Università degli Studi di Salerno

Dottorato di Ricerca in Informatica e Ingegneria dell'Informazione
Ciclo 30 – a.a 2016/2017

TESI DI DOTTORATO / PH.D. THESIS

Delayed-Input and Non-Malleable Cryptographic Protocols

LUISA SINISCALCHI

SUPERVISOR: **PROF. IVAN VISCONTI**

PHD PROGRAM DIRECTOR: **PROF. PASQUALE CHIACCHIO**

Dipartimento di Ingegneria dell'Informazione ed Elettrica
e Matematica Applicata
Dipartimento di Informatica

Contents

to my family.

2

Abstract

A major goal in the design of cryptographic protocols is to reduce the number of communication rounds. Since a cryptographic protocol usually consists of a composition and interplay of some subprotocols and cryptographic primitives, the natural approach to save rounds consists in playing all subprotocols in parallel. Unfortunately this approach often fails since a subprotocol in order to start could require as input the output of another subprotocol. In such cases the two subprotocols must be played sequentially therefore penalizing the overall round complexity.

In this thesis we provide *delayed-input* cryptographic protocols that can be played in parallel with other subprotocols even in the above scenario where the output of a subprotocol is required as input by the other subprotocol. We show the actual impact of our *delayed-input* cryptographic protocols by improving the round efficiency of various applications.

More precisely, this thesis includes the following results:

1. The first OR-composition technique for Σ-protocols that requires only one statement to be fixed when the protocol starts, while the other statement can be defined in the last round. Our OR-composition technique does not require computational assumptions.

2. The first efficient 4-round resettable witness indistinguishable argument of knowledge. We make use of subexponential hardness assumptions and of our OR-composition technique. Previous constructions required 5 rounds.

3. The first 4-round delayed-input (i.e., the theorem and the witness can be used just to compute the last round of the protocol) one-many (also many-many synchronous) non-malleable zero-knowledge (NMZK) argument of knowledge Π_{NMZK} from one-way functions.

4. The first 4-round (round optimal for black-box simulation) multi-party coin tossing protocol from one-to-one one-way functions. This construction makes use of Π_{NMZK}. Previous constructions required much strong computational assumptions.

5. The first 3-round concurrent non-malleable commitment scheme from subexponentially hard one-way permutations. The protocol is also delayed input and public coin.

Introduction

Reducing the round complexity of cryptographic protocols is an important task in Cryptography. Many results have been published in the past in order to show techniques that allow to save communication rounds. However, the minimal number of rounds required to realize several fundamental cryptographic primitives is still an open question.

In this work we investigate and improve the round complexity of some notorious cryptographic protocols.

We start from the well known witness-indistinguishable proof system proposed by Lapidot and Shamir [LS90] (LS) that is a powerful tool to improve the round efficiency of applications, since it allows to specify in the last round the statement that is being proved (this is a *delayed-input* property). Because of this special feature, many round-optimal protocols have been designed leveraging on LS. However these round-optimal protocols require expensive NP reductions due to the fact that LS works for the NP-completely language Hamiltonicity.

One of the goals of this thesis is to reduce the round complexity of cryptographic protocols while also trying to avoid NP reductions. We note that efficient witness-indistinguishable proof systems exist for many interesting languages through the OR-composition technique of Cramer, Damgård and Schoenmakers (CDS)[CDS94], that can be applied to all Σ-protocols. The CDS OR-composition technique has found countless applications as building block for designing efficient protocols. Unfortunately, the CDS OR-composition technique does not enjoy the delayed-input property and this limitation sometimes penalizes the round complexity of the higher-level protocols that make use of it.

In the first part of this work we show an unconditional OR-composition technique for Σ-protocols, that requires only one statement to be fixed when the protocol starts, while the other statement can be defined in the last round. This seemingly partial version of the delayed-input property is sufficient for many applications, since often one of the statements is fixed before the proof starts. Concretely, we show how our new OR-composition technique can directly improve the round complexity of two higher-level protocols obtaining: 1) an efficient perfect quasi-polynomial time simulatable argument system that improves the round complexity of a construction of Pass [Pas03] from four to three rounds; 2) an efficient resettable WI argument that improves the round complexity of a construction of Scafuro and Visconti [SV12] from five to four rounds.

Then, we move our target to round efficiency of non-malleable protocols. Indeed, the round complexity of commitment schemes secure against man-in-the-middle attacks has been the focus of extensive research for about 25 years. Very recently, Goyal, Pandey and Richelson [GPR16] showed that 3 rounds are sufficient for (one-left, one-right) non-malleable commitments, leaving open the question of obtaining a similar results w.r.t. concurrent man-in-the-middle adversaries.

In this work we solve the above open problem by showing how to transform any 3-round (one-left one-right) non-malleable commitment scheme (with some extractability property) into a 3-round concurrent non-malleable commitment scheme. Our transform makes use of complexity leveraging and can be instantiated with the construction of [GPR16] giving a 3-round concurrent non-malleable commitment scheme from one-way permutations (OWPs) secure w.r.t. subexponential-time adversaries. We also show one more candidate to instantiate our compiler, that relies on sub-exponential OWPs. In more details, we propose a different approach for 3-round one-one non-malleable commitments that can be instantiated with a limited form of non-malleability enjoyed by both a subprotocol of [GRRV14] and a subprotocol of [GPR16] (therefore we can instantiate our result in two completely different ways).

Our 3-round concurrent non-malleable commitment scheme can be used for 3-round arguments of knowledge and in turn for 3-round identification schemes secure against concurrent man-in-the-middle attacks.

After our work, Khurana [Khu17] presented a 3-round concurrent non-malleable commitment scheme under number-theoretic assumptions. Then a work of Khurana, Sahai [KS17] and a work of Lin, Pass, Soni [LPS17] obtained a 2-round concurrent non-malleable commitment scheme under stronger complexity-theoretic assumptions.

Next, keeping in mind that the delayed-input property is important to improve the round complexity of cryptographic protocols, we design a non-malleable zero-knowledge argument system that needs the statement and the witness only in the last round. Then, we show how to obtain a construction that is round optimal when security is prove through black-box simulation for multi-party coin tossing relying on our new tool. In more details we start from the following two results in the state-of-the art.

1. 4-round non-malleable zero knowledge (NMZK): Goyal et al. in [GRRV14] showed the first 4-round one-one NMZK argument from one-way functions (OWFs). Their construction requires the prover to know the instance and the witness already at the 2nd round.
2. 4-round multi-party coin tossing (MPCT): Garg et al. in [GMPP16] showed the first 4-round protocol for MPCT. Their result crucially relies on 3-round 3-robust parallel non-malleable commitments. So far there is no candidate construction for such a commitment scheme under standard polynomial-time hardness assumptions.

We improve the state-of-the art on NMZK and MPCT by presenting the following two results:

1. a *delayed-input* 4-round one-*many* NMZK argument Π_{NMZK} from OWFs; moreover Π_{NMZK} is also a *delayed-input* many-many *synchronous* NMZK argument.

2. a 4-round MPCT protocol Π_{MPCT} from one-to-one OWFs; Π_{MPCT} uses Π_{NMZK} as subprotocol and exploits the special properties (e.g., delayed input, many-many synchronous) of Π_{NMZK}.

Both Π_{NMZK} and Π_{MPCT} make use of a special proof of knowledge that offers additional security guarantees when played in parallel with other protocols. The new technique behind such a proof of knowledge is of independent interest.

The results described in this thesis have all been published in IACR conferences. In particular, our OR-composition appears in TCC 2016-A ([CPS$^+$16a]). The results on non-malleable commitments are published in CRYPTO 2016 and CRYPTO 2017 ([COSV16, COSV17b]). Finally, the 4-round one-many NMZK argument and his application appear in TCC 2017 ([COSV17a]).

Chapter 1

Definitions

1.1 Notation, Definitions and Tools

We denote the security parameter by λ and use "$|$" as concatenation operator (i.e., if a and b are two strings then by $a|b$ we denote the concatenation of a and b). For a finite set Q, $x \leftarrow Q$ denotes the algorithm that chooses x from Q with uniform distribution. Usually we use the abbreviation PPT that stays for probabilistic polynomial-time. We use $\mathsf{poly}(\cdot)$ to indicate a generic polynomial function of the input.

A *polynomial-time relation* Rel (or *polynomial relation*, in short) is a subset of $\{0,1\}^* \times \{0,1\}^*$ such that membership of (x, w) in Rel can be decided in time polynomial in $|x|$. For $(x, w) \in \mathsf{Rel}$, we call x the *instance* and w a *witness* for x. For a polynomial-time relation Rel, we define the $\mathcal{NP}$-language L_{Rel} as $L_{\mathsf{Rel}} = \{x | \exists w : (x, w) \in \mathsf{Rel}\}$. Analogously, unless otherwise specified, for an $\mathcal{NP}$-language L we denote by $\mathsf{Rel_L}$ the corresponding polynomial-time relation (that is, $\mathsf{Rel_L}$ is such that $L = L_{\mathsf{Rel_L}}$). Following [GMY06], we define $\hat{L}_{\mathsf{Rel}}$ to be the input language that includes both L_{Rel} and all well formed instances that do not have a witness. More formally, $L_{\mathsf{Rel}} \subseteq \hat{L}_{\mathsf{Rel}}$ and membership in $\hat{L}_{\mathsf{Rel}}$ can be tested in polynomial time. We implicitly assume that the verifier of a protocol for relation Rel executes the protocol only if the common input x belongs to $\hat{L}_{\mathsf{Rel}}$ and rejects immediately common inputs not in $\hat{L}_{\mathsf{Rel}}$.

Let A and B be two interactive probabilistic algorithms A and B. We denote by $\langle A(\alpha), B(\beta)\rangle(\gamma)$ the distribution of B's output after running on private input β with A using private input α, both running on common input γ. Typically, one of the two algorithms receives 1^λ as input. A *transcript* of $\langle A(\alpha), B(\beta)\rangle(\gamma)$ consists of the messages exchanged during an execution where A receives a private input α, B receives a private input β and both A and B receive a common input γ. Moreover, we will refer to the *view* of A as the messages it received during the execution of $\langle A(\alpha), B(\beta)\rangle(\gamma)$, along with its randomness and its input. We denote by A_r an algorithm A that receives as randomness r. We say that a protocol (A, B) is public coin if B sends to A random bits only.

A function $\nu(\cdot)$ from non-negative integers to reals is called negligible, if for every constant $c > 0$ and all sufficiently large $\lambda \in \mathbb{N}$ we have $\nu(\lambda) < \lambda^{-c}$.

1.1.1 Standard Definitions

Definition 1.1.1 (One-way function (OWF)). *A function $f : \{0,1\}^\star \rightarrow \{0,1\}^\star$ is called one way if the following two conditions hold:*

- *there exist a deterministic polynomial-time algorithm that on input y in the domain of f outputs $f(y)$;*
- *for every* PPT *algorithm $\mathcal{A}$ there exists a negligible function ν, such that for every auxiliary input $z \in \{0,1\}^{\mathsf{poly}(\lambda)}$:*

$$\text{Prob}\left[\, y \leftarrow \{0,1\}^\star : \mathcal{A}(f(y), z) \in f^{-1}(f(y)) \,\right] < \nu(\lambda).$$

We say that a OWF f is a one-way permutation (OWP) *if f is a permutation.*

We will require that an algorithm that runs in time $\tilde{T} = 2^{\lambda^\alpha}$ for some positive constant $\alpha < 1$, can invert a OWP f. In this case we say that f is $\tilde{T}$-breakable.

Definition 1.1.2 (Computational indistinguishability)**.** *Let* $X = \{X_\lambda\}_{\lambda\in\mathbb{N}}$ *and* $Y = \{Y_\lambda\}_{\lambda\in\mathbb{N}}$ *be ensembles, where* X_λ*'s and* Y_λ*'s are probability distribution over* $\{0,1\}^l$*, for same* $l = \mathsf{poly}(\lambda)$*. We say that* $X = \{X_\lambda\}_{\lambda\in\mathbb{N}}$ *and* $Y = \{Y_\lambda\}_{\lambda\in\mathbb{N}}$ *are* computationally indistinguishable*, denoted* $X \approx Y$*, if for every* PPT *distinguisher* $\mathcal{D}$ *there exist a negligible function* ν *such that for sufficiently large* $\lambda \in \mathbb{N}$*,*

$$\Big|\text{Prob}\left[\, t \leftarrow X_\lambda : \mathcal{D}(1^\lambda, t) = 1 \,\right] - \text{Prob}\left[\, t \leftarrow Y_\lambda : \mathcal{D}(1^\lambda, t) = 1 \,\right]\Big| < \nu(\lambda).$$

We note that in the usual case where $|X_\lambda| = \Omega(\lambda)$ and λ can be derived from a sample of X_λ, it is possible to omit the auxiliary input 1^λ. In this work we also use the definition of *Statistical Indistinguishability.* This definition is the same as Definition 1.1.2 with the only difference that the distinguisher $\mathcal{D}$ is unbounded. In this case use $X \equiv_s Y$ to denote that two ensembles are statistically indistinguishable.

We note that in the usual case where $|X_\lambda| = \Omega(\lambda)$ and the length λ can be derived from a sample of X_λ, it is possible to omit the auxiliary input 1^λ.

Definition 1.1.3 (Delayed-input proof/argument system)**.** *A pair of* PPT *interactive algorithms* $\Pi = (\mathcal{P}, \mathcal{V})$ *constitutes a* proof system *(resp., an* argument system*) for an* $\mathcal{NP}$*-language* L*, if the following conditions hold:*

Completeness: *For every* $x \in L$ *and* w *such that* $(x, w) \in \mathsf{Rel_L}$*, it holds that:*

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

Soundness: *For every interactive (resp.,* PPT *interactive) algorithm* $\mathcal{P}^\star$*, there exists a negligible function* ν *such that for every* $x \notin L$ *and every* z*:*

$$\text{Prob}\left[\, \langle \mathcal{P}^\star(z), \mathcal{V}\rangle(x) = 1 \,\right] < \nu(|x|).$$

A proof/argument system $\Pi = (\mathcal{P}, \mathcal{V})$ *for an* $\mathcal{NP}$*-language* L*, enjoys* delayed-input *completeness if* $\mathcal{P}$ *needs* x *and* w *only to compute the last round of* Π *and* $\mathcal{V}$ *needs* x *only to compute the output. Before that,* $\mathcal{P}$ *and* $\mathcal{V}$ *run having as input only the size of* x*.*

The notion of delayed-input completeness was defined in [CPS+16a]. We say that the transcript τ of an execution of $(\mathcal{P}, \mathcal{V})$ is *accepting* if $\mathcal{V}$ outputs 1. An interactive protocol $\Pi = (\mathcal{P}, \mathcal{V})$ is *public coin* if, at every round, $\mathcal{V}$ at each round simply tosses a predetermined number of coins (random challenge) and sends them to $\mathcal{P}$.

Definition 1.1.4 (Witness Indistinguishable (WI))**.** *An argument/proof system* $\Pi = (\mathcal{P}, \mathcal{V})$*, is* Witness Indistinguishable (WI) *for a relation* Rel *if, for every malicious* PPT *verifier* $\mathcal{V}^\star$*, there exists a negligible function* ν *such that for all* x, w, w' *such that* $(x, w) \in \mathsf{Rel}$ *and* $(x, w') \in \mathsf{Rel}$ *it holds that:*

$$\left|\text{Prob}\left[\ \langle \mathcal{P}(w), \mathcal{V}^\star\rangle(x) = 1\ \right] - \text{Prob}\left[\ \langle \mathcal{P}(w'), \mathcal{V}^\star\rangle(x) = 1\ \right]\right| < \nu(|x|).$$

The notion of a *perfect* WI proof system is obtained by requiring $\nu(|x|) = 0$.

Obviously one can generalize the above definitions of WI to their natural adaptive-input variants, where the adversarial verifier can select the statement and the witnesses adaptively, before the prover plays the last round.

In this thesis we also consider a definition where the WI property of LS still holds against a distinguisher with running time bounded by $T = 2^{\lambda^\alpha}$ for some constant positive constant $\alpha < 1$. In this case we say that the instantiation of LS is T-witness indistinguishable (T-WI).

Definition 1.1.5 (Proof of Knowledge [LP11b])**.** *A proof system* $\Pi = (\mathcal{P}, \mathcal{V})$ *is a* proof of knowledge *(PoK) for the relation* Rel_L *if there exist a probabilistic expected polynomial-time machine* E*, called the extractor, such that for every algorithm* $\mathcal{P}^\star$*, there exists a negligible function* $\nu(\lambda)$*, every statement* $x \in \{0,1\}^\lambda$*, every randomness* $r \in \{0,1\}^\star$ *and every auxiliary input* $z \in \{0,1\}^\star$*,*

$$\text{Prob}\left[\ \langle \mathcal{P}_r^\star(z), \mathcal{V}\rangle(x) = 1\ \right] \leq \text{Prob}\left[\ w \leftarrow \mathsf{E}^{\mathcal{P}_r^\star(z)}(x) : (x, w) \in \mathsf{Rel_L}\ \right] + \nu(\lambda).$$

We also say that an argument system Π *is a* argument of knowledge *(AoK) if the above condition holds w.r.t. any* PPT $\mathcal{P}^\star$*.*

In this work we also consider the *adaptive-PoK* property. The adaptive-PoK property ensures that the PoK property still holds when a malicious prover can choose the statement adaptively at the last round. In this case, to be consistent with Definition 1.1.5 where the extractor algorithm E takes as input the statement proved by $\mathcal{P}^\star$, we have to consider a different extractor algorithm. This extractor algorithm takes as input the randomness r of $\mathcal{P}$, the randomness r' of $\mathcal{V}$ and outputs the witness for $x \in L$, where x is selected by $\mathcal{P}_r^\star$ when interacting with $\mathcal{V}_{r'}$.

In this work we use the 3-round public-coin WI Proof of Knowledge (WIPoK) proposed by Lapidot and Shamir [LS90] (LS protocol), that relays on OWPs. LS enjoys delayed-input completeness since the inputs for both $\mathcal{P}$ and $\mathcal{V}$ are needed only to play the last round, and only the length of the instance is needed earlier. The LS protocol is also sound when a malicious prover can choose the statement adaptively at the third round. We refer to this property as adaptive soundness. LS also enjoys the property of adaptive PoK and adaptive WI. We also use a variant of LS that relies on OWFs only. The additional round is indeed needed to instantiate the commitment scheme used in LS under any OWF. The reader can find more details in Section 4.4.

1.1.2 Number-Theoretic Assumptions

We define *group generator* algorithms to be probabilistic polynomial-time algorithms that take as input security parameter 1^λ and output $(\mathcal{G}, q, g)$, where $\mathcal{G}$ is (the description of) a cyclic group of order q and g is a generator of $\mathcal{G}$. We assume that membership in $\mathcal{G}$ and its group operations can be performed in time polynomial in the length of q and that there is an efficient procedure to randomly select elements from $\mathcal{G}$. Moreover, with a slight abuse of notation, we will use $\mathcal{G}$ to denote the group and its description.

We consider the sub-exponential versions of the DLog and of the DDH assumptions that posit the hardness of the computation of discrete logarithms and of breaking the Decisional Diffie-Hellman assumption with respect to the group generator algorithm IG that, on input λ, randomly selects a λ-bit prime q such that $p = 2q + 1$ is also prime and outputs the order q group $\mathcal{G}$ of the quadratic residues modulo p along with a random generator g of $\mathcal{G}$. The strong versions of the two assumptions posit the hardness of the same problems even if p (and q) and generator g are chosen adversarially. More precisely:

Assumption 1 (DLog Assumption)*.* There exists a constant α such that for every probabilistic algorithm A running in time 2^{λ^α} the following probability is a negligible function of λ

$$\text{Prob}\left[\ (\mathcal{G}, q, g) \leftarrow \mathsf{IG}(1^\lambda); y \leftarrow \mathbb{Z}_q : A(g^y) = y\ \right].$$

Assumption 2 (Strong DLog Assumption [CD08])*.* Consider a pair of probabilistic algorithms (A_0, A_1) such that A_0, on input 1^λ, outputs $(\mathcal{G}, q, g)$, where $\mathcal{G}$ is the group of the quadratic residues modulo p, where p is prime, $p = 2q + 1$, q is a λ-bit prime and $g \in \mathcal{G}$, along with some auxiliary information $\mathtt{aux}$. There exists a constant α such that for any such pair (A_0, A_1) running in time 2^{λ^α} the following probability is a negligible function of λ:

$$\text{Prob}\left[\ ((\mathcal{G}, q, g), \mathtt{aux}) \leftarrow A_0(1^\lambda); y \leftarrow \mathbb{Z}_q : A_1(g^y, \mathtt{aux}) = y\ \right].$$

We next introduce the DDH Assumption and the Strong DDH Assumption which imply the DLog Assumption and the Strong DLog Assumption, respectively.

Assumption 3 (DDH Assumption)*.* There exists a constant α such that, for every probabilistic algorithm A running in time 2^{λ^α}, the following is a negligible function of λ

$$\Big|\text{Prob}\left[\ (\mathcal{G}, q, g) \leftarrow \mathsf{IG}(1^\lambda); x, y, z \leftarrow \mathbb{Z}_q : A((\mathcal{G}, q, g), g^x, g^y, g^z) = 1\ \right] - \text{Prob}\left[\ (\mathcal{G}, q, g) \leftarrow \mathsf{IG}(1^\lambda); x, y, z \leftarrow \mathbb{Z}_q : A((\mathcal{G}, q, g), g^x, g^y, g^{xy}) = 1\ \right]\Big|.$$

Assumption 4 (Strong DDH Assumption)*.* Consider a pair of probabilistic algorithms (A_0, A_1) such that A_0, on input 1^λ, outputs $(\mathcal{G}, q, g)$, where $\mathcal{G}$ is the group of the quadratic residues modulo p, where p is prime, $p = 2q + 1$, q is a λ-bit prime and $g \in \mathcal{G}$, along with some auxiliary information $\texttt{aux}$. There exists a constant α such that, for any such pair (A_0, A_1) running in time 2^{λ^α}, the following is a negligible function of λ

$$\begin{aligned}\Big|\text{Prob}\ \big[\{((\mathcal{G}, q, g), \texttt{aux}) \leftarrow A_0(1^\lambda); x, y, z \leftarrow \mathbb{Z}_q :\\ A_1((\mathcal{G}, q, g), g^x, g^y, g^z, \texttt{aux}) = 1\big] -\\ \text{Prob}\ \big[((\mathcal{G}, q, g), \texttt{aux}) \leftarrow A_0(1^\lambda); x, y, z \leftarrow \mathbb{Z}_q :\\ A_1((\mathcal{G}, q, g), g^x, g^y, g^{xy}, \texttt{aux}) = 1\big]\Big|\end{aligned}$$

Following [HKR$^+$14] and [CPS$^+$16b] we also use the following DDH variant.

Assumption 5 (1DDH)*.* There exists a constant α such that, for every probabilistic algorithm A running in time 2^{λ^α}, the following is a negligible function of λ

$$\begin{aligned}\Big|\text{Prob}\left[\ (\mathcal{G}, q, g) \leftarrow \mathsf{IG}(1^\lambda); \alpha, \beta \leftarrow \mathbb{Z}_q : \mathcal{A}((\mathcal{G}, q, g), g^x, g^y, g^{xy+1}) = 1\ \right] -\\ \text{Prob}\left[\ (\mathcal{G}, q, g) \leftarrow \mathsf{IG}(1^\lambda); \alpha, \beta \leftarrow \mathbb{Z}_q : \mathcal{A}((\mathcal{G}, q, g), g^x, g^y, g^{xy}) = 1\ \right]\Big|.\end{aligned}$$

In the rest of the work we will refer to a tuple $T = \big((\mathcal{G}, q, g), A = g^x, B = g^y, C = g^{xy}\big)$ as a DH tuple, to $T = \big((\mathcal{G}, q, g), A = g^x, B = g^y, C = g^{xy+1}\big)$ as a 1-non-DH tuple and $T = \big((\mathcal{G}, q, g), A = g^x, B = g^y, C = g^z\big)$ as a non-DH tuple.

Lemma 1.1.6. *Assumption 5 is implied by the DDH assumption.*

Proof. It is easy to see that under the DDH assumption, randomly selected 1-non-DH tuples are indistinguishable from randomly selected non-DH tuples. Indeed, let $T = ((\mathcal{G}, q, g), A, B, C)$ be any tuple and consider tuple $T' = ((\mathcal{G}, q, g), A, B, C \cdot g)$. Then we have that if T is a randomly selected DH tuple, then T' is a randomly selected 1-non-DH tuple; whereas, if T is a randomly selected non-DH tuple then T' is statistically close to a randomly selected non-DH tuple. Moreover, by transitivity, we have that, under the DDH assumption, randomly selected 1-non-DH tuples are indistinguishable from randomly selected DH tuples. □

1.1.3 Σ-Protocols

We consider *3-move protocols* Π for a polynomial-time relation Rel. Protocol Π is played by a prover $\mathcal{P}$ and a verifier $\mathcal{V}$ that receive a common input x. $\mathcal{P}$ receives as an additional private input a witness w for x and the security parameter 1^λ in unary. The protocol Π has the following form:

1. $\mathcal{P}$ executes algorithm P_1 on common input x, private input w, security parameter 1^λ and randomness R obtaining $a = P_1(x, w, 1^\lambda; R)$ and sends a to $\mathcal{V}$.

2. $\mathcal{V}$, after receiving a from $\mathcal{P}$, chooses a random *challenge* $c \leftarrow \{0,1\}^l$ and sends c to $\mathcal{P}$.

3. $\mathcal{P}$ executes algorithm P_2 on input x, w, R, c and sends $z \leftarrow P_2(x, w, R, c)$ to $\mathcal{V}$.

4. $\mathcal{V}$ executes and outputs $\mathsf{V}(x, a, c, z)$ (i.e., $\mathcal{V}$'s decision to accept ($b = 1$) or reject ($b = 0$)).

We call (P_1, P_2, V) the algorithms *associated* with Π and l the challenge length such that, wlog, the challenge space $\{0,1\}^l$ is composed of 2^l different challenges.

