aborted. Let $k^* \in \{0,1,2\}$ be

such that event EVENT_{k^*} occurs in the main thread. Then, the condition for breaking the loop is to find a right look-ahead thread in which event EVENT_{k^*} occurs.

i:2 Compute $\widetilde{w}_{P,i} \coloneqq \mathsf{SWI.Ext}(\widetilde{x}_{P,i}, \widetilde{\tau}_{\mathsf{swi},i}, \widetilde{x}'_{P,i}, \widetilde{\tau}'_{\mathsf{swi},i})$ for (i) the SWI statement $\widetilde{x}_{P,i}$ and transcript $\widetilde{\tau}_{\mathsf{swi},i}$ defined based on $(\widetilde{x}_i, \widetilde{\mathsf{m}}_{i,1}, \widetilde{\mathsf{m}}_{i,2}, \widetilde{\mathsf{m}}_{i,3}, \widetilde{\mathsf{m}}_{i,4})$ and (ii) the SWI statement $\widetilde{x}'_{P,i}$ and transcript $\widetilde{\tau}'_{\mathsf{swi},i}$ defined based on $(\widetilde{x}'_i, \widetilde{\mathsf{m}}_{i,1}, \widetilde{\mathsf{m}}_{i,2}, \widetilde{\mathsf{m}}'_{i,3}, \widetilde{\mathsf{m}}'_{i,4})$. If $(\widetilde{x}_{P,i}, \widetilde{w}_{P,i}) \in \mathbf{R}_{\mathbf{L}_P}$, let $\widetilde{w}_i \coloneqq \widetilde{w}_{P,i}$. Otherwise, let $\widetilde{w}_i \coloneqq \bot_{\mathsf{fail}}$.

Final output. After completing the extraction part, $\mathcal{SE}$ outputs $(\mathsf{sim\text{-}view}, \{\widetilde{w}_i\}_{i\in[m(\lambda)]})$.

We analyze $\mathcal{SE}$ in the rest of this section. Fix any PPT man-in-the-middle adversary $\mathcal{A}$ and a polynomial m such that $\mathcal{A}$ participates in at most $m(\lambda)$ right interactions. Fix any PPT witness-selecting machine M.

5.4 Running Time of Simulator-Extractor

First, we obtain an upper bound for the expected running time of the simulation part of $\mathcal{SE}$. Toward this end, it suffices to analyze the number of the left look-ahead threads generated during the simulation part. Let R be the random variable representing the randomness used by $\mathcal{SE}$ until the point just before sampling a Round-2 message m_2 for $\mathcal{A}$ in the left interaction in the main thread, and fix arbitrary randomness r. Let p denote the probability that $\mathcal{A}$ outputs an accepting Round-3 message m_3 in the left interaction in the main thread under the condition $R = r$. Then, since each left look-ahead thread is generated identically to the main thread, it follows that under the condition $R = r$, the expected number of the left look-ahead threads generated during the simulation part is bounded above by $p \cdot 2/p = 2$. Since r is arbitrary, the expected number of left look-ahead threads during the simulation part is bounded above by 2.

Next, we obtain an upper bound for the expected running time of the extraction part of $\mathcal{SE}$. Toward this end, it suffices to analyze the time spent in the right rewinding in each accepting right interaction. Fix any $i \in [m(\lambda)]$. Let R be the random variable representing the randomness used by $\mathcal{SE}$ until the point just before sampling a SWI challenge message $\widetilde{\mathsf{swi}}_{i,3}$ in Round i-3 in the main thread, and fix arbitrary randomness r. Let $T_{\mathrm{sim}}(r)$ denote the expected running time of the simulation part of $\mathcal{SE}$ under the condition $R = r$. For each $k \in \{0, 1, 2\}$, let p_k denote the probability that event EVENT_k occurs in the simulation part of $\mathcal{SE}$ under the condition $R = r$. For each $j > 0$, let X_j denote the random variable representing the time spent in the j-th right look-ahead thread of the i-th right interaction when $R = r$. Then, under the condition

$R = r$, the expected time spent in the right rewinding of the i-th right interaction is

$$T_{\text{ext}}(r) \coloneqq \mathbb{E}\left[\sum_{j=1}^{\infty} X_j\right] = \sum_{j=1}^{\infty} \mathbb{E}\left[X_j\right] = \sum_{j=1}^{\infty} \sum_{\substack{0 \le k \le 2 \\ p_k \neq 0}} p_k (1-p_k)^{j-1} T_{\text{sim}}(r)$$
$$= \sum_{\substack{0 \le k \le 2 \\ p_k \neq 0}} p_k \cdot \frac{1}{p_k} \cdot T_{\text{sim}}(r) \le 3 \cdot T_{\text{sim}}(r).$$

Recalling that the expected running time of the simulation part of $\mathcal{SE}$ is bounded by $\mathsf{poly}(\lambda)$ (and therefore $\mathbb{E}_r\left[T_{\text{sim}}(r)\right] \le \mathsf{poly}(\lambda)$), the expected time spent in the right rewinding of the i-th right interaction is

$$\mathop{\mathbb{E}}_{r}\left[T_{\text{ext}}(r)\right] \le 3 \mathop{\mathbb{E}}_{r}\left[T_{\text{sim}}(r)\right] \le \mathsf{poly}(\lambda).$$

5.5 Statistical Simulation Property of Simulator-Extractor

For any $\lambda \in \mathbb{N}$ and $(z, z') \in Z_{\mathcal{A},M}(\lambda)$, consider the following hybrid experiments H_0, H_1, H_2.

- Hybrid H_0 is identical to the real man-in-the-middle experiment. That is, $\mathcal{A}(1^\lambda, z)$ interacts with a single honest prover and $m(\lambda)$ honest verifiers as described in Section 3.4 with the help of the witness-selecting machine $M(\cdot, \cdot, z')$. The output of the experiment is the joint view of $\mathcal{A}$ and the honest verifiers.

- Hybrid H_1 is an experiment where the simulator-extractor $\mathcal{SE}(1^\lambda, z)$ is executed with the following modifications.

  1. Only the simulation part is executed.
  2. In Step 4:4 of the simulation part, the SWI response message swi_4 is computed as in the honest prover algorithm, i.e., by $\mathsf{swi}_4 \leftarrow \mathsf{SWI.P}_2(\mathsf{st}_{\mathsf{swi}}, x_P, w, \mathsf{swi}_3)$ using the witness w provided by M.[24]

  The output of the experiment is the simulated view $\mathsf{sim\text{-}view}$ computed in $\mathcal{SE}$.

- Hybrid H_2 is identical to H_1 except that Step 4:4 of the simulation part is executed as in $\mathcal{SE}$, i.e., swi_4 is computed by $\mathsf{swi}_4 \leftarrow \mathsf{SWI.P}_2(\mathsf{st}_{\mathsf{swi}}, x_P, w_{\mathsf{td}}, \mathsf{swi}_3)$ using the witness w_{td} that is extracted through the left rewinding.

First, we observe that the outputs of H_0 and H_1 are statistically indistinguishable. Toward this end, it suffices to show that in H_1, the simulator-extractor $\mathcal{SE}$ outputs fail in Step 4:3 of the simulation part with negligible probability. Recall that in Step 4:3 of the simulation part, $\mathcal{SE}$ outputs fail only if $\mathsf{WI.Ext}((x'_{V,0}, x'_{V,1}), \tau'_{\mathsf{wi}}, (x''_{V,0}, x''_{V,1}), \tau''_{\mathsf{wi}})$ fails to output a valid witness for $x'_{V,0} \in \mathbf{L}_V \vee x'_{V,1} \in \mathbf{L}_V$, which in turn occurs only if the two challenge messages $\mathsf{wi}'_2, \mathsf{wi}''_2$ in the WI transcripts $\tau'_{\mathsf{wi}}, \tau''_{\mathsf{wi}}$ satisfy

[24] That is, the witness w is obtained by internally running M on input the statement chosen by $\mathcal{A}$, the current view of $\mathcal{A}$, and auxiliary input z'.

$\mathsf{WI.EQ}(\mathsf{wi}'_2, \mathsf{wi}''_2) = 1$. Given that the number of the left look-ahead threads, denoted by N_{left}, exceeds $2 \cdot 2^{\lambda_V/2}$ with probability at most $2^{-\lambda_V/2}$,[25] the delayed-input special soundness of WI implies that we have $\mathsf{WI.EQ}(\mathsf{wi}'_2, \mathsf{wi}''_2) = 1$ with probability at most

$$\Pr\left[N_{\mathrm{left}} > 2 \cdot 2^{\lambda_V/2}\right] + \Pr\left[\mathsf{WI.EQ}(\mathsf{wi}'_2, \mathsf{wi}''_2) = 1 \;\middle|\; N_{\mathrm{left}} \leq 2 \cdot 2^{\lambda_V/2}\right]$$
$$\leq 2^{-\lambda_V/2} + 2 \cdot 2^{\lambda_V/2} \cdot 2^{-\lambda_V} \leq \mathsf{negl}(\lambda),$$

where the last inequality follows from $\lambda_V = \Omega(\lambda)$.[26]

Next, we observe that the outputs of H_1 and H_2 are statistically indistinguishable. First, let us observe that in H_2, the witness used for SWI in Step 4:4 of the simulation part is indeed a valid witness. When $\mathcal{SE}$ does not abort in Step 4:3 of the simulation part, the witness $w_{\mathsf{td}} = (\beta, \text{i-dec})$ satisfies $\mathsf{NMC.R_D}(\text{i-}\tau'_\beta, \text{i-dec}) \neq \bot$ for the β-th inner NMC commitment $\text{i-}\tau'_\beta = (\text{i-}\mathsf{nm}_{\beta,1}, \text{i-}\mathsf{nm}'_{\beta,2}, \text{i-}\mathsf{nm}'_{\beta,3})$ that is contained in $(\mathsf{m}_1, \mathsf{m}'_2, \mathsf{m}'_3)$. Thus, by the correctness of the unique decommitment candidate property of NMC, we have $\mathsf{NMC.R_{D,weak}}(\text{i-}\mathsf{nm}_{\beta,1}, \text{i-dec}) = 1$, and therefore, w_{td} is a valid witness for SWI in the main thread. Then, since H_2 differs from H_1 only in the witness used for SWI, the indistinguishability follows directly from the adaptive-input statistical witness indistinguishability of SWI.[27]

Now, since the output of H_0 is identical to the view in the real man-in-the-middle experiment whereas the output of H_2 is identical to the view simulated by $\mathcal{SE}$, the statistical simulation property of $\mathcal{SE}$ follows.

5.6 Witness Extractability of Simulator-Extractor

Finally, we analyze the witness extractability of $\mathcal{SE}$. For each $i \in [m(\lambda)]$, let us say that event BAD_i occurs in the execution of $\mathcal{SE}$ if (i) $\mathcal{SE}$ does not abort, (ii) in the main thread, the i-th right interaction is accepting and uses a different tag from the left interaction, and (iii) the i-th extracted witness $\widetilde{w}_i$ is not a valid witness for the i-th right statement $\widetilde{x}_i$. Assume for contradiction that there exists a polynomial q such that for infinitely many $\lambda \in \mathbb{N}$, there exist $(z, z') \in Z_{\mathcal{A},M}(\lambda)$ and $i^* \in [m(\lambda)]$ such that BAD_{i^*} occurs with probability at least $1/q(\lambda)$. Fix any such q, λ, (z, z'), and i^*. For simplicity (and without loss of generality), we assume that $\mathcal{A}$ schedules the messages in the left and right interactions in the following way.

- Round 4 of every right interaction is scheduled at the end of the experiment.
- Round 2 of the left interaction is scheduled immediately after Round 1 of the left interaction, and Round 4 of the left interaction is scheduled immediately after Round 3 of the left interaction.[28]

[25] This follows from Markov's inequality since the expected number of the left look-ahead threads is at most 2.

[26] Recall that λ_V is the length of the statements proven by the verifier in WI.

[27] To see that the left rewinding does not interfere with the adaptive-input statistical witness indistinguishability of SWI, observe that the left rewinding only involves sampling new commit messages for SWI (which can be done by anyone knowing the statement length).

[28] In other words, no messages belonging to the right interactions are scheduled between Rounds 1 and 2 (resp., 3 and 4) of the left interaction.

- Round 1 of each right interaction is scheduled immediately after the point where $\mathcal{A}$ selects a tag for the corresponding right interaction, and Round 3 of each right interaction is scheduled immediately after Round 2 of the corresponding right interaction.

Under this assumption, the i^*-th right interaction must be scheduled in one of the following ways (see Figure 5).[29]

Schedule 1. Rounds 1 and 2 of the left interaction are scheduled between Rounds i^*-1 and i^*-2, and Rounds 3 and 4 of the left interaction are scheduled between Rounds i^*-3 and i^*-4. (In other words, the i^*-th right interaction is synchronized with the left interaction.)

Schedule 2. Round 2 of the left interaction is scheduled before the beginning of the i^*-th right interaction, and Rounds 3 and 4 of the left interaction are scheduled between Rounds i^*-3 and i^*-4.

Schedule 3. Round 4 of the left interaction is scheduled before Round i^*-2.

Schedule 4. Round 1 of the left interaction is scheduled after Round i^*-3.

For each $j \in \{1, \ldots, 4\}$, let $\mathsf{BAD}_{i^*,j}$ denote the event that (i) BAD_{i^*} occurs and (ii) the i^*-th right interaction is scheduled in Schedule j in the main thread. Since we have $\Pr[\mathsf{BAD}_{i^*}] \geq 1/q(\lambda)$, there exists $j^* \in \{1, \ldots, 4\}$ such that $\Pr[\mathsf{BAD}_{i^*,j^*}] \geq 1/4q(\lambda)$.