The triple (a, c, z) of messages exchanged is called a *3-move transcript.* A 3-move transcript is *honest* if a, z correspond to the messages computed running the honest algorithms, respectively, of P_1 and P_2, and c is a random string, in $\{0,1\}^l$. A 3-move transcript (a, c, z) is *accepting* for x if and only if $\mathsf{V}(x, a, c, z) = 1$. Two accepting 3-move transcripts (a, c, z) and (a', c', z') for an instance x constitute a *collision* if $a = a'$ and $c \neq c'$.

Definition 1.1.7 (Σ-protocol [CDS94])**.** *A 3-move protocol* Π *with challenge length* l *is a* Σ-protocol *for a relation* Rel *if it enjoys the following properties:*

1. ***Completeness****. If $(x, w) \in \mathsf{Rel}$ then all honest 3-move transcripts for (x, w) are accepting.*

2. ***Special Soundness****. There exists an efficient algorithm* Extract *that, on input x and a collision for x, outputs a witness w such that $(x, w) \in \mathsf{Rel}$.*

3. ***Special Honest-Verifier Zero Knowledge (SHVZK)****. There exists a PPT simulator algorithm* Sim *that takes as input $x \in L_{\mathsf{Rel}}$, security parameter 1^λ and $c \in \{0, 1\}^l$ and outputs an accepting transcript (a, c, z) for x where c is the challenge. Moreover, for all l-bit strings c, the distribution of the output of the simulator on input (x, c) is computationally indistinguishable from the distribution of the 3-move honest transcript obtained when $\mathcal{V}$ sends c as challenge and $\mathcal{P}$ runs on common input x and any private input w such that $(x, w) \in \mathsf{Rel}$.*

   We say that Π *is* Perfect *when the two distributions are identical.*

Not to overburden the descriptions of protocols and simulators, we will omit the specification of the security parameter when it is clear from the context.

In the rest of the thesis, we will call a 3-move protocol that enjoys Completeness, Special Soundness and Honest-Verifier Zero Knowledge (HVZK[1]) a $\tilde{\Sigma}$-*protocol.* The next theorem shows that SHVZK can be added to a 3-move protocol with HVZK without any significant penalty in terms of efficiency.

Theorem 1 ([Dam10]). Suppose relation Rel admits a 3-move protocol Π' that is HVZK (resp., perfect HVZK). Then Rel admits a 3-move protocol Π that is SHVZK (resp., perfect SHVZK) and has the same efficiency.

[1]Recall that HVZK requires the existence of a simulator that generates a full transcript. This is a seemingly weaker requirement than SHVZK where the challenge is an input for the simulator.

Proof. Let l be the challenge length of Π', let $(P'_1, P'_2, \mathsf{V}')$ be the algorithms associated with Π' and let Sim' be the simulator for Π'. Consider the following algorithms.

1. P_1, on input $(x, w) \in \mathsf{Rel}$, security parameter 1^λ and randomness R_1, parses R_1 as (r_1, c'') where $|c''| = l$, computes $a' \leftarrow P'_1(x, w, 1^\lambda; r_1)$, and outputs $a = (a', c'')$.

2. P_2, on input $(x, w) \in \mathsf{Rel}$, R_1 and randomness R_2 parses R_1 as (r_1, c''), c, sets $c' = c \oplus c''$, computes $z' \leftarrow P'_2(x, w, r_1, c'; R_2)$, and sends it to $\mathcal{V}$.

3. V, on input x, $a = (a', c'')$, c and z', returns the output of $\mathsf{V}'(x, a', c \oplus c'', z')$ to decide whether to accept or not.

Consider the following PPT simulator Sim that, on input an instance x and a challenge c, runs Sim' on input x and obtains (a', c', z'). Then Sim sets $c'' = c \oplus c'$ and $a = (a', c'')$ and outputs (a, c, z'). It is easy to see that if Sim' is a HVZK (resp. perfect HVZK) simulator for Π' then Sim is a SHVZK (resp. perfect SHVZK) simulator for Π. □

Theorem 2. ([Dam10]) Let $\Pi = (\mathcal{P}, \mathcal{V})$ be a Σ-protocol for relation Rel_L with negligible soundness error[2], then Π is a proof of knowledge for Rel_L.

Definition 1.1.8 (Delayed-input Σ-protocol)**.** *A Σ-protocol $\Pi = (\mathcal{P}, \mathcal{V})$ with $\mathcal{P}$ running PPT algorithms (P_1, P_2) is an delayed-input Σ-protocol if P_1 takes as input only the length of the common instance and P_2 takes as input the common instance x, the witness w, the randomness R_1 used by P_1 and the challenge c received from the verifier.*

Definition 1.1.9 (Delayed-witness Σ-protocol)**.** *A Σ-protocol $\Pi = (\mathcal{P}, \mathcal{V})$ for a relation* Rel *with associated algorithms (P_1, P_2, V) is a delayed-witness Σ-protocol if P_1 takes as input only the common instance x.*

[2]The soundness error represents the probability of a malicious prover to convince the verifier of a false statement.

In a *Chameleon Σ-protocol*, the prover can compute the first message by using the simulator and thus knowing only the input but not the witness. Once the challenge has been received, the prover can compute the last message (thus completing the interaction) by using the witness w (which is thus used only to compute the last message) and the coin tosses used by the simulator to compute the first message.

Definition 1.1.10 (Chameleon Σ-protocol). *A Σ-protocol Π for polynomial-time relation Rel is a* Chameleon *Σ-protocol if there exists an SHVZK simulator Sim and an algorithm $\mathsf{P_{sim}}$ satisfying the following property:*

Delayed Indistinguishability*: for all pairs of challenges c_0 and c_1 and for all $(x, w) \in \mathsf{Rel}$, the following two distributions $\{R \leftarrow \{0,1\}^{|x|^d}; (a, z_0) \leftarrow \mathsf{Sim}(x, c_0; R); z_1 \leftarrow \mathsf{P_{sim}}((x, c_0, R), w, c_1) : (x, a, c_1, z_1)\}$ and $\{(a, z_1) \leftarrow \mathsf{Sim}(x, c_1) : (x, a, c_1, z_1)\}$ are indistinguishable, where Sim is the SHVZK simulator and d is such that Sim, on input an λ-bit instance, uses at most λ^d random coin tosses. If the two distributions above are identical then we say that delayed indistinguishability is perfect, and Π is a* Perfect *Chameleon Σ-protocol.*

We remark that a Chameleon Σ-protocol Π has two modes of operations: the standard mode when $\mathcal{P}$ runs P_1 and P_2, and a *delayed* mode when $\mathcal{P}$ uses Sim and $\mathsf{P_{sim}}$. Moreover, observe that since Sim is a simulator for Π, it follows from the delayed-indistinguishability property that, for all challenges c and $\tilde{c}$ and common inputs x, distribution

$$\{R \leftarrow \{0,1\}^{|x|^d}; (a, \tilde{z}) \leftarrow \mathsf{Sim}(x, \tilde{c}; R); z \leftarrow \mathsf{P_{sim}}((x, \tilde{c}, R), w, c) : (a, c, z)\}$$

is indistinguishable from

$$\{R \leftarrow \{0,1\}^{|x|^d}; a \leftarrow P_1(x, w; R); z \leftarrow P_2(x, w, R, c) : (a, c, z)\}.$$

That is, the two modes of operations of Π are indistinguishable. This property make us able to claim that if Π is WI when a WI challenger interacts with an adversary using (P_1, P_2), then Π is WI even when the pair $(\mathsf{Sim}, \mathsf{P_{sim}})$ is used. Finally, we observe that Chameleon Σ-protocols do exist and Schnorr's protocol [Sch89] is one example. When considering the algorithms associated to a Chameleon Σ-protocol, we will add $\mathsf{P_{sim}}$.

Theorem 3 ([CDS94]). Every Perfect $\tilde{\Sigma}$-protocol[3] is Perfect WI.

In this work we are studying delayed-input protocols, therefore we also consider the SHVZK property and the special-soundness property that hold when a player has the freedom of specify the theorem and the witness in the last round of the protocol.

Definition 1.1.11. *A delayed-input 3-round protocol* $\Pi = (\mathcal{P}, \mathcal{V})$ *for relation* $\mathsf{Rel_L}$ *enjoys* adaptive-input special soundness *if there exists a polynomial time algorithm such that, for any pair of accepting transcripts* $(\mathsf{a}, \mathsf{c}_1, \mathsf{z}_1)$ *for input* x_1 *and* $(\mathsf{a}, \mathsf{c}_2, \mathsf{z}_2)$ *for input* x_2 *with* $\mathsf{c}_1 \neq \mathsf{c}_2$*, outputs witnesses* w_1 *and* w_2 *such that* $(x_1, w_1) \in \mathsf{Rel_L}$ *and* $(x_2, w_2) \in \mathsf{Rel_L}$.

Definition 1.1.12. *A delayed-input 3-round protocol* $\Pi = (\mathcal{P}, \mathcal{V})$ *for relation* $\mathsf{Rel_L}$ *enjoys* adaptive-input Special Honest Verifier Zero-knowledge (adaptive-input Special HVZK) *if there exists a two phases* PPT *simulator algorithm* Sim *that works as follow:*

1. $\mathsf{a} \leftarrow \mathsf{Sim}(1^\lambda, \mathsf{c}, \kappa; \rho)$*, where* 1^λ *is the security parameter,* c *is the challenge* κ *is the size of the instance to be proved and the randomness* ρ*;*
2. $\mathsf{z} \leftarrow \mathsf{Sim}(x, \rho)$[4]*, where* x *is the instance to be proved.*

Π *is adaptive-input Special HVZK if any* $x \in L$ *and for any* $\mathsf{c} \in \{0,1\}^\lambda$*, the distribution of the transcripts* $(\mathsf{a}, \mathsf{c}, \mathsf{z})$*, computed by* Sim*, is computationally indistinguishable from the distribution of a transcript obtained when* $\mathcal{V}$ *sends* c *as challenge and* $\mathcal{P}$ *runs on common input* x *and any* w *(available only in the third round) such that* $(x, w) \in \mathsf{Rel_L}$.

[3]We remind the reader that we call a 3-move protocol that enjoys Completeness, Special Soundness and Honest-Verifier Zero Knowledge (HVZK) a $\tilde{\Sigma}$-protocol.

[4]To not overburden the notation we omit the randomness when we use the adaptive-input Special HVZK[5] simulator

1.2 Commitment Schemes

Definition 1.2.1 (Commitment Scheme)**.** *Given a security parameter 1^λ, a commitment scheme* $(\mathsf{Sen}, \mathsf{Rec})$ *is a two-phase protocol between two* PPT *interactive algorithms, a sender* Sen *and a receiver* Rec*. In the commitment phase* Sen *on input a message m interacts with* Rec *to produce a commitment* $\mathtt{com}$*. In the decommitment phase,* Sen *sends to* Rec *a decommitment information* $\mathtt{d}$ *such that* Rec *accepts m as the commitment of* $\mathtt{com}$*.*

Formally, we say that $\mathtt{CS} = (\mathsf{Sen}, \mathsf{Rec})$ *is a perfectly binding commitment scheme if the following properties hold:*

Correctness:

- *Commitment phase. Let* $\mathtt{com}$ *be the commitment of the message m (i.e.,* $\mathtt{com}$ *is the transcript of an execution of* $\mathtt{CS} = (\mathsf{Sen}, \mathsf{Rec})$ *where* Sen *runs on input a message m). Let* $\mathtt{d}$ *be the private output of* Sen *in this phase.*
- *Decommitment phase[6].* Rec *on input m and* $\mathtt{d}$ *accepts m as decommitment of* $\mathtt{com}$*.*

Hiding: *for a* PPT *adversary $\mathcal{A}$ and a randomly chosen bit $b \in \{0,1\}$, consider the following hiding experiment* $\mathsf{ExpHiding}^b_{\mathcal{A},\mathtt{CS}}(\lambda)$*:*

- *Upon input 1^λ, the adversary $\mathcal{A}$ outputs a pair of messages m_0, m_1 that are of the same length.*
- Sen *on input the message m_b interacts with $\mathcal{A}$ to produce a commitment of m_b.*
- *$\mathcal{A}$ outputs a bit b' and this is the output of the experiment.*

For any PPT *adversary $\mathcal{A}$, there exist a negligible function ν, such that:*

$$\Big|\mathrm{Prob}\Big[\ \mathsf{ExpHiding}^0_{\mathcal{A},\mathtt{CS}}(\lambda) = 1\ \Big] - \mathrm{Prob}\Big[\ \mathsf{ExpHiding}^1_{\mathcal{A},\mathtt{CS}}(\lambda) = 1\ \Big]\Big| < \nu(\lambda).$$

[6] In this work we consider a non-interactive decommitment phase only.

Binding: *for every commitment* `com` *generated during the commitment phase by a possibly malicious unbounded sender* $\mathsf{Sen}^\star$ *interacting with an honest receiver* Rec*, there exists at most one message* m *that* Rec *accepts as decommitment of* `com`*.*

We also consider the definition of a commitment scheme where the hiding property still holds against an adversary $\mathcal{A}$ running in time bounded by $T = 2^{\lambda^\alpha}$ for some positive constant $\alpha < 1$. In this case we will say that a commitment scheme is T-hiding. We will also say that a commitment scheme is $\tilde{T}$-breakable to specify that an algorithm running in time $\tilde{T} = 2^{\lambda^\beta}$, for some positive constant $\beta < 1$, recovers the (if any) only message that can be successfully decommitment.

In the rest of the work we also use a non-interactive commitment schemes, with secure parameter λ. In this case we consider a commitment scheme as a pair of PPT algorithms $(\mathsf{NISen}, \mathsf{NIRec})$ where:

- NISen takes as input $(m; \sigma)$, where $m \in \{0,1\}^{\mathsf{poly}(\lambda)}$ is the message to be committed and $\sigma \leftarrow \{0,1\}^\lambda$ is randomness, and outputs the commitment `com` and the decommitment `dec`;
- NIRec takes as input (`dec`, `com`, m) and outputs 1 if it accepts m as a decommitment of `com` and 0 otherwise.

3-Round extractable commitment schemes. Informally, a 3-round commitment scheme is extractable if there exists an efficient extractor that having black-box access to any efficient malicious sender $\mathrm{ExSen}^\star$ that successfully performs the commitment phase, outputs the only committed string that can be successfully decommitted.

Definition 1.2.2 (Extractable Commitment Scheme [GLOV12])**.** *A 3-round perfectly (resp. statistically) binding commitment scheme* `ExCS` $= (\mathrm{ExSen}, \mathrm{ExRec})$ *is an* extractable commitment scheme *if given oracle access to any malicious sender* $\mathrm{ExSen}^\star$*, there exists an expected* PPT *extractor* `ExtCom` *that outputs a pair* $(\tau, \sigma^\star)$ *such that the following properties hold:*

- **Simulatability:** *τ is identically distributed to the view of* ExSen* *(when interacting with an honest* ExRec*) in the commitment phase.*
- **Extractability:** *the probability that there exists a decommitment of τ to a message m', where $m \neq m'$ is 0 (resp. negligible).*

If the definition holds only against honest sender, then we said that the commitment scheme is *honest extractable.*

1.2.1 t-Instance-Dependent Trapdoor Commitment Schemes

In this section, for integer $t \geq 2$, we define the notion of a t-Instance-Dependent Trapdoor Commitment scheme associated with a polynomial-time relation Rel.

Definition 1.2.3 (t-Instance-Dependent Trapdoor Commitment scheme). *Let $t \geq 2$ be an integer and let Rel be a polynomial-time relation. A t-*Instance-Dependent Trapdoor Commitment *(a t-IDTC, in short) scheme for Rel with message space M is a triple of PPT algorithms* $(\mathsf{TCom}, \mathsf{TDec}, \mathsf{TFake})$ *where TCom is the randomized* commitment *algorithm that takes as input security parameter 1^λ, an instance $x \in \hat{L}_{\mathsf{Rel}}$ (with $|x| = \mathsf{poly}(\lambda)$) and a message $m \in M$ and outputs* commitment $\mathtt{com}$, decommitment $\mathtt{dec}$, *and* auxiliary information $\mathtt{rand}$*; TDec is the* verification *algorithm that takes as input $(x, \mathtt{com}, \mathtt{dec}, m)$ and decides whether m is the decommitment of $\mathtt{com}$; TFake is the randomized* equivocation *algorithm that takes as input $(x, w) \in \mathsf{Rel}$, messages m_1 and m_2 in M, commitment $\mathtt{com}$ of m_1 with respect to instance x and associated auxiliary information $\mathtt{rand}$ and produces decommitment information $\mathtt{dec}_2$ such that TDec, on input $(x, \mathtt{com}, \mathtt{dec}_2, m_2)$, outputs 1.*

A t-Instance-Dependent Trapdoor Commitment scheme has the following properties:

- **Correctness***: for all $x \in \hat{L}_{\mathsf{Rel}}$, all $m \in M$, it holds that*

$$\text{Prob}\left[(\mathtt{com}, \mathtt{dec}, \mathtt{rand}) \leftarrow \mathsf{TCom}(1^\lambda, x, m) : \mathsf{TDec}(x, \mathtt{com}, \mathtt{dec}, m) = 1\right] = 1.$$

- t-**Special Extract**: *there exists an efficient algorithm* $\mathsf{ExtractTCom}$ *that, on input x, commitment* $\mathtt{com}$*, pairs* $(\mathtt{dec}_i, m_i)_{i=1}^t$ *of openings and messages such that*
    - *for* $1 \le i < j \le t$ *we have that* $m_i \neq m_j$*;*
    - $\mathsf{TDec}(x, \mathtt{com}, \mathtt{dec}_i, m_i) = 1$*, for* $i = 1, \ldots, t$*;*

  outputs w such that $(x, w) \in \mathsf{Rel}$.

- **Hiding (resp., Perfect Hiding)**: *for every PPT (resp., unbounded) adversary $\mathcal{A}$ there exists a negligible function ν (resp., $\nu(\cdot) = 0$) such that, for all $x \in L_{\mathsf{Rel}}$ and all $m_0, m_1 \in M$, it holds that*

$$\mathrm{Prob}\big[b \leftarrow \{0,1\}; (\mathtt{com}, \mathtt{dec}, \mathtt{rand}) \leftarrow \mathsf{TCom}(1^\lambda, x, m_b) : \\ b = \mathcal{A}(x, \mathtt{com}, m_0, m_1)\big] \le \frac{1}{2} + \nu(\lambda).$$

- **Trapdoorness**: *the following two families of probability distributions are indistinguishable:*

$$\{(\mathtt{com}, \mathtt{dec}_1, \mathtt{rand}) \leftarrow \mathsf{TCom}(1^\lambda, x, m_1); \\ \mathtt{dec}_2 \leftarrow \mathsf{TFake}(x, w, m_1, m_2, \mathtt{com}, \mathtt{rand}) : (\mathtt{com}, \mathtt{dec}_2)\}$$

  and

$$\{(\mathtt{com}, \mathtt{dec}_2, \mathtt{rand}) \leftarrow \mathsf{TCom}(1^\lambda, x, m_2) : (\mathtt{com}, \mathtt{dec}_2)\}$$

  over all families $\{(x, w, m_1, m_2)\}$ *such that* $(x, w) \in \mathsf{Rel}$ *and* $m_1, m_2 \in M$.

  The perfect trapdoorness *property requires the two probability distributions to coincide for all* (x, w, m_1, m_2) *such that* $(x, w) \in \mathsf{Rel}$ *and* $m_1, m_2 \in M$.

Definition 1.2.4 (WI t-Instance-Dependent Trapdoor Commitment scheme). *A t-Instance-Dependent Trapdoor Commitment scheme is a WI t-Instance-Dependent Trapdoor Commitment if the following property holds:*

- **WI-Trapdorness**: *for every PPT (resp., unbounded) adversary $\mathcal{A}$ there exists a negligible function ν (resp., $\nu(\cdot) = 0$) such that, for all $x \in L_{\mathsf{Rel}}$, all $m_1, m_2 \in M$ and for all w_0, w_1 s.t. $(x, w_0) \in \mathsf{Rel}$ and $(x, w_1) \in \mathsf{Rel}$, it holds that*

$$\begin{aligned}&\text{Prob}\big[b \leftarrow \{0,1\}; (\texttt{com}, \texttt{dec}, \texttt{rand}) \leftarrow \mathsf{TCom}(1^\lambda, x, m_1),\\ &(w_1, w_2, m_2) \leftarrow \mathcal{A}(x, \texttt{com})\ \texttt{dec}' \leftarrow \mathsf{TFake}(x, w_b, m_1, m_2, \texttt{com}, \texttt{rand}) :\\ &b = \mathcal{A}(x, \texttt{com}, \texttt{dec}')\big] \leq \frac{1}{2} + \nu(\lambda).\end{aligned}$$

Definition 1.2.5 (Perfectly Binding t-Instance-Dependent Trapdoor Commitment scheme)**.** *A t-Instance-Dependent Trapdoor Commitment scheme is a Perfectly Binding t-Instance-Dependent Trapdoor Commitment if the following property holds:*

- **Binding**: *for all $x \notin \hat{L}_{\mathsf{Rel}}$ and for every commitment* $\texttt{com}$ *there exists at most one message $m \in M$ for which there exists a valid decommitment* $\texttt{dec}$*; that is, such that* $\mathsf{TDec}(x, \texttt{com}, \texttt{dec}, m) = 1$.

1.2.2 Non-Malleable Commitment Schemes

Here we follow [LPV08][7]. Let $\Pi = (\mathsf{Sen}, \mathsf{Rec})$ be a statistically binding commitment scheme. Consider MiM adversaries that are participating in left and right sessions in which $\mathsf{poly}(\lambda)$ commitments take place. We compare between a MiM and a simulated execution. In the MiM execution the adversary $\mathcal{A}$, with auxiliary information z, is simultaneously participating in $\mathsf{poly}(\lambda)$ left and right sessions. In the left sessions the MiM adversary $\mathcal{A}$ interacts with Sen receiving commitments to values $m_1, \dots, m_{\mathsf{poly}(\lambda)}$ using identities $\texttt{id}_1, \dots, \texttt{id}_{\mathsf{poly}(\lambda)}$ of its choice. In the right session $\mathcal{A}$ interacts with Rec attempting to commit to a sequence of related values $\tilde{m}_1, \dots, \tilde{m}_{\mathsf{poly}(\lambda)}$ again using identities of its choice $\tilde{\texttt{id}}_1, \dots, \tilde{\texttt{id}}_{\mathsf{poly}}(\lambda)$. If any of the right commitments is invalid, or

[7]In this work we will consider only NM commitments w.r.t. commitments.

undefined, its value is set to $\perp$. For any i such that $\tilde{\mathtt{id}}_i = \mathtt{id}_j$ for some j, set $\tilde{m}_i = \perp$ (i.e., any commitment where the adversary uses the same identity of one of the honest senders is considered invalid). Let $\mathsf{mim}_{\Pi}^{\mathcal{A},m_1,\ldots,m_{\mathsf{poly}(\lambda)}}(z)$ denote a random variable that describes the values $\tilde{m}_1,\ldots,\tilde{m}_{\mathsf{poly}(\lambda)}$ and the view of $\mathcal{A}$, in the above experiment. In the simulated execution, an efficient simulator S directly interacts with Rec. Let $\mathsf{sim}_{\Pi}^{S}(1^{\lambda},z)$ denote the random variable describing the values $\tilde{m}_1,\ldots,\tilde{m}_{\mathsf{poly}(\lambda)}$ committed by S, and the output view of S; whenever the view contains in the i-th right session the same identity of any of the identities of the left session, then $\tilde{m}_i$ is set to $\perp$.

In all the work we denote by $\tilde{\delta}$ a value associated with the right session (where the adversary $\mathcal{A}$ plays with a receiver NMRec) where δ is the corresponding value in the left session. For example, the sender commits to v in the left session while $\mathcal{A}$ commits to $\tilde{v}$ in the right session.

Definition 1.2.6 (Concurrent NM commitment scheme [LPV08])**.** *A commitment scheme is* concurrent NM with respect to commitment *(or a many-many NM commitment scheme) if, for every* PPT *concurrent MiM adversary $\mathcal{A}$, there exists a* PPT *simulator S such that for all $m_i \in \{0,1\}^{\mathsf{poly}(\lambda)}$ for $i = \{1,\ldots,\mathsf{poly}(\lambda)\}$ the following ensembles are computationally indistinguishable:*

$$\{\mathsf{mim}_{\Pi}^{\mathcal{A},m_1,\ldots,m_{\mathsf{poly}(\lambda)}}(z)\}_{z\in\{0,1\}^{\star}} \approx \{\mathsf{sim}_{\Pi}^{S}(1^{\lambda},z)\}_{z\in\{0,1\}^{\star}}.$$

As in [LPV08] we also consider relaxed notions of concurrent non-malleability: one-many and one-one NM commitment schemes. In a one-many NM commitment scheme, $\mathcal{A}$ participates in one left and polynomially many right sessions. In a one-one (i.e., a stand-alone secure) NM commitment scheme, we consider only adversaries $\mathcal{A}$ that participate in one left and one right session. We will make use of the following proposition of [LPV08].

Proposition 1.2.7. *Let* $(\mathsf{Sen},\mathsf{Rec})$ *be a one-many NM commitment scheme. Then,* $(\mathsf{Sen},\mathsf{Rec})$ *is also a concurrent (i.e., many-many) NM commitment scheme.*

Definition 1.2.8 (*weak* NM commitment scheme). *A commitment scheme is* weak *one-one (resp., one-many) non-malleable if it is a one-one (resp., one-many) NM commitment scheme with respect to MiM adversary that when receiving a well formed commitment in the left session, except with negligible probability computes well formed commitments (i.e., the computed commitments can be opened to messages $\neq \perp$) in the right sessions.*

In the rest of the thesis, following [GRRV14], we assume that identities are known before the protocol begins, though strictly speaking this is not necessary, as the identities do not appear in the protocol until after the first committer message. The MiM can choose his identity adversarially as long as it differs from the identities used by honest senders. As already observed in previous work, when the identity is selected by the sender the id-based definitions guarantee non-malleability as long as the MiM does not behave like a proxy (an unavoidable attack). Indeed the sender can pick as `id` the public key of a signature scheme signing the transcript. The MiM will have to use a different `id` or to break the signature scheme.

Definition 1.2.9 (*synchronous* NM commitment scheme). *A commitment scheme is* synchronous *one-one (resp., one-many) non-malleable if it is one-one (resp., one-many) NM with respect to synchronous MiM adversaries*[8].

We also consider the definition of a NM commitment scheme secure against a MIM $\mathcal{A}$ running in time bounded by $T = 2^{\lambda^\alpha}$ for some positive constant $\alpha < 1$. In this case we will say that a commitment scheme is T-non-malleable. We will also say that an NM commitment scheme is $\tilde{T}$-breakable to specify that an algorithm which runs in time $\tilde{T} = 2^{\lambda^\beta}$, for some positive constant $\beta < 1$, can maul the committed message.

For some of our propose we use a 4-round synchronous honest-extractable non-malleable commitment. That is, a commitment

[8]Following [LP11b] we say that a MiM is *synchronous* if it "aligns" the left and the right sessions; that is, whenever it receives message i on the left, it directly sends message i on the right, and vice versa.

scheme that enjoys 1) non-malleability only against synchronous adversaries, 2) is extractable w.r.t. honest sender (honest-extractable) and 3) is public-coin. The non-malleable commitment Π provided in Figure 2 of [GPR16] enjoys non-malleability against synchronous adversary (as proved in Theorem 1 of [GPR16]), is public coin and can be instantiated in 4 rounds relying on OWFs (the protocol can be squeezed to 3 rounds using OWPs). Also, as stated in Section 5 of [GPR16], given a commitment computed by the sender of Π one can rewind the sender in order to obtain a new accepting transcript with the same first round (resp., first two rounds if we consider the instantiation that relies on OWFs) in order to extract a message m. Moreover, if the sender is honest, then it is possible to claim that m is the actual message committed by the sender. We remark that in this case we do not require any form of extractability against malicious senders.

1.3 Delayed-Input Non-Malleable Zero Knowledge.

Following [LP11a] we use a definition that gives to the adversary the power of adaptive-input selection. More precisely, in [LP11a] the adversary selects the instance and then a Turing machine outputs the witness in exponential time. Here we slightly deviate (similarly to [DDO$^+$01]) by 1) requiring the adversary to output also the witness and 2) allowing the adversary to make this choice at the last round. This choice is due to our application where delayed-input non-malleable zero knowledge is used. Indeed we will show that this definition is enough for our propose. More precisely our definition (similarly to [COSV17b]) we will allow the adversary to explicitly select the statement, and as such the adversary will provide also the witness for the prover. The simulated game however will filter out the witness so that the simulator will receive only the instance. This approach strictly follows the one of [DDO$^+$01] where adaptive-input selection is explicitly allowed and managed in a similar way. As final remark, our definition will require the existence of a black-box simulator since a non-black-box simulator could retrieve from the code of the adversary the witness for the adaptively generated statement. The non-black-box simulator could then run the honest prover procedure, therefore canceling completely the security flavor of the simulation paradigm.

Let $\Pi = (\mathcal{P}, \mathcal{V})$ be a delayed-input interactive argument system for a $\mathcal{NP}$-language L with witness relation $\mathsf{Rel_L}$. Consider a PPT MiM adversary $\mathcal{A}$ that is simultaneously participating in one left session and $\mathsf{poly}(\lambda)$ right sessions. Before the execution starts, $\mathcal{P}, \mathcal{V}$ and $\mathcal{A}$ receive as a common input the security parameter in unary 1^λ. Additionally $\mathcal{A}$ receives as auxiliary input $z \in \{0,1\}^\star$. In the left session $\mathcal{A}$ verifies the validity of a statement x (chosen adaptively in the last round of Π) by interacting with $\mathcal{P}$ using identity $\mathtt{id}$ of $\mathcal{A}$'s choice. In the right sessions $\mathcal{A}$ proves the validity of the statements $\tilde{x}_1 \ldots, \tilde{x}_{\mathsf{poly}(\lambda)}$[9] (chosen adaptively in the last round of Π) to the honest verifiers $\mathcal{V}_1, \ldots, \mathcal{V}_{\mathsf{poly}(\lambda)}$, using identities $\tilde{\mathtt{id}}_1, \ldots, \tilde{\mathtt{id}}_{\mathsf{poly}(\lambda)}$ of $\mathcal{A}$'s choice.

More precisely in the left session $\mathcal{A}$, before the last round of Π is executed, adaptively selects the statement x to be proved and the witness w, s.t. $(x, w) \in \mathsf{Rel_L}$, and sends them to $\mathcal{P}$[10].

Let $\mathsf{View}^{\mathcal{A}}(1^\lambda, z)$ denote a random variable that describes the view of $\mathcal{A}$ in the above experiment.

Definition 1.3.1 (Delayed-input NMZK). *A delayed-input argument system $\Pi = (\mathcal{P}, \mathcal{V})$ for an $\mathcal{NP}$-language L with witness relation $\mathsf{Rel_L}$ is delayed-input non-malleable zero knowledge (NMZK) if for any MiM adversary $\mathcal{A}$ that participates in one left session and $\mathsf{poly}(\lambda)$ right sessions, there exists a expected* PPT *machine $S(1^\lambda, z)$ such that:*

1. *The probability ensembles $\{S^1(1^\lambda, z)\}_{\lambda \in \mathbb{N}, z \in \{0,1\}^\star}$ and $\{\mathsf{View}^{\mathcal{A}}(1^\lambda, z)\}_{\lambda \in \mathbb{N}, z \in \{0,1\}^\star}$ are computationally indistinguishable over λ, where $S^1(1^\lambda, z)$ denotes the first output of $S(1^\lambda, z)$.*

[9]We denote (here and in the rest of the work) by $\tilde{\delta}$ a value associated with the right session where δ is the corresponding value in the left session.

[10]The witness w sent by $\mathcal{A}$ will be just ignored by the simulator.

2. Let $(\mathsf{View}, w_1, \ldots, w_{\mathsf{poly}(\lambda)})$ *denote the output of* $S(1^\lambda, z)$*, for some* $z \in \{0,1\}^\star$*. Let* $\tilde{x}_1, \ldots, \tilde{x}_{\mathsf{poly}(\lambda)}$ *be the right-session statements appearing in* View *and let* $\mathtt{id}$ *and* $\tilde{\mathtt{id}}_1, \ldots, \tilde{\mathtt{id}}_{\mathsf{poly}(\lambda)}$ *be respectively the identities used in the left and right sessions appearing in* View*. Then for every* $i \in \{1, \ldots, \mathsf{poly}(\lambda)\}$*, if the i-th right session is accepting and* $\mathtt{id} \neq \tilde{\mathtt{id}}_i$*, then* $\tilde{w}_i$ *is s.t.* $(\tilde{x}_i, \tilde{w}_i) \in \mathsf{Rel_L}$*.*

The above definition of NMZK allows the adversary to select statements adaptively in the last round both in left and in the right sessions. Therefore any argument system that is NMZK according to the above definition enjoys also adaptive-input argument of knowledge. Following [LP11b] we say that a MiM is *synchronous* if it "aligns" the left and the right sessions; that is, whenever it receives message i on the left, it directly sends message i on the right, and vice versa. In our work we also consider the notion of *delayed-input many-many synchronous NMZK*, that is equal to the notion of delayed-input NMZK except that polynomially many left and right sessions are played in synchronously.

In the rest of the work, following [GRRV14], we assume that identities are known before the protocol begins, though strictly speaking this is not necessary, as the identities do not appear in the protocol until after the first prover message. The MiM can choose his identity adversarially as long as it differs from the identities used by honest senders. As already observed in previous works, when the identity is selected by the sender the id-based definitions guarantee non-malleability as long as the MiM does not behave like a proxy (an unavoidable attack). Indeed the sender can pick as id the public key of a signature scheme signing the transcript. The MiM will have to use a different id or to break the signature scheme.

1.4 Two-Party Computation

Here we recall some useful definitions for one of our application. Our Multi-Party Computation (MPC) protocol for coin tossing is secure in the same model used in [GMPP16], therefore some definitions are taken almost verbatim from [GMPP16]. Always following Garg et al. we only recall the security definition for the the two party case. The description naturally extends to multi party case as well (details can be found in [Gol09]).

Two-party protocol. A two-party protocol problem is cast by specifying a random process that maps pairs of inputs to pairs of outputs (one for each party). We refer to such a process as a functionality and denote it $F : \{0,1\}^* \times \{0,1\}^* \rightarrow \{0,1\}^* \times \{0,1\}^*$ where $F = (F_1, F_2)$. That is, for every pair of inputs (x, y), the output-pair is a random variable $(F_1(x, y), F_2(x, y))$ ranging over pairs of strings. The first party (with input x) wishes to obtain $F_1(x, y)$ and the second party (with input y) wishes to obtain $F_2(x, y)$.

Adversarial behavior. Loosely speaking, the aim of a secure two-party protocol is to protect an honest party against dishonest behavior by the other party. In this work, we consider malicious adversaries who may arbitrarily deviate from the specified protocol. When considering malicious adversaries, there are certain undesirable actions that cannot be prevented. Specifically, a party may refuse to participate in the protocol, may substitute its local input (and use instead a different input) and may abort the protocol prematurely. One ramification of the adversary's ability to abort, is that it is impossible to achieve fairness. That is, the adversary may obtain its output while the honest party does not. In this work we consider a static corruption model, where one of the parties is adversarial and the other is honest, and this is fixed before the execution begins.

Communication channel. In our result we consider a secure simultaneous message exchange channel in which all parties can simultaneously send messages over the channel at the same communication round but allowing a rushing adversary. Moreover, we assume an asynchronous network[11] where the communication is open and delivery of messages is not guaranteed. For simplicity,

[11]The fact that the network is asynchronous means that the messages are not necessarily delivered in the order which they are sent.

we assume that the delivered messages are authenticated. This can be achieved using standard methods.

Execution in the ideal model. An ideal execution proceeds as follows. Each party obtains an input, denoted w ($w = x$ for P_1, and $w = y$ for P_2). An honest party always sends w to the trusted party. A malicious party may, depending on w, either abort or send some $w' \in \{0,1\}^{|w|}$ to the trusted party. In case it has obtained an input pair (x, y), the trusted party first replies to the first party with $F_1(x, y)$. Otherwise (i.e., in case it receives only one valid input), the trusted party replies to both parties with a special symbol $\perp$. In case the first party is malicious it may, depending on its input and the trusted party's answer, decide to stop the trusted party by sending it $\perp$ after receiving its output. In this case the trusted party sends $\perp$ to the second party. Otherwise (i.e., if not stopped), the trusted party sends $F_2(x, y)$ to the second party. Outputs: An honest party always outputs the message it has obtained from the trusted party. A malicious party may output an arbitrary (probabilistic polynomial-time computable) function of its initial input and the message obtained from the trusted party.

Let $F : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^* \times \{0,1\}^*$ be a functionality where $F = (F_1, F_2)$ and let $S = (S_1, S_2)$ be a pair of non-uniform probabilistic expected polynomial-time machines (representing parties in the ideal model). Such a pair is admissible if for at least one $i \in \{0,1\}$ we have that S_i is honest (i.e., follows the honest party instructions in the above-described ideal execution). Then, the joint execution of F under S in the ideal model (on input pair (x, y) and security parameter λ), denoted $\mathsf{IDEAL}_{F,S(z)}(1^\lambda, x, y)$ is defined as the output pair of S_1 and S_2 from the above ideal execution.

Execution in the real model. We next consider the real model in which a real (two-party) protocol is executed (and there exists no trusted third party). In this case, a malicious party may follow an arbitrary feasible strategy; that is, any strategy implementable by non-uniform probabilistic polynomial-time machines. In particular, the malicious party may abort the execution at any point in time (and when this happens prematurely, the other party is left with no output). Let F be as above and let Π be a two-party protocol for computing F. Furthermore, let $A = (A_1, A_2)$ be a pair of non-uniform probabilistic polynomial-time machines (representing parties in the real model). Such a pair is admissible if for at least one $i \in \{0,1\}$ we have that A_i is honest (i.e., follows the strategy specified by Π). Then, the joint execution of Π under A in the real model, denoted $\mathsf{REAL}_{\Pi,\mathcal{A}(z)}(1^\lambda)$, is defined as the output pair of A_1 and A_2 resulting from the protocol interaction.

Definition 1.4.1 (secure two-party computation)**.** *Let F and Π be as above. Protocol Π is said to securely compute F (in the malicious model) if for every pair of admissible non-uniform probabilistic polynomial-time machines $A = (A_1, A_2)$ that run with auxiliary input z for the real model, there exists a pair of admissible non-uniform probabilistic expected polynomial-time machines $S = (S_1, S_2)$ (that use z as auxiliary input) for the ideal model, such that:*

$$\mathsf{REAL}_{\Pi,\mathcal{A}(z)}(1^\lambda) \approx \mathsf{IDEAL}_{f,S(z)}(1^\lambda).$$

Chapter 2

Improved OR Composition

2.1 Overview of the Chapter

Witness-indistinguishable (WI) proofs. WI[1] proofs are fundamental for the design of cryptographic protocols, particularly when they are also proofs of knowledge (PoK). In a WIPoK the prover $\mathcal{P}$ proves knowledge of a witness certifying the veracity of a statement $x \in L$ to a verifier $\mathcal{V}$. WIPoKs can be used directly in some applications (e.g., in identification schemes) or can be a building block for stronger security notions (e.g., for zero-knowledge proofs using the FLS [FLS90] paradigm or for round-optimal secure computation [KO04]).

Round complexity of cryptographic protocols has been extensively studied both for its practical relevance and for its natural and conceptual interest. Regarding WIPoKs, we know from Blum's protocol [Blu86a] that 3-round WIPoKs exist for all $\mathcal{NP}$ languages under the sole assumptions that one-way permutations exist. This result is obtained by designing a WIPoK for the language of Hamiltonian graphs and then by leveraging on the NP-completeness of the language of Hamiltonian graphs. Under stronger cryptographic assumptions, 2-round WI proofs, called ZAPs, and non-interactive WI (NIWI) proofs have been shown

[1]We will use WI to mean both "witness indistinguishability" and "witness indistinguishable".

in [DN00, GOS06, BP15]. Neither ZAPs nor NIWI proofs are PoKs.

Since $\mathcal{NP}$ reductions are extremely expensive, several practical interactive PoKs have been designed for languages that are used in real-world cryptographic protocols (e.g., for proving knowledge of a discrete logarithm (DLog)). The study of such ad-hoc protocols mainly concentrates on a standardized form of a 3-round PoK referred to as Σ-protocol [Dam10, Sch89].

Σ-protocols. A Σ-protocol for an $\mathcal{NP}$ language L with witness relation R_L is a 3-round proof system jointly run by a prover $\mathcal{P}$ and a verifier $\mathcal{V}$ in which $\mathcal{P}$ proves knowledge of a witness w for $x \in L$. In a Σ-protocol the only message sent by $\mathcal{V}$ is a random string called challenge. Such proof systems have two very useful properties: special soundness, which is a strong form of proof of knowledge, and special honest-verifier zero knowledge (SHVZK). The latter property basically says the following: if the challenge is known in advance, then by just knowing also the theorem, it is possible to generate an accepting transcript without using the witness. This is formalized through the existence of a special simulator, called the SHVZK simulator that, on input *a theorem* x and a challenge c, will output (a, z) such that (a, c, z) is an accepting 3-message transcript for x and is indistinguishable from the transcript produced by the honest prover when the challenge is c. Blum's protocol for Graph Hamiltonicity is an example of a Σ-protocol. Another popular example of Σ-protocols is Schnorr's protocol [Sch89] for proving knowledge of a discrete logarithm.

The security provided by the SHVZK property is clearly insufficient as it gives no immediate guarantees against verifiers who deviates from the protocol. Despite of this, the success of Σ-protocols and their impact in various constructions [Lin15, LP15, GK15, AOS13, SV12, PS96] is a fact. This is due to a breakthrough of Cramer et al. [CDS94] that adds WI to the security of Σ-protocol.

OR composition of Σ-protocols. Let L be a language that admits a Σ-protocol Π_L. In [CDS94] it is shown how to use Π_L

and its properties to construct a new Σ-protocol, Π_L^{OR}, for proving the OR composition of theorems in L *avoiding* the $\mathcal{NP}$ reduction by crucially exploiting the honest-verifier zero-knowledge (HVZK[2]) property of Π_L. The rationale behind the transformation can be informally explained as follows. The prover wishes to prove a statement of the form $((x_0 \in L) \vee (x_1 \in L))$. The naïve idea of simply running Π_L twice in parallel would not work because the prover knows only one of the witnesses, say w_b, and cannot compute two accepting transcripts without knowing w_{1-b}. However, due to the HVZK property, the prover can generate an accepting transcript for $x_{1-b} \in L$ even without knowing w_{1-b}, by running the HVZK simulator Sim associated with Π_L. Indeed, Sim "only" needs in input the theorem x_{1-b} and will output the entire transcript, challenge included. The trick is then to generate the challenges for the two executions of Π_L, in such a way that the prover can control the challenge of exactly one of them (but not both), and set it to the value generated by Sim. Note that, if running the algorithm of Sim is as efficient as running the algorithm of $\mathcal{P}$, then the composed protocol is efficient. We stress that this OR-composition technique preserves SHVZK and will refer to it as the CDS-OR technique.

A very interesting property of this transformation, besides the fact that it does not need $\mathcal{NP}$ reduction, is that if Sim is a simulator for perfect HVZK then Π_L^{OR} is WI (this was shown in [CDS94]). This result was further extended by Garay et al. [GMY06] that noted that the CDS-OR technique can be used also for Σ-protocols that are computational HVZK. In this case the relation proved is slightly different, namely, starting with a relation Rel_L and instances x_0 and x_1, the resulting Π_L^{OR} protocol is computational WI for the relation $\mathsf{Rel}_L^{\mathsf{OR}} = \{((x_0, x_1), w) : ((x_0, w) \in \mathsf{Rel}_L \wedge (x_1 \in L)) \vee ((x_1, w) \in \mathsf{Rel}_L \wedge (x_0 \in L))\}$.

Delayed-input proofs. Often in cryptographic protocols there is a preamble phase that has the purpose of establishing, at least

[2]HVZK requires the existence of a simulator that by receiving in input the theorem gives in output an accepting triple (a, c, z). Clearly HVZK is implied by SHVZK.

in part, a statement to be proven with a WI proof. In such cases, since one of the statements is fully specified only when the preamble is completed, the WI proof can start only after the preamble ends. Hence, the overall round complexity of protocols that follow this paradigm amounts to the sum of the round complexity of the preamble and of the WI proof.

In [LS90][3], Lapidot and Shamir (and later on Feige et al. in [FLS90]) show a 3-round proof of knowledge for Hamiltonian Graphs which has the special property that enjoys the *delayed input* property. In more details, the prover can compute the first round of the proof, *without* knowing the theorem to be proved (that is, the graph) but only needs to know its size (that is, the number of vertices). Such a 3-round protocol is a Σ-protocol (and thus satisfies the SHVZK property) and is a WI proof. We will refer to this protocol as LS.

The delayed-input property directly improves the round complexity of all the cryptographic protocols that follow the paradigm described above. The reason is that now the WI proof can start even if the preamble that generates the statement is not completed yet. It is worthy to note that in many applications the preamble serves as a mean to generate some trapdoor theorem, that is used only in the security proof. The "honest" theorem instead is typically known already at the beginning of the protocol. This technique has been used extensively and, most notably, it led to the celebrated FLS paradigm that upgrades any WI proof system into a zero-knowledge (ZK) proof system.

The delayed-input property of LS has been instrumental to provide round-efficient constructions from general assumptions, such as: 4-round (optimal) secure 2PC where only one player gets the output (5 rounds when both players get the output) [KO04], 4-round resettable WI arguments [YZ07, SV12], 4-round (optimal) resettable ZK for $\mathcal{NP}$ in the $\mathtt{BPK}$ model [YZ07, SV12].

Despite being so influential to achieve round efficiency for cryptographic protocols, the power of LS unfortunately vanishes as soon as practical constructions are desired. Indeed, similarly to

[3]See [OV12] for a detailed description of [LS90].

Blum's protocol, LS is crucially based on specific properties of Hamiltonian graphs. Thus, when used to prove more natural languages, which is the case of most of the applications using WI proofs, it requires to perform rather inefficient $\mathcal{NP}$ reductions.

Efficient protocols and limits of the CDS-OR technique. A natural question is what happens if we want to avoid the $\mathcal{NP}$ reduction and we try to use the CDS-OR technique to construct delayed-input adaptive WI proofs. A bit more specifically, we know that there exist Σ-protocols that are delayed input. Schnorr's protocol [Sch89] for DLog is such an example since the first message can be computed without knowing the instance, but only a group generator. Thus the question is what happens if we apply the CDS-OR technique to an delayed-input Σ-protocol. Do we obtain a WI Σ-protocol that is delayed input as well?

Unfortunately, the answer is negative. The CDS-OR technique does *not* preserve the delayed-input property, not even when used to compose two Σ-protocols that are both delayed input. To see why, recall that the CDS-OR composition technique when applied to Σ-protocol Π_L for language L requires the prover to compute two accepting transcripts, one of which is computed by running the HVZK simulator Sim. Recall that Sim needs in input the theorem to be proved. Hence, to prove knowledge of a witness for the compound theorem $(x_0 \in L \vee x_1 \in L)$, the prover, who knows one witness, say w_b, needs to know also x_{1-b} already at the first round to be able to run the simulator. Thus, in the CDS-OR technique the prover can successfully complete the protocol if and only if *both*[4] instances are specified already at the first round.

Because of this missing feature, the CDS-OR technique has limited power in allowing one to obtain round-efficient/optimal cryptographic protocols, compared to the number of rounds obtained by using LS. As such, in some cases when focusing on efficient constructions, the *best* round-complexity that we can achieve using efficient Σ-protocols and avoiding $\mathcal{NP}$ reductions needs at

[4]To see why, note that the WI property requires that the prover would be able to prove any of the two theorems, and thus potentially use the simulator on either x_0 or x_1.

least one additional round, therefore requiring at least 5-round if one wants to match the previously mentioned applications (e.g., 5-round resettable ZK for $\mathcal{NP}$ in the BPK model [YZ07, SV12] and 5-round resettable WI [YZ07, SV12]) argument systems.

Additionally, we note that the CDS-OR technique is the bottleneck in the round-complexity of the 4-round straight-line perfect simulatable in quasi-polynomial time argument shown by Pass in [Pas03]. This argument uses quasi-polynomial time simulation and, potentially, it would only need three rounds as any Σ-protocol. The additional first round is required precisely to define the trapdoor theorem. Hence, the following natural question arises:

> *Given a language L with an delayed-input Σ-protocol Π_L, is it possible to design an efficient Witness Indistinguishable Σ-protocol Π^L_{OR} for proving knowledge of a witness certifying that $(x_0 \in L \vee x_1 \in L)$ that does* not *require knowledge of both x_0 and x_1 to play the first round?*

2.1.1 Our Contribution

In this work we answer the above question positively for a large class of Σ-protocols that includes *all* Σ-protocols used in efficient constructions. Specifically, we propose a new OR-composition technique for Σ-protocols that relaxes the need of having both instances fixed before the Σ-protocol starts. Our technique allows the composition of Σ-protocols for different languages and leads to improved round complexity in previous efficient constructions based on CDS-OR technique. Namely, we describe the following two results that we obtain by making use of our new OR-composition technique:

- Efficient 3-round straight-line perfect quasi-polynomial time simulatable argument system for a large class of useful languages. The previous construction required four rounds [Pas03].

- Efficient 4-round rWI argument system. Previous constructions required five rounds [YZ07, SV12].

Our new technique can also be used to replace LS towards obtaining efficient round-optimal resettable zero-knowledge arguments in the BPK model (using the constructions of [YZ07, SV12]).

Finally, we provide a precise classification of the Σ-protocols that can be used in our new OR-composition technique. In the following paragraphs we first provide a high-level description our OR-composition technique, then we discuss the applications in more details.

2.1.2 Our Techniques

Overview. We start by defining the setting we are considering. Let L_0 and L_1 be any pair of languages admitting Σ-protocols Π_0 and Π_1. We want to construct a Σ-protocol Π_L^{OR} for the language $L = L_0 \vee L_1$. An instance of L is a pair (x_0, x_1) and we want only x_0 to be specified before Π_L^{OR} starts while x_1 is specified only upon the last round of the protocol[5]. We assume that Π_1 is an *delayed-input* Σ-protocol and thus the first prover message of Π_1 can be computed without knowing x_1. As mentioned earlier this property is satisfied by popular Σ-protocols such as the ones for Discrete Log, Diffie-Hellman triples, and of course, LS itself.

Now, recall that the problem with the CDS-OR technique was that a prover needs to run Sim to compute the first round of the protocol, and this necessarily requires knowledge of *both* theorems before the protocol starts. We want instead that the prover uses only knowledge of x_0.

We solve this problem by introducing a new OR-composition technique that does not require the prover to run Sim on x_1 already in the first round. Instead, our technique allows the prover to wait and take action only in the third round when x_1 is finally defined.

Our starting point is the well known fact that given any Σ-protocol there exists an instance-dependent trapdoor commitment

[5]Like LS, we will just need the size of x_1 to be known when Π_L^{OR} starts.

(IDTC) scheme where the witness for the membership of the instance in the language can be used as a trapdoor to open a committed message as any desired message, as in [DG03]. Our next observation is that, instead of having the prover send the first round for protocol Π_1 in the clear, we can have him send a commitment to it, and such commitment can be computed using an instance-dependent trapdoor commitment based on Π_0 with respect to instance x_0. Recall that this is possible, as in our setting we assume that Π_1 is an delayed-input Σ-protocol, so the prover can honestly compute the first message of Π_1 without knowing x_1. Therefore, the first round of our Π_L^{OR} protocol, is simply an IDTC of a honest Π_1's first round.

Later on, upon receiving the challenge c from the verifier, and after the theorem x_1 is defined, the prover computes the third round as follows. If she has received a witness for x_0, then she will run Sim on input (x_1, c) to compute an accepting transcript of Π_1 for x_1. Then, using the witness w_0 she will equivocate the commitment sent in the first round, according to the message output by Sim. Otherwise, if she has received a witness for x_1 then she does not need to equivocate: she will honestly open the commitment, and honestly compute the third message of Π_1. Therefore, the third round of Π_L^{OR}, simply consists of an opening of the IDTC together with the third message of Π_1.