5.6.1 Synchronous Case ($j^* = 1$)

First, we consider the case that $\Pr[\mathsf{BAD}_{i^*,1}] \geq 1/4q(\lambda)$. We start by making several simplifications to $\mathcal{SE}$. Consider the following hybrid simulator-extractors $\mathcal{SE}_1, \ldots, \mathcal{SE}_5$.

1. $\mathcal{SE}_1$ is identical to $\mathcal{SE}$ except that $\mathcal{SE}_1$ outputs fail and aborts if the i^*-th right interaction is not scheduled in Schedule 1 in the main thread.

2. $\mathcal{SE}_2$ is identical to $\mathcal{SE}_1$ except that in the extraction part, $\mathcal{SE}_2$ lets $\widetilde{w}_i := \bot$ for every $i \neq i^*$ and skips every step related to such $\widetilde{w}_i$ (i.e., Steps i:1 and i:2 for every $i \neq i^*$).

3. $\mathcal{SE}_3$ is identical to $\mathcal{SE}_2$ except that when defining the i^*-th witness $\widetilde{w}_{i^*}$ in the extraction part, $\mathcal{SE}_3$ only generates a single right look-ahead thread (rather than keep generating right look-ahead threads until EVENT_{k^*} occurs in the current right look-ahead thread). If EVENT_{k^*} occurs in the unique right look-ahead thread, $\mathcal{SE}_3$ defines $\widetilde{w}_{i^*}$ as in $\mathcal{SE}_2$. Otherwise, $\mathcal{SE}_3$ outputs fail and aborts.

4. $\mathcal{SE}_4$ is identical to $\mathcal{SE}_3$ except that in the right look-ahead thread, $\mathcal{SE}_4$ completes SWI in the left interaction by reusing the witness that it used in the main thread. Specifically, consider the case that EVENT_0 or EVENT_1 occurs in the main thread.

[29] When the left interaction is aborted, we think the left interaction is completed at the point of being aborted by exchanging empty messages for the remaining rounds.

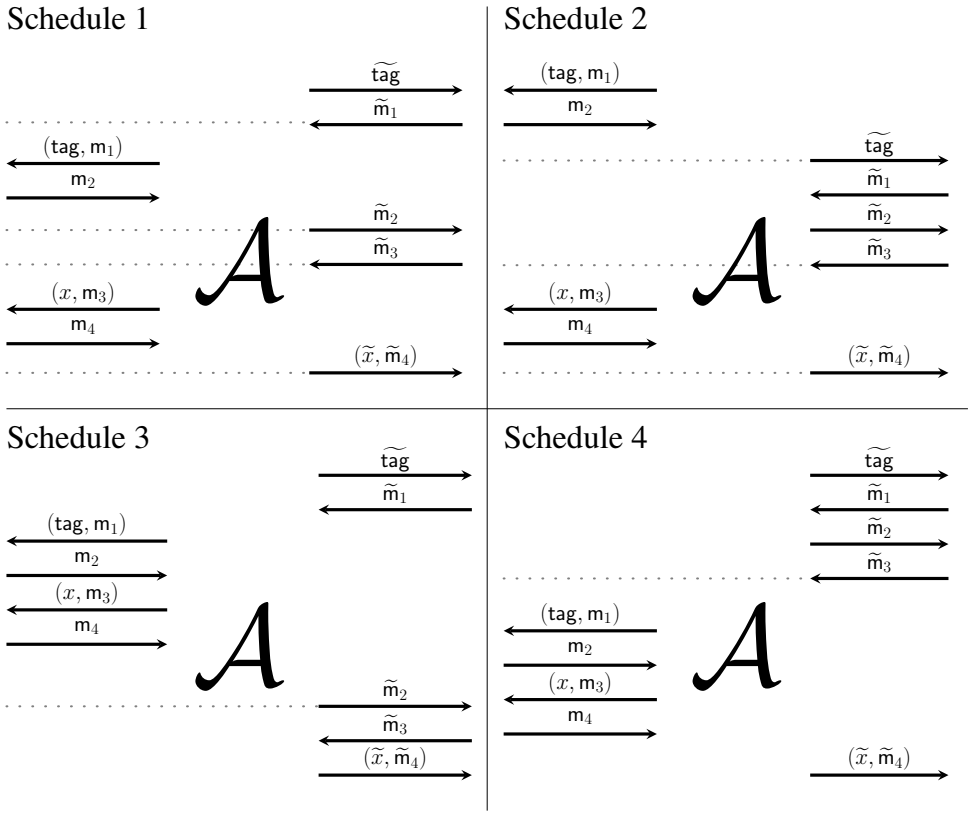

Figure 5: Four possible message schedules among the messages in the left and the i^*-th right interactions.

From the definitions of EVENT_0 and EVENT_1, it follows that in the main thread, Round 4 of the left interaction is successfully completed after Round i^*-3 by extracting a witness w_{td} for SWI using the left rewinding and $\mathsf{WI.Ext}$. Then, when completing Round 4 of the left interaction in the right look-ahead thread, $\mathcal{SE}_4$ no longer re-executes the left rewinding and $\mathsf{WI.Ext}$ to obtain a (potentially different) witness for SWI; instead, it simply reuses w_{td}. Furthermore, in the right look-ahead thread, $\mathcal{SE}_4$ no longer checks whether EVENT_{k^*} occurs. Instead, $\mathcal{SE}_4$ checks whether a new accepting message is obtained in Round i^*-4: if yes, $\mathcal{SE}_4$ defines the i^*-th witness $\widetilde{w}_{i^*}$ as in $\mathcal{SE}_3$ using $\mathsf{SWI.Ext}$; if no, $\mathcal{SE}_4$ outputs fail and aborts. ($\mathcal{SE}_4$ also outputs fail and aborts if w_{td} is not a valid witness for SWI in the right look-ahead thread.)

5. $\mathcal{SE}_5$ is identical to $\mathcal{SE}_4$ except that $\mathcal{SE}_5$ only generates two left look-ahead threads in the simulation part (rather than keep generating left look-ahead threads until obtaining two new accepting Round-3 messages). If two new accepting Round-3 messages are obtained in the two left look-ahead threads, $\mathcal{SE}_5$ completes the simulation part as in $\mathcal{SE}_4$. Otherwise, $\mathcal{SE}_5$ outputs fail and aborts.

The full description of the final hybrid simulator-extractor $\mathcal{SE}' := \mathcal{SE}_5$ is provided in Appendix B. It is not difficult to see that $\mathsf{BAD}_{i^*,1}$ still occurs with non-negligible

probability in $\mathcal{SE}'$. For completeness, we prove the following in Appendix C.

Claim 4. *There exists a polynomial q' (determined based on q) such that event $\mathsf{BAD}_{i^*,1}$ occurs with probability at least $1/q'(\lambda)$ in $\mathcal{SE}'$.*

Since the extraction part is only performed for the i^*-th right interaction in $\mathcal{SE}'$, in what follows we refrain from mentioning i^* in our notation. (For example, when we say "the right interaction," we refer to the i^*-th right interaction, and we write BAD_1 rather than $\mathsf{BAD}_{i^*,1}$.) We comment that $\mathcal{SE}'$ runs in strict polynomial time because it generates only a fixed number of threads.

Next, we observe that the extracted witness $\widetilde{w}$ is computationally independent of the bit $\widetilde{\beta}$ that $\mathcal{SE}'$ uses for $\mathsf{WI}.\widehat{\mathsf{P}}_1$ in Round 1 of the right interaction.

Claim 5. *Let $\widetilde{\beta}$ denote the bit used by $\mathcal{SE}'$ for $\mathsf{WI}.\widehat{\mathsf{P}}_1$ in Round 1 of the right interaction, and let $\{\widetilde{\mathsf{i\text{-}nm}}_{b,1}\}_{b\in\{0,1\}}$ denote the NMC messages sent by $\mathcal{SE}'$ in Round 1 of the right interaction. Let us say that event $\widehat{\mathsf{BAD}}$ occurs in an execution of $\mathcal{SE}'$ if (i) BAD_1 occurs and (ii) the extracted witness $\widetilde{w}$ is a valid witness for $(\widetilde{\mathsf{i\text{-}nm}}_{0,1}, \widetilde{\mathsf{i\text{-}nm}}_{1,1}) \in \boldsymbol{L}_{\mathsf{td}}$ and of the form $(1-\widetilde{\beta}, *)$. Then, there exists a polynomial $\widehat{q}$ (determined based on q') such that when λ is sufficiently large, event $\widehat{\mathsf{BAD}}$ occurs with probability at least $1/\widehat{q}(\lambda)$.*

Proof. We show that $\widehat{\mathsf{BAD}}$ occurs with probability roughly half of the probability that BAD_1 occurs. By definition, when BAD_1 occurs (and therefore the extracted witness $\widetilde{w}$ is not a valid witness for $\widetilde{x} \in \mathbf{L}$), either $\widetilde{w} = \bot_{\mathsf{fail}}$ or $\widetilde{w}$ is a valid witness for $(\widetilde{\mathsf{i\text{-}nm}}_{0,1}, \widetilde{\mathsf{i\text{-}nm}}_{1,1}) \in \mathbf{L}_{\mathsf{td}}$. First, we have $p_{\mathsf{fail}} := \Pr\left[\mathsf{BAD}_1 \wedge \widetilde{w} = \bot_{\mathsf{fail}}\right] \leq \mathsf{negl}(\lambda)$ because of the adaptive-input computational special soundness of SWI.[30] To analyze the case that $\widetilde{w}$ is a valid witness for $(\widetilde{\mathsf{i\text{-}nm}}_{0,1}, \widetilde{\mathsf{i\text{-}nm}}_{1,1}) \in \mathbf{L}_{\mathsf{td}}$, let us define p (resp., p') to be the probability that (i) BAD_1 occurs and (ii) $\widetilde{w}$ is a valid witness for $(\widetilde{\mathsf{i\text{-}nm}}_{0,1}, \widetilde{\mathsf{i\text{-}nm}}_{1,1}) \in \mathbf{L}_{\mathsf{td}}$ and of the form $(1-\widetilde{\beta}, *)$ (resp., $(\widetilde{\beta}, *)$). Then, we have $|p - p'| \leq \mathsf{negl}(\lambda)$ because of the adaptive-input rewinding OR-indistinguishability of the hybrid prover of WI.[31] Now, using $p_{\mathsf{fail}} + p + p' = \Pr\left[\mathsf{BAD}_1\right] \geq 1/q'(\lambda)$, we obtain $\Pr\left[\widehat{\mathsf{BAD}}\right] = p \geq (1/q'(\lambda) - \mathsf{negl}(\lambda))/2$. □

We proceed to case analysis depending on which trapdoor witness is used to complete SWI of the left interaction in the main thread. For each $\beta \in \{0,1\}$, let us say that event $\widehat{\mathsf{BAD}}_\beta$ occurs if (i) $\widehat{\mathsf{BAD}}$ occurs and (ii) SWI of the left interaction is completed using a witness of form $w_{\mathsf{td}} = (\beta, *)$ in the main thread. Similarly, let us say that event $\widehat{\mathsf{BAD}}_2$ occurs if (i) $\widehat{\mathsf{BAD}}$ occurs and (ii) the left interaction is aborted in the main thread. Given Claim 5, there exists $k^* \in \{0,1,2\}$ such that $\widehat{\mathsf{BAD}}_{k^*}$ occurs with

[30] To see that the left rewinding in $\mathcal{SE}'$ does not interfere with the adaptive-input computational special soundness of SWI in the right interaction, observe that the left rewinding can only require sampling a new challenge message for SWI (which can be done without knowing the internal state of the SWI verifier since SWI is a public-coin protocol).

[31] We require the rewinding version of the adaptive-input OR-indistinguishability since the left rewinding can require sampling new response messages for WI in the right interaction. To see that the right rewinding is not problematic, observe that the right look-ahead thread reuses the same WI response message as the main thread.

probability at least $1/3\widehat{q}(\lambda)$. In the following, we focus on the case $k^* \in \{0,1\}$. (The case of $k^* = 2$ can be handled analogously.)

Next, we design a series of hybrid experiments in which $\mathcal{SE}'$ is gradually modified, and we show that $\widehat{\mathsf{BAD}}_{k^*}$ occurs with non-negligible probability in each hybrid.

- Hybrid H_0 is the experiment where $\mathcal{SE}'(1^\lambda, z)$ is executed. From the observation above, $\widehat{\mathsf{BAD}}_{k^*}$ occurs with probability at least $\varepsilon_0 := 1/3\widehat{q}(\lambda)$ in H_0.
- Hybrid H_1 is identical to H_0 except that in the main thread, the WI response message $\widetilde{\mathsf{wi}}_3$ in the right interaction is computed by $\widetilde{\mathsf{wi}}_3 \leftarrow \mathsf{WI.P}_2(\widetilde{\mathsf{st}}_{\mathsf{wi}}, \widetilde{x}_{V,0}, \widetilde{x}_{V,1}, \widetilde{w}_{V,\widetilde{\beta}}, \widetilde{\mathsf{wi}}_2)$ using the real prover algorithm (rather than by $\widetilde{\mathsf{wi}}_3 \leftarrow \mathsf{WI.}\widehat{\mathsf{P}}_2(\widetilde{\mathsf{st}}_{\mathsf{wi}}, \widetilde{\mathsf{aux}}_{\mathsf{wi}}, \widetilde{x}_{V,0}, \widetilde{x}_{V,1}, \widetilde{w}_{V,0}, \widetilde{w}_{V,1}, \widetilde{\mathsf{wi}}_2)$ using the hybrid prover algorithm).[32]
- Hybrid H_2 is identical to H_1 except in how the trapdoor witness w_{td} is computed in the simulation part. Concretely, when the Round-3 message m_3 is accepting in the left interaction in the main thread, the left look-ahead threads are no longer generated (and therefore WI.Ext is no longer executed either), and w_{td} is computed by brute force as follows.
  1. Extract the committed values of the k^*-th outer NMC commitment by brute force. If the k^*-th outer NMC commitment is invalid (i.e., cannot be decommitted to any value), output fail and abort. Otherwise, let i-dec denote the extracted value.
  2. Let $w_{\mathsf{td}} := (k^*, \text{i-dec})$.
- Hybrid H_3 is identical to H_2 except that in the main thread, the $(1-\widetilde{\beta})$-th outer NMC commitment in the right interaction is computed by committing to a random string (i.e., by running $\mathsf{NMC.S}_1(1^\lambda, \widetilde{\mathsf{tag}}, r')$ for random $r' \in \{0,1\}^\lambda$ rather than $\mathsf{NMC.S}_1(1^\lambda, \widetilde{\mathsf{tag}}, \widetilde{\text{i-dec}}_{1-\widetilde{\beta}})$ in Round 1 of the right interaction in the main thread).

Claim 6. *$\widehat{\mathsf{BAD}}_{k^*}$ occurs in H_1 with probability at least $\varepsilon_1 := (\varepsilon_0/2)^3 - \mathsf{negl}(\lambda)$.*

Proof. Recall that H_1 differs from H_0 in that the real prover algorithm of WI is used in Round 3 of the right interaction in the main thread (while the hybrid prover algorithm continues to be used in the left look-ahead threads). We show this claim by relying on Lemma 1 (the property shown for WI in Section 4).

Informally, we use Lemma 1 as follows. Recall that Lemma 1 is stated about three Turing machines, one is executed to obtain intermediate information $\mathsf{st}_{\mathcal{A}}$ and the remaining two are executed independently of each other on input $\mathsf{st}_{\mathcal{A}}$. Accordingly, we consider decomposing the execution of H_0 (resp., H_1) into three components.

- The first component runs the main thread until the point just before completing Round 3 of the right interaction (more concretely, just before obtaining the WI response message $\widetilde{\mathsf{wi}}_3$ in the right interaction).

[32] We emphasize that the switch to the real prover algorithm is made only in the main thread. In the left look-ahead threads, the WI response message in the right interaction is computed as in H_1, i.e., using the hybrid prover algorithm.

- Given the partial transcript generated by the first component, the second component obtains a trapdoor witness w_{td} by generating two left look-ahead threads and invoking $\mathsf{WI.Ext}$.
- Given the partial transcript generated by the first component and the trapdoor witness w_{td} generated by the second component, the third component runs the rest of the main thread (starting from the generation of the WI response message $\widetilde{\mathsf{wi}}_3$ in the right interaction) and the subsequent extraction part of $\mathcal{SE}'$.

Note that H_0 and H_1 differ only in the third component, where a message obtained by the hybrid prover of WI is used in H_0 while a message obtained by the real prover is used in H_1. We cannot use Lemma 1 yet since the second and third components are not independent (the third component receives the trapdoor witness w_{td} from the second component). To make Lemma 1 applicable, we use the following consequence of the unique decommitment candidate property: the transcript generated by the first component already determines the trapdoor witness w_{td} extracted by the second component. Specifically, since we have $\Pr\left[\widehat{\mathsf{BAD}}_{k^*}\right] \geq \varepsilon_0$ in H_0, the following occur with probability at least ε_0 when H_0 is decomposed as above.

- In the first component, the k^*-th inner NMC commit message $\mathsf{i\text{-}nm}_{k^*,1}$ in Round 1 of the left interaction uniquely determines a decommitment candidate $\mathsf{i\text{-}dec}$ satisfying $\mathsf{NMC.R}_{\mathsf{D},\mathsf{weak}}(\mathsf{i\text{-}nm}_{k^*,1}, \mathsf{i\text{-}dec}) = 1$.
- In the second component, a valid trapdoor witness w_{td} of the form $(k^*, *)$ is obtained, and furthermore, it satisfies $w_{\mathsf{td}} = (k^*, \mathsf{i\text{-}dec})$ for $\mathsf{i\text{-}dec}$ determined in the first component.[33]
- In the third component, after the execution of $\mathcal{SE}'$ is completed by using $w_{\mathsf{td}} = (k^*, \mathsf{i\text{-}dec})$ for SWI in the left interaction, $\widehat{\mathsf{BAD}}_{k^*}$ occurs in the emulated execution of $\mathcal{SE}'$.

Now, observe that the above continues to hold even when the third component is executed independently of the second one as follows: instead of having it use the trapdoor witness that is extracted in the second component, have it use the trapdoor witness that is defined based on the decommitment candidate that is determined in the first component. Now that the second and third components are independent, we can use Lemma 1 to show that the above occurs with non-negligible probability even in H_1, which implies that $\widehat{\mathsf{BAD}}_{k^*}$ occurs in H_1 with non-negligible probability.

Formally, to use Lemma 1, we consider the following Turing machines.

- $\mathcal{A}_1(1^\lambda, \mathsf{wi}_1, z)$ internally invokes $\mathcal{SE}'$ and emulates H_0 with the following differences.
  - In the main thread, wi_1 is forwarded as the WI commit message in Round 1 of the right interaction.

[33] To see the furthermore part, observe that by the uniqueness of valid decommitment candidates, a valid trapdoor witness of the form $(k^*, *)$ must be equal to $(k^*, \mathsf{i\text{-}dec})$.

– In the main thread, when inner NMC commit messages $\{\text{i-nm}_{b,1}\}_{b\in\{0,1\}}$ are obtained in Round 1 of the left interaction, a decommitment candidate i-dec such that $\mathsf{NMC}.\mathsf{R}_{\mathsf{D},\mathsf{weak}}(\text{i-nm}_{k^*,1}, \text{i-dec}) = 1$ is obtained by brute force.

– The emulation of H_0 is suspended at the point where the WI response message is to be computed in Round 3 of the right interaction in the main thread. Let $\widetilde{x}_{V,0}, \widetilde{x}_{V,1}, \widetilde{w}_{V,0}, \widetilde{w}_{V,1}, \widetilde{\mathsf{wi}}_2$ be the WI statements etc. that are to be used to generate the WI response message, and let $\mathsf{st}_{\mathcal{SE}}$ be the current state of $\mathcal{SE}'$.

Then, $\mathcal{A}_1$ lets $\mathsf{st}_{\mathcal{A}} := (\mathsf{st}_{\mathcal{SE}}, \text{i-dec})$ and outputs $(\widetilde{x}_{V,0}, \widetilde{x}_{V,1}, \widetilde{w}_{V,0}, \widetilde{w}_{V,1}, \widetilde{\mathsf{wi}}_2, \mathsf{st}_{\mathcal{A}})$.

- $\widehat{\mathcal{A}}_2^{\mathcal{O}(\mathsf{st},\mathsf{aux},\cdot)}(\mathsf{st}_{\mathcal{A}})$ parses $\mathsf{st}_{\mathcal{A}}$ as $\mathsf{st}_{\mathcal{A}} := (\mathsf{st}_{\mathcal{SE}}, \text{i-dec})$ and continues the emulation of H_0 as follows.

  1. Using $\mathsf{st}_{\mathcal{SE}}$, reconstruct the main thread until the point just before Round 2 of the left interaction.
  2. Obtain a trapdoor witness w_{td} by generating two left look-ahead threads and running $\mathsf{WI}.\mathsf{Ext}$ as in H_0, where the oracle $\mathcal{O}$ is used whenever a new WI response message is required in the left look-ahead threads.

  Then, $\widehat{\mathcal{A}}_2$ outputs 1 if and only if the trapdoor witness w_{td} satisfies $w_{\mathsf{td}} = (k^*, \text{i-dec})$.

- $\mathcal{A}_2(\mathsf{st}_{\mathcal{A}}, \mathsf{wi}_3)$ parses $\mathsf{st}_{\mathcal{A}}$ as $\mathsf{st}_{\mathcal{A}} := (\mathsf{st}_{\mathcal{SE}}, \text{i-dec})$ and continues the emulation of H_0 as follows.

  1. Using $\mathsf{st}_{\mathcal{SE}}$, reconstruct the main thread until the point at which the WI response message is to be computed in Round 3 of the right interaction. Then, complete Round 3 of the right interaction by using wi_3 as the WI response message.
  2. Emulate the rest of H_0 honestly except that in the main thread, if the Round-3 message m_3 in the left interaction is accepting, then Round 4 is completed by using $(k^*, \text{i-dec})$ as the witness for SWI (without generating left look-ahead threads).

  Then, $\mathcal{A}_2$ outputs 1 if and only if $\widehat{\mathsf{BAD}}_{k^*}$ occurs.

Note that when wi_3 is computed by the hybrid prover of WI, the adversary $(\mathcal{A}_1, \widehat{\mathcal{A}}_2, \mathcal{A}_2)$ perfectly emulates H_0 when a trapdoor witness of the form $(k^*, *)$ is obtained in the left interaction. Thus, we have

$$\Pr\left[\begin{array}{l|l} \mathcal{A}_2(\mathsf{st}_{\mathcal{A}}, \mathsf{wi}_3) = 1 & (\mathsf{wi}_1, \mathsf{st}, \mathsf{aux}) \leftarrow \mathsf{WI}.\widehat{\mathsf{P}}_1(1^\lambda, \beta) \\ \wedge\ \widehat{\mathcal{A}}_2^{\mathcal{O}(\mathsf{st},\mathsf{aux},\cdot)}(\mathsf{st}_{\mathcal{A}}) = 1 & (x_0, x_1, w_0, w_1, \mathsf{wi}_2, \mathsf{st}_{\mathcal{A}}) \leftarrow \mathcal{A}_1(1^\lambda, \mathsf{wi}_1, z) \\ & \mathsf{wi}_3 \leftarrow \mathsf{WI}.\widehat{\mathsf{P}}_2(\mathsf{st}, \mathsf{aux}, x_0, x_1, w_0, w_1, \mathsf{wi}_2) \end{array}\right] \geq \varepsilon_0.$$

Thus, we can use Lemma 1 to obtain

$$\Pr\left[\begin{array}{l|l} \mathcal{A}_2(\mathsf{st}_{\mathcal{A}}, \mathsf{wi}_3) = 1 & (\mathsf{wi}_1, \mathsf{st}, \mathsf{aux}) \leftarrow \mathsf{WI}.\widehat{\mathsf{P}}_1(1^\lambda, \beta) \\ \wedge\, \widehat{\mathcal{A}}_2^{\mathcal{O}(\mathsf{st},\mathsf{aux},\cdot)}(\mathsf{st}_{\mathcal{A}}) = 1 & (x_0, x_1, w_0, w_1, \mathsf{wi}_2, \mathsf{st}_{\mathcal{A}}) \leftarrow \mathcal{A}_1(1^\lambda, \mathsf{wi}_1, z) \\ & \boxed{\mathsf{wi}_3 \leftarrow \mathsf{WI}.\mathsf{P}_2(\mathsf{st}, x_0, x_1, w_\beta, \mathsf{wi}_2)} \end{array}\right] \geq \left(\frac{\varepsilon_0}{2}\right)^3 - \mathsf{negl}(\lambda). \tag{13}$$

Note that when wi_3 is computed by the real prover of WI, the adversary $(\mathcal{A}_1, \widehat{\mathcal{A}}_2, \mathcal{A}_2)$ perfectly emulates H_1 when a trapdoor witness of the form $(k^*, *)$ is obtained in the left interaction. Thus, from (13), $\widehat{\mathsf{BAD}}_{k^*}$ occurs in H_1 with probability at least $\varepsilon_1 := (\varepsilon_0/2)^3 - \mathsf{negl}(\lambda)$. □

Claim 7. *$\widehat{\mathsf{BAD}}_{k^*}$ occurs in H_2 with probability at least $\varepsilon_2 := \varepsilon_1$.*

Proof. By definition, when m_3 is rejecting, H_2 is identical to H_1. We therefore focus on the case that m_3 is accepting, and we show that when $\widehat{\mathsf{BAD}}_{k^*}$ occurs in H_1, the same trapdoor witness w_{td} is used in H_1 and H_2. Suppose $\widehat{\mathsf{BAD}}_{k^*}$ occurs in H_1. Let $w^*_{\mathsf{td}} = (k^*, \text{i-dec}^*)$ denote the witness used in H_1 in Round 4 of the left interaction in the main thread. By the way in which w^*_{td} is defined, i-dec^* is guaranteed to be the (unique) committed value of the k^*-th outer NMC commitment. Thus, when the committed value of the k^*-th outer NMC commitment is extracted by brute force, the extracted value agrees with i-dec^*. Thus, even when the trapdoor witness w^*_{td} is computed by brute force as in H_2, event $\widehat{\mathsf{BAD}}_{k^*}$ occurs with probability at least ε_1. □

Claim 8. *$\widehat{\mathsf{BAD}}_{k^*}$ occurs in H_3 with probability at least $\varepsilon_3 := \varepsilon_2 - \mathsf{negl}(\lambda)$.*

Proof. This claim follows directly from the synchronous non-malleability of NMC since the brute-force extraction can be efficiently emulated in the non-malleability security experiment. (To see that the left rewinding is not problematic, observe that the left look-ahead threads are no longer generated in H_2 and H_3. To see that the right rewinding is not problematic, observe that the right look-ahead thread reuses the same NMC response message as the main thread.) □

Now, we derive a contradiction by breaking the non-malleability of NMC using H_3. Toward this end, it suffices to consider a man-in-the-middle adversary that internally emulates H_3 while (i) using the external committer to generate the $(1 - \widetilde{\beta})$-th inner NMC commitment in the right interaction and (ii) using the external receiver for the k^*-th outer NMC commitment in the left interaction. Since this adversary perfectly emulates the experiment H_3, event $\widehat{\mathsf{BAD}}_{k^*}$ occurs in the internally emulated experiment H_3 with non-negligible probability. Furthermore, when event $\widehat{\mathsf{BAD}}_{k^*}$ occurs in the internally emulated experiment H_3, the adversary can obtain the value that the external committer committed to. Thus, the adversary can trivially break the non-malleability of NMC.[34]

[34]Note that the $(1 - \widetilde{\beta})$-th inner NMC commitment in the right interaction only needs to be non-malleable with the k^*-th outer NMC in the left interaction, and it does not need to be non-malleable with

5.6.2 Asynchronous Case ($j^* \in \{2,3,4\}$)

The case of $j^* = 2$ can be analyzed in a similar (and somewhat simpler) way to the synchronous case. In particular, it suffices to make the following modifications.