Now note that this idea works only if we have a special IDTC scheme that has the following strong trapdoor property: a sender can equivocate even a commitment that has been computed honestly. Unfortunately, this property is not satisfied in general by any trapdoor commitment based on Σ-protocols, but only for some. This would restrict the class of Σ-protocols that we can use as L_0 in our technique. For example, this class would not contain Blum's protocol.

Our next contribution is the construction of IDTC schemes that satisfy this strong trapdoor property, for a large class of Σ-protocols. Towards this goal, we define the notion of a t-IDTC scheme which are IDTCs for which the ability to open a commitment in t ways implies knowledge of a witness for the instance associated with the commitment. Next, we construct 2-IDTC and 3-IDTC schemes based on two different classes of Σ-protocols, the union of which includes all the Σ-protocols that are commonly used in cryptographic protocols. Finally, we provide a general OR-composition technique for any pair of languages L_0 and L_1 such that L_0 has a t-IDTC scheme and L_1 has an delayed-input Σ-protocol.

t-instance-dependent trapdoor commitment scheme. For integer $t \geq 2$, a t-IDTC scheme for a polynomial-time relation Rel admitting Σ-protocol Π_{Rel} is a triple $(\mathsf{TCom}, \mathsf{TDec}, \mathsf{TFake})$ where TCom, TDec are the honest commitment/decommitment procedures and TFake is the equivocation procedure that, given a witness for an instance x, equivocates any commitment with respect to x computed by TCom. The crucial differences between a t-IDTC scheme and a regular trapdoor commitment scheme are:
(a) the trapdoor property is strong in the sense that knowledge of the trapdoor (that is, the witness of the instance x) allows to equivocate even commitments that have been honestly computed;
(b) the binding property is relaxed: in a t-IDTC scheme, the sender can open the same commitment in $t-1$ different ways, even without the trapdoor. This relaxation allows us to build an IDTC scheme from a wider class of Σ-protocols, which will cover all the Σ-protocols that have been used in literature.

Constructing a 2-IDTC scheme. A 2-IDTC scheme can be straight-forwardly constructed from any Σ-protocol Π_0 that has the following property: even if the first message a_0 was computed by the SHVZK simulator Sim, an accepting z_0 can be efficiently computed, for every challenge c_0, by using knowledge of the witness and of the randomness used by Sim to produce a_0. We call the Σ-protocols that satisfy this property, *chameleon* Σ-protocols, and we denote by $\mathsf{P}_{\mathsf{sim}}$ the special prover strategy that can answer any challenge even starting from a simulated a_0.

More precisely, given a chameleon Σ-protocol Π_0 for a language L_0, one can construct a 2-IDTC scheme as follows. Let $x_0 \in L_0$. To commit to a message m, the sender runs $\mathsf{Sim}(x_0, m; r_0)$ and obtains a_0, z_0. The commitment is the value a_0. The opening is the pair m, z_0. The commitment is accepted iff (x_0, a_0, m, z_0) is accepting. To equivocate a_0, as a message m', run the special prover algorithm $\mathsf{P}_{\mathsf{sim}}((x_0, m, r_0), w_0, m')$ and obtain an accepting z_0.

Constructing a 3-IDTC scheme. We now discuss a different committing strategy that works for Σ-protocols in which the simulated first message a_0 can only be continued for the challenge specified by Sim, even if a witness is made available. Blum's protocol for Hamiltonicity is an example of a Σ-protocol with this property.

To commit to m, the sender sends a pair (a_0, a'_0) where, with probability $1/2$, a_0 is obtained by running $\mathsf{Sim}(x_0, m)$ while a'_0 is computed by running the prover of Π_0, and with probability $1/2$ the above order is inverted. One can think of a commitment as composed of two threads: a *simulated* thread and a *honest* thread. To open the commitment, the prover sends m and z^*, and the verifier accepts the decommitment if m, z^* are accepting for one of the threads; namely, the verifier checks that either (a_0, m, z^*) or (a'_0, m, z^*) is accepting for $x_0 \in L_0$. To equivocate (a_0, a'_0) to a message m', the sender simply continues the thread of the honest prover, using m' as challenge and computes z^* using the witness. Clearly, a malicious sender can open in two different ways even when $x_0 \notin L$. Nevertheless, three openings allow the extraction of the witness for x_0.

When our OR-composition technique is instantiated with a 3-IDTC scheme we have that the resulting protocol is still WI since no power is added to the verifier. However the protocol is *not* a Σ-protocol since the special-soundness property is not guaranteed. The reason is that, in a 3-IDTC scheme the sender can open the commitment in two different ways even without having the trapdoor (i.e., the witness for $x_0 \in L_0$). Therefore, for any challenge c sent by $\mathcal{V}$, the fact that the commitment of a_1 can be opened in two ways gives a malicious prover $\mathcal{P}^*$ two chances (a_1, c, z_1) and (a'_1, c, z'_1) to successfully complete the protocol for a false statement x_1. Nevertheless, this extra freedom does not hurt soundness as both openings (i.e., a_1 and a'_1) are fixed in advance, and thus when x_1 is not an instance of the language there exist only two challenges c' and c'' that would allow $\mathcal{P}^*$ to succeed. When the challenge is long enough the success probability of $\mathcal{P}^*$ is therefore negligible.

Our construction when starting from a 3-IDTC scheme is 3-special sound (i.e., answering to 3 challenges allows one to compute a witness efficiently), and therefore it is a proof of knowledge when the challenge is long enough.

2.1.3 Discussion

What really matters. Our new OR-composition technique works only when the theorem that has not been defined yet (i.e., x_1, admits an delayed-input Σ-protocol). We stress that this is not a limitation for the applications that we have in mind. In fact, in all *efficient* protocols that make use of delayed-input proofs that we are aware of, the preamble has always the purpose of generating the trapdoor theorem. In practical scenarios[6] L_1 usually corresponds to DLog or DDH. The fact that we can not have Blum's Σ-protocol for L_1 when L_1 is the language of Hamiltonian graphs, is therefore not relevant as the actual language of interest is L_0.

Comparison with the CDS-OR technique. We remark that even in the extremely simplified case where:

1. the two instances x_0 and x_1 are for the same language L,
2. L admits an *delayed-input* Σ-protocol Π_L which is also SHVZK,
3. Π_L is chameleon and thus one can compute the first message using Sim and then continue with the prover to answer to arbitrary challenges,

[6]These are the only scenarios of interest for our work since if practicality is not desired than one can just rely on the LS Σ-protocol and use $\mathcal{NP}$ reductions.

4. the prover knows in advance the witness w and instance x_b for which she will be able to honestly complete the protocol,

the CDS-OR technique fails in obtaining a Σ-protocol (or a WIPoK) for the OR composition of instances of L if any one of the instances is not known when the protocol starts.

Beyond Schnorr's protocol. The works of Cramer [Cra96], Cramer and Damgård [CD98], and Maurer [Mau09, Mau15] showed that a protocol (referred to as the *Pre-Image Protocol*) for proving knowledge of a pre-image of a group homomorphism unifies and generalizes a large number of protocols in the literature. Classic Σ-protocols, such as Schnorr's protocol [Sch89] and the Guillou-Quisquater protocol [GQ88], are particular cases of this abstraction. We show that the *Pre-Image Protocol* is a chameleon Σ-protocol and can thus be used in our construction.

What is in and what is out. As mentioned previously, the Σ-protocol for L_1 can be any *delayed-input* Σ-protocol. We now discuss which Σ-protocols can be used to instantiate L_0 in our OR transform. For this purpose, we identify four classes of Σ-protocols and we prove that any Σ-protocol that falls in any of the first three classes can be used in our OR transform (by instantiating either a 2-IDTC or a 3-IDTC scheme).

We also identify a class of Σ-protocols that is not suitable for any of our techniques. Luckily, we have no example of natural Σ-protocols that fall in this class, and in order to prove the separation we had to construct a very contrived scheme. The four classes are listed below.

- *(Class 1)* Σ-protocols that *are* Chameleon and *do not* require the witness to compute the first round. This class of Σ-protocols can be used to construct both 2-IDTC and 3-IDTC schemes.

- *(Class 2)* Σ-protocols that *are* Chameleon and *require* the prover to use the witness already to compute the first round. This class of Σ-protocols can be used to construct a 2-IDTC scheme.

- *(Class 3)* Σ-protocols that *are not* Chameleon but *do not require* the prover to use the witness in the first round. This class of Σ-protocols can be used to construct a 3-IDTC scheme.

- *(Class 4)* Σ-protocols that *are not* Chameleon and *require* the witness to be used already in the first round. This class of Σ-protocols can not be used in our techniques.

The delayed-input features. We stress here that our techniques allow to start and complete an efficient OR composition of two Σ-protocols (with the discussed restrictions) provided that one instance x_0 is known and another one x_1 will be known later. Having a witness for the first or the second instance always allows $\mathcal{P}$ to convince $\mathcal{V}$. This contrasts with the CDS-OR technique where knowing a witness for x_0 would block $\mathcal{P}$ immediately since $\mathcal{P}$ would need immediately x_1 to continue, but x_1 will not be available until the third round.

2.1.4 Applications

Our new OR-composition technique does not provide the full power of LS because it needs one theorem to be known before the protocol starts. However, as we show below, this seemingly weaker property suffices to improve the round-complexity of some of the previous constructions based on the CDS-OR technique. Such constructions aim to efficiently[7] transform a Σ-protocol for a relation Rel into a *round-efficient* argument with more appealing features.

Efficient 3-round straight-line perfect quasi-polynomial time simulatable argument system. We achieve this result directly, using the construction of Pass [Pas03] and replacing the CDS-OR technique with our technique. As a result the first round of the verifier of [Pas03] can be postponed, therefore reducing the

[7]By *efficiently* we mean that no $\mathcal{NP}$ reduction is needed and only a constant number of modular exponentiations are added. We do not discuss the practicality of the resulting constructions.

round complexity from four to three rounds. Our construction works for all languages admitting a perfect chameleon Σ-protocol.

Efficient 4-round resettable WI arguments. Security against reset attacks has been extensively studied in literature [CGGM00, DN00, COSV12, OV12, COP$^+$14]. It is well known [CGGM00] how to transform a Σ-protocol into a resettable WI protocol: the verifier commits to the challenge c using a perfectly hiding commitment scheme and sends it to the prover in the first round; the prover then computes its messages with randomness derived by applying a pseudo-random function (PRF) on the commitment received. Soundness follows directly from the soundness of the Σ-protocol due to the perfect hiding of the commitment. WI follows from the fact that the protocol is zero knowledge against a stand-alone verifier and thus concurrent WI. Then the use of the PRF and the fact that all messages of the verifier are committed in advance upgrades concurrent WI to resettable WI. This approach, however, generates a 5-round protocol.

Achieving the same result *efficiently*, namely, avoiding $\mathcal{NP}$ reductions, in only four rounds is non-trivial. The reason is that if we attempt to replace the 2-round perfectly hiding commitment with a non-interactive commitment, we lose the unconditional soundness property, and then it is not clear how to argue about computational soundness. More specifically, black-box extraction of the witness is not possible (black-box extraction and resettable WI can not coexist) and the adversarial prover could try to maul the commitment of the verifier and adaptively generate the first round of the Σ-protocol. In fact, even allowing complexity-leveraging arguments (and thus, straight-line extraction), constructing a 4-round WI argument system that avoids $\mathcal{NP}$ reductions and adds only a few modular exponentiations to the underlying Σ-protocol has remained so far an open problem.

We solve this problem by using our new OR-composition technique. We have the verifier commit to the challenge in the first round, but then later, instead of sending the decommitment, she will directly send the challenge and prove that either the challenge is the correct opening of the commitment or she solved some hard puzzle (in our construction, computing the Discrete Log of a random group element chosen by the prover). The puzzle is sent by the prover in the second round and it will be solved by the reduction in super-polynomial time in the proof of soundness.

This trick has been proposed in literature in various forms [Pas03, DPV04] and we are using the form used in [DPV04] where the puzzle is sent only in the second round. [DPV04] must use the LS transform and therefore needs $\mathcal{NP}$-reduction. As explained earlier, going through LS *was* necessary as the CDS-OR transform can be applied only if both statements are fixed at the beginning.

Our new OR transform solves precisely this problem, and it allows the verifier to start the proof before the puzzle is defined, and this proof can be done efficiently without $\mathcal{NP}$ reductions.

Resettable WI follows from the CGGM transformation and the WI property of the proof generated by the prover. The groups used for the commitment of the challenge and for the puzzle sent by the prover, will be chosen appropriately so that the hardness of computing discrete logarithms are different and guarantee that our reductions work (i.e., we make use of complexity leveraging).

4-Round (optimal round) resettable ZK argument systems in the BPK model. Our new OR-composition technique can find various other applications. Indeed, wherever there is a round-efficient (but otherwise inefficient) construction based on the use of LS without a corresponding efficient construction with the *same* round complexity, then our technique constitutes a powerful tool towards achieving computationally efficient and round-efficient constructions. For instance, the 4-round (optimal round) resettable ZK argument systems in the BPK model provided in [YZ07, SV12], consists (roughly) of the parallel execution of a (resettable) WI protocol from the prover to the verifier, where the prover proves that either $x \in L$ or he knows the secret key associated to the public identity of the verifier, and a 3-round (resettably-sound) WI protocol from the verifier to the prover, where $\mathcal{V}$ proves knowledge of the secret key associate to its public key, or knowledge of the solution of a puzzle computed by the prover. When instantiated with efficient Σ-protocols, such construction requires

5-rounds, where the additional round, from the prover to the verifier, is used to send the puzzle necessary for the verifier to start a proof using the CDS-OR technique. We observe that this setting closely resembles the setting of the 4-round resettable WI ($\mathsf{r}\mathcal{WI}$) protocol that we provide in this work. As such, one could directly instantiate the proof provided by the prover of the BPK model, with our 4-round $\mathsf{r}\mathcal{WI}$ protocol, and have the verifier just prove knowledge of its secret keys, thus avoiding the need of the additional first round.

Other applications. In [BKZZ16] the authors introduce a new two-party protocol called Proof of Work or Knowledge (PoWorK). In this setting a prover can convince a verifier that he has either performed a work or he knows a witness for a public theorem. Also, a malicious verifier is unable to distinguish which strategy has been used by the prover. As also mentioned in [BKZZ16], our OR-composition can be used to implements PoWorK based on discrete logarithm assumption.

2.2 OR Composition of $\tilde{\Sigma}$-protocols: the CDS-OR Transform

In this section we describe the CDS-OR [CDS94] transform in details. Let Π be a $\tilde{\Sigma}$-protocol for polynomial-time relation Rel with challenge length l, associated algorithms (P_1, P_2, V) and HVZK simulator Sim. The CDS-OR transform constructs a $\tilde{\Sigma}$-protocol Π_{OR} with associated algorithms $(\mathsf{P}_1^{\mathsf{OR}}, \mathsf{P}_2^{\mathsf{OR}}, \mathsf{V}_{\Sigma}^{\mathsf{OR}})$ for the relation

$$\mathsf{Rel}_{\mathsf{OR}} = \Big\{((x_0, x_1), w) : \Big((x_0, w) \in \mathsf{Rel} \wedge x_1 \in \hat{L}_{\mathsf{Rel}}\Big) \\ \mathsf{OR} \Big((x_1, w) \in \mathsf{Rel} \wedge x_0 \in \hat{L}_{\mathsf{Rel}}\Big) \Big\}.$$

We describe Π_{OR} below.

Protocol 1. CDS-OR Transform.
Common input: (x_0, x_1).
$\mathcal{P}$'s private input: (b, w) with $b \in \{0, 1\}$ and $(x_b, w) \in \mathsf{Rel}$.

$\mathsf{P}_1^{\mathsf{OR}}((x_0, x_1), (b, w); R_1)$. *Set* $a_b = P_1(x_b, w; R_1)$. *Compute* $(a_{1-b}, c_{1-b}, z_{1-b}) \leftarrow \mathsf{Sim}(x_{1-b})$. *Output* (a_0, a_1).

$P_2^{\mathsf{OR}}((x_0, x_1), (b, w), c, R_1)$. *Set* $c_b = c \oplus c_{1-b}$. *Compute* $z_b \leftarrow P_2(x_b, w, c_b, R_1)$. *Output* $((c_0, c_1), (z_0, z_1))$.

$\mathsf{V}_\Sigma^{\mathsf{OR}}((x_0, x_1), (a_0, a_1), c, ((c_0, c_1), (z_0, z_1)))$. $\mathsf{V}_\Sigma^{\mathsf{OR}}$ *accepts if and only if* $c = c_0 \oplus c_1$ *and* $\mathsf{V}(x_0, a_0, c_0, z_0) = 1$ *and* $\mathsf{V}(x_1, a_1, c_1, z_1) = 1$.

Theorem 4 ([CDS94, GMY06]). If Π is a $\tilde{\Sigma}$-protocol for Rel then Π_{OR} is a $\tilde{\Sigma}$-protocol for $\mathsf{Rel}_{\mathsf{OR}}$ and is WI for relation

$$\begin{aligned}\mathsf{Rel}'_{\mathsf{OR}} = \{((x_0, x_1), w) : ((x_0, w) \in \mathsf{Rel} \wedge x_1 \in L_{\mathsf{Rel}})\\ \mathsf{OR}((x_1, w) \in \mathsf{Rel} \wedge x_0 \in L_{\mathsf{Rel}})\}.\end{aligned}$$

Moreover, if Π is a Perfect $\tilde{\Sigma}$-protocol for Rel then Π^{OR} is WI for $\mathsf{Rel}_{\mathsf{OR}}$.

It is possible to extend the above construction to handle two different relations Rel_0 and Rel_1 that admit $\tilde{\Sigma}$-protocols. Indeed by Theorem 21, we can assume, wlog, that Rel_0 and Rel_1 have $\tilde{\Sigma}$-protocols Π_0 and Π_1 with the same challenge length. Hence, the construction outlined above can be used to construct $\tilde{\Sigma}$-protocol $\Pi^{\mathsf{OR}}_{\mathsf{Rel}_0,\mathsf{Rel}_1}$ for relation

$$\begin{aligned}\mathsf{Rel}_{\mathsf{OR}} = \{((x_0, x_1), w) : ((x_0, w) \in \mathsf{Rel} \wedge x_1 \in \hat{L}_{\mathsf{Rel}})\\ \mathsf{OR}((x_1, w) \in \mathsf{Rel} \wedge x_0 \in \hat{L}_{\mathsf{Rel}})\}.\end{aligned}$$

We have the following theorem.

Theorem 5. If Π_0 and Π_1 are $\tilde{\Sigma}$-protocols for Rel_0 and Rel_1, respectively, then $\Pi^{\mathsf{OR}}_{\mathsf{Rel}_0,\mathsf{Rel}_1}$ is a $\tilde{\Sigma}$-protocol for relation $\mathsf{Rel}_{\mathsf{OR}}$ and is WI for relation

$$\begin{aligned}\mathsf{Rel}'_{\mathsf{OR}} = \{((x_0, x_1), w) : ((x_0, w) \in \mathsf{Rel} \wedge x_1 \in L_{\mathsf{Rel}_0}\\ \mathsf{OR}((x_1, w) \in \mathsf{Rel} \wedge x_0 \in L_{\mathsf{Rel}_1})\}.\end{aligned}$$

Moreover, if Π_0 and Π_1 are Perfect $\tilde{\Sigma}$-protocols for Rel_0 and Rel_1 then Π^{OR} is Perfect WI for $\mathsf{Rel}_{\mathsf{OR}}$.

We remark that if Π_0 and Π_1 are Σ-protocols then the CDS-OR transform yields a Σ-protocol for $\mathsf{Rel}_{\mathsf{OR}}$ and the equivalent of Theorem 5 (and of Theorem 4) holds.

2.3 Constructing t-IDTC Scheme

2.3.1 Constructing a 2-IDTC Scheme from a Chameleon Σ-protocol.

Let $\Pi = (\mathcal{P}, \mathcal{V})$ with associated algorithms $(P_1, P_2, \mathsf{V}, \mathsf{P}_{\mathsf{sim}})$ be a Chameleon Σ-protocol for polynomial-time relation Rel with a security parameter 1^λ. Let l be the challenge length of Π and let Sim be a SHVZK simulator associated to Π. We construct a 2-IDTC scheme $(\mathsf{TCom}_\Pi, \mathsf{TDec}_\Pi, \mathsf{TFake}_\Pi)$ for Rel with messages space $M = \{0,1\}^l$ for $x \in \hat{L}_R$ as follows.

Protocol 2. 2-IDTC *scheme from Chameleon* Σ*-protocol* Π.

- $\mathsf{TCom}_\Pi(1^\lambda, x, m_1)$: On input x and $m_1 \in M$, pick randomness R and compute $(a, z) \leftarrow \mathsf{Sim}(x, m_1; R)$. Output $\mathtt{com} = a$, $\mathtt{dec} = z$ and $\mathtt{rand} = R$;

- $\mathsf{TDec}_\Pi(x, \mathtt{com}, \mathtt{dec}, m_1)$: On input $x, \mathtt{com}, \mathtt{dec}$ and m_1, run $b = \mathsf{V}(x, \mathtt{com}, m_1, \mathtt{dec})$ and accept m_1 as the decommitted message iff $b = 1$.

- TFake_Π: On input $(x, w) \in \mathsf{Rel}$, messages $m_1, m_2 \in M$, commitment $\mathtt{com}$ for m_1 and $\mathtt{rand}$ for $\mathtt{com}$, output $z = \mathsf{P}_{\mathsf{sim}}((x, m_1, \mathtt{rand}), w, m_2)$.

Theorem 6. If Π is a Chameleon Σ-protocol for Rel then Protocol 2 is a 2-IDTC scheme for Rel. Moreover, if Π is Perfect then so is Protocol 2.

Proof. Correctness follows directly from the Completeness property of Π.

2-Special-Extract. Suppose $\mathtt{com}$ is a commitment with respect to instance x and let $\mathtt{dec}_1$ and $\mathtt{dec}_2$ be two openings of $\mathtt{com}$ as messages $m_1 \neq m_2$, respectively. Then, triplets $(\mathtt{com}, m_1, \mathtt{dec}_1)$ and $(\mathtt{com}, m_2, \mathtt{dec}_2)$ are accepting transcripts for Π on common input x with the same first round; that is, they constitute a collision for Π. Therefore, we define algorithm $\mathsf{ExtractTCom}$ to be the algorithm that runs algorithm $\mathsf{Extract}$ (that exists by the special soundness of Π) on input the collision. $\mathsf{ExtractTCom}$ returns the witness for x computed by $\mathsf{Extract}$.
(Perfect) Trapdoorness. It follows from the (perfect) delayed-indistinguishability property of Π as well as the (perfect) Hiding property.

□

Theorem 7. If Π is a Chameleon Σ-protocol for Rel that enjoys the (perfect) WI property, then the 2-IDTC scheme for Rel is (perfect) WI 2-IDTC for Rel.

Proof. We recall that if Π is (perfect) WI he is (perfect) WI even when the pair $(\mathsf{Sim}, \mathsf{P}_{\mathsf{sim}})$ is used (that is when 2-IDTC is playing in the trapdoor mode). If exists an adversary that can contradict the WI-Trapdorness of WI 2-IDTC it is possible to use it to make a reduction to the WI of Π. More in details, let $x \in \hat{L}_R$ be the statement on input to the procedures of 2-IDTC. The reduction works as a proxy between the adversary and the challenger of the WI of Π ($\mathcal{CH}$). The reduction chooses at random a messages $m_1 \in M$ and sends it to $\mathcal{CH}$ receiving back a. Then the reduction send $\mathtt{com} = a$ to the adversary. Upon receiving $m_2 \in M$ and the witnesses w_0, w_1 s.t. $(x, w_0) \in \mathsf{Rel}$ and $(x, w_1) \in \mathsf{Rel}$ from the adversary the reduction sends m_2, w_0, w_1 to $\mathcal{CH}$. The reduction obtains z from $\mathcal{CH}$ and he sets $\mathtt{dec} = z$. The reduction sends $\mathtt{dec}$ to the adversary and outputs the output of the adversary.

If Π is perfect WI then the WI 2-IDTC enjoys the perfect WI-Trapdorness property. □

Theorem 8. If Π is a Chameleon Σ-protocol for Rel, then the 2-IDTC scheme for Rel is a perfect binding 2-IDTC for Rel.

Proof. The property of perfect binding follows from the special soundness of Π. □

2.3.2 Constructing a 3-IDTC Scheme.

Let Rel be a polynomial-time relation as above admitting a delayed-witness Σ-protocol Π with associated algorithms (P_1, P_2, V) and security parameter 1^λ. Let l denote the challenge length of Π. We construct a 3-IDTC scheme for message space $M = \{0,1\}^l$ for $x \in \hat{L}_R$, as follows.

Protocol 3. 3-IDTC *scheme.*

- TCom_Π: On input 1^λ, x and $m_1 \in M$, pick randomness R and compute $(a_0, z) \leftarrow \mathsf{Sim}(x, m_1)$ and $a_1 \leftarrow P_1(x; R)$. Let $\mathtt{com}_0 = a_0$ and $\mathtt{com}_1 = a_1$. Output $\mathtt{com} = (\mathtt{com}_b, \mathtt{com}_{1-b})$ for a randomly selected bit b, $\mathtt{dec} = z$ and $\mathtt{rand} = R$.

- TDec_Π: On input $x, \mathtt{com} = (\mathtt{com}_0, \mathtt{com}_1), \mathtt{dec}$ and m_1, accept m_1 if and only if either $\mathsf{V}(x, \mathtt{com}_0, m_1, \mathtt{dec}) = 1$ or $\mathsf{V}(x, \mathtt{com}_1, m_1, \mathtt{dec}) = 1$.

- TFake_Π: On input $(x, w) \in \mathsf{Rel}$, messages $m_1, m_2 \in M$, commitment $\mathtt{com}$ for m_1 and $\mathtt{rand}$ for $\mathtt{com}$, output $z \leftarrow P_2(x, w, \mathtt{rand}, m_2)$.

Theorem 9. If Π is a delayed-witness Σ-protocol for Rel, with the associated algorithms (P_1, P_2, V), then Protocol 3 is a 3-IDTC scheme for Rel. Moreover, if Π is Perfect then so is Protocol 3.

Proof. Correctness follows from the completeness of Π.