1. Hybrid simulator-extractor $\mathcal{SE}_1$ outputs fail and aborts if the i^*-th right interaction is not scheduled in Schedule 2 (rather than Schedule 1) in the main thread.
2. Claim 5 is proved by relying on the non-rewinding version of the adaptive-input OR-indistinguishability of the hybrid prover of WI.[35]
3. Hybrids $H_1, \ldots, H_3$ are replaced with the following hybrids H'_1, H'_2.
   - Hybrid H'_1 is identical to H_0 except that in the right interaction in the main thread, the WI messages $\widetilde{\mathsf{wi}}_1$ and $\widetilde{\mathsf{wi}}_3$ are computed by using the real prover algorithms.
   - Hybrid H'_2 is identical to H'_1 except that in the right interaction in the main thread, the $(1-\widetilde{\beta})$-th outer NMC commitment is computed by committing to a random string of an appropriate length.

   The hybrids H'_1, H'_2 are analyzed as in the proof of soundness (Section 5.2) using the indistinguishability of the real and hybrid provers of WI and the hiding property of NMC.

The case of $j^* \in \{3,4\}$ can be analyzed in a similar (and much simpler) way. The key point is that in this case, either the right rewinding can only require re-executing the left interaction from the beginning or it does not require any new message in the left interaction at all. Specifically, we replace hybrid simulator-extractor $\mathcal{SE}'$ with one in which (i) the execution is aborted if the i^*-th right interaction is not scheduled in Schedule 3 or 4, (ii) the witness extraction is only done from the i^*-th right interaction, (iii) only a single right look-ahead thread is generated during the right rewinding, and (iv) the prover messages in the left interaction are generated using the honest prover algorithm (with the help of the witness-selecting machine). As before, we can show that event BAD_{i^*,j^*} occurs in this simulator-extractor with non-negligible probability. Then, since there is no left look-ahead thread in this simulator-extractor, we can derive a contradiction as in the proof of soundness (Section 5.2).

References

[Bar02] Boaz Barak. Constant-round coin-tossing with a man in the middle or realizing the shared random string model. In *43rd FOCS*, pages 345–355. IEEE Computer Society Press, November 2002.

the inner NMC commitments in the left interaction. Because of this asymmetry, the inner commitments actually do not need to satisfy full-fledged non-malleability. (For example, if the outer NMC commitments satisfy a sufficiently strong extractability property, we can use any non-interactive commitment scheme for the inner commitments.) We chose to use NMC for the inner commitment for simplicity.

[35] The non-rewinding version is sufficient since in Schedule 2, the left rewinding can only require sampling a new set of commit, challenge, and response messages of WI (which can be done by anyone using fresh randomness).

[BCC88] Gilles Brassard, David Chaum, and Claude Crépeau. Minimum disclosure proofs of knowledge. *Journal of Computer and System Sciences*, 37(2):156–189, 1988.

[BCO+24] Vincenzo Botta, Michele Ciampi, Emmanuela Orsini, Luisa Siniscalchi, and Ivan Visconti. Black-box (and fast) non-malleable zero knowledge. In Leonid Reyzin and Douglas Stebila, editors, *CRYPTO 2024, Part IX*, volume 14928 of *LNCS*, pages 458–490. Springer, Cham, August 2024.

[Blu82] Manuel Blum. Coin flipping by telephone. In *Proc. IEEE Spring COMPCOM*, pages 133–137, 1982.

[BPS06] Boaz Barak, Manoj Prabhakaran, and Amit Sahai. Concurrent non-malleable zero knowledge. In *47th FOCS*, pages 345–354. IEEE Computer Society Press, October 2006.

[CLOS02] Ran Canetti, Yehuda Lindell, Rafail Ostrovsky, and Amit Sahai. Universally composable two-party and multi-party secure computation. In *34th ACM STOC*, pages 494–503. ACM Press, May 2002.

[CLP13] Kai-Min Chung, Huijia Lin, and Rafael Pass. Constant-round concurrent zero knowledge from P-certificates. In *54th FOCS*, pages 50–59. IEEE Computer Society Press, October 2013.

[CLP15] Kai-Min Chung, Huijia Lin, and Rafael Pass. Constant-round concurrent zero-knowledge from indistinguishability obfuscation. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *CRYPTO 2015, Part I*, volume 9215 of *LNCS*, pages 287–307. Springer, Berlin, Heidelberg, August 2015.

[CLP20] Rohit Chatterjee, Xiao Liang, and Omkant Pandey. Improved black-box constructions of composable secure computation. In Artur Czumaj, Anuj Dawar, and Emanuela Merelli, editors, *ICALP 2020*, volume 168 of *LIPIcs*, pages 28:1–28:20. Schloss Dagstuhl, July 2020.

[COS22] Michele Ciampi, Emmanuela Orsini, and Luisa Siniscalchi. Four-round black-box non-malleable schemes from one-way permutations. In Eike Kiltz and Vinod Vaikuntanathan, editors, *TCC 2022, Part II*, volume 13748 of *LNCS*, pages 300–329. Springer, Cham, November 2022.

[COSV17a] Michele Ciampi, Rafail Ostrovsky, Luisa Siniscalchi, and Ivan Visconti. Delayed-input non-malleable zero knowledge and multi-party coin tossing in four rounds. In Yael Kalai and Leonid Reyzin, editors, *TCC 2017, Part I*, volume 10677 of *LNCS*, pages 711–742. Springer, Cham, November 2017.

[COSV17b] Michele Ciampi, Rafail Ostrovsky, Luisa Siniscalchi, and Ivan Visconti. Delayed-input non-malleable zero knowledge and multi-party coin tossing in four rounds. Cryptology ePrint Archive, Report 2017/931, Version 20181022:155022, 2017. A preprint version of [COSV17a].

[COSV17c] Michele Ciampi, Rafail Ostrovsky, Luisa Siniscalchi, and Ivan Visconti. Four-round concurrent non-malleable commitments from one-way functions. In Jonathan Katz and Hovav Shacham, editors, *CRYPTO 2017, Part II*, volume 10402 of *LNCS*, pages 127–157. Springer, Cham, August 2017.

[CPS+16a] Michele Ciampi, Giuseppe Persiano, Alessandra Scafuro, Luisa Siniscalchi, and Ivan Visconti. Improved OR-composition of sigma-protocols. In Eyal Kushilevitz and Tal Malkin, editors, *TCC 2016-A, Part II*, volume 9563 of *LNCS*, pages 112–141. Springer, Berlin, Heidelberg, January 2016.

[CPS+16b] Michele Ciampi, Giuseppe Persiano, Alessandra Scafuro, Luisa Siniscalchi, and Ivan Visconti. Online/offline OR composition of sigma protocols. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part II*, volume 9666 of *LNCS*, pages 63–92. Springer, Berlin, Heidelberg, May 2016.

[DDN00] Danny Dolev, Cynthia Dwork, and Moni Naor. Nonmalleable cryptography. *SIAM Journal on Computing*, 30(2):391–437, 2000.

[DPP97] Ivan Damgård, Torben P. Pedersen, and Birgit Pfitzmann. On the existence of statistically hiding bit commitment schemes and fail-stop signatures. *Journal of Cryptology*, 10(3):163–194, June 1997.

[FKP19] Cody Freitag, Ilan Komargodski, and Rafael Pass. Non-uniformly sound certificates with applications to concurrent zero-knowledge. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part III*, volume 11694 of *LNCS*, pages 98–127. Springer, Cham, August 2019.

[FS90a] Uriel Feige and Adi Shamir. Witness indistinguishable and witness hiding protocols. In *22nd ACM STOC*, pages 416–426. ACM Press, May 1990.

[FS90b] Uriel Feige and Adi Shamir. Zero knowledge proofs of knowledge in two rounds. In Gilles Brassard, editor, *CRYPTO'89*, volume 435 of *LNCS*, pages 526–544. Springer, New York, August 1990.

[GGJS12] Sanjam Garg, Vipul Goyal, Abhishek Jain, and Amit Sahai. Concurrently secure computation in constant rounds. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 99–116. Springer, Berlin, Heidelberg, April 2012.

[GK96] Oded Goldreich and Hugo Krawczyk. On the composition of zero-knowledge proof systems. *SIAM Journal on Computing*, 25(1):169–192, 1996.

[GMR89] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on Computing*, 18(1):186–208, 1989.

[Gol01] Oded Goldreich. *Foundations of Cryptography: Basic Tools*, volume 1. Cambridge University Press, Cambridge, UK, 2001.

[Goy11] Vipul Goyal. Constant round non-malleable protocols using one way functions. In Lance Fortnow and Salil P. Vadhan, editors, *43rd ACM STOC*, pages 695–704. ACM Press, June 2011.

[GPR15] Vipul Goyal, Omkant Pandey, and Silas Richelson. Textbook non-malleable commitments. Cryptology ePrint Archive, Report 2015/1178, Version 20170521:195131, 2015. A preprint version of [GPR16].

[GPR16] Vipul Goyal, Omkant Pandey, and Silas Richelson. Textbook non-malleable commitments. In Daniel Wichs and Yishay Mansour, editors, *48th ACM STOC*, pages 1128–1141. ACM Press, June 2016.

[GR19] Vipul Goyal and Silas Richelson. Non-malleable commitments using Goldreich-Levin list decoding. In David Zuckerman, editor, *60th FOCS*, pages 686–699. IEEE Computer Society Press, November 2019.

[GRRV21] Vipul Goyal, Silas Richelson, Alon Rosen, and Margarita Vald. An algebraic approach to nonmalleability. *SIAM Journal on Computing*, 50(5):1537–1579, 2021.

[HM96] Shai Halevi and Silvio Micali. Practical and provably-secure commitment schemes from collision-free hashing. In Neal Koblitz, editor, *CRYPTO'96*, volume 1109 of *LNCS*, pages 201–215. Springer, Berlin, Heidelberg, August 1996.

[JP14] Abhishek Jain and Omkant Pandey. Non-malleable zero knowledge: Black-box constructions and definitional relationships. In Michel Abdalla and Roberto De Prisco, editors, *SCN 14*, volume 8642 of *LNCS*, pages 435–454. Springer, Cham, September 2014.

[Kiy20] Susumu Kiyoshima. Statistical concurrent non-malleable zero-knowledge from one-way functions. *Journal of Cryptology*, 33(3):1318–1361, July 2020.

[KLP22] Allen Kim, Xiao Liang, and Omkant Pandey. A new approach to efficient non-malleable zero-knowledge. In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022, Part IV*, volume 13510 of *LNCS*, pages 389–418. Springer, Cham, August 2022.

[LP11a] Huijia Lin and Rafael Pass. Concurrent non-malleable zero knowledge with adaptive inputs. In Yuval Ishai, editor, *TCC 2011*, volume 6597 of *LNCS*, pages 274–292. Springer, Berlin, Heidelberg, March 2011.

[LP11b] Huijia Lin and Rafael Pass. Constant-round non-malleable commitments from any one-way function. In Lance Fortnow and Salil P. Vadhan, editors, *43rd ACM STOC*, pages 705–714. ACM Press, June 2011.

[LP15] Huijia Lin and Rafael Pass. Constant-round nonmalleable commitments from any one-way function. *Journal of the ACM*, 62(1), March 2015.

[LPTV10] Huijia Lin, Rafael Pass, Wei-Lung Dustin Tseng, and Muthuramakrishnan Venkitasubramaniam. Concurrent non-malleable zero knowledge proofs. In Tal Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 429–446. Springer, Berlin, Heidelberg, August 2010.

[LPV08] Huijia Lin, Rafael Pass, and Muthuramakrishnan Venkitasubramaniam. Concurrent non-malleable commitments from any one-way function. In Ran Canetti, editor, *TCC 2008*, volume 4948 of *LNCS*, pages 571–588. Springer, Berlin, Heidelberg, March 2008.

[LS91] Dror Lapidot and Adi Shamir. Publicly verifiable non-interactive zero-knowledge proofs. In Alfred J. Menezes and Scott A. Vanstone, editors, *CRYPTO'90*, volume 537 of *LNCS*, pages 353–365. Springer, Berlin, Heidelberg, August 1991.

[MU10] Jörn Müller-Quade and Dominique Unruh. Long-term security and universal composability. *Journal of Cryptology*, 23(4):594–671, October 2010.

[OOR+14] Claudio Orlandi, Rafail Ostrovsky, Vanishree Rao, Amit Sahai, and Ivan Visconti. Statistical concurrent non-malleable zero knowledge. In Yehuda Lindell, editor, *TCC 2014*, volume 8349 of *LNCS*, pages 167–191. Springer, Berlin, Heidelberg, February 2014.

[OPV10] Rafail Ostrovsky, Omkant Pandey, and Ivan Visconti. Efficiency preserving transformations for concurrent non-malleable zero knowledge. In Daniele Micciancio, editor, *TCC 2010*, volume 5978 of *LNCS*, pages 535–552. Springer, Berlin, Heidelberg, February 2010.

[Pas11] Rafael Pass. Limits of provable security from standard assumptions. In Lance Fortnow and Salil P. Vadhan, editors, *43rd ACM STOC*, pages 109–118. ACM Press, June 2011.

[PPS15] Omkant Pandey, Manoj Prabhakaran, and Amit Sahai. Obfuscation-based non-black-box simulation and four message concurrent zero knowledge for NP. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *TCC 2015, Part II*, volume 9015 of *LNCS*, pages 638–667. Springer, Berlin, Heidelberg, March 2015.

[PR08a] Rafael Pass and Alon Rosen. Concurrent nonmalleable commitments. *SIAM Journal on Computing*, 37(6):1891–1925, 2008.

[PR08b] Rafael Pass and Alon Rosen. New and improved constructions of nonmalleable cryptographic protocols. *SIAM Journal on Computing*, 38(2):702–752, 2008.

[Ven14] Muthuramakrishnan Venkitasubramaniam. On adaptively secure protocols. In Michel Abdalla and Roberto De Prisco, editors, *SCN 14*, volume 8642 of *LNCS*, pages 455–475. Springer, Cham, September 2014.

A Proof of Theorem 3

A.1 Preliminaries

We start by recalling the definitions of several basic primitives.

A.1.1 Non-interactive Commitments

Definition 10 (Non-interactive commitment)**.** *A pair of polynomial-time Turing machines* $\mathsf{Com} = (\mathsf{Com.S}, \mathsf{Com.R_D})$*, where* $\mathsf{Com.S}$ *is probabilistic and* $\mathsf{Com.R_D}$ *is deterministic, is called a* non-interactive commitment scheme *if it satisfies the following.*

Correctness. *For every* $\lambda \in \mathbb{N}$ *and* $b \in \{0,1\}$*,*

$$\Pr\left[\mathsf{Com.R_D}(1^\lambda, \mathsf{com}, b, \mathsf{dec}) = 1 \mid (\mathsf{com}, \mathsf{dec}) \leftarrow \mathsf{Com.S}(1^\lambda, b)\right] = 1.$$

Perfect binding. *For any* $\lambda \in \mathbb{N}$ *and* $\mathsf{com} \in \{0,1\}^*$*, there is no* $(\mathsf{dec}_0, \mathsf{dec}_1) \in (\{0,1\}^*)^2$ *satisfying* $\mathsf{Com.R_D}(1^\lambda, \mathsf{com}, 0, \mathsf{dec}_0) = 1 \wedge \mathsf{Com.R_D}(1^\lambda, \mathsf{com}, 1, \mathsf{dec}_1) = 1$*.*

Computational hiding. *For every* $\lambda \in \mathbb{N}$ *and* $b \in \{0,1\}$*, let* $\mathsf{com}_{\lambda,b}$ *be the random variable representing the first output of* $\mathsf{Com.S}(1^\lambda, b)$*. Then, the ensembles* $\{\mathsf{com}_{\lambda,0}\}_{\lambda\in\mathbb{N}}$ *and* $\{\mathsf{com}_{\lambda,1}\}_{\lambda\in\mathbb{N}}$ *are computationally indistinguishable.*

It is known that a non-interactive commitment scheme can be constructed from any injective one-way function [Blu82].

A.1.2 Non-interactive Instance-dependent Trapdoor Commitment

Definition 11 (Non-interactive instance-dependent trapdoor commitment)**.** *Let* $\boldsymbol{L}$ *be an NP language. A tuple of polynomial-time Turing machines* $\mathsf{TC} = (\mathsf{TC.S}, \mathsf{TC.R_D}, \mathsf{TC.FakeS}, \mathsf{TC.FakeOpen})$*, where* $\mathsf{TC.S}$ *and* $\mathsf{TC.FakeS}$ *are probabilistic and* $\mathsf{TC.R_D}$ *and* $\mathsf{TC.FakeOpen}$ *are deterministic, is called a* non-interactive instance-dependent trapdoor commitment scheme *for* $\boldsymbol{L}$ *if it satisfies the following.*

Correctness. *For every* $\lambda \in \mathbb{N}$*,* $x \in \{0,1\}^*$*, and* $m \in \{0,1\}^*$*,*

$$\Pr\left[\mathsf{TC.R_D}(1^\lambda, x, \mathsf{tcom}, m, \mathsf{tdec}) = 1 \mid (\mathsf{tcom}, \mathsf{tdec}) \leftarrow \mathsf{TC.S}(1^\lambda, x, m)\right] = 1.$$

Perfect binding. *For any* $\lambda \in \mathbb{N}$*,* $x \notin \boldsymbol{L}$*,* $\mathsf{tcom} \in \{0,1\}^*$*, and* $(m_0, m_1) \in (\{0,1\}^*)^2$ *such that* $m_0 \neq m_1$*, there is no* $(\mathsf{tdec}_0, \mathsf{tdec}_1) \in (\{0,1\}^*)^2$ *satisfying* $\mathsf{TC.R_D}(1^\lambda, x, \mathsf{tcom}, m_0, \mathsf{tdec}_0) = 1 \wedge \mathsf{TC.R_D}(1^\lambda, x, \mathsf{tcom}, m_1, \mathsf{tdec}_1) = 1$*.*

Equivocality correctness. *For every* $\lambda \in \mathbb{N}$, $(x, w) \in \boldsymbol{R_L}$, *and* $m \in \{0,1\}^*$,

$$\Pr\left[\mathsf{TC.R_D}(1^\lambda, x, \mathsf{tcom}, m, \mathsf{tdec}) = 1 \;\middle|\; \begin{array}{l} (\mathsf{tcom}, \mathsf{st}) \leftarrow \mathsf{TC.FakeS}(1^\lambda, x) \\ \mathsf{tdec} \coloneqq \mathsf{TC.FakeOpen}(\mathsf{st}, w, m) \end{array}\right] = 1.$$

Equivocality hiding. *For every pair of PPT Turing machines* $(\mathcal{V}_1, \mathcal{V}_2)$*, there exists a negligible function* negl *such that for every* $\lambda \in \mathbb{N}$ *and* $z \in \{0,1\}^*$*, the random variable* out *satisfies* $\Pr[\mathsf{out} = 1] \leq 1/2 + \mathsf{negl}(\lambda)$ *in the following experiment.*

1. *Compute* $(x, w, m, \mathsf{st}_\mathcal{V}) \leftarrow \mathcal{V}_1(1^\lambda, z)$*. If* $(x, w) \notin \boldsymbol{R_L}$*, abort.*
2. *Sample a random bit* $\mathsf{ch} \in \{0,1\}$*. If* $\mathsf{ch} = 0$*, compute* $(\mathsf{tcom}, \mathsf{tdec})$ *by* $(\mathsf{tcom}, \mathsf{tdec}) \leftarrow \mathsf{TC.S}(1^\lambda, x, m)$*. If* $\mathsf{ch} = 1$*, compute* $(\mathsf{tcom}, \mathsf{tdec})$ *by* $(\mathsf{tcom}, \mathsf{st}) \leftarrow \mathsf{TC.FakeS}(1^\lambda, x)$ *and* $\mathsf{tdec} \coloneqq \mathsf{TC.FakeOpen}(\mathsf{st}, w, m)$*.*
3. *Compute* $\widetilde{\mathsf{ch}} \leftarrow \mathcal{V}_2(\mathsf{st}_\mathcal{V}, \mathsf{tcom}, \mathsf{tdec})$ *and let* $\mathsf{out} \coloneqq 1$ *if and only if* $\widetilde{\mathsf{ch}} = \mathsf{ch}$*.*

It is known that a non-interactive instance-dependent trapdoor commitment scheme can be constructed for an NP-complete language (Graph Hamiltonicity) from any non-interactive commitment scheme [FS90b, CLOS02].

A.1.3 3-round Adaptive-input Special-sound Special-HVZK Proof

Definition 12 (3-round adaptive-input special-sound special-HVZK proof)**.** *Let* $\boldsymbol{L}$ *be an NP language. A tuple of PPT Turing machines* $(\mathsf{P}_1, \mathsf{P}_2, \mathsf{V}, \mathsf{V_D})$ *is called a* 3-round adaptive-input special-sound special honest-verifier zero-knowledge (special-HVZK) proof *if it satisfies the following.*

Delayed-input correctness. *For every* $(x, w) \in \boldsymbol{R_L}$*,*

$$\Pr\left[\mathsf{V_D}(x, \tau) = 1 \;\middle|\; \begin{array}{l} (\mathsf{m}_1, \mathsf{st}) \leftarrow \mathsf{P}_1(1^{|x|}) \\ \mathsf{m}_2 \leftarrow \mathsf{V}(1^{|x|}) \\ \mathsf{m}_3 \leftarrow \mathsf{P}_2(\mathsf{st}, x, w, \mathsf{m}_2) \\ \tau \coloneqq (\mathsf{m}_1, \mathsf{m}_2, \mathsf{m}_3) \end{array}\right] = 1.$$

Delayed-input special soundness. *There exists a deterministic polynomial-time Turing machine* Ext *(called the* extractor*) such that the following holds.*

- *Fix any* $x_0 \in \{0,1\}^*$*,* $x_1 \in \{0,1\}^*$*,* $\tau_0 \coloneqq (\mathsf{m}_{0,1}, \mathsf{m}_{0,2}, \mathsf{m}_{0,3}) \in (\{0,1\}^*)^3$*, and* $\tau_1 \coloneqq (\mathsf{m}_{1,1}, \mathsf{m}_{1,2}, \mathsf{m}_{1,3}) \in (\{0,1\}^*)^3$*. Suppose that* $\mathsf{m}_{0,1} = \mathsf{m}_{1,1}$*,* $\mathsf{m}_{0,2} \neq \mathsf{m}_{1,2}$*, and* $\mathsf{V_D}(x_b, \tau_b) = 1$ *for every* $b \in \{0,1\}$*. Then,* $\mathsf{Ext}(x_0, \tau_0, x_1, \tau_1)$ *outputs a value* w_0 *such that* $(x_0, w_0) \in \boldsymbol{R_L}$*.*

Delayed-input special HVZK. *There exists a PPT Turing machine* Sim *(called the* simulator*) such that for every sequence* $\{w_x, \mathsf{m}_{x,2}\}_{x \in \boldsymbol{L}}$ *such that* $w_x \in \boldsymbol{R_L}(x)$ *and* $\mathsf{m}_{x,2} \in \{0,1\}^*$*, the following two ensembles are computationally indistinguishable.*

- $\{(\mathsf{m}_1, \mathsf{m}_{x,2}, \mathsf{m}_3) \mid (\mathsf{m}_1, \mathsf{st}) \leftarrow \mathsf{P}_1(1^{|x|}); \mathsf{m}_3 \leftarrow \mathsf{P}_2(\mathsf{st}, x, w_x, \mathsf{m}_{x,2})\}_{x \in \boldsymbol{L}}$.

- $\{(\mathsf{m}_1, \mathsf{m}_{x,2}, \mathsf{m}_3) \mid (\mathsf{m}_1, \mathsf{m}_3) \leftarrow \mathsf{Sim}(x, \mathsf{m}_{x,2})\}_{x \in \boldsymbol{L}}$.

It is known that a 3-round adaptive-input special-sound special-HVZK proof can be constructed from any non-interactive commitment scheme. Specifically, it suffices to augment the delayed-input proof of knowledge of Lapidot and Shamir [LS91] with a trick by Ciampi et al. [COSV17b, Appendix C.1] to obtain delayed-input special soundness. The resulting scheme has a public-coin verifier.

A.2 Protocol Description

Fix any NP language $\mathbf{L}$. We use the following building blocks.

- A non-interactive perfectly binding commitment scheme $\mathsf{Com} = (\mathsf{Com.S}, \mathsf{Com.R_D})$.
- A non-interactive instance-dependent trapdoor commitment scheme $\mathsf{TC} = (\mathsf{TC.S}, \mathsf{TC.R_D}, \mathsf{TC.FakeS}, \mathsf{TC.FakeOpen})$ for the NP language $\mathbf{L}_{\mathsf{tcom}} := \{(\lambda, \mathsf{com}, b) \mid \exists \mathsf{dec} \text{ s.t. } \mathsf{Com.R_D}(1^\lambda, \mathsf{com}, b, \mathsf{dec}) = 1\}$. In words, $(\lambda, \mathsf{com}, b) \in \mathbf{L}_{\mathsf{tcom}}$ if and only if com is a valid Com-commitment to b with security parameter λ.
- A 3-round adaptive-input special-sound special-HVZK proof $\mathsf{LS} = (\mathsf{LS.P}_1, \mathsf{LS.P}_2, \mathsf{LS.V}, \mathsf{LS.V_D})$ for $\mathbf{L}$. Let LS.Sim be the simulator for the special HVZK property and LS.Ext be the extractor for the delayed-input special soundness.

At a high level, the protocol proceeds as follows. To prove a statement $x_0 \in \mathbf{L} \vee x_1 \in \mathbf{L}$, the prover generates two LS proofs, one for the statement $x_0 \in \mathbf{L}$ and the other for the statement $x_1 \in \mathbf{L}$. To complete these two proofs while having only a witness for $x_\beta \in \mathbf{L}$, the prover avoids sending messages in the clear in the first round. Instead, the prover commits to the messages in the first round and reveals them in the last round after receiving challenge messages from the verifier. The key point is that the prover makes the commitments by using TC so that the commitment is binding in the proof for $x_\beta \in \mathbf{L}$ and is equivocal in the proof for $x_{1-\beta} \in \mathbf{L}$. On the one hand, the prover can complete the proof for $x_\beta \in \mathbf{L}$ since the prover has a valid witness for the statement. On the other hand, the prover can complete the proof for $x_{1-\beta} \in \mathbf{L}$ by using the SHVZK simulator of LS. Specifically, for the latter, the prover generates an accepting transcript by using the SHVZK simulator and opens the commitments accordingly using the equivocality of TC.