3-*Special Extract.* It follows from the special soundness of Π. Assume that the committer generates 3 accepting openings $\mathtt{dec}_1$, $\mathtt{dec}_2$ and $\mathtt{dec}_3$, for distinct messages m_1, m_2 and m_3, for the same commitment $\mathtt{com}$ computed w.r.t. x. In this case, we have three accepting transcript for Π and therefore at least two of them must share the same first message, i.e., it is a collision. Thus we can run the extractor Extract for Π on the collision and obtain a witness for x.

Trapdoorness. It follows from the SHVZK property of Π. We prove this property via hybrid arguments.

The first hybrid, $\mathcal{H}_1$ is the real execution, where a honest prover commits to a message following the honest commitment and decommitment procedure, without using the trapdoor. More formally, in the hybrid $\mathcal{H}_1$ the prover performs the following steps:

- On input x and $m_1, m_2 \in M$, the prover selects random coin tosses R and computes $(a_0, z) \leftarrow \mathsf{Sim}(x, m_2)$, $a_1 \leftarrow P_1(x; R)$. It picks $b \leftarrow \{0,1\}$ and sends $\texttt{com} = (a_b, a_{1-b})$, $\texttt{dec} = z$, m_2.

The second hybrid $\mathcal{H}_2$ is equal to $\mathcal{H}_1$ with the difference that a_0 is computed using the algorithm P_1 and z using P_2. Formally:

- On input x and $m_1, m_2 \in M$, the prover selects random coin tosses $R = (r_1, r_2)$ and computes $a_0 \leftarrow P_1(x; r_1)$, $z \leftarrow P_2(x, w, r_1, m_2)$ and $a_1 \leftarrow P_1(x; r_2)$. It picks $b \leftarrow \{0,1\}$ and sends $\texttt{com} = (a_b, a_{1-b})$, $\texttt{dec} = z$, m_2.

Due to the SHVZK property of Π, $\mathcal{H}_1$ is indistinguishable from $\mathcal{H}_2$. Now we consider the hybrid $\mathcal{H}_3$ in which a_1 is computed using $\mathsf{Sim}(x, m_2)$. Formally:

- On input x and $m_1, m_2 \in M$, the prover selects random coin tosses R and computes $a_0 \leftarrow P_1(x; R)$, $z \leftarrow P_2(x, w, R, m_2)$ and $(a_1, \overline{z}) \leftarrow \mathsf{Sim}(x, m_1)$. It picks $b \leftarrow \{0,1\}$ and sends $\texttt{com} = (a_b, a_{1-b})$, $\texttt{dec} = z$, m_2.

Even in this case, we can claim that $\mathcal{H}_3$ is indistinguishable from $\mathcal{H}_2$ because of the SHVZK of Π. The proof ends with the observation that $\mathcal{H}_3$ is the experiment in which a sender commits to a message m_1 and opens to m_2 using the trapdoor.

If Π is a perfect SHVZK protocol, then the sequence of hybrids produces identical distributions. $\square$

Theorem 10. If, Π is a is a delayed-witness Σ-protocol for Rel that enjoys (perfect) WI property, with the associated algorithms (P_1, P_2, V), then the 3-IDTC scheme for Rel is a (perfect) WI 3-IDTC for Rel.

Proof. If exists an adversary against the WI-Trapdorness property of WI 3-IDTC it is possible to use it to make a reduction to the WI of Π. More in details, let $x \in \hat{L}_R$ be the statement on input to the procedures of 3-IDTC. The reduction works as a proxy between the adversary and the challenger of the WI of Π ($\mathcal{CH}$). Upon receiving a_0 from $\mathcal{CH}$ the reduction chooses at random a messages $m_1 \in M$ and runs $(a_1, \overline{z}) \leftarrow \mathsf{Sim}(x, m_1)$ and forwards $\mathtt{com} = (a_b, a_{1-b})$ to the adversary, where $b \in \{0, 1\}$. Upon receiving the reduction $m_2 \in M$ and the witnesses w_0, w_1 s.t. $(x, w_0) \in \mathsf{Rel}$ and $(x, w_1) \in \mathsf{Rel}$ from the adversary sends m_2, w_0, w_1 to $\mathcal{CH}$. The reduction obtains z from $\mathcal{CH}$ and he sets $\mathtt{dec} = z$. The reduction sends $\mathtt{dec}$ to the adversary and he outputs the output of the adversary. If Π is perfect WI then the WI 3-IDTC enjoys the perfect WI-Trapdorness property. □

Theorem 11. If Π is a delayed-witness Σ-protocol for Rel, with the associated algorithms (P_1, P_2, V), then the 3-IDTC scheme for Rel is a perfect binding 3-IDTC for Rel.

Proof. The property of binding follows from the special soundness of Π. □

2.4 A New OR-Composition Technique

In this section we formally describe our new OR transform. Let Rel_0 be a relation admitting a t-IDTC scheme,
$I = (\mathsf{TCom}_{\Pi_0}, \mathsf{TDec}_{\Pi_0}, \mathsf{TFake}_{\Pi_0})$, with $t = 2$ or $t = 3$, and Rel_1 a relation admitting an delayed-input Σ-protocol Π_1 with associated algorithms $(P_1^1, P_2^1, \mathsf{V}^1)$ and simulator Sim^1. We show a Σ-protocol

Π^{OR} for the OR relation:

$$\begin{aligned}\mathsf{Rel}_{\mathsf{OR}} = \{((x_0, x_1), w) : ((x_0, w) \in \mathsf{Rel}_0 \wedge x_1 \in \hat{L}_{\mathsf{Rel}_1}) \\ \mathsf{OR}\ ((x_1, w) \in \mathsf{Rel}_1 \wedge x_0 \in \hat{L}_{\mathsf{Rel}_0})\}.\end{aligned}$$

We denote by $(P_1^{\mathsf{OR}}, P_2^{\mathsf{OR}}, \mathsf{V}^{\mathsf{OR}})$ the algorithms associated with Π^{OR}. We assume that the initial common input is x_0. The other input x_1 and the witness w for (x_0, x_1) are made available to the prover only after the challenge has been received. We let $b \in \{0, 1\}$ be such that $(x_b, w) \in \mathsf{Rel}_b$ and assume that the message space of the t-IDTC scheme I includes all possible first-round messages of Π_1. Note that for the constructions of the t-IDTC scheme we provide, the message space coincides with the set of challenges of the underlying Σ-protocol and, in Section 2.7.1, we show that the challenge length of a Σ-protocol can be easily expanded/reduced.

We remind that prover algorithm P_2^{OR} receives as further input the randomness $(R_1, \mathtt{rand}_1)$ used by P_1^{OR} to produce the first-round message.

Protocol 4. *Protocol* Π^{OR} *for* $\mathsf{Rel}_{\mathsf{OR}}$.

Common input: $(x_0, 1^\lambda)$, where 1^λ is the security parameter.

1. $P_1^{\mathsf{OR}}(x_0, 1^\lambda)$. Pick random R_1 and compute $a_1 \leftarrow P_1^1(1^\lambda; R_1)$. Then commit to a_1 by running $(\mathtt{com}, \mathtt{dec}_1, \mathtt{rand}_1) \leftarrow \mathsf{TCom}_{\Pi_0}(1^\lambda, x_0, a_1)$. Output $\mathtt{com}$.
2. $P_2^{\mathsf{OR}}((x_0, x_1), c, (w, b), (\mathtt{rand}_1, R_1))$ (with $(x_b, w) \in \mathsf{Rel}_b$).
   If $b = 1$, compute $z_1 \leftarrow P_2^1(x_1, w, R_1, c)$ and output $(\mathtt{dec}_1, a_1, z_1)$.
   If $b = 0$, compute $(a_2, z_2) \leftarrow \mathsf{Sim}^1(x_1, c)$, $\mathtt{dec}_2 \leftarrow$ $\mathsf{TFake}_{\Pi_0}(x_0, w, a_1, a_2, \mathtt{com}, \mathtt{rand}_1)$ and output $(\mathtt{dec}_2, a_2, z_2)$.
3. V^{OR}, on input (x_0, x_1), $\mathtt{com}$, c, and $(\mathtt{dec}, a, z))$ received from Π^{OR}, outputs 1 iff

$$\mathsf{TDec}_{\Pi_0}(x_0, \mathtt{com}, \mathtt{dec}, a) = 1 \text{ and } \mathsf{V}^1(x_1, a, c, z) = 1;$$

Theorem 12. If Rel_0 admits a 2-IDTC (resp., 3-IDTC) scheme and if Rel_1 admits an delayed-input Σ-protocol, then Π^{OR} is a Σ-protocol (resp., is a 3-round public-coin SHVZK PoK) for relation $\mathsf{Rel}_{\mathsf{OR}}$.

Proof. Completeness follows by inspection. We next prove the properties of Protocol 4 when instantiated with a 2-IDTC and 3-IDTC schemes.

Proof for the construction based on the 2-IDTC **scheme.** *Special Soundness.* It follows from the special soundness of the underlying Σ-protocol Π_1 and the 2-Special Extract of the 2-IDTC scheme. More formally, consider a collision $(\mathtt{com}, c, (\mathtt{dec}, a, z))$ and $(\mathtt{com}, c', (\mathtt{dec}', a', z'))$ for input (x_0, x_1). We observe that:

- if $a = a'$ then (a, c, z) and (a', c', z') is a collision for Π_1 for input x_1; then we can obtain a witness w_1 for x_1 by the Special Soundness property of Π_1;

- if $a \neq a'$, then $\mathtt{dec}$ and $\mathtt{dec}'$ are two openings of $\mathtt{com}$ with respect to x_0 for messages $a \neq a'$; then we can obtain a witness w_0 by the 2-Special Extract of the 2-IDTC scheme.

SHVZK property. Consider simulator $\mathsf{Sim}^{\mathsf{OR}}$ that, on input (x_0, x_1) and challenge c, sets $(a, c, z) \leftarrow \mathsf{Sim}_1(x_1, c)$ and $(\mathtt{com}, \mathtt{dec}) \leftarrow \mathsf{TCom}_{x_0}(a)$, and outputs $(\mathtt{com}, c, (\mathtt{dec}, a, z))$. Next, we show that the transcript generated by $\mathsf{Sim}^{\mathsf{OR}}$ is indistinguishable from the one generated by a honest prover.

Let us first consider the case in which the prover of Π^{OR} receives a witness for x_1. In this case, if we sample a random distribution $(\mathtt{com}, c, (\mathtt{dec}, a, z))$ of Π^{OR} on input (x_0, x_1) constrained to c being the challenge we have that (a, c, z) has the same distribution as in random transcript of Π_1 on input x_1 constrained to c being the challenge; moreover, $(\mathtt{com}, \mathtt{dec})$ is a pair of commitment and decommitment of a with respect to x_0. By the property SHVZK of Π_1, this distribution is indistinguishable from (a, c, z) computed as $\mathsf{Sim}_1(x_1, c)$ which is exactly as in the output $\mathsf{Sim}^{\mathsf{OR}}$.

Let us now consider the case in which the prover of Π^{OR} receives a witness for x_0. If we sample a random distribution $(\mathtt{com}, c, (\mathtt{dec}, a, z))$ of Π^{OR} on input (x_0, x_1) constrained to c being the challenge we have that (a, c, z) are distributed exactly as in the output of $\mathsf{Sim}^{\mathsf{OR}}$ (that is by running Sim_1 on input x_1 and c). In addition, in the output of $\mathsf{Sim}^{\mathsf{OR}}$, $(\mathtt{com}, \mathtt{dec})$ are commitment and decommitment of a

whereas in the view of Π^{OR} they are computed by means of TFake algorithm. However, the two distributions are indistinguishable by the trapdoorness of the Instance-Dependent Trapdoor Commitment.

Proof for the construction based on the 3-IDTC scheme.

3-*Special Soundness.* This property ensures that there exists an efficient algorithm that, given three accepting transcripts, (a, c_0, z_0), (a, c_1, z_1), (a, c_2, z_2) with $c_i \neq c_j$ for $1 \leq i < j \leq 3$, for the same common input, outputs a witness for x.

Consider three accepting transcripts for Π^{OR} and input (x_0, x_1):

$$(\texttt{com}, c_1, (\texttt{dec}_1, a_1, z_1)), \qquad (\texttt{com}, c_2, (\texttt{dec}_2, a_2, z_2))$$

and

$$(\texttt{com}, c_3, (\texttt{dec}_3, a_3, z_3)).$$

We observe that:

- if $a_i = a_j$ for some $i \neq j$ then (a_i, c_i, z_i) and (a_j, c_j, z_j) is a collision for Π_1 for input x_1; thus we can obtain a witness w_1 for x_1 by the Special Soundness property of Π_1;

- if $a_i \neq a_j$ for all $i \neq j$, then, $\texttt{dec}_1$ and $\texttt{dec}_2$ and $\texttt{dec}_3$ are three openings of the same $\texttt{com}$ with respect to x_0 for messages a_1, a_2 and a_3; then we can obtain a witness w_0 for x_0 by the 3-Special Extract of the 3-IDTC scheme.

We stress that having a long enough challenge, 3-special soundness implies the proof of knowledge property.

SHVZK property. This is similar to the proof for the construction based on 2-IDTC. □

2.4.1 Witness Indistinguishability of Our Transform

In this section we discuss the *adaptive WI* property of Π^{OR}. Roughly speaking, adaptive WI means that in the WI experiment the adversary $\mathcal{A}$ is not forced to choose *both* theorems x_0 and x_1 at the onset of the experiment. Rather, she can choose theorem x_1 and witnesses w_0, w_1 adaptively, *after* seeing the first message of Π^{OR} played by the prover on input x_0. After x_1, w_0, w_1 have been selected by $\mathcal{A}$, the experiment randomly selects $b \leftarrow \{0,1\}$. The prover then receives x_1 and w_b and proceeds to complete the protocol. The adversary wins the game if she can guess b with probability non-negligibly greater than 1/2. More formally, we consider adaptive WI for polynomial-time relation

$$\begin{aligned}\mathsf{Rel}^p_{\mathsf{OR}} = ((x_0, x_1), w) : ((x_0, w) &\in \mathsf{Rel}_0 \wedge x_1 \in \hat{L}_{\mathsf{Rel}_1}) \\ \mathsf{OR}((x_1, w) &\in \mathsf{Rel}_1 \wedge x_0 \in \hat{L}_{\mathsf{Rel}_0})\end{aligned}$$

and for the weaker relation

$$\begin{aligned}\mathsf{Rel}^c_{\mathsf{OR}} = ((x_0, x_1), w) : ((x_0, w) &\in \mathsf{Rel}_0 \wedge x_1 \in L_{\mathsf{Rel}_1}) \\ \mathsf{OR}((x_1, w) &\in \mathsf{Rel}_1 \wedge x_0 \in L_{\mathsf{Rel}_0})\end{aligned}$$

The Adaptive WI experiment, $\mathsf{ExpWI}^\delta_{\mathcal{A}}(x_0, \lambda, \mathsf{aux})$ with $\delta \in \{c, p\}$, is parameterized by PPT adversary $\mathcal{A}$ and has three inputs: instance x_0, security parameter λ, and auxiliary information aux for $\mathcal{A}$.

$\mathsf{ExpWI}^\delta_{\mathcal{A}}(x_0, \lambda, \mathsf{aux})$:

1. $a = P^{\mathsf{OR}}_1(x_0, 1^\lambda; R_1)$, for random coin tosses R_1;
2. $\mathcal{A}(x_0, a, \mathsf{aux})$ outputs $((x_1, w_0, w_1), c, \mathtt{state})$ such that $((x_0, x_1), w_0), ((x_0, x_1), w_1) \in \mathsf{Rel}^\delta_{\mathsf{OR}}$;
3. $b \leftarrow \{0,1\}$;
4. $z \leftarrow P^{\mathsf{OR}}_2((x_0, x_1), w_b, R_1, c)$;
5. $b' \leftarrow \mathcal{A}(z, \mathtt{state})$;
6. If $b = b'$ then output 1 else output 0.

We set

$$\mathsf{Adv}^\delta_{\mathcal{A}}(x_0, \lambda, \mathsf{aux}) = \left| \mathrm{Prob}\left[\ \mathsf{ExpWI}^\delta_{\mathcal{A}}(x_0, \lambda, \mathsf{aux}) = 1\ \right] - \frac{1}{2} \right|$$

Definition 2.4.1. Π^{OR} *is* Adaptive Witness Indistinguishable *(resp.,* Adaptive Perfect Witness Indistinguishable*) if for every adversary $\mathcal{A}$ there exists a negligible function ν such that for all* aux *and* x_0 *it holds that* $\mathsf{Adv}^c_{\mathcal{A}}(x_0, \lambda, \mathsf{aux}) \leq \nu(\lambda)$ *(resp.,* $\mathsf{Adv}^p_{\mathcal{A}}(x_0, \lambda, \mathsf{aux}) = 0$*).*

Next, in Theorem 13, we prove the Adaptive Perfect WI of Π^{OR} when both Π_0 and Π_1 are perfect SHVZK. When one of Π_0 and Π_1 is not perfect, we would like to prove that Π^{OR} is Adaptive WI. In Theorem 14 we prove a weaker form of Adaptive WI in which the adversary is restricted in his choice of witnesses (w_0, w_1) for relation $\mathsf{Rel}^c_{\mathsf{OR}}$.

Theorem 13. If Π_0 and Π_1 are perfect SHVZK then Π^{OR} is Adaptive Perfect Witness Indistinguishable.

Proof. The proof considers the following three cases:

Case 1. $(x_0, w_0) \in \mathsf{Rel}_0$ and $(x_1, w_1) \in \mathsf{Rel}_1$;

Case 2. $(x_1, w_0) \in \mathsf{Rel}_1$ and $(x_1, w_1) \in \mathsf{Rel}_1$;

Case 3. $(x_0, w_0) \in \mathsf{Rel}_0$ and $(x_0, w_1) \in \mathsf{Rel}_0$.

For each case we present a sequence of hybrids and prove that pairs of consecutive hybrids are perfectly indistinguishable.

Case 1. The first hybrid experiment $\mathcal{H}_1(x_0, \lambda, \mathsf{aux})$ is the original experiment $\mathsf{ExpWI}^p_{\mathcal{A}}(x_0, \lambda, \mathsf{aux})$ in which $b = 1$ (and thus $\mathcal{P}$ uses witness w_1). That is,

- In Step 1 of $\mathsf{ExpWI}^p_{\mathcal{A}}(x_0, \lambda, \mathsf{aux})$, the following steps are executed:
    1. $a = P^1_1(1^\lambda; R_1)$, for random coin tosses R_1;
    2. set $a' = a$;
    3. $(\mathtt{com}, \mathtt{dec}, \mathtt{rand}) \leftarrow \mathsf{TCom}_{\Pi_0}(x_0, 1^\lambda, a)$ and outputs $\mathtt{com}$.
- In Step 4 of $\mathsf{ExpWI}^p_{\mathcal{A}}(x_0, \lambda, \mathsf{aux})$, the following steps are executed:

1. $z \leftarrow P_2^1(x_1, w_1, c, R_1)$;
2. set $\mathtt{dec}' = \mathtt{dec}$;
3. output $(\mathtt{dec}', a', z)$.

The second hybrid experiment $\mathcal{H}_2(x_0, \lambda, \mathsf{aux})$ differs from $\mathcal{H}_1(x_0, \lambda, \mathsf{aux})$ in the way a' and $\mathtt{dec}'$ are computed. More specifically,

- Step 1 of $\mathsf{ExpWI}^{p}_{\mathcal{A}}(x_0, \lambda, \mathsf{aux})$ stays the same.

  1. $a = P_1^1(1^\lambda; R^*)$, for random coin tosses R^*;
  2. $a' = P_1^1(1^\lambda; R_1)$, for random coin tosses R_1;
  3. $(\mathtt{com}, \mathtt{dec}, \mathtt{rand}) \leftarrow \mathsf{TCom}_{\Pi_0}(x_0, 1^\lambda, a)$ and outputs $\mathtt{com}$.

- In Step 4 of $\mathsf{ExpWI}^{p}_{\mathcal{A}}(x_0, \lambda, \mathsf{aux})$, the following steps are executed:

  1. $z \leftarrow P_2^1(x_1, w_1, c, R_1)$;
  2. $\mathtt{dec}' \leftarrow \mathsf{TFake}_{\Pi_0}(x_0, w_0, a, a', \mathtt{com}, \mathtt{rand})$;
  3. $(\mathtt{dec}', a', z)$.

The trapdoorness of the instance-dependent trapdoor commitment scheme based on Π_0 and perfect WI of Π_0 guarantee that $\mathcal{H}_1(x_0, \lambda, \mathsf{aux})$ and $\mathcal{H}_2(x_0, \lambda, \mathsf{aux})$ are perfectly indistinguishable for all λ.

The third hybrid experiment $\mathcal{H}_3(x_0, \lambda, \mathsf{aux})$ differs from $\mathcal{H}_2(x_0, \lambda, \mathsf{aux})$ in the way a' and z are computed. More specifically,

- Step 1 of $\mathsf{ExpWI}^{p}_{\mathcal{A}}(x_0, \lambda, \mathsf{aux})$ stays the same.

  1. $a = P_1^1(1^\lambda; R_1)$, for random coin tosses R_1;
  2. $(\mathtt{com}, \mathtt{dec}, \mathtt{rand}) \leftarrow \mathsf{TCom}_{\Pi_0}(x_0, 1^\lambda, a)$ and outputs $\mathtt{com}$.

- In Step 4 of $\mathsf{ExpWI}^{p}_{\mathcal{A}}(x_0, \lambda, \mathsf{aux})$, the following steps are executed:

  1. $(a', z) \leftarrow \mathsf{Sim}^1(x_1, c)$;
  2. $\mathtt{dec}' \leftarrow \mathsf{TFake}_{\Pi_0}(x_0, w_0, a, a', \mathtt{com}, \mathtt{rand})$;

3. $(\mathtt{dec}', a', z)$.

By the perfect SHVZK of Π_1, we have that $\mathcal{H}_2(x_0, \lambda, \mathsf{aux})$ and $\mathcal{H}_3(x_0, \lambda, \mathsf{aux})$ are perfectly indistinguishable for all λ. The proof ends with the observation that $\mathcal{H}_3(x_0, \lambda, \mathsf{aux})$ is exactly experiment $\mathsf{ExpWI}^{p}_{\mathcal{A}}(x_0, \lambda, \mathsf{aux})$ when $b = 0$.

Case 2. The first hybrid experiment $\mathcal{H}_1(x_0, \lambda, \mathsf{aux})$ is again the original experiment $\mathsf{ExpWI}^{p}_{\mathcal{A}}(x_0, \mathsf{aux})$ in which $b = 1$ (and thus $\mathcal{P}$ uses witness w_1). The second hybrid experiment $\mathcal{H}_2(x_0, \lambda, \mathsf{aux})$ differs from $\mathcal{H}_1(x_0, \lambda, \mathsf{aux})$ in the way z is computed (using as input w_0 instead of w_1 when P_2 is executed). More specifically,

- In Step 1 of $\mathsf{ExpWI}^{p}_{\mathcal{A}}(x_0, \lambda, \mathsf{aux})$, the following steps are executed:
  1. $a = P^1_1(1^\lambda; R_1)$, for random coin tosses R_1;
  2. $(\mathtt{com}, \mathtt{dec}, \mathtt{rand}) \leftarrow \mathsf{TCom}_{\Pi_0}(x_0, 1^\lambda, a)$ and outputs $\mathtt{com}$.
- In Step 4 of $\mathsf{ExpWI}^{p}_{\mathcal{A}}(x_0, \lambda, \mathsf{aux})$, the following steps are executed:
  1. $z \leftarrow P^1_2(x_1, w_0, c, R_1)$;
  2. output $(\mathtt{dec}, a, z)$

The Perfect $\mathcal{WI}$ property of Π_1 implies that $\mathcal{H}_1(x_0, \lambda, \mathsf{aux})$ is perfectly indistinguishable from $\mathcal{H}_2(x_0, \lambda, \mathsf{aux})$. The proof ends with the observation that $\mathcal{H}_2(x_0, \lambda, \mathsf{aux})$ is exactly the experiment $\mathsf{ExpWI}^{p}_{\mathcal{A}}(x_0, \lambda, \mathsf{aux})$ when $b = 0$. $\square$

Case 3. The first hybrid experiment $\mathcal{H}_1(x_0, \lambda, \mathsf{aux})$ is again the original experiment $\mathsf{ExpWI}^{p}_{\mathcal{A}}(x_0, \lambda, \mathsf{aux})$ in which $b = 1$ (and thus $\mathcal{P}$ uses witness w_1). The second hybrid experiment $\mathcal{H}_2(x_0, \lambda, \mathsf{aux})$ differs from $\mathcal{H}_1(x_0, \lambda, \mathsf{aux})$ in the way TFake is executed (namely, using as input w_0 instead of w_1). More specifically,

- Step 1 of $\mathsf{ExpWI}^{p}_{\mathcal{A}}(x_0, \lambda, \mathsf{aux})$ stays the same.

1. $a = P_1^1(1^\lambda; R_1)$, for random coin tosses R_1;
2. $(\mathtt{com}, \mathtt{dec}, \mathtt{rand}) \leftarrow \mathsf{TCom}_{\Pi_0}(x_0, 1^\lambda, a)$ and outputs $\mathtt{com}$.

- In Step 4 of $\mathsf{ExpWI}^p_{\mathcal{A}}(x_0, \lambda, \mathsf{aux})$, the following steps are executed:

1. $(a', z) \leftarrow \mathsf{Sim}^1(x_1, c)$;
2. $\mathtt{dec}' \leftarrow \mathsf{TFake}_{\Pi_0}(x_0, w_0, a, a', \mathtt{com}, \mathtt{rand})$;
3. $(\mathtt{dec}', a', z)$.

Note that Π_0 is perfect SHVZK which implies that Π_0 is perfect WI and from Theorems 7 and 10 we can conclude that the IDTC based on Π_0 is also WI-IDTC. Therefore the WI-Trapdorness of the WI-IDTC scheme based on Π_0 implies that $\mathcal{H}_1(x_0, \lambda\mathsf{aux})$ is perfectly indistinguishable from $\mathcal{H}_2(x_0, \lambda\mathsf{aux})$ for all λ. The proof ends with the observation that $\mathcal{H}_2(x_0, \lambda, \mathsf{aux})$ is exactly experiment $\mathsf{ExpWI}^p_{\mathcal{A}}(x_0, \lambda, \mathsf{aux})$ when $b = 0$.