The formal description of the protocol, $\mathsf{WI} = (\mathsf{WI.P}_1, \mathsf{WI.P}_2, \mathsf{WI.V}, \mathsf{WI.V_D})$, is given in Figure 6.

A.3 Security proof

Let $(\mathsf{WI.}\widehat{\mathsf{P}}_1, \mathsf{WI.}\widehat{\mathsf{P}}_2, \mathsf{WI.}\widehat{\mathsf{P}}_{1.5})$ be defined as in Figure 7. We prove each of the required properties below.

For each $b \in \{0,1\}$, let $(x_b, w_b) \in \mathbf{R}_{\mathbf{L}}$ be an instance of the NP language $\mathbf{L}$ and a corresponding witness. Suppose $|x_0| = |x_1|$. Let $\lambda := |x_0|$ denote the security parameter.

Round 1. $\mathsf{WI.P}_1(1^\lambda, \beta)$:

1. Compute $(\mathsf{com}, \mathsf{dec}) \leftarrow \mathsf{Com.S}(1^\lambda, 1-\beta)$.
2. Compute $(\mathsf{ls}_{\beta,1}, \mathsf{st}_{\mathsf{ls},\beta}) \leftarrow \mathsf{LS.P}_1(1^\lambda)$.
3. Compute $(\mathsf{tcom}_\beta, \mathsf{tdec}_\beta) \leftarrow \mathsf{TC.S}(1^\lambda, x_{\mathsf{tcom},\beta}, \mathsf{ls}_{\beta,1})$ and $(\mathsf{tcom}_{1-\beta}, \mathsf{st}_{\mathsf{tcom},1-\beta}) \leftarrow \mathsf{TC.FakeS}(1^\lambda, x_{\mathsf{tcom},1-\beta})$, where $x_{\mathsf{tcom},b} := (\lambda, \mathsf{com}, b)$ is viewed as an instance of $\mathbf{L}_{\mathsf{tcom}}$ for each $b \in \{0,1\}$ so that $x_{\mathsf{tcom},\beta} \notin \mathbf{L}_{\mathsf{tcom}}$ and $x_{\mathsf{tcom},1-\beta} \in \mathbf{L}_{\mathsf{tcom}}$.
4. Output $\mathsf{wi}_1 := (\mathsf{com}, \{\mathsf{tcom}_b\}_{b\in\{0,1\}})$ and $\mathsf{st} := (\beta, \mathsf{dec}, \mathsf{ls}_{\beta,1}, \mathsf{st}_{\mathsf{ls},\beta}, \mathsf{tdec}_\beta, \mathsf{st}_{\mathsf{tcom},1-\beta})$.

Round 2. $\mathsf{WI.V}(1^\lambda)$:

1. $\forall b \in \{0,1\}$: Compute $\mathsf{ls}_{b,2} \leftarrow \mathsf{LS.V}(1^\lambda)$.
2. Output $\mathsf{wi}_2 := \{\mathsf{ls}_{b,2}\}_{b\in\{0,1\}}$.

Round 3. $\mathsf{WI.P}_2(\mathsf{st}, x_0, x_1, w_\beta, \mathsf{wi}_2)$:

1. Compute $\mathsf{ls}_{\beta,3} \leftarrow \mathsf{LS.P}_2(\mathsf{st}_{\mathsf{ls},\beta}, x_\beta, w_\beta, \mathsf{ls}_{\beta,2})$.
2. Compute $(\mathsf{ls}_{1-\beta,1}, \mathsf{ls}_{1-\beta,3}) \leftarrow \mathsf{LS.Sim}(x_{1-\beta}, \mathsf{ls}_{1-\beta,2})$.
3. Compute $\mathsf{tdec}_{1-\beta} := \mathsf{TC.FakeOpen}(\mathsf{st}_{\mathsf{tcom},1-\beta}, w_{\mathsf{tcom}}, \mathsf{ls}_{1-\beta,1})$, where $w_{\mathsf{tcom}} := \mathsf{dec}$ is viewed as a witness for $x_{\mathsf{tcom},1-\beta} \in \mathbf{L}_{\mathsf{tcom}}$.
4. Output $\mathsf{wi}_3 := \{\mathsf{ls}_{b,1}, \mathsf{ls}_{b,3}, \mathsf{tdec}_b\}_{b\in\{0,1\}}$.

Verification. $\mathsf{WI.V}_{\mathsf{D}}(x_0, x_1, \tau)$, where $\tau = (\mathsf{wi}_1, \mathsf{wi}_2, \mathsf{wi}_3)$:

1. Output 1 if and only if both of the following hold.
   - $\forall b \in \{0,1\}$: $\mathsf{TC.R}_{\mathsf{D}}(1^\lambda, x_{\mathsf{tcom},b}, \mathsf{tcom}_b, \mathsf{ls}_{b,1}, \mathsf{tdec}_b) = 1$.
   - $\forall b \in \{0,1\}$: $\mathsf{LS.V}_{\mathsf{D}}(x_b, \tau_{\mathsf{ls},b}) = 1$, where $\tau_{\mathsf{ls},b} := (\mathsf{ls}_{b,1}, \mathsf{ls}_{b,2}, \mathsf{ls}_{b,3})$.

Figure 6: 3-round adaptive-input OR-indistinguishable special-sound proof for $\mathbf{L}$.

Delayed-input correctness. Delayed-input correctness can be verified by inspection.

Delayed-input special soundness. Define the function EQ and the extractor Ext as follows.

- $\mathsf{EQ}(\mathsf{wi}_2, \mathsf{wi}_2')$: Parse wi_2 as $\mathsf{wi}_2 = \{\mathsf{ls}_{b,2}\}_{b\in\{0,1\}}$ and wi_2' as $\mathsf{wi}_2' = \{\mathsf{ls}_{b,2}'\}_{b\in\{0,1\}}$. Then, return 1 if and only if $\exists b \in \{0,1\}$ s.t. $\mathsf{ls}_{b,2} = \mathsf{ls}_{b,2}'$.

Alternative Round 1. $\widehat{\mathsf{WI.P}}_1(1^\lambda, \beta)$:

1. Compute $(\mathsf{com}, \mathsf{dec}) \leftarrow \mathsf{Com.S}(1^\lambda, 1-\beta)$.
2. $\forall b \in \{0,1\}$: Compute $(\mathsf{ls}_{b,1}, \mathsf{st}_{\mathsf{ls},b}) \leftarrow \mathsf{LS.P}_1(1^\lambda)$.
3. Compute $(\mathsf{tcom}_\beta, \mathsf{tdec}_\beta) \leftarrow \mathsf{TC.S}(1^\lambda, x_{\mathsf{tcom},\beta}, \mathsf{ls}_{\beta,1})$ and $(\mathsf{tcom}_{1-\beta}, \mathsf{st}_{\mathsf{tcom},1-\beta}) \leftarrow \mathsf{TC.FakeS}(1^\lambda, x_{\mathsf{tcom},1-\beta})$, where $x_{\mathsf{tcom},b} := (\lambda, \mathsf{com}, b)$ is viewed as an instance of $\mathbf{L}_{\mathsf{tcom}}$ for each $b \in \{0,1\}$ so that $x_{\mathsf{tcom},\beta} \notin \mathbf{L}_{\mathsf{tcom}}$ and $x_{\mathsf{tcom},1-\beta} \in \mathbf{L}_{\mathsf{tcom}}$.
4. Output $\mathsf{wi}_1 := (\mathsf{com}, \{\mathsf{tcom}_b\}_{b\in\{0,1\}})$, $\mathsf{st} := (\beta, \mathsf{dec}, \mathsf{ls}_{\beta,1}, \mathsf{st}_{\mathsf{ls},\beta}, \mathsf{tdec}_\beta, \mathsf{st}_{\mathsf{tcom},1-\beta})$, and $\mathsf{aux} := (\mathsf{ls}_{1-\beta,1}, \mathsf{st}_{\mathsf{ls},1-\beta})$.

Alternative Round 3. $\widehat{\mathsf{WI.P}}_2(\mathsf{st}, \boxed{\mathsf{aux}}, x_0, x_1, \boxed{w_0}, \boxed{w_1}, \mathsf{wi}_2)$:

1. Compute $\mathsf{ls}_{\beta,3} \leftarrow \mathsf{LS.P}_2(\mathsf{st}_{\mathsf{ls},\beta}, x_\beta, w_\beta, \mathsf{ls}_{\beta,2})$.
2. Compute $\mathsf{ls}_{1-\beta,3} \leftarrow \mathsf{LS.P}_2(\mathsf{st}_{\mathsf{ls},1-\beta}, x_{1-\beta}, w_{1-\beta}, \mathsf{ls}_{1-\beta,2})$.
3. Compute $\mathsf{tdec}_{1-\beta} := \mathsf{TC.FakeOpen}(\mathsf{st}_{\mathsf{tcom},1-\beta}, w_{\mathsf{tcom}}, \mathsf{ls}_{1-\beta,1})$, where $w_{\mathsf{tcom}} := \mathsf{dec}$ is viewed as a witness for $x_{\mathsf{tcom},1-\beta} \in \mathbf{L}_{\mathsf{tcom}}$.
4. Output $\mathsf{wi}_3 := \{\mathsf{ls}_{b,1}, \mathsf{ls}_{b,3}, \mathsf{tdec}_b\}_{b\in\{0,1\}}$.

. . . . . .

Round 1.5. $\widehat{\mathsf{WI.P}}_{1.5}(1^\lambda, \beta)$:

1. Compute $(\mathsf{ls}_{1-\beta,1}, \mathsf{st}_{\mathsf{ls},1-\beta}) \leftarrow \mathsf{LS.P}_1(1^\lambda)$.
2. Output $\mathsf{aux} := (\mathsf{ls}_{1-\beta,1}, \mathsf{st}_{\mathsf{ls},1-\beta})$.

Figure 7: Alternative prover algorithms of WI.

- $\mathsf{Ext}(x, \tau, x', \tau')$:
  1. Parse x, τ, x', and τ' as

$$
\begin{aligned}
x &= (x_0, x_1),\\
\tau &= (\mathsf{wi}_1, \mathsf{wi}_2, \mathsf{wi}_3)\\
&= ((\mathsf{com}, \{\mathsf{tcom}_b\}_{b\in\{0,1\}}), \{\mathsf{ls}_{b,2}\}_{b\in\{0,1\}}, \{\mathsf{ls}_{b,1}, \mathsf{ls}_{b,3}, \mathsf{tdec}_b\}_{b\in\{0,1\}}),\\
x' &= (x'_0, x'_1), \text{ and}\\
\tau' &= (\mathsf{wi}'_1, \mathsf{wi}'_2, \mathsf{wi}'_3)\\
&= ((\mathsf{com}', \{\mathsf{tcom}'_b\}_{b\in\{0,1\}}), \{\mathsf{ls}'_{b,2}\}_{b\in\{0,1\}}, \{\mathsf{ls}'_{b,1}, \mathsf{ls}'_{b,3}, \mathsf{tdec}'_b\}_{b\in\{0,1\}}).
\end{aligned}
$$

     If $\mathsf{wi}_1 \neq \mathsf{wi}'_1$, abort.
  2. Find $b^* \in \{0,1\}$ such that $\mathsf{ls}_{b^*,1} = \mathsf{ls}'_{b^*,1}$. If such b^* does not exist, abort.

3. Return $\mathsf{LS.Ext}(x_{b^*}, (\mathsf{ls}_{b^*,1}, \mathsf{ls}_{b^*,2}, \mathsf{ls}_{b^*,3}), x'_{b^*}, (\mathsf{ls}'_{b^*,1}, \mathsf{ls}'_{b^*,2}, \mathsf{ls}'_{b^*,3}))$.

First, we analyze EQ. Since LS has a public-coin verifier, we can assume without loss of generality that the following holds for any $\lambda \in \mathbb{N}$.[36]

$$\forall b \in \{0,1\} : \Pr\left[\mathsf{ls}_{b,2} = \mathsf{ls}'_{b,2} \;\middle|\; \begin{array}{l} \mathsf{ls}_{b,2} \leftarrow \mathsf{LS.V}(1^\lambda) \\ \mathsf{ls}'_{b,2} \leftarrow \mathsf{LS.V}(1^\lambda) \end{array}\right] \leq \frac{1}{2^{\lambda+1}}.$$

Thus, from the union bound, we have

$$\begin{aligned}
&\Pr\left[\mathsf{EQ}(\{\mathsf{ls}_{b,2}\}_{b\in\{0,1\}}, \{\mathsf{ls}'_{b,2}\}_{b\in\{0,1\}}) = 1 \;\middle|\; \begin{array}{l} \{\mathsf{ls}_{b,2}\}_{b\in\{0,1\}} \leftarrow \mathsf{WI.V}(1^\lambda) \\ \{\mathsf{ls}'_{b,2}\}_{b\in\{0,1\}} \leftarrow \mathsf{WI.V}(1^\lambda) \end{array}\right] \\
&= \Pr\left[\exists b \in \{0,1\} \text{ s.t. } \mathsf{ls}_{b,2} = \mathsf{ls}'_{b,2} \;\middle|\; \begin{array}{l} \{\mathsf{ls}_{b,2}\}_{b\in\{0,1\}} \leftarrow \mathsf{WI.V}(1^\lambda) \\ \{\mathsf{ls}'_{b,2}\}_{b\in\{0,1\}} \leftarrow \mathsf{WI.V}(1^\lambda) \end{array}\right] \\
&\leq \frac{1}{2^{\lambda+1}} + \frac{1}{2^{\lambda+1}} = \frac{1}{2^\lambda}.
\end{aligned}$$

Next, we analyze Ext. Fix any

$$\begin{aligned}
x &= (x_0, x_1), \\
\tau &= (\mathsf{wi}_1, \mathsf{wi}_2, \mathsf{wi}_3) \\
&= ((\mathsf{com}, \{\mathsf{tcom}_b\}_{b\in\{0,1\}}), \{\mathsf{ls}_{b,2}\}_{b\in\{0,1\}}, \{\mathsf{ls}_{b,1}, \mathsf{ls}_{b,3}, \mathsf{tdec}_b\}_{b\in\{0,1\}}), \\
x' &= (x'_0, x'_1), \text{ and} \\
\tau' &= (\mathsf{wi}'_1, \mathsf{wi}'_2, \mathsf{wi}'_3) \\
&= ((\mathsf{com}', \{\mathsf{tcom}'_b\}_{b\in\{0,1\}}), \{\mathsf{ls}'_{b,2}\}_{b\in\{0,1\}}, \{\mathsf{ls}'_{b,1}, \mathsf{ls}'_{b,3}, \mathsf{tdec}'_b\}_{b\in\{0,1\}}).
\end{aligned}$$

Suppose we have

$$\mathsf{wi}_1 = \mathsf{wi}'_1, \tag{14}$$

$$\mathsf{EQ}(\mathsf{wi}_2, \mathsf{wi}'_2) = 0, \text{ and} \tag{15}$$

$$\mathsf{WI.V}_{\mathsf{D}}(x_0, x_1, \tau) = 1 \wedge \mathsf{WI.V}_{\mathsf{D}}(x'_0, x'_1, \tau') = 1. \tag{16}$$

Let $\lambda := |x_0|$. First, we note that there always exists $b^* \in \{0,1\}$ such that $\mathsf{ls}_{b^*,1} = \mathsf{ls}'_{b^*,1}$ (i.e., Ext never aborts). Indeed, the perfect binding of Com guarantees that there exists $b^* \in \{0,1\}$ such that $(\lambda, \mathsf{com}, b^*) \notin \mathbf{L}_{\mathsf{tcom}}$, and for any such b^*, the perfect binding of TC implies $\mathsf{ls}_{b^*,1} = \mathsf{ls}'_{b^*,1}$. Next, we observe that Ext outputs a valid witness. Fix any $b^* \in \{0,1\}$ such that $\mathsf{ls}_{b^*,1} = \mathsf{ls}'_{b^*,1}$. By definition, Equation (15) implies $\mathsf{ls}_{b^*,2} \neq \mathsf{ls}'_{b^*,2}$ and Equation (16) implies $\mathsf{LS.V}_{\mathsf{D}}(x_{b^*}, (\mathsf{ls}_{b^*,1}, \mathsf{ls}_{b^*,2}, \mathsf{ls}_{b^*,3})) = 1 \wedge \mathsf{LS.V}_{\mathsf{D}}(x'_{b^*}, (\mathsf{ls}'_{b^*,1}, \mathsf{ls}'_{b^*,2}, \mathsf{ls}'_{b^*,3})) = 1$. Thus, the delayed-input special soundness of LS guarantees that $\mathsf{LS.Ext}(x_{b^*}, (\mathsf{ls}_{b^*,1}, \mathsf{ls}_{b^*,2}, \mathsf{ls}_{b^*,3}), x'_{b^*}, (\mathsf{ls}'_{b^*,1}, \mathsf{ls}'_{b^*,2}, \mathsf{ls}'_{b^*,3}))$ outputs a value w such that $(x_{b^*}, w) \in \mathbf{R}_{\mathbf{L}}$.

Adaptive-input OR indistinguishability. Adaptive-input OR indistinguishability can be shown by combining (i) indistinguishability between the real and hybrid provers and (ii) adaptive-input OR-indistinguishability of the hybrid prover (both of which are shown below).

[36] In particular, it suffices to assume that LS is run sufficiently many times in parallel.

Delayed-input correctness of hybrid prover. Delayed-input correctness of the hybrid prover can be verified by inspection.

Indistinguishability between real and hybrid provers. The equivalence between $\mathsf{WI}.\widehat{\mathsf{P}}_1$ and $(\mathsf{WI}.\mathsf{P}_1, \mathsf{WI}.\widehat{\mathsf{P}}_{1.5})$ can be verified by inspection. For the indistinguishability between $\mathsf{WI}.\mathsf{P}_2$ and $(\mathsf{WI}.\widehat{\mathsf{P}}_{1.5}, \mathsf{WI}.\widehat{\mathsf{P}}_2)$, observe that the only difference between the two experiments is that in the first one, $\mathsf{ls}_{1-\beta,1}$ and $\mathsf{ls}_{1-\beta,3}$ are generated by $\mathsf{LS}.\mathsf{Sim}$ whereas in the second one, $\mathsf{ls}_{1-\beta,1}$ and $\mathsf{ls}_{1-\beta,3}$ are generated by $\mathsf{LS}.\mathsf{P}_1$ and $\mathsf{LS}.\mathsf{P}_2$. Thus, the indistinguishability follows directly from the delayed-input special HVZK of LS.

Adaptive-input (rewinding) OR-indistinguishability of hybrid prover. We consider the following hybrid prover by modifying $\mathsf{WI}.\widehat{\mathsf{P}}_1$ and $\mathsf{WI}.\widehat{\mathsf{P}}_2$.

- $\mathsf{WI}.\widehat{\mathsf{P}}'_1(1^\lambda, \beta)$:
  1. Compute $(\mathsf{com}, \mathsf{dec}) \leftarrow \mathsf{Com}.\mathsf{S}(1^\lambda, 1-\beta)$.
  2. $\forall b \in \{0,1\}$: Compute $(\mathsf{ls}_{b,1}, \mathsf{st}_{\mathsf{ls},b}) \leftarrow \mathsf{LS}.\mathsf{P}_1(1^\lambda)$.
  3. $\boxed{\forall b \in \{0,1\}\text{: Compute } (\mathsf{tcom}_b, \mathsf{tdec}_b) \leftarrow \mathsf{TC}.\mathsf{S}(1^\lambda, x_{\mathsf{tcom},b}, \mathsf{ls}_{b,1}).}$
  4. Output $\mathsf{wi}_1 := (\mathsf{com}, \{\mathsf{tcom}_b\}_{b\in\{0,1\}})$, $\mathsf{st} := (\beta, \mathsf{dec}, \mathsf{ls}_{\beta,1}, \mathsf{st}_{\mathsf{ls},\beta}, \mathsf{tdec}_\beta, \boxed{\mathsf{tdec}_{1-\beta}})$, and $\mathsf{aux} := (\mathsf{ls}_{1-\beta,1}, \mathsf{st}_{\mathsf{ls},1-\beta})$.
- $\mathsf{WI}.\widehat{\mathsf{P}}'_2(\mathsf{st}, \mathsf{aux}, x_0, x_1, w_0, w_1, \mathsf{wi}_2)$:
  1. Compute $\mathsf{ls}_{\beta,3} \leftarrow \mathsf{LS}.\mathsf{P}_2(\mathsf{st}_{\mathsf{ls},\beta}, x_\beta, w_\beta, \mathsf{ls}_{\beta,2})$.
  2. Compute $\mathsf{ls}_{1-\beta,3} \leftarrow \mathsf{LS}.\mathsf{P}_2(\mathsf{st}_{\mathsf{ls},1-\beta}, x_{1-\beta}, w_{1-\beta}, \mathsf{ls}_{1-\beta,2})$.
  3. ~~Compute $\mathsf{tdec}_{1-\beta} := \mathsf{TC}.\mathsf{FakeOpen}(\mathsf{st}_{\mathsf{tcom},1-\beta}, w_{\mathsf{tcom}}, \mathsf{ls}_{1-\beta,1})$, where $w_{\mathsf{tcom}} :=$ dec is viewed as a witness for $x_{\mathsf{tcom},1-\beta} \in \mathbf{L}_{\mathsf{tcom}}$.~~
  4. Output $\mathsf{wi}_3 := \{\mathsf{ls}_{b,1}, \mathsf{ls}_{b,3}, \mathsf{tdec}_b\}_{b\in\{0,1\}}$.

Using the equivocality hiding of TC, we obtain

$$\Pr\left[\beta = \widetilde{\beta} \;\middle|\; \begin{array}{l} \beta \leftarrow \{0,1\} \\ (\mathsf{wi}_1, \mathsf{st}, \mathsf{aux}) \leftarrow \mathsf{WI}.\widehat{\mathsf{P}}_1(1^\lambda, \beta) \\ \widetilde{\beta} \leftarrow \mathcal{V}^{\mathcal{O}(\mathsf{st},\mathsf{aux},\cdot)}(1^\lambda, \mathsf{wi}_1, z) \end{array}\right]$$
$$\leq \Pr\left[\beta = \widetilde{\beta} \;\middle|\; \begin{array}{l} \beta \leftarrow \{0,1\} \\ (\mathsf{wi}_1, \mathsf{st}, \mathsf{aux}) \leftarrow \boxed{\mathsf{WI}.\widehat{\mathsf{P}}'_1}(1^\lambda, \beta) \\ \widetilde{\beta} \leftarrow \mathcal{V}^{\boxed{\mathcal{O}'}(\mathsf{st},\mathsf{aux},\cdot)}(1^\lambda, \mathsf{wi}_1, z) \end{array}\right] + \mathsf{negl}(\lambda),$$

where the oracle $\mathcal{O}'$ is defined as follows.

- $\underline{\mathcal{O}'(\mathsf{st}, \mathsf{aux}, x_0, x_1, w_0, w_1, \mathsf{wi}_2)}$: If $|x_b| = \lambda$ and $(x_b, w_b) \in \mathbf{R}_{\mathbf{L}}$ for every $b \in \{0,1\}$, return $\boxed{\mathsf{WI}.\widehat{\mathsf{P}}'_2}(\mathsf{st}, \mathsf{aux}, x_0, x_1, w_0, w_1, \mathsf{wi}_2)$. Otherwise, return $\bot$.

Also, using the hiding of Com, we obtain

$$\Pr\left[\beta = \widetilde{\beta} \;\middle|\; \begin{array}{l} \beta \leftarrow \{0,1\} \\ (\mathsf{wi}_1, \mathsf{st}, \mathsf{aux}) \leftarrow \mathsf{WI}.\widehat{\mathsf{P}}'_1(1^\lambda, \beta) \\ \widetilde{\beta} \leftarrow \mathcal{V}^{\mathcal{O}'(\mathsf{st},\mathsf{aux},\cdot)}(1^\lambda, \mathsf{wi}_1, z) \end{array}\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda).$$

Combining the above, we obtain the following as desired.

$$\Pr\left[\beta = \widetilde{\beta} \;\middle|\; \begin{array}{l} \beta \leftarrow \{0,1\} \\ (\mathsf{wi}_1, \mathsf{st}, \mathsf{aux}) \leftarrow \mathsf{WI}.\widehat{\mathsf{P}}_1(1^\lambda, \beta) \\ \widetilde{\beta} \leftarrow \mathcal{V}^{\mathcal{O}(\mathsf{st},\mathsf{aux},\cdot)}(1^\lambda, \mathsf{wi}_1, z) \end{array}\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda).$$

B Full Description of Hybrid Simulator-extractor $\mathcal{SE}'$

Hybrid simulator-extractor $\mathcal{SE}'$.

Simulation part. On input $(1^\lambda, z)$, the simulator-extractor $\mathcal{SE}'$ internally invokes $\mathcal{A}(1^\lambda, z)$ and emulates the left and right interactions for $\mathcal{A}$ in the following way.
In each right interaction, $\mathcal{SE}'$ interacts with $\mathcal{A}$ using the honest verifier algorithms. That is, $\mathcal{SE}'$ interacts with $\mathcal{A}$ in the i-th right interaction as follows.
Round i-1. Receiving $\widetilde{\mathsf{tag}}_i$, compute $(\widetilde{\mathsf{m}}_{i,1}, \widetilde{\mathsf{st}}_{V,i}) \leftarrow \mathsf{V}_1(1^\lambda, \widetilde{\mathsf{tag}}_i)$ and send $\widetilde{\mathsf{m}}_{i,1}$ to $\mathcal{A}$.

Round i-2. Receiving $\widetilde{\mathsf{m}}_{i,2}$, compute $\widetilde{\mathsf{m}}_{i,3} \leftarrow \mathsf{V}_2(\widetilde{\mathsf{st}}_{V,i}, \widetilde{\mathsf{m}}_{i,2})$ and send $\widetilde{\mathsf{m}}_{i,3}$ to $\mathcal{A}$.

In the left interaction, $\mathcal{SE}'$ interacts with $\mathcal{A}$ as follows.

Round 2. Receiving $(\mathsf{tag}, \mathsf{m}_1)$, compute $(\mathsf{m}_2, \mathsf{st}_P) \leftarrow \mathsf{P}_1(1^\lambda, \mathsf{tag}, \mathsf{m}_1)$ and send m_2 to $\mathcal{A}$.

Round 4. Receiving (x, m_3), do the following.