Next we consider the computational case in which one of Π_0 and Π_1 is not Perfect SHVZK (but they are both SHVZK).

Theorem 14. If Π_0 and Π_1 are SHVZK then Π^{OR} is Adaptive Witness Indistinguishable with respect to adversaries that output (x_1, w_0, w_1) such that at least one of w_0 and w_1 is a witness for $x_1 \in L_{\mathsf{Rel}_1}$.

Proof. We prove this theorem by considering the following two cases:

Case 1. $(x_0, w_0) \in \mathsf{Rel}_0$ and $(x_1, w_1) \in \mathsf{Rel}_1$;

Case 2. $(x_1, w_0) \in \mathsf{Rel}_1$ and $(x_1, w_1) \in \mathsf{Rel}_1$.

Case 1. In this case the proof follows closely the one of Case 1 of Theorem 13, with the difference that hybrids here are only computationally indistinguishable.

Case 2. In this case we show that there exists $\mathcal{A}'$ for Case 1 that has the same success probability of $\mathcal{A}$. Suppose indeed that both w_0 and w_1 are witnesses for x_1 and that $\mathcal{A}$ breaks the Adaptive WI property of Π^{OR}. Then, by definition of $\mathsf{Rel}^c_{\mathsf{OR}}$ and by Definition 2.4.1, there exists $\mathcal{A}'$ that has in his description a witness w_2 for x_0. Indeed, the output of $\mathcal{A}$ interacting with $\mathcal{P}((x_0, x_1), w_2)$ would necessarily be distinguishable from the output of the interaction with either $\mathcal{P}((x_0, x_1), w_0)$ or $\mathcal{P}((x_0, x_1), w_1)$. Therefore $\mathcal{A}'$ would contradict Case 1 and thus there exists no successful $\mathcal{A}$ for Case 2. □

In Theorem 14 we force the adversary of the Adaptive WI to output two witnesses w_0 and w_1, where at least one of w_0 and w_1 is a witness for instance $x_1 \in L_{\mathsf{Rel}_1}$. In fact we can not demonstrate the Adaptive WI of Π^{OR} when both witnesses are for instance $x_0 \in L_{\mathsf{Rel}_0}$, because Π_0 is SHVZK, so we are not guarantee that is WI (Theorem 20). Therefore we can not rely on this property to demonstrate that Π^{OR} is adaptive WI. Furthermore, we can not use the same idea of Case 2 of Theorem 14 because the instance x_1 is adaptively chosen. However, using the transformation described in Section 2.2 we can obtain from Π_0 a Σ-protocol $\overline{\Pi}_0$ for Rel_0 that is WI (note that in Case 3 we are guaranted that $x_0 \in L_{\mathsf{Rel}_0}$). We can use $\overline{\Pi}_0$ to construct a WI-IDTC following Theorems 7 and 10. Finally we will apply our OR-transformation on the WI-IDTC constructed from $\overline{\Pi}_0$ and Π_1 obtaining an OR-composition Π^{OR} that is adaptive WI with no restriction. In more details, we have the following theorem.

Theorem 15. If $\overline{\Pi}_0$ and Π_1 are SHVZK then Π^{OR} is Adaptive Witness Indistinguishable.

Proof. We prove this theorem by considering the following three cases:

Case 1 $(x_0, w_0) \in \mathsf{Rel}_0$ and $(x_1, w_1) \in \mathsf{Rel}_1$;

Case 2 $(x_1, w_0) \in \mathsf{Rel}_1$ and $(x_1, w_1) \in \mathsf{Rel}_1$;

Case 3 $(x_0, w_0) \in \mathsf{Rel}_0$ and $(x_0, w_1) \in \mathsf{Rel}_0$.

Case 1 and Case 2. For this two cases the proofs are the same, respectively, of the Case 1 and Case 2. of Theorem 14.

Case 3. In this case the proof follows closely the one of Case 3 of Theorem 13, with the difference that hybrid experiments ($\mathcal{H}_1(x_0, \lambda, \mathsf{aux})$ and $\mathcal{H}_2(x_0, \lambda, \mathsf{aux})$) are only computationally indistinguishable. In more details if it exists a distinguisher between these hybrid experiments we can made a reduction to the WI-Trapdorness of WI-IDTC construct from $\overline{\Pi}_0$. The reduction works as a proxy between the malicious verifier $\mathcal{V}^\star$ and the challenger of the WI of Π ($\mathcal{CH}$) w.r.t. the messages of the WI-IDTC and for all the other messages he follows the steps described in $\mathcal{H}_1(x_0, \lambda, \mathsf{aux})$ (that are equal to the steps of $\mathcal{H}_2(x_0, \lambda, \mathsf{aux})$). More in details, the reduction obtains a commit $\mathtt{com}$ of a random message from $\mathcal{CH}$ that he sends to $\mathcal{V}^\star$. Then upon receiving x_1, c, w_0, w_1 from $\mathcal{V}^\star$ the reduction computes the step 3 of $\mathcal{H}_1$ obtaining (a', z) and forwards a' and the witnesses w_0, w_1 to $\mathcal{CH}$. Upon receiving $\mathtt{dec}$ from $\mathcal{CH}$ sent $\mathtt{dec}, a', z$ to $\mathcal{V}^\star$, and outputs the output of $\mathcal{V}^\star$. It easy to see that if $\mathcal{CH}$ uses the witness w_1 the reduction acts as described in $\mathcal{H}_1$, otherwise acts as described in $\mathcal{H}_2$. This observation conclude the proof. □

2.5 Applications

In this section, we describe the application of our new OR-composition technique for constructing a 3-round straight-line perfect quasi-polynomial time simulatable argument system, an efficient 4-round resettable WI argument system and an efficient 4-round resettable zero knowledge with concurrent soundness argument system in the BPK model. For the last application we provide only an informal protocol description.

2.5.1 A 3-Round Efficient Perfectly Simulatable Argument System

In [Pas03], Pass introduced relaxed notion of zero knowledge and knowledge extraction in which the simulator and the extractor are allowed to run in quasi-polynomial time. Allowing the simulator to run in quasi-polynomial time typically dispenses with the need of rewinding the verifier; that is, the simulator is *straight-line*. In [Pas03], Pass first describes the following 2-round perfect ZK argument for any language L. The verifier $\mathcal{V}$ sends a value $Y = f(y)$ for a randomly chosen y where f is a sub-exponentially hard OWF and the first round of a ZAP protocol. The prover $\mathcal{P}$ then sends a commitment to $(y'|w')$ and uses the second round of the ZAP to prove that either $y' = f^{-1}(y)$ or w' is a witness for $x \in L$. If language L admits a Σ-protocol Π_L then the above construction can be implemented as an efficient 4-round argument with quasi-polynomial time simulation. The function f is concretely instantiated to be an exponentiation in a group in which the Discrete Log problem is hard and the ZAP is replaced with the CDS-OR composition of Π_L and Schnorr's Σ-protocol for the Discrete Log.

Note that Schnorr's Σ-protocol is delayed input and thus we can use it as Σ-protocol Π_1 in our OR transform in conjunction with any Chameleon Σ-protocol Π_0. One drawback of reducing to three round the result of [Pas03] is that we can use only a perfect Σ-protocol since our goal is to obtain perfect WI in just three rounds.

2.5.1.1 Preliminary Definitions

We start by providing some useful definitions.

Simulation in quasi-polynomial time. Since the verifier in an interactive argument is often modeled as a PPT machine, the classical zero-knowledge definition requires that the simulator runs also in (expected) polynomial time. In [Pas03], the simulator is allowed to run in time $\lambda^{\mathtt{poly}(\mathtt{log}(\lambda))}$. Loosely speaking, we say that an interactive argument is $\lambda^{\mathtt{poly}(\mathtt{log}(\lambda))}$-perfectly simulatable if for any adversarial verifier there exists a simulator running in time $\lambda^{\mathtt{poly}(\mathtt{log}(\lambda))}$, where λ is the size of the statement being proved, whose output is identically distributed to the output of the adversarial verifier.

Definition 2.5.1 (One-way functions for sub-exponential circuits. [Pas03]). *A function $f : \{0,1\}^* \rightarrow \{0,1\}^*$ is called one-way for sub-exponential circuits if there exists a constant α such that the following two condition holds:*

- *there exist a deterministic polynomial-time algorithm that on input y outputs $f(y)$;*
- *for every probabilistic algorithm $\mathcal{A}$ with running time bounded by 2^{λ^α}, all sufficiently large λ's, and every auxiliary input $z \in \{0,1\}^{\mathsf{poly}(\lambda)}$*

$$\text{Prob}\left[\, y \overset{R}{\leftarrow} \{0,1\}^* : \mathcal{A}(f(y), z) \in f^{-1}(f(y)) \,\right] < \frac{1}{\mathsf{poly}(2^{\lambda^\alpha})}.$$

Now we define straight-line $T(\lambda)$-perfectly simulatable interactive arguments.

For our result we consider a one-way functions for sub-exponential circuits that has an efficiently recognizable range (as in [Pas03])

Definition 2.5.2 (straight-line $T(\lambda)$ simulatability, Def. 31 of [Pas04a]). *Let $T(\lambda)$ be a class of functions that is closed under composition with any polynomial. We say that an interactive argument (proof) $(\mathcal{P}, \mathcal{V})$ for the language $L \in \mathcal{NP}$, with the witness relation Rel_L, is straight-line $T(\lambda)$-simulatable if for every PPT machine $\mathcal{V}^\star$ there exists a probabilistic simulator S with running time bounded by $T(\lambda)$ such that the following two ensembles are computationally indistinguishable (when the distinguish gap is a function in $\lambda = |x|$)*

- $\{(\langle \mathcal{P}(w), \mathcal{V}^\star(z)\rangle(x))\}_{z\in\{0,1\}^*, x\in L}$ *for arbitrary w s.t. $(x,w) \in \mathsf{Rel}_L$*
- $\{(\langle S, \mathcal{V}^\star(z)\rangle(x))\}_{z\in\{0,1\}^*, x\in L}$

We note that the above definition is very restrictive. In fact, the simulator is supposed to act as a cheating prover, with its only advantage being the possibility of running in time $T(\lambda)$, instead of in polynomial time. Trivially, it does not exist a straight-line $T(\lambda)$-simulatable proof for non-trivial languages (this should be contrasted with straight-line simulatable interactive arguments, which instead does exist).

The following theorem shows the importance of straight-line $\lambda^{\mathtt{poly(log(\lambda))}}$-perfect simulatability by connecting it to concurrent composition of arguments.

Theorem 16. If the interactive argument $\Pi = (\mathcal{P}, \mathcal{V})$ is straight-line $\lambda^{\mathtt{poly(log(\lambda))}}$-simulatable then it is also straight-line concurrent $\lambda^{\mathtt{poly(log(\lambda))}}$-simulatable.

2.5.1.2 The Protocol

For any $\mathcal{NP}$-language L we consider the perfect chameleon Σ-protocol Π_L for the relation R_L. Also we consider the Schnorr Σ-protocol Π^{DLOG} for the following relation $\mathsf{DLOG} = \{((\mathcal{G}, q, g, Y), y) : g^y = Y\}$ with the associated $\mathcal{NP}$-language L_{DLOG}, over groups $\mathcal{G}$ of prime-order q, and use our OR-composition technique to obtain a new perfect Σ-protocol $\Pi^{\mathsf{OR}} = (\mathcal{P}^{\mathsf{OR}}, \mathcal{V}^{\mathsf{OR}})$ for the relation $\mathsf{Rel}_{\mathsf{OR}} = \{((x_L, x_{\mathsf{DLOG}}), w) : ((x_L, w) \in \mathsf{Rel}_L \wedge x_{\mathsf{DLOG}} \in \hat{L}_{\mathsf{DLOG}})\mathsf{OR}$ $((x_{\mathsf{DLOG}}, w) \in \mathsf{DLOG} \wedge x_L \in \hat{L}_{\mathsf{Rel}_L})\}$ with challenge length $l = \lambda$ and associated algorithms P_1^{OR}, P_2^{OR} and V^{OR}.

Let f be a sub-exponentially hard one-way function that has an efficiently recognizable range implemented using DLog as described before, with the only change that for some constant α s.t. $0 < \lambda < 1$, f is one-way w.r.t. circuits of size 2^{λ^α}. Let $L \in \mathcal{NP}$ and $k = \frac{1}{\alpha} + 1$. Our 3-round straight-line quasi-polynomial time simulatable argument system for $x \in L$ is the following.

Protocol 5. *A 3-round straight-line quasi-polynomial time simulatable argument system.*

Common input: An instance x of a language $L \in \mathcal{NP}$ with witness relation R_L and a perfect chameleon Σ-protocol, and 1^λ as security parameter.

Private input: $\mathcal{P}$ has w as a private input, s.t. $(x, w) \in \mathsf{Rel}_L$.
Round 1. $\mathcal{P} \to \mathcal{V}$:

1. On input a randomness R_1, $\mathcal{P}$ uniformly chooses (p, q, g) where $p = 2q + 1$ is a safe prime and g is a generator of a group $\mathcal{G}_q$ of size q. We remark that (p, q, g) are parameters selected so that the function $f(y) = g^y$ is a one-way function for some constant α w.r.t circuits of size 2^{λ^α}.
2. $\mathcal{P}$ computes $a \leftarrow P_1^{\mathsf{OR}}((x, 1^{\lambda^\alpha}); R_1)$.
3. $\mathcal{P}$ sends (p, q, g) and a to $\mathcal{V}$.

Round 2. $\mathcal{V} \to \mathcal{P}$:

1. $\mathcal{V}$ chooses $y \leftarrow \mathbb{Z}_q$ and computes $Y = g^y$.
2. $\mathcal{V}$ chooses $c \leftarrow \{0, 1\}^l$.
3. $\mathcal{V}$ sends c and Y to $\mathcal{P}$.

Round 3. $\mathcal{P} \to \mathcal{V}$:

1. $\mathcal{P}$ computes $z \leftarrow P_2^{\mathsf{OR}}((x, ((p, q, g), Y)), w, c, R_1)$.
2. $\mathcal{P}$ sends z to $\mathcal{V}$.
3. $\mathcal{V}$ accepts if and only if $\mathsf{V}^{\mathsf{OR}}((x, ((p, q, g), Y)), a, c, z) = 1$.

We remark that we are using the same assumption of [CD08] that allows the adversary of DLog to generate the DLog parameters while the challenger selects the random element of the group.

Theorem 17. If Π^{OR} is a perfect Σ-protocol for OR composition of Rel_L and DLOG, then Protocol 5 is a 3-round straight-line perfectly $\lambda^{O(\log^k \lambda)}$-simulatable argument of knowledge.

Proof. Completeness follows directly from the completeness of Π^{OR}.

Soundness/knowledge extraction. We show that Π is an argument of knowledge; this directly implies soundness. The claim follows from the fact that the argument system Π^{OR} used is a proof of knowledge when the challenge is long enough, and from the fact that a PPT adversary only finds a pre-image to Y (for f) with negligible probability. More formally, we construct a polynomial-time extractor E for every polynomial-time $\mathcal{P}^\star$ for protocol Π. E internally incorporates $\mathcal{P}^\star$ and each time Π^{OR} proves a new theorem it proceeds as follows. E invokes the extractor E^{OR} for Π^{OR}. E outputs whatever E^{OR} outputs. By the proof knowledge property of Π^{OR}, the output of E will either be a witness w for the statement proved, or the pre-image of Y. If E outputs w, we are done. Otherwise, if it outputs y with non-negligible probability, then we can construct a reduction that breaks the DLog assumption (still in the form proposed by [CD08]).

Quasi-polynomial time perfect simulation. Consider a straight-line simulator Sim that computes the first round as the honest prover. This is possible because Π^{OR} does not need any witness to computes the first round. After the simulator receives Y it checks that Y has a pre-image. Sim thereafter performs an exhaustive search to find a pre-image y of a value Y for the function f. To perform this task Sim tries all possible values $y' \in \{0,1\}^{\log^k \lambda}$ and checks if $f(y') = Y$. This thus takes time $\mathsf{poly}(2^{\log^k \lambda})$, since the time it takes to evaluate the function f is a polynomial in λ. After having found a value y such that $f(y) = Y$, Sim uses y as witness to complete the execution of Π^{OR} (instead of using a real witness for x, as the honest prover would do). Clearly the running time of Sim is bounded by $\lambda^{O(\log^k \lambda)}$. We proceed to show that the output of the simulator is identically distributed to the output of any adversarial verifier in a real execution with an honest prover. Note that the only difference between a real execution and a simulated execution is in the choice of the witness used in the last stage of the protocol. Therefore, from the perfect adaptive WI property of Π^{OR} we have that the output of the simulated execution is identically distributed to the output of the real execution. $\square$

2.5.2 Efficient Resettable WI Argument System

In this section, we show how to efficiently transform *any* Σ-protocol Π into a resettable WI ($\mathsf{r}\mathcal{WI}$) argument system at the cost of adding only one extra round and a constant number of modular exponentiations.

Resettable witness indistinguishability was introduced in [CGGM00]. Very roughly, a resetting verifier is a PPT adversary that is able to interact with the prover polynomially many times with possibly distinct inputs forcing the prover to execute the protocol using the same randomness several times. Namely, we can think of a prover under a reset attack as being equipped with a vector of inputs $\bar{x}$ and a vector of random tapes $\vec{\omega}$. The malicious verifier can adaptively start a new interaction with the prover by specifying the input x_i and the randomness ω_j to be used in the interaction. Moreover, the malicious verifier has complete control over the schedule of the messages. A concurrent verifier is a restricted form of resetting verifier that cannot start two interactions with the same randomness. A formal definition is found in [CGGM00].

An informal description. Let Rel be a polynomial-time relation with Σ-protocol Π. Now we give an high-level overview of our 4-round $\mathsf{r}\mathcal{WI}$ argument system $\Pi^{\mathcal{WI}} = (\mathcal{P}, \mathcal{V})$ for Rel. The basic idea is to have $\mathcal{V}$ that commits to the challenge of Π in the first round of $\Pi^{\mathcal{WI}}$. Subsequently, $\mathcal{P}$ and $\mathcal{V}$ play Σ-protocol Π where at the second round (corresponding to the 3 round of $\Pi^{\mathcal{WI}}$), $\mathcal{V}$ decommits to the challenge. To prove $\mathsf{r}\mathcal{WI}$ we need that the commitment computed in the first round to be perfectly binding for $\mathcal{V}^\star$ so that reset attacks are ineffective, then we can demonstrate the $\mathsf{r}\mathcal{WI}$ of $\Pi^{\mathcal{WI}}$ relying on the WI of Π. Instead, to prove soundness against $\mathcal{P}^\star$ we need that the commitment computed in the first round enjoys the property of trapdoorness. In this way we can rewind $\mathcal{P}^\star$ and using the special soundness of Π to reach a contradiction.

In order to implement this aspect we use a perfectly binding

2-IDTC for the following relation:

$$\mathsf{DDH} = \Big\{ \big(((\mathcal{G}, q, g), A = g^a, B = g^b, C = g^{ab}), b \big) : A^b = C \Big\}.$$

over groups $\mathcal{G}$ of prime-order q.

The perfectly binding 2-IDTC is perfectly binding when an instance T is s.t. $T \notin L_{\mathsf{DDH}}$, and it enjoys the property of perfect trapdoorness when $T \in L_{\mathsf{DDH}}$. Therefore we need to ensure that $\mathcal{V}^\star$ to compute the commitment of the challenge can not choose an instance T s.t. $T \in L_{\mathsf{DDH}}$. For this reason we add a protocol $\Pi^{1-\mathsf{DDH}}$ to ensure that $T \notin T_{\mathsf{DDH}}$. In more details, the protocol $\Pi^{1-\mathsf{DDH}}$ proves that T has the following form $T = ((\mathcal{G}, q, g), A = g^a, B = g^b, C = g^{ab+1})$, and this implies that $T \notin L_{\mathsf{DDH}}$. On the other hand in the proof of soundness we want to cheat, in order to use the trapdoorness of perfectly binding 2-IDTC. We could use our OR-composition technique combining $\Pi^{1-\mathsf{DDH}}$ with Schnorr's protocol, obtaining Π^{OR} that $T \notin L_{\mathsf{DDH}}$ or it knows the DLog of a challenge sent by the receiver. Note that our OR-composition is crucial because the instance for the Schnorr's protocol is available only in the second round of $\Pi^{\mathcal{WI}}$. In the proof of soundness we can break the DLog challenge running in super-polynomial time therefore succeeding in equivocate the commitment in order to rewinding the malicious prover $\mathcal{P}^\star$. Rewinding $\mathcal{P}^\star$ we obtain a collision for a false statement x and therefore by the special soundness of Π, it is possible to extract a witness w for s.t $(x, w) \in \mathsf{Rel}_L$ and reaching a contradiction. Instead $\mathcal{V}^\star$ will not be able to decommit to a different message since otherwise we could use $\mathcal{V}^\star$ to break in polynomial time the DLog challenge.

Formal description of our 4-round efficient resettable WI. Let Π^{DLOG} be an delayed-input Σ-protocol for the discrete log polynomial-time relation

$$\mathsf{DLOG} = \{((\mathcal{G}, q, g, Y), y) : g^y = Y\}$$

over groups $\mathcal{G}$ of prime-order q; e.g., Schnorr's Σ-protocol [Sch89].

Consider also the following polynomial-time relation

$$\mathsf{1DDH} = \Big\{\big(((\mathcal{G}, q, g), A = g^a, B = g^b, C = g^{ab+1}), b{+}1\big) : A^{b+1} = C\Big\}.$$

over groups $\mathcal{G}$ of prime-order q. We will refer to a tuple $T = \big((\mathcal{G}, q, g), A = g^a, B = g^b, C = g^{ab}\big)$ as DH tuple and $T' = \big((\mathcal{G}, q, g), A = g^a, B = g^b, C = g^{ab+1}\big)$ a 1-non-DH tuple.

Let Π^{DDH} be a Chameleon Σ-protocol for DDH ([Ped91]), and the Chameleon Σ-protocol Π^{1DDH} for $\mathsf{1DDH}$ described in Section 2.7.2.

From Theorem 8 follows that we can use Π^{DDH} to build a perfectly binding 2-IDTC scheme $(\mathsf{TCom}_{\Pi^{\mathsf{DDH}}}, \mathsf{TDec}_{\Pi^{\mathsf{DDH}}}, \mathsf{TFake}_{\Pi^{\mathsf{DDH}}})$ for DDH.

Note that Π^{1DDH} is Chameleon Σ-protocol and Π^{DLOG} is delayed-input Σ-protocol, therefore we can apply Theorem 12 and construct the protocol Π^{OR}, with the associated algorithms $(P_1^{\mathsf{OR}}, P_2^{\mathsf{OR}}, \mathsf{V}^{\mathsf{OR}})$, for the following polynomial-time relation

$$\mathsf{Rel}_{\mathsf{OR}} = \Big\{((x_0, x_1), w) : \big((x_0, w) \in \mathsf{DLOG} \wedge x_1 \in \hat{L}_{\mathsf{1DDH}}\big)\mathsf{OR}$$
$$\big((x_1, w) \in \mathsf{1DDH} \wedge x_0 \in \hat{L}_{\mathsf{DLOG}}\big)\Big\}.$$

We denote by x the common input and by w the witness received by $\mathcal{P}$ s.t. $(x, w) \in \mathsf{Rel}$. We also consider a Σ-protocol Π, with the associated algorithm (P_1, P_2, V) for the polynomial-time relation Rel.

Security parameters are λ, $\hat{\lambda}$. Given the security parameter λ we choose $\hat{\lambda}$ s.t. $2^{\hat{\lambda}} < 2^{\lambda} \cdot \mathsf{poly}(\lambda)$, where an adversary that runs in time $2^{\hat{\lambda}}$ can compute the discrete log of an element $Y \in \mathbb{Z}_{\hat{q}}$ and the 1DDH assumption still holds against adversary that running in time $2^{\hat{\lambda}}$.

The security of our proposed $\mathsf{r}\mathcal{WI}$ $\Pi^{\mathcal{WI}} = (\mathcal{P}, \mathcal{V})$ for Rel is based on the 1DDH assumption and on a strengthening of it (see Assumption 5).

Protocol 6.
Public input: instance x and security parameter 1^{λ}.
Private input to $\mathcal{P}$: witness w such $(x, w) \in \mathsf{Rel}$.
Round 1: from $\mathcal{V}$ to $\mathcal{P}$.

1.1 Committing to the challenge c of Π

$\mathcal{V}$ randomly selects $(\mathcal{G}, q, g) \leftarrow \mathsf{IG}(1^\lambda)$ and $(\hat{\mathcal{G}}, \hat{q}, \hat{g}) \leftarrow \mathsf{IG}(1^{\hat{\lambda}})$ and sends them to $\mathcal{P}$;

$\mathcal{V}$ randomly selects $r_1, r_2, \leftarrow \mathbb{Z}_q$, computes $A = g^{r_1}, B = g^{r_2}, C = A^{r_2} \cdot g$ and sets $T = ((\mathcal{G}, q, g), A, B, C)$;

$\mathcal{V}$ randomly selects $c \leftarrow \mathbb{Z}_q$ and computes $(\mathtt{com}, \mathtt{dec}, \mathtt{rand}) \leftarrow \mathsf{TCom}_{\Pi^{\mathsf{DDH}}}(1^\lambda, T, c)$;

$\mathcal{V}$ sends $((\hat{\mathcal{G}}, \hat{q}, \hat{g}), T, \mathtt{com})$ to $\mathcal{P}$;

1.2 Computing first round of Π^{OR}.

$\mathcal{V}$ randomly selects coin tosses R_1, sets $a^{\mathsf{OR}} = P_1^{\mathsf{OR}}(T; R_1)$ and sends a^{OR} to $\mathcal{P}$;

Round 2: from $\mathcal{P}$ to $\mathcal{V}$.