4:1 Like the honest prover, check whether m_3 is accepting, i.e., whether $\mathsf{WI}.\mathsf{V}_\mathsf{D}(x_{V,0}, x_{V,1}, \tau_\mathsf{wi}) \stackrel{?}{=} 1$ for the WI statements $x_{V,0}, x_{V,1}$ and transcript τ_wi defined based on $(\mathsf{m}_1, \mathsf{m}_2, \mathsf{m}_3)$. If the check fails, abort the left interaction.

4:2 If the check passes, do the following.

First left rewinding. Rewind $\mathcal{A}$ to the point just before sending m_2 to $\mathcal{A}$ in Round 2. Then, sample a new Round-2 message m'_2 using $\mathsf{P}_1(1^\lambda, \mathsf{tag}, \mathsf{m}_1)$ with fresh randomness, send m'_2 to $\mathcal{A}$, and continue interacting with $\mathcal{A}$ until obtaining a new Round-3 message m'_3. (During the interaction with $\mathcal{A}$, the right interactions are simulated as before using fresh randomness.) Let $(x'_{V,0}, x'_{V,1})$ and τ'_wi denote the WI statements and transcript defined based on $(\mathsf{m}_1, \mathsf{m}'_2, \mathsf{m}'_3)$.

Second left rewinding (same as above). Rewind $\mathcal{A}$ to the point just before sending m_2 to $\mathcal{A}$ in Round 2. Then, sample a new Round-2 message m_2'' using $\mathsf{P}_1(1^\lambda, \mathsf{tag}, \mathsf{m}_1)$ with fresh randomness, send m_2'' to $\mathcal{A}$, and continue interacting with $\mathcal{A}$ until obtaining a new Round-3 message m_3''. (During the interaction with $\mathcal{A}$, the right interactions are simulated as before using fresh randomness.) Let $(x''_{V,0}, x''_{V,1})$ and τ''_{wi} denote the WI statements and transcript defined based on $(\mathsf{m}_1, \mathsf{m}_2'', \mathsf{m}_3'')$.

If either m_3' or m_3'' is not accepting, output fail and abort. Otherwise, restore the state of $\mathcal{A}$ to the one that $\mathcal{A}$ had before the above rewinding.

4:3 Compute $w_V \coloneqq \mathsf{WI.Ext}((x'_{V,0}, x'_{V,1}), \tau'_{\mathsf{wi}}, (x''_{V,0}, x''_{V,1}), \tau''_{\mathsf{wi}})$. If $\exists \beta \in \{0,1\}$ s.t. $(x'_{V,\beta}, w_V) \in \mathbf{R}_{\mathbf{L}_V}$, parse w_V as $(\text{i-dec}, \text{o-dec})$ and let $w_{\mathsf{td}} \coloneqq (\beta, \text{i-dec})$. If there is no such β, output fail and abort.

4:4 Compute $\mathsf{swi}_4 \leftarrow \mathsf{SWI.P}_2(\mathsf{st}_{\mathsf{swi}}, x_P, w_{\mathsf{td}}, \mathsf{swi}_3)$ by using w_{td} along with the SWI prover state $\mathsf{st}_{\mathsf{swi}}$, statement x_P, and challenge swi_3 that are defined based on $(x, \mathsf{m}_1, \mathsf{m}_2, \mathsf{m}_3)$. Then, send $\mathsf{m}_4 \coloneqq \mathsf{swi}_4$ to $\mathcal{A}$.

Let sim-view be the joint view of $\mathcal{A}$ and the honest verifiers emulated in the above. We call the execution of $\mathcal{A}$ recorded in sim-view the *main thread*, and we call the executions of $\mathcal{A}$ generated during the left rewinding the *left look-ahead threads*. We call w_{td} (the witness used for SWI in Step 4:4) the *trapdoor witness*. If the messages in the left and the i^*-th right interaction are not scheduled in Schedule 1 in the main thread, output fail and abort.

Extraction part. After completing the simulation part, $\mathcal{SE}'$ defines witness candidates $\{\widetilde{w}_i\}_{i\in[m(\lambda)]}$ in the following way. For every $i \in [m(\lambda)] \setminus \{i^*\}$, $\mathcal{SE}'$ lets $\widetilde{w}_i \coloneqq \bot$. Also, $\mathcal{SE}'$ lets $\widetilde{w}_{i^*} \coloneqq \bot$ if the i^*-th right interaction is rejecting or uses the same tag as the left interaction. Otherwise, $\mathcal{SE}'$ defines $\widetilde{w}_{i^*}$ as follows. Let $\widetilde{x}_{i^*}$ and $\widetilde{\mathsf{m}}_{i^*,4}$ denote the statement and message sent by $\mathcal{A}$ in Round i^*-4.

i^*:1 Do the following.

Right rewinding. Rewind $\mathcal{A}$ to the point just before sending $\widetilde{\mathsf{m}}_{i^*,3}$ to $\mathcal{A}$ in Round i^*-3. Then, define a new Round-3 message $\widetilde{\mathsf{m}}'_{i^*,3}$ as follows: parse $\widetilde{\mathsf{m}}_{i^*,3}$ as $\widetilde{\mathsf{m}}_{i^*,3} = (\{\widetilde{\text{i-nm}}_{b,3}, \widetilde{\text{o-nm}}_{b,3}\}_{b\in\{0,1\}}, \widetilde{\mathsf{wi}}_3, \widetilde{\mathsf{swi}}_3)$; then, sample a new challenge message $\widetilde{\mathsf{swi}}'_3 \leftarrow \mathsf{SWI.V}_2(1^{\lambda_P})$ for SWI and let $\widetilde{\mathsf{m}}'_{i^*,3} \coloneqq (\{\widetilde{\text{i-nm}}_{b,3}, \widetilde{\text{o-nm}}_{b,3}\}_{b\in\{0,1\}}, \widetilde{\mathsf{wi}}_3, \widetilde{\mathsf{swi}}'_3)$. Then, send $\widetilde{\mathsf{m}}'_{i^*,3}$ to $\mathcal{A}$ and continue interacting with $\mathcal{A}$ until obtaining a new Round-4 message $\widetilde{\mathsf{m}}'_{i^*,4}$ along with a new statement $\widetilde{x}'_{i^*}$ in Round i^*-4. (During the interaction with $\mathcal{A}$, the left and other right interactions are simulated as before with the following exception. If EVENT_0 or EVENT_1 occurred in the main thread, then when

Round 4 of the left interaction is simulated, Steps 4:2 and 4:3 are skipped, and Step 4:4 is executed by reusing the witness w_{td} that was used in the main thread.)

Let us call the interactions with $\mathcal{A}$ during the right rewinding the *right look-ahead thread*. If either of the following holds, output fail and abort.

- In the main thread, EVENT_0 or EVENT_1 occurs, and in the right look-ahead thread, either the new Round-4 message $\widetilde{\mathsf{m}}'_{i^*,4}$ is not accepting or w_{td} is not a valid witness for SWI.
- In the main thread, EVENT_2 occurs, and in the right look-ahead thread, EVENT_2 does not occur.

i^*:2 Compute $\widetilde{w}_{P,i^*} := \mathsf{SWI.Ext}(\widetilde{x}_{P,i^*}, \widetilde{\tau}_{\mathsf{swi},i^*}, \widetilde{x}'_{P,i^*}, \widetilde{\tau}'_{\mathsf{swi},i^*})$ for (i) the SWI statement $\widetilde{x}_{P,i^*}$ and transcript $\widetilde{\tau}_{\mathsf{swi},i^*}$ defined based on $(\widetilde{x}_{i^*}, \widetilde{\mathsf{m}}_{i^*,1}, \widetilde{\mathsf{m}}_{i^*,2}, \widetilde{\mathsf{m}}_{i^*,3}, \widetilde{\mathsf{m}}_{i^*,4})$ and (ii) the SWI statement $\widetilde{x}'_{P,i^*}$ and transcript $\widetilde{\tau}'_{\mathsf{swi},i^*}$ defined based on $(\widetilde{x}'_{i^*}, \widetilde{\mathsf{m}}_{i^*,1}, \widetilde{\mathsf{m}}_{i^*,2}, \widetilde{\mathsf{m}}'_{i^*,3}, \widetilde{\mathsf{m}}'_{i^*,4})$. If $(\widetilde{x}_{P,i^*}, \widetilde{w}_{P,i^*}) \in \mathbf{R}_{\mathbf{L}_P}$, let $\widetilde{w}_{i^*} := \widetilde{w}_{P,i^*}$. Otherwise, let $\widetilde{w}_{i^*} := \bot_{\mathsf{fail}}$.

Final output. After completing the extraction part, $\mathcal{SE}'$ outputs $(\mathsf{sim\text{-}view}, \{\widetilde{w}_i\}_{i\in[m(\lambda)]})$.

C Proof of Claim 4

Proof. First, we analyze the probability that $\mathsf{BAD}_{i^*,1}$ occurs in $\mathcal{SE}_1$. By definition, $\mathsf{BAD}_{i^*,1}$ occurs only when the i^*-th right interaction is scheduled in Schedule 1 in the main thread. Thus, aborting in the other cases does not decrease the probability that $\mathsf{BAD}_{i^*,1}$ occurs. Therefore, $\mathsf{BAD}_{i^*,1}$ occurs in $\mathcal{SE}_1$ with probability at least $\varepsilon_1 := 1/q(\lambda)$.

Next, we analyze the probability that $\mathsf{BAD}_{i^*,1}$ occurs in $\mathcal{SE}_2$. Once the simulation part is completed, the steps defining $\widetilde{w}_{i^*}$ are independent of the steps defining $\widetilde{w}_i$ for any $i \neq i^*$. Thus, skipping the steps defining $\widetilde{w}_i$ for every $i \neq i^*$ does not decrease the probability that $\mathsf{BAD}_{i^*,1}$ occurs. Therefore, $\mathsf{BAD}_{i^*,1}$ occurs in $\mathcal{SE}_2$ with probability at least $\varepsilon_2 := \varepsilon_1$.

Next, we analyze the probability that $\mathsf{BAD}_{i^*,1}$ occurs in $\mathcal{SE}_3$. Since the expected number of the right look-ahead threads is 1, Markov's inequality implies that the number of the right look-ahead threads exceeds $2/\varepsilon_2$ with probability at most $\varepsilon_2/2$. Then, since each right look-ahead thread is generated identically, it follows that even when only a single right look-ahead thread is generated, $\mathsf{BAD}_{i^*,1}$ occurs with probability at least $(\varepsilon_2 - \varepsilon_2/2) \cdot (2/\varepsilon_2)^{-1} = (\varepsilon_2/2)^2$. Therefore, $\mathsf{BAD}_{i^*,1}$ occurs in $\mathcal{SE}_3$ with probability at least $\varepsilon_3 := (\varepsilon_2/2)^2$.

Next, we analyze the probability that $\mathsf{BAD}_{i^*,1}$ occurs in $\mathcal{SE}_4$. Clearly, $\mathcal{SE}_4$ is identical to $\mathcal{SE}_3$ when neither EVENT_0 nor EVENT_1 occurs in the main thread. We therefore focus on the case that either EVENT_0 or EVENT_1 occurs in the main thread, and we

show that when $\mathsf{BAD}_{i^*,1}$ occurs in $\mathcal{SE}_3$, it also occurs in $\mathcal{SE}_4$. Let $\{\text{i-nm}_{b,1}\}_{b\in\{0,1\}}$ denote the inner NMC messages that are given in Round 1 of the left interaction, and let w_{td} and w'_{td} denote the witnesses that are used in Round 4 of the left interaction in the main and right look-ahead threads in $\mathcal{SE}_3$. Suppose $\mathsf{BAD}_{i^*,1}$ occurs in $\mathcal{SE}_3$. By the definition of $\mathsf{BAD}_{i^*,1}$, there exists $\beta \in \{0,1\}$ such that EVENT_β occurs both in the main and right look-ahead threads. Thus, w_{td} and w'_{td} can be parsed as $w_{\mathsf{td}} = (\beta, \text{i-dec})$ and $w'_{\mathsf{td}} = (\beta, \text{i-dec}')$ such that $\mathsf{NMC}.\mathsf{R}_{\mathsf{D},\mathsf{weak}}(\text{i-nm}_{\beta,1}, \text{i-dec}) = 1$ and $\mathsf{NMC}.\mathsf{R}_{\mathsf{D},\mathsf{weak}}(\text{i-nm}_{\beta,1}, \text{i-dec}') = 1$. Therefore, the unique decommitment candidate property of NMC implies $\text{i-dec} = \text{i-dec}'$, and thus, $w_{\mathsf{td}} = w'_{\mathsf{td}}$. Hence, using w_{td} instead of w'_{td} in the right look-ahead thread does not decrease the probability that $\mathsf{BAD}_{i^*,1}$ occurs. Therefore, $\mathsf{BAD}_{i^*,1}$ occurs in $\mathcal{SE}_4$ with probability at least $\varepsilon_4 := \varepsilon_3$.

Finally, we analyze the probability that $\mathsf{BAD}_{i^*,1}$ occurs in $\mathcal{SE}_5$. Since the expected number of the left look-ahead threads is 2, Markov's inequality implies that the number of the left look-ahead threads exceeds $2 \cdot 2/\varepsilon_4$ with probability at most $\varepsilon_4/2$. Then, since each left look-ahead thread is generated identically, it follows that even when only two left look-ahead threads are generated, $\mathsf{BAD}_{i^*,1}$ occurs with probability at least $(\varepsilon_4 - \varepsilon_4/2) \cdot (4/\varepsilon_4)^{-2} = 2 \cdot (\varepsilon_4/4)^3$. Therefore, $\mathsf{BAD}_{i^*,1}$ occurs in $\mathcal{SE}_5$ with probability at least $\varepsilon_5 := 2 \cdot (\varepsilon_4/4)^3$. □