2.1 Picking randomness.

$\mathcal{P}$ reads ω from its random tape, computes $\bar{R} = F_\omega(x, T, (\hat{\mathcal{G}}, \hat{q}, \hat{g}), \mathtt{com}, a^{\mathsf{OR}})$ and parses it as $\bar{R} = (R_2, R_3)$;

2.2 Computing first round of Π.

$\mathcal{P}$ sets $a = P_1(x, w; R_2)$ and sends it to $\mathcal{V}$;

2.3 Preparing second input for Π^{OR} and sending the challenge for Π^{OR}.

$\mathcal{P}$ randomly selects $y \leftarrow \mathbb{Z}_{\hat{q}}$, sets $Y = \hat{g}^y$ and sends it to $\mathcal{V}$;

$\mathcal{P}$ randomly selects $c^{\mathsf{OR}} \leftarrow \{0,1\}^{l^{\mathsf{OR}}}$ and sends it to $\mathcal{V}$;

Round 3: from $\mathcal{V}$ to $\mathcal{P}$.

3.1 Opening commitment of challenge for Π.

$\mathcal{V}$ sends $(c, \mathtt{dec})$ to $\mathcal{P}$;

3.2 Computing third round of Π^{OR}.

$\mathcal{V}$ randomly selects coin tosses R_4 and sets $w^{\mathsf{OR}} = r_2$ and computes and sends $z^{\mathsf{OR}} = P_2^{\mathsf{OR}}((T, (\hat{\mathcal{G}}, \hat{q}, \hat{g}, Y), c^{\mathsf{OR}}, w^{\mathsf{OR}}, R_1; R_4)$;

Round 4: from $\mathcal{P}$ to $\mathcal{V}$.

4.1 Verification of Π^{OR}.

If $\mathsf{V}^{\mathsf{OR}}((T,((\hat{\mathcal{G}},\hat{q},\hat{g},Y)),a^{\mathsf{OR}},c^{\mathsf{OR}},z^{\mathsf{OR}})=0$, $\mathcal{P}$ aborts;

4.2 Checking the decommitment.

If $\mathsf{TDec}_{\Pi^{\mathsf{DDH}}}(T,\mathtt{com},\mathtt{dec},c)=0$ then $\mathcal{P}$ aborts;

4.3 Compute third round of Π.

$\mathcal{P}$ computes $z=P_2(x,w,c,R_2;R_3)$ and sends it to $\mathcal{V}$;

$\mathcal{V}$'s decision: $\mathcal{V}$ accepts if and only if $\mathsf{V}(x,a,c,z)=1$.

Theorem 18. If polynomial-time relation Rel admits a Σ-protocol Π then, under the DDH assumption against sub-exponential adversary, $\Pi^{\mathcal{WI}}=(\mathcal{P},\mathcal{V})$ is an argument system $\mathsf{r}\mathcal{WI}$ for Rel.

Completeness is straightforward.

Soundness. Let $\mathcal{P}^\star$ be an efficient algorithm that starts $\mathsf{poly}(\lambda)$ concurrent interactions with an honest verifier $\mathcal{V}$. We show that for any common input $x\in\{0,1\}^\lambda$, if $\mathcal{P}^\star$, for $x\notin L_{\mathsf{Rel}}$, makes the honest verifier $\mathcal{V}$ accept with non-negligible probability in one of the $\mathsf{poly}(\lambda)$ concurrent interactions, then we can reach a contradiction using the special soundness of Π.

In order to do that we consider the following four hybrid experiments $\mathcal{H}_0,\mathcal{H}_1,\mathcal{H}_2$ and $\mathcal{H}_3$.

The first hybrid experiment $\mathcal{H}_0(\mathtt{aux})$ is simply the real game between $\mathcal{P}^\star$ and the honest verifier $\mathcal{V}$. Note that in $\mathcal{H}_0(\mathtt{aux})$ $\mathcal{P}^\star$ can convince $\mathcal{V}$ of a false statement, $x\notin L_{\mathsf{Rel}}$, in one of the $\mathsf{poly}(\lambda)$ concurrent interactions, it comes from the assumption.

In the second hybrid experiment, $\mathcal{H}_1(\mathtt{aux})$, $\mathcal{P}^\star$ interacts in $\mathsf{poly}(\lambda)$ concurrent interactions with a verifier $\mathcal{V}_1$ that, for every interactions, computes the following steps. Upon receiving Y, $\mathcal{V}_1$, running in time at most $2^{\hat{\lambda}}$, computes the discrete logarithm y of Y (i.e. $Y=g^y$). Then, $\mathcal{V}_1$ sets $w^{\mathsf{OR}}=y$ at Step 3.2. Therefore, $\mathcal{V}_1$ uses y instead of r_2 to compute message z^{OR}. $\mathcal{H}_0(\mathtt{aux})$ and $\mathcal{H}_1(\mathtt{aux})$ are statistical indistinguishable because of the perfect adaptive WI of Π^{OR}, it still holds that $\mathcal{P}^\star$, for $x \notin L_{\mathsf{Rel}}$, makes the honest verifier $\mathcal{V}$ accept with non-negligible probability in one of the $\mathsf{poly}(\lambda)$ concurrent interactions. Therefore there exists a portion non-negligible of randomness used by $\mathcal{P}^\star$ for which he declares an x s.t. $x \notin L_{\mathsf{Rel}}$ in one of the $\mathsf{poly}(\lambda)$ concurrent interactions, and he makes the honest verifier accept with non-negligible probability in that interaction. Let `i` be the index of this interaction. Furthermore let `rand` be a prefix of the randomness for which $\mathcal{P}^\star$ declares an x s.t. $x \notin L_{\mathsf{Rel}}$ in the `i`-th interaction.

The next hybrid $\mathcal{H}_2$ takes as auxiliary input (`aux`) the prefix of randomness `rand` and the index `i`. The verifier $\mathcal{V}_2$ interacts with $\mathcal{P}^\star$ that is executed using the randomness that have as a prefix `rand`. Until the interaction with index `i` $\mathcal{V}_2$ acts as $\mathcal{V}_1$. In the `i`-th interaction $\mathcal{V}_2$ modifies the Step 1.1 choosing T as a DH-tuple. In more details, in the Step 1.1 of the `i`-th interaction $\mathcal{V}_2$ picks at random $r_1, r_2, \leftarrow \mathbb{Z}_q$, computes $A = g^{r_1}, B = g^{r_2}, C = A^{r_2}$ and sets $T = ((\mathcal{G}, q, g), A, B, C)$. $\mathcal{V}_2$ computes all the other steps as $\mathcal{V}_1$ does.

Lemma 2.5.3. *In the hybrid $\mathcal{H}_2$(`aux`) $\mathcal{P}^\star$, for $x \notin L_{\mathsf{Rel}}$, makes the honest verifier $\mathcal{V}$ accept with non-negligible probability in the interaction with index* `i`.

Proof. From the hybrid $\mathcal{H}_1$(`aux`) follows that $\mathcal{P}^\star$ fed with the randomness that have as a prefix `rand` in the interaction with index `i` declares statement x s.t. $x \notin L_{\mathsf{Rel}}$ and makes the honest verifier accept with non-negligible probability. We observe that x is chosen before the interaction with index `i` is started, and before that the hybrids $\mathcal{H}_1$(`aux`) and $\mathcal{H}_2$(`aux`) are identical distributed. Therefore, if in $\mathcal{H}_1$(`aux`) $\mathcal{P}^\star$, running using the prefix of randomness `rand`, chooses a statement x s.t. $x \notin L_{\mathsf{Rel}}$ in the interaction with index `i` it continues, with non negligible probability, to do the same in $\mathcal{H}_2$(`aux`).

Now, we can argue that the interaction with index `i` is completed by $\mathcal{P}^\star$ with non-negligible probability also in $\mathcal{H}_2$(`aux`). Suppose by contradiction that is not the case, then we can construct an adversary $\mathcal{A}$ that breaks the Assumption 5 (i.e. 1DDH assumption). In more details, let $\mathcal{C}_{\mathtt{1DDH}}$ be the challenger of 1DDH game. $\mathcal{A}$ acts as $\mathcal{V}_2$ against $\mathcal{P}^\star$ (i$\mathcal{P}^\star$ is fed with the randomnesses of the form $r = \mathtt{rand}||r'$) except that in the interaction with index $\mathtt{i}$, for witch he has the following behavior. Upon receiving $T^\star = ((\mathcal{G}, q, g), A^\star, B^\star, C^\star)$ from $\mathcal{C}_{\mathtt{1DDH}}$ $\mathcal{A}$ sets $T = T^\star$, then he continues the interaction with $\mathcal{P}^\star$ computing all the other steps as $\mathcal{V}_2$ does. If $\mathcal{P}^\star$ completes that in the interaction with index $\mathtt{i}$ $\mathcal{A}$ outputs 1, otherwise he outputs a bit b, where $b \in \{0, 1\}$. Observe that if $T^\star$ is a 1-non-DH tuple then $\mathcal{A}$ acts as $\mathcal{V}_1$ of $\mathcal{H}_1(\mathtt{aux})$ otherwise he acts as $\mathcal{V}_2$ of $\mathcal{H}_2(\mathtt{aux})$.

The running time of the adversary $\mathcal{A}$ is dominated by $2^{\hat{\lambda}}$ that is the time needed to compute the discrete logarithm of Y but the 1DDH assumption still holds against adversary that runs in time $2^{\hat{\lambda}}$, therefore we reach a contradiction.

These two observations conclude the proof. $\square$

The next hybrid $\mathcal{H}_3(\mathtt{aux})$ takes as auxiliary input ($\mathtt{aux}$) the prefix of randomness $\mathtt{rand}$ and the index $\mathtt{i}$ of the interaction. The verifier $\mathcal{V}_3$ acts as the verifier $\mathcal{V}_2$ excepted that in the interaction with index $\mathtt{i}$. In the Step 3.1 of the $\mathtt{i}$-th interaction he runs the trapdoor procedure TFake in order to equivocate the opening of the commitment computed at first round.

It follows from the statistically trapdoorness of 2-IDTC that $\mathcal{H}_2$ and $\mathcal{H}_3$ are statistically close. Therefore $\mathcal{P}^\star$ feed with the randomness that have as a prefix $\mathtt{rand}$ in the $\mathtt{i}$-th interaction chooses a statement $x \notin L_{\mathsf{Rel}}$ and has a non-negligible probability of making the verifier accept in this interaction. $\mathcal{V}_3$ can rewinds $\mathcal{P}^\star$ at the ending of the second round and equivocate the opening of the commitment made in the first round w.r.t. different values: c, and $\hat{c}$ respectively before and after the rewind. Note that the distribution seen by $\mathcal{P}^\star$ before and after the rewind is statistical close therefore $\mathcal{V}_3$ obtains with non-negligible probability two transcripts (a, c, z) and $(a, \hat{c}, \hat{z})$ that are a collision for x. It follows from the special soundness of Π, that is possible to extract (in super-polynomial time) a witness for $x \in L_{\mathsf{Rel}}$. Contradiction.

Proof of rWI. The idea of the proof for rWI is very simple. We consider the prover $\mathcal{P}_1$ that, when instructed to use randomness with index j and input with index i and receives first message $\texttt{msg}$, checks first if a tuple $(j, i, \texttt{msg}, R)$ has been stored in a previous step. If such a tuple is found then R is used as source of randomness; otherwise, a fresh R is selected and tuple $(j, i, \texttt{msg}, R)$ is stored. Clearly, by pseudo-randomness, $\mathcal{P}_1$ is indistinguishable from the honest prover $\mathcal{P}$.

Now we observe that the resetting verifier $\mathcal{V}^\star$ is performing an attack on $\mathcal{P}_1$ in which two distinct interactions use the same randomness iff they share j, the input and the first message. More precisely, we say that interactions t_1 and t_2 between $\mathcal{P}_1$ and $\mathcal{V}^\star$ are a *collision* if they share the input, the randomness used by $\mathcal{P}_1$ and the first message and $\mathcal{V}^\star$ opens the commitment in the first messages in two different ways. If $\mathcal{V}^\star$ has a non-negligible probability of producing a transcript then we can break the Strong DLog Assumption (see Assumption 2). Consider algorithm DL that receives as input $(\bar{x}, \bar{w}^0, \bar{w}^1)$ for which $\mathcal{V}^\star$ distinguishes $(\bar{x}, \bar{w}^0)$ from $(\bar{x}, \bar{w}^1)$. DL interacts with $\mathcal{V}^\star$ and at the start of the interaction guesses two interactions $t_1 < t_2$ (in the hope they constitute a collision). In all interactions other than t_1 and t_2, DL runs just like $\mathcal{P}_1$. When DL receives the discrete logarithm parameters from $\mathcal{V}^\star$ as part of the first message of interaction t_1, it forwards them to the challenger of the discrete logarithm and receives Y (and the task is to compute the discrete logarithm of Y) and uses it as part of the second message of interaction t_1. When interaction t_2 is activated DL checks if the first message is the same as the first message of interaction t_1. If this is the case (and this happens with non-negligible probability) DL continues and sends the same second message, including Y. Notice that $\mathcal{V}^\star$ expects to receive the same message since it thinks it is interacting with $\mathcal{P}_1$ that is using the same randomness. Otherwise, DL aborts. Then DL rewinds $\mathcal{V}^\star$ and sends a different challenge for the Π^{OR} protocol in each of the two interactions. In at least one of them $\mathcal{V}^\star$ has opened the commitment to a different message than the one committed in the first message. This means that $\mathcal{V}^\star$ has used knowledge of the discrete logarithm of Y to complete the interaction and thus DL can extract it.

Finally, let us consider the case in which $\mathcal{V}^\star$ produces a collision in his resetting attack only with negligible probability. This means that $\mathcal{V}^\star$ is conducting a successful concurrent WI attack on the argument system. Standard arguments show that this contradicts the WI of Π[8].

2.5.3 Efficient 4-Round Resettable Zero Knowledge in the BPK model

Let Rel be a polynomial-time relation with Σ-protocol $\Pi = (\mathcal{P}, \mathcal{V})$. In this section we give an informal description of an efficient 4-round argument system (Π^{BPK}) for Rel, that is resettable zero knowledge and concurrently sound in the BPK model.

In our protocol we consider that each entry of the verifier's identities file is an element of a group $\mathcal{G}$ in which is hard to compute the discrete logarithm. $\Pi^{\mathsf{BPK}} = (\mathcal{P}, \mathcal{V})$ for Rel consists of the interleaved execution of two protocols: $\Pi^{\mathcal{WI}}$, in which $\mathcal{P}$ acts a prover, and Π^{OR}, in which $\mathcal{V}$ acts as a prover. Π^{OR} is the Σ-protocol obtained from the OR composition of two Schnorr's Σ-protocols Π_0 and Π_1. Π^{OR} is used by the verifier $\mathcal{V}$ to prove that she knows either the discrete logarithm of a selected identity $\mathtt{id}$, or the discrete logarithm of an elements sent by the prover $\mathcal{P}$ at the second round of Π^{BPK}. $\Pi^{\mathcal{WI}}$ is the rWI argument system described in Section 2.5.2, and it is used by the prover $\mathcal{P}$ to show that she know either the witness for the theorem x to be proved or the discrete logarithm of $\mathtt{id}$.

To prove concurrent soundness of Π^{BPK} we use the same arguments of Section 2.5.2, with the difference that at the end of the proof we can extract (in sub-exponential time) the discrete logarithm of $\mathtt{id}$ (instead of a witness for x), and construct a reduction using complexity leveraging to break the assumption that

[8]We are implicitly using the fact that the Σ-protocol Π is WI. This is certainly true if Π is perfect [CDS94]. If Π is only computational then we consider self OR composition of Π which by [GMY06] is WI.

it is hard to compute the discrete logarithm of an elements in $\mathcal{G}$.

To prove r$\mathcal{ZK}$ we consider a simulator $\mathcal{B}$ that rewinds the verifier $\mathcal{V}$ to get the discrete logarithm of `id` (used as a witness by $\mathcal{V}$ to run Π^{OR}) and uses this witness to complete the execution of the protocol $\Pi^{\mathcal{WI}}$. We observe that $\mathcal{B}$ works correctly only if the first round of $\Pi^{\mathcal{WI}}$ can be computed without using the witness. In this case we have no problem, because we can construct $\Pi^{\mathcal{WI}}$ starting form a Σ-protocol Π' that enjoys this property. More specifically, Π' is the result of an OR composition (using [CDS94]) of Π and of a Schnorr's Σ-protocol that is delayed witness. It is easy to see that the property of delayed witness of Schnorr's Σ-protocol holds even in Π' and in turn in $\Pi^{\mathcal{WI}}$. The last observation make us able to conclude the proof sketch.

2.5.4 Proof of Work of Knowledge

In [BKZZ16] the authors introduce a new class of prover verifier protocol called Proof of Work or Knowledge (PoWorK). Following [BKZZ16], a PoWorK protocol $\Pi = (\mathcal{P}, \mathcal{V})$ has two mode operations: (a) the Proof of Knowledge (PoK) mode, where $\mathcal{P}$ convinces $\mathcal{V}$ that he knows a witness for some instance, or (b) the Proof of WorK (PoW) mode, where $\mathcal{P}$ makes calls to the puzzle solving algorithm to solve a certain puzzle. Furthermore a malicious verifier $\mathcal{V}^\star$ does not distinguish between the PoK mode and PoW mode. As a counterpart if a malicious prover $\mathcal{P}^\star$ has a running time (in number of steps) less than a specified parameter f calibrated according to the hardness of the puzzle, then there exists a knowledge extractor that extract the witness for the instance proved by $\mathcal{P}^\star$ (f-soundness).

In order to formalize the work aspect in [BKZZ16] the authors give the notion of puzzle system. A puzzle system PuzSys is a tuple of algorithms $\mathsf{PuzSys} = (\mathsf{Sample}, \mathsf{Solve}, \mathsf{Verify})$ that are defined in the following way. Sample on input the security parameter 1^λ and the hardness factor h outputs a puzzle `puz`; Solve on input the security parameter 1^λ, a hardness factor h and a puzzle instance `puz` outputs a potential solution `sol`; Verify on input the security parameter 1^λ, a hardness factor h a puzzle instance puz, and solution $\mathtt{sol}$ outputs 0 or 1.

Loosely speaking in [BKZZ16] require that the algorithms Sample and Verify are efficient. Furthermore, it is difficult to compute a solution for a sampled puzzle, that is, if puzzle system is f-hard, then the Solve algorithm can not take less of f-steps of computation.

Using our OR-composition we can also implement PoWorK protocol, where the prover proves either the knowledge of a witness for some instance or a solution for a puzzle.

More formally, let Π be a Chameleon Perfect Σ-protocol (or a delayed-witness Perfect Σ-protocol) for Rel, and let $\mathsf{PuzSys} = (\mathsf{Sample}, \mathsf{Solve}, \mathsf{Verify})$ a puzzle system, that has an associated delayed-input Perfect Σ-protocol Π^{P} for the relation $\mathsf{Puzzle} = \{\mathsf{puz} \leftarrow \mathsf{Sample}(1^\lambda, h) : \exists\ \mathtt{sol}$ s.t. $\mathsf{Verify}(1^\lambda, h, \mathsf{puz}, \mathtt{sol}) = 1\}$.

We can use Π^{P} and Π as input of our OR-composition described in Section 2.4 and obtaining a $\Pi^{\mathsf{OR}} = (\mathcal{P}^{\mathsf{OR}}, \mathcal{V}^{\mathsf{OR}})$ for the following relation

$$\mathsf{Rel}_{\mathsf{OR}} = \Big\{((x_0, x_1), w) : \big((x_0, w) \in L \wedge x_1 \in \hat{L}_{\mathsf{Puzzle}}\big)\mathsf{OR}$$
$$\big((x_1, w) \in \mathsf{Puzzle} \wedge x_0 \in \hat{L}\big)\Big\}$$

where the statement x_1 (the puzzle) is sent by $\mathcal{V}$ at the second round.

First of all we observe that $\mathcal{P}^{\mathsf{OR}}$ can work in PoK mode and PoW mode. When $\mathcal{P}^{\mathsf{OR}}$ works in PoK mode he computes the third round of Π^{OR} using the witness w_0 s.t. $(x_0, w_0) \in \mathsf{Rel}_L$. Instead when $\mathcal{P}^{\mathsf{OR}}$ works in the PoW mode, he runs Solve obtaining a witness w_1 s.t. $(x_1, w_1) \in \mathsf{Puzzle}$ and he computes the third round of Π^{OR} using the witness w_1 (i.e. a solution for the puzzle). The perfect indistinguishability follows from the perfect adaptive WI property of Π^{OR}, and f-soundness follows from the PoK of Π^{OR}.

As the PoWorK protocols constructed in [BKZZ16], also our contraction is 3-round and public coin. Note, also, that we do not require the additional feature for the puzzle system that in [BKZZ16] is called "density" and it is needed in [BKZZ16].

Instantiation of the puzzle system. In [BKZZ16] one of the construction of the puzzle system is based on discrete logarithm problem. In that case we need a delayed-input Σ-protocol for Puzzle relation that corresponds to the DLog relation, therefore we can use the Schnorr's Σ-protocol.

2.6 More About OR-Composition

In this section we give some more details about our OR-compostion. In particular we show that the pre-image protocol described in [Mau15, Cra96] is a Chameleon Σ-Protocol, therefore can be used to instantiate our OR-composition. Then, we provide a precise classification of the Σ-Protocols that can be used in our new OR-composition technique. Finally we discuss the efficiency of our technique.

2.6.1 More About Chameleon Σ-Protocols

Here we described the pre-image protocol for proving knowledge of a pre-image of a value in the range of a homomorphic function. This protocol is an abstraction of a large class of protocols like Schnorr's [Sch89] protocol and Guillou-Quisquater [GQ88]. This abstraction is first described in [Cra96, CD98] and later observed in [Mau15].

Let $(\mathcal{G}, \star)$ and $(\mathcal{H}, \otimes)$ be two groups whose operations are efficiently computable, and let $f : \mathcal{G} \to \mathcal{H}$ be a one-way homomorphism from $\mathcal{G}$ to $\mathcal{H}$. That is, $f(x \star y) = f(x) \otimes f(y)$.

The pre-image protocol Π for relation $\mathsf{Rel} = \{(x, w) : x = f(w)\}$ with associated algorithm $(P_1, P_2, \mathsf{V}))$ and with challenge length l is described below:

Common input: (description of) $\mathcal{G}$ and $\mathcal{H}$ and $x \in \mathcal{H}$;

Prover's private input: w such that $x = f(w)$.

- Algorithm P_1.

  On input $(x, w) \in \mathsf{Rel}$ and random coin tosses R_1, P_1 picks $k \leftarrow \mathcal{G}$, sets $a \leftarrow f(k)$ and outputs a.

- Algorithm P_2.

  On input $(x, w) \in \mathsf{Rel}$, k, and challenge c, P_2 computes and outputs $z = k \star w^c$.

- Algorithm V.

  On input (x, a, c, z), V outputs 1 iff $f(z) = a \otimes x^c$.

The simulator Sim of Π on input instance x and challenge c works as follows:

- randomly pick $z \leftarrow \mathcal{G}$;
- compute $a = f(z) \otimes x^{-c}$;
- return (a, z).

Theorem 3 of [Mau15] describes the two conditions under which the pre-image protocol is a Σ-protocol. Specifically, for integer y, $u \in \mathcal{G}$ and $(x, w) \in \mathsf{Rel}$ we have:

- $\gcd(c_1 - c_2, y) = 1$, for all challenges $c_1 \neq c_2 \in \{0,1\}^l$;
- $f(u) = x^y$.

Theorem 19. The Pre-Image Protocol is a Chameleon Σ-protocol.

Proof. We describe algorithm $\mathsf{P_{sim}}$. Let $(a, \widetilde{z})$ be the output of Sim on input x and challenge $\widetilde{c}$. PPT algorithm $\mathsf{P_{sim}}$ on input $x, \widetilde{c}$ and the witness w for x and challenge c, computes and outputs $z = \widetilde{z} \star w^{-\widetilde{c}} \star w^c$.

The triple (a, c, z) is an accepting transcript because the test $(f(z) = a \otimes x^c)$ of V is successful. Indeed we have

$$a \otimes x^c = f(\widetilde{z}) \otimes x^{-\widetilde{c}} \otimes x^c = f(\widetilde{z}) \otimes f(w)^{-\widetilde{c}} \otimes f(w)^c = f(\widetilde{z} \star w^{-\widetilde{c}} \star w^c) = f(z).$$

We show now the property of indistinguishability for Chameleon Σ-protocols. We prove that for all pairs of challenges $\widetilde{c}$ and c and for all $(x, w) \in \mathsf{Rel}$ the two following distributions are indistinguishable:

- (a, c, z), where $a = f(z) \otimes x^{-c}$;
- (a, c, z), where $a = f(\widetilde{z}) \otimes x^{-\widetilde{c}}$ $z = \widetilde{z} \star w^{-\widetilde{c}} \star w^{c}$.

From the SHVZK property follows that the first distribution is perfect indistinguishable from a real transcript $a = f(k)$, c, $z = k \star w^{c}$ (where k is a random element of $\mathcal{G}$) produced by Π.

We note that $\widetilde{z}$ is a random element of $\mathcal{G}$, so $\widetilde{z} \star w^{-\widetilde{c}}$ is also a random element of $\mathcal{G}$, for this reason we can set $\bar{k} = \widetilde{z} \star w^{-\widetilde{c}}$, and obtain that the second distribution $a = f(\bar{k})$, c, $z = \bar{k} \star w^{c}$ is identically distributed to a transcript given in output by Π. $\square$

2.6.2 Classification of Σ-Protocols

We provide examples for the four classes of Σ-protocols mentioned in Section 2.1. Table 2.1 summarizes the classes of Σ-protocols that can be used to construct either a 2-IDTC scheme or a 3-IDTC scheme, or both, and the class of Σ-protocols that cannot be used to instantiate any of our IDTC schemes.

	(Class 1)	*(Class 2)*	*(Class 3)*	*(Class 4)*
2-IDTC	Yes	Yes	No	No
3-IDTC	Yes	No	Yes	No

Figure 2.1 Class of Σ-protocols

Examples of Class 1 and Class 3 Σ-protocols. The Class 1 is the class of Σ-protocols that are Chameleon and that are delayed-witness Σ-protocols. Schnorr's Σ-protocol for DLog is an example of Class 1 Σ-protocol. Indeed, its first round consists of the prover sending a random group element. Moreover, even if this value is computed by a simulator, knowledge of the witness and of the randomness used by the simulator suffices for the prover to answer any challenge.

The Class 3 is the class of Σ-protocols that are not Chameleon and that are delayed-witness Σ-protocols. For instance, Blum's Σ-protocol for Hamiltonian graphs belongs to Class 3. In fact this Σ-protocol requires the prover only to know the graph to compute the first round.

An example of a Class 2 Σ-protocol. Recall that Class 2 is the class of Σ-protocols that are Chameleon and that are not delayed-witness Σ-protocols. We construct a Class 2 Σ-protocol $\Pi = (\mathcal{P}, \mathcal{V})$ for the relation $\mathsf{DLOG} = \{((\mathcal{G}, q, g, Y), y) : g^y = Y\}$ by using Pedersen's commitment scheme [Ped91] as a 2-IDTC scheme. The TCom algorithm of the 2-IDTC scheme for DLOG based on Pedersen's commitment takes as input the description of a cyclic group $\mathcal{G}$ of order q, a generator g of $\mathcal{G}$ and an element $Y \in \mathcal{G}$. To commit to m, TCom selects $r \leftarrow \mathbb{Z}_q$ uniformly at random and returns $\mathtt{com} = g^r \cdot Y^m, \mathtt{dec} = r, \mathtt{rand} = r$. The decommitment algorithm TDec is straightforward and the trapdoor algorithm TFake, knowing the discrete log of Y, can open the commitment $\mathtt{com}$ as any message m'.

We are now ready to describe our proposed Class 2 Σ-protocol for DLOG with common input $(\mathcal{G}, q, g, Y)$. In the first round, $\mathcal{P}$ computes $(\mathtt{com}_p, \mathtt{dec}_p^0, \mathtt{rand}) \leftarrow \mathsf{TCom}(\mathcal{G}, g, Y, 0)$ and then uses TFake to compute an opening $\mathtt{dec}_p^1$ of $\mathtt{com}_p$ as 1. Finally, $\mathcal{P}$ computes $(\mathtt{com}_0, \mathtt{dec}_0, \mathtt{rand}_0) \leftarrow \mathsf{TCom}(\mathcal{G}, g, Y, \mathtt{dec}_p^0)$ and $(\mathtt{com}_1, \mathtt{dec}_1, \mathtt{rand}_1)$ $\leftarrow \mathsf{TCom}(\mathcal{G}, g, Y, \mathtt{dec}_p^1)$ and sends $(\mathtt{com}_p, \mathtt{com}_0, \mathtt{com}_1)$ to $\mathcal{V}$ that replies with a one bit challenge b. $\mathcal{P}$ answers by sending $\mathtt{dec}_p^b$ and $\mathtt{dec}_b$. The simulator Sim for the SHVZK receives an instance $(\mathcal{G}, q, g, Y)$ and a bit b and computes $(\mathtt{com}_p, \mathtt{dec}_p^b, \mathtt{rand}) \leftarrow \mathsf{TCom}((\mathcal{G}, q, g, Y), b)$. Commitment $\mathtt{dec}_p^b$ is committed twice obtaining $\mathtt{com}_0$ and $\mathtt{com}_1$ and only $\mathtt{com}_b$ is opened. Notice that, since Pedersen's commitment is perfectly hiding, then the simulation of Sim is perfect. Clearly, $\mathcal{P}$ needs the witness of Y for computing the first round. Moreover, the proposed protocol is Chameleon since Sim commits twice to the same opening of the commitment $\mathtt{com}_p$ but then $\mathcal{P}$, once the discrete log of Y becomes available, can computed $\mathtt{dec}_p^{1-b}$ and then open $\mathtt{com}_{1-b}$ as $\mathtt{dec}_p^{1-b}$.

Examples of Class 4 Σ-protocols. Recall that Class 4 is the class of Σ-protocols that are not Chameleon and that are not delayed-witness Σ-protocols.

As an example of a Class 4 Σ-protocol we consider the protocol obtained from the Class 2 Σ-protocol described in the previous paragraph in which $\mathtt{dec}_p^0$ and $\mathtt{dec}_p^1$ are committed by using a (non-interactive) commitment scheme that is perfectly binding e computationally hiding instead of a Pedersen's commitment scheme. For example, the ElGamal encryption scheme can be used to construct such a commitment scheme. In this case, the Σ-protocol obtained is only computational HVZK.

2.6.3 Efficiency

We now give a briefly comparison our OR transform and the CDS-OR transform in terms of number of modular exponentiations that they involve.

For this comparison we consider a protocol Π^{OR} that proves the knowledge of one out of two discrete logarithms. Therefore the OR transforms has on input Π_0 and Π_1 both corresponding to Schnorr's Σ-protocol.

We consider two cases:

- 1st case: Π_0 is Schnorr's Σ-protocol for relation $\mathsf{DLOG} = \{((\mathcal{G}', q', g', Y'), y') : g^{y'} = Y'\}$, where q' is prime and $\mathcal{G}'$ is a group of order q' of the quadratic residues modulo p' s.t. $p' = 2q' + 1$, where $|p'| = 2048$ and $|q'| = 2047$. Π_1 is Schnorr's Σ-protocol for relation $\mathsf{DLOG} = \{((\mathcal{G}, q, g, Y), y) : g^y = Y\}$, where q is prime and G is a group of order q of the quadratic residues modulo p s.t. $p = 2q + 1$, where $|p| = 1024$ and $|q| = 1023$.
- 2nd case: Π_0 and Π_1 are both like Π_1 described in the 1st case.

In both cases we instantiate t$-$IDTC from Π_0.

The execution of Schnorr's Σ-protocol costs 1 modular exponentiation, while the execution of the simulator of Schnorr's Σ-protocol costs 2 modular exponentiations. By exponentiation mod p or exponentiation mod p', we indicate, respectively the exponentiation modulo a prime of 1024 bits and the exponentiation modulo a prime of 2048 bits.

To evaluate the cost of our OR transform, we first note that the number of modular exponentiations of our OR transform are different when it is instantiated using a 2$-$IDTC scheme or 3$-$IDTC scheme, even when the 2$-$IDTC scheme and 3$-$IDTC schemes are constructed from the same Σ-protocol. In particular if the scheme are constructed from Schnorr's Σ-protocol run the algorithm TCom costs 2 modular exponentiations in the case of 2$-$IDTC and it costs 3 modular exponentiations in the case of a 3$-$IDTC.

From this observation follows that to compute the first round of our our OR transform we need 1 exponentiation mod p to compute the first round of Schnorr's Σ-protocol plus the cost to execute TCom.

To compute the third round of our OR transform we need 2 exponentiations mod p when we run equivocal procedure TFake, because we need to execute again a simulator of Π_1. Otherwise no other exponentiations is required. Therefore in the worst case to compute the third round of our OR transform we need 2 exponentiations mod p.

Summing up:

- in the 1st case our OR transform costs 3 exponentiations mod p plus 2 exponentiations mod p' if we use a 2$-$IDTC scheme (or 3 exponentiations mod p' if we use a 3$-$IDTC scheme).

- in the 2nd case our OR transform costs 3 exponentiations mod p plus 4 exponentiations mod p if we use a 2$-$IDTC scheme (or 6 exponentiations mod p if we use a 3$-$IDTC scheme). Note that in this case to commit to a first round of Schnorr's Σ-protocol where $|a| = 1024$ bits, we need to run twice the TCom procedure.

The CDS-OR transform costs in both cases 3 modular exponentiations. In the 1st case the 2 exponentiations are mod p and 1 exponentiation is mod p', in the 2nd case all the exponentiations are mod p.

2.7 More About Σ-Protocols

In this last section we will give some additional theorems on Σ-protocol, that have been used during the Chapter. Furthermore, we will described a Σ-protocol for the 1DDH relation.

Theorem 20. For every relation Rel such that $L_{\mathsf{Rel}} \notin \mathsf{BPP}$ there exist Σ-protocols that are not WI.

Proof. Let $\Pi' = (\mathcal{P}', \mathcal{V}')$ be a Σ-protocol for the relation Rel with challenge length l and let $(P'_1, P'_2, \mathsf{V}')$ be the triple of PPT algorithms associated to Π'. We use these algorithms to describe a Σ-protocol Π with associated algorithms (P_1, P_2, V) that is not WI. Consider $(x, w) \in \mathsf{Rel}$.

1. P_1 on input (x, w) and randomness R_1 parses it as (r_1, c_p) where c_p is an l-bit string, computes $a' \leftarrow P'_1(x, w; r_1)$, and outputs $a = (a', c_p)$.

2. P_2, on input (x, w), R_1, a challenge c computes $z' \leftarrow P'_2(x, w, r_1, c)$ and if $c = c_p$ then it also sets $z = w$ otherwise it sets $z = z'$; finally it outputs z.

3. V, on input x, $a = a', c_p$, c and z, makes the following steps: in case c is different from c_p it outputs $\mathsf{V}'(x, a', c, z)$ otherwise it output 1 iff $(x, z) \in \mathsf{Rel}$.

We now check that Π is a Σ-protocol.

- **Completeness**: The completeness of Π follows from the completeness of Π' except when c is equal to c_p. In this case $\mathcal{P}$ has a witness and sends it to $\mathcal{V}$ that still accepts.
- **Special Soundness**: $\mathsf{Extract}$ on input a collision $(a = (a', c_p), c_1, z_1)$ $(a = (a', c_p), c_2, z_2)$ works as follows:
  – if c_1 and c_2 are different from c_p then it runs the extractor $\mathsf{Extract}'$ of Π' on input x and a collision (a', c_1, z_1) (a', c_2, z_2) returning its output.
  – if c_1 is equal to c_p, it outputs z_1 while instead if c_2 is equal to c_p it outputs z_2.
- **SHVZK** $\mathsf{Sim}(x, c)$ of Π works as follows:
  – computes $(a', z') \leftarrow \mathsf{Sim}'(x, c)$, where $\mathsf{Sim}'(x, c)$ is the simulator of Π';
  – picks $c_p \leftarrow \{0, 1\}^l$.
  – if c_p is equal to c then it aborts, otherwise it outputs $(a = (a', c_p), z')$.

We prove that Π is computational SHVZK, namely: for any l-bit string c, the transcript given in output by $\mathsf{Sim}(x, c)$ is computationally indistinguishable from a honest transcript where the challenge is c and $\mathcal{P}$ runs on common input x and private input w such that $(x, w) \in \mathsf{Rel}$.

Suppose there exists a distinguisher $\mathcal{A}$ for the SHVZK of Π, then we can show a distinguisher $\mathcal{A}'$ for the SHVZK of Π'.

$\mathcal{A}'$ runs $\mathcal{A}$ that outputs a pair (x, w) and a challenge c. $\mathcal{A}'$ then asks the challenger of SHVZK to produce a transcript (either honest or simulated) for instance x, witness w and challenge c. $\mathcal{A}'$ obtains from the challenger a pair (a', z') such that (a', c, z') is an accepting transcript. $\mathcal{A}'$ picks randomly an l-bit string c', sets $a = a'|c'$, $z = z'$, feeds (a, z) to $\mathcal{A}$, and outputs what $\mathcal{A}$ outputs.

We note that the success probability of $\mathcal{A}'$ is statistically close to the one of $\mathcal{A}$ since the probability that c is equal to c' is negligible and this case is the only deviation among the two distributions.

We finally note that Π' is not WI since an adversarial verifier $\mathcal{V}^\star$ can obtain a witness by just sending a challenge c that is equal to c_p. As a consequence $\mathcal{V}^\star$ can get and output a witness for $x \in L$ during an execution with $\mathcal{P}$. Clearly no PPT simulator can produce the same output unless $L \in \mathsf{BPP}$. $\square$

2.7.1 Challenge Length of Σ-Protocols

In this section we show how one can reduce or stretch the size of the challenge in a Σ-protocol and in a $\tilde{\Sigma}$-protocol.

Challenge-length amplification. The challenge of a Σ-protocol can be extended through parallel repetition.

Lemma 2.7.1. *[CDS94, Dam10] Let Π be a Σ-protocol (resp. $\tilde{\Sigma}$-protocol) for relation Rel and challenge length l. Running Π k-times in parallel for the same instance x corresponds to running Σ-protocol (resp. $\tilde{\Sigma}$-protocol) for Rel with challenge length $k \cdot l$.*

Challenge-length reduction.

Lemma 2.7.2. *[Dam10] Given a Σ-protocol of challenge length l for the relation Rel, is possible to construct a Σ-protocol, for the same relation Rel with challenge length l' where $l' < l$.*

We now show that Lemma 2.7.2 it is true even when we consider a $\tilde{\Sigma}$-protocol. One possibility to obtain this result is to convert the $\tilde{\Sigma}$-protocol in a Σ-protocol, and then use Lemma 2.7.2. We show how to obtain the same result without first converting the $\tilde{\Sigma}$-protocol to a Σ-protocol.

Lemma 2.7.3. *For any $\tilde{\Sigma}$-protocol $\Pi = (\mathcal{P}, \mathcal{V})$, for a relation Rel with challenge length l, simulator Sim, and the associated triple (P_1, P_2, V), there exists a $\tilde{\Sigma}$-protocol $\Pi' = (\mathcal{P}', \mathcal{V}')$, for the same relation Rel, with challenge length l', where $l' < l$ and with the same efficiency.*

Proof. We show Π' by presenting the associated triple $(P_1', P_2', \mathsf{V}')$ of efficient PPT algorithms.

1. P_1' on input (x, w) and randomness R_1 computes and outputs $a \leftarrow P_1(x, w; R_1)$.

2. P_2' on input (x, w), $c \in \{0,1\}^{l'}$, R_1 and randomness R_2, parses R_2 as (pad, R_2') where pad is an $(l - l')$-bit string, sets $c' = c|pad$, computes $z \leftarrow P_2(x, w, R_1, c'; R_2')$ and outputs $z' = (z, pad)$.

3. V' on input $x, a, z' = (z, pad)$ and c, outputs the output of $\mathsf{V}(a, c|pad, z)$.

Completeness follows directly from the completeness of Π.

HVZK We can consider the simulator Sim', that on input x runs as follows:

- runs $(a, c, z) \leftarrow \mathsf{Sim}(x)$;

- sets pad equal to the last $l - l'$ bits of c, and sets c' equal to the fist l' bits of c;

- outputs $(a, c', (z, pad))$.

Special soundness follows directly from the special soundness of Π. □

From Lemma 2.7.1, 2.7.2 and 2.7.3, we can claim the following theorem.

Theorem 21. Suppose that relation Rel has a Σ-protocol ($\tilde{\Sigma}$-protocol) Π. Then, for any challenge length l, Rel admits a Σ-protocol ($\tilde{\Sigma}$-protocol) Π' with challenge length l'. If $l' \leq l$ than Π' is almost as efficient as Π. Otherwise the communication and computation complexities of Π' are l'/l times the ones of Π.

2.7.2 Σ-protocol for the 1DDH Relation.

Let $\mathsf{1DDH}$ be the following relation

$$\mathsf{1DDH} = \Big\{ \big(((\mathcal{G}, q, g), A = g^a, B = g^b, C = g^{ab+1}), b{+}1\big) : A^{b+1} = C \Big\}$$

and let Π^{DDH} be a Chameleon Σ-protocol for DDH ([Ped91]). We can use Π^{DDH} to build a Chameleon Σ-protocol Π^{1DDH} for $\mathsf{1DDH}$ in the following way. $\mathcal{P}$ and $\mathcal{V}$ on common input $T = \big((\mathcal{G}, q, g), A, B, C\big)$ construct the tuple $T' = \big((\mathcal{G}, q, g), A', B', C'/g\big)$, then they engage Π^{DDH} on input T'.

Theorem 22. Π^{1DDH} is a Chameleon Σ-protocol for $\mathsf{1DDH}$.

It is easy to observe that if T is a 1-non-DH tuple, then T' is a DH-tuple and $\mathcal{P}$ has a witness for it. Furthermore the completeness, the SHVZK and the special soundness of Π^{1DDH} follow from the completeness, the SHVZK and the special soundness of Π^{DDH}. Finally, Π^{1DDH} is a Chameleon Σ-protocol because Π^{DDH} is a Chameleon Σ-protocol.

Chapter 3

Non-Malleable Commitment Schemes

3.1 Overview of the Chapter

Commitment schemes are fundamental in Cryptography. They require a sender to fix a message that can not be changed anymore, but that will remain hidden to a receiver until the sender decides to reveal it.

In order to model modern real-world adversaries, commitment schemes have been proposed with additional security properties. Here we consider the intriguing question of constructing a scheme that remains secure against man-in-the-middle (MiM) attacks: a non-malleable (NM) commitment scheme [DDN91].

This fascinating setting is much harder to deal with than the classic stand-alone setting. Indeed while we know 1-round and 2-round regular commitment schemes under various assumptions ([GL89, NY89, Ped91, Nao91, HM96]), Pass proved that NM commitments[1] require at least 3 rounds [Pas13] when security is proved through a black-box reduction to a falsifiable (polynomial) hardness assumption. Instead in different and more controversial models (e.g., assuming the existence of a trusted random string, by

[1]We consider the notion of NM commitment w.r.t. commitment.

modeling hash functions as random oracles, by weakening the security definition admitting an inefficient challenger) we know constructions of non-interactive NM commitments [DG03, PPV08].

The round complexity of NM commitment schemes in the standard model has puzzled researchers for long time. Starting from the construction of [DDN91] that required a logarithmic number of rounds, various constant-round schemes were proposed [Bar02, PR05b, PR05a, PR08a, PW10, LP11b, Goy11, GLOV12, GRRV14, BGR$^+$15, LP15, COSV17b, COSV16]. Interestingly Ciampi et al. in [COSV17b] show a 4-round commitment scheme that is secure also when the adversary mounts a concurrent MiM attack, a setting that corresponds to what can actually happen when sender and receiver are connected through a communication network like the Internet. In such a much more interesting setting a MiM adversary receives multiple commitments from senders and sends his commitments to multiple receivers.

3.1.1 Towards 3-Round (Concurrent) NM Commitments

The existence of 3-round NM commitment schemes is an important question first because 3 is the best possible constant (in light of the lower bound of [Pas13]), and second because 3 is the smallest number of rounds for a primitive that often makes use of commitment schemes: proofs of knowledge.

The importance of obtaining 3-round (and not just any constant-round) NM commitments motivated the following works. The result of [GPR16] showed that 3 rounds are sufficient for (one-left, one-right) non-malleable commitments. Then, the work of [Khu17] realizes a 3-round concurrent non-malleable commitment under number theoretic assumptions. Furthermore the work of [KS17] and the work of [LPS17], decrease the round complexity of concurrent non-malleable commitment to two, but they require stronger complexity theoretic assumptions. Therefore the following natural and important question remains open.

Main Open Question: Can we construct a 3-round concurrent non-

malleable commitment scheme under standard generic assumptions, rather than specific number-theoretic assumptions?

3.1.2 Other 3-Round Challenges

We list here 3 other interesting settings where no 3-round construction is known against concurrent MiM adversaries.

- Proofs[2] of knowledge are very useful in Cryptography. They have been studied in particular when there are only 3 rounds and the verifier just sends random bits (e.g., Σ-protocols [CDS94, Dam10], Blum's protocol for Hamiltonicity [Blu86b]). Despite their importance, there is no construction for 3-round proofs of knowledge (PoK) that is sufficiently secure under concurrent MiM attacks. This is due to the fact that such attacks are in general extremely difficult to deal with. Even though there exist constructions with a constant number of rounds, the case of just 3 rounds so far has remained unsolved.

- Lapidot and Shamir in [LS90] proposed a 3-round public-coin delayed-input WIPoK for NP: the LS protocol. As we said before, when a PoK is used as sub-protocol the delayed-input feature is instrumental to give a better round complexity to the external protocol. An additional features of delayed-input protocols is that they allow to shift large part of the computation to an off-line phase. Unfortunately the LS protocol and the PoKs of [CPS$^+$16b, CPS$^+$16a] are not secure against concurrent MiM attacks and this penalizes those applications where both round complexity and security against concurrent MiM attacks are important.

- We notice that identification schemes have been often proposed (e.g., [FFS87]) through the paradigm of proving "knowl-

[2]For simplicity in the informal part of the work we will not make a strict distinction between proofs and arguments. In the formal part we will use appropriate terms.

edge" of a secret[3]. Under this formulation there are constant-round constructions that are proven secure against concurrent MiM attacks [BFGM01]. However no 3-round scheme known in literature is proven secure in presence of a concurrent MiM adversary.

3.1.3 Our Contribution

In this work we study 3-round commitment scheme in presence of concurrent MiM attacks and solve in the positive the above open problems.

3-Round concurrent NM commitment schemes and more. In the main result of this chapter, we show a transform that on input any 3-round NM commitment scheme[4] gives a 3-round concurrent NM commitment scheme. The construction of [GPR16] can be used to instantiate our transform, therefore obtaining a 3-round concurrent NM commitment scheme based on any one-way permutation secure against subexponential-time adversaries. This result solves the main open question. Moreover our scheme (still when instantiated with the one of [GPR16] and using a proper one-way permutation) is public coin and (if desired[5]) has the delayed-input property.

Our transform extends the security of the underlying commitment scheme to multiple receivers. It is known that this implies security also with multiple senders [LPV08]. The crucial idea of our transform is to combine the underlying NM commitment

[3]Other notions based on signature or decryption capabilities are considered weaker since in some applications the verifier wants to make sure that the prover is the actual entity matching the announced identity. Indeed without a PoK a prover could give some partial information about his secret to others that can still succeed in convincing the verifier, even though they do not know the full secret.

[4]We also require the scheme to be extractable. Extractability often comes for free since it is commonly used in the non-malleability proof.

[5]Our transform can be instantiated in two ways. In the former the message to commit is required already when playing the first round, while in the latter the message to commit is required when playing the third round only.

scheme along with a one-time pad, to produce a commitment of a message that by itself, in case of a malleability attack, will have sufficient structure to be recognized by a distinguisher in the session in which it appears. Therefore a successful concurrent MiM even playing multiple commitments with multiple receivers will have to maul the underlying commitment scheme in at least one session. Since the message has sufficient structure with respect to that single session, we are able to translate the concurrent MiM attack into a non-concurrent MiM that violates the security of the underlying (non-concurrent) NM commitment scheme. We will implement the idea of committing to a message with structure by forcing a successful concurrent MiM to commit to the solution of a puzzle in at least one session. We will use complexity leveraging to show that the attack of the concurrent MiM is indistinguishable from the attack of a polynomial-time simulator that plays with receivers only.

Furthermore we propose a different approach for 3-round one-one non-malleable commitments that can be instantiated with a limited form of non-malleability enjoyed by both a subprotocol of [GRRV14] and a subprotocol of [GPR16] (therefore we can instantiate our result in two completely different ways). The main result of the chapter can still be instantiated using our 3-round one-one non-malleable commitment scheme.

Our 3-round one-one non-malleable commitment scheme combines some ideas of the first mentioned result of the chapter along with the concept of weak non-malleable commitment. In particular we start with a scheme that is one-one non-malleable only against synchronous adversaries that do not commit to $\perp$. Note that both a subprotocol of [GRRV14] and a subprotocol of [GPR16] satisfy this security property. Considering this notion we construct a compiler that, on input a 3-round synchronous weak one-one NM commitment scheme, gives as output a 3-round extractable one-one NM commitment scheme assuming OWPs secure against subexponential-time adversaries.

3-round arguments of knowledge and ID schemes against concurrent MiM attacks. We notice that our 3-round concurrent NM commitment scheme is a commit-and-prove argument of knowledge (AoK). This means that one can see our scheme as a commitment followed by an AoK about the committed value. By applying a simple change to the statement of the underlying AoK we obtain a 3-round concurrent NM witness-indistinguishable AoK (concurrent NMWIAoK) a notion introduced in [OPV08] and later on extended in [LPV09]. We stress that the delayed-input and public-coin properties of our commitment scheme are preserved by our concurrent NMWIAoK.

Notice that AoKs under standard assumptions require at least 3 rounds. The simulation-based notion for concurrent non-malleable arguments of knowledge (i.e., concurrent NM zero knowledge) requires at least a polylogarithmic number of rounds [CKPR01, BPS06] when the simulator is black box. In [OPV08] it is shown how to get concurrent NM zero knowledge (NMZK) in the bare public-key (BPK) model [CGGM00, MR01] with just two executions of a concurrent NMWIAoK. Therefore here we directly obtain a round-efficient concurrent NMZKAoK in the BPK model. Notice also that by making use of the delayed-input feature the simulator can extend a main thread avoiding issues due to aborting adversaries as discussed in [SV12, ORSV13].

Finally, we notice that one can get an identification scheme secure in the PoK sense in the concurrent[6] setting of [BFGM01] as well as under the stronger definition based on matching conversations of [BR93, Kat02] naturally extended to multiple concurrent sessions. Following [OPV08, COSV12], the key idea consists in using an identity that has two possible secrets such that knowledge of one witness does not allow to compute the other one in polynomial time. Then, by using our implementation of a concurrent NMWIAoK for proving knowledge of a secret associated to such identity we obtain a 3-round identification scheme secure against concurrent MiM attacks.

[6]In [BFGM01] a notion called CR2 is proposed to deal with concurrent MiM attacks and reset attacks. Reset attack were also considered in the notion CR1+ introduced in [BPSV08]. Since reset attacks are out of the scope of this work, we will focus on concurrent MiM attacks only.

Other applications. In [GMPP16] it is showed that the existence of a 3-round 3-robust parallel non-malleable commitment scheme implies the existence of a 4-round protocol for secure multi-party coin tossing for polynomially many coins and of a 4-round secure two-party protocol for any functionality in the simultaneous channel. One of the candidate instantiations of such special commitment scheme is our 3-round concurrent non-malleable commitment scheme.

3.2 3-Round Concurrent Non-Malleable Commitments

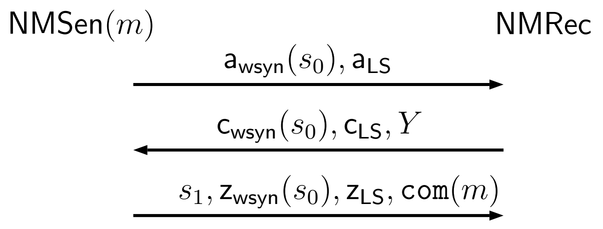
In this section we show the main result of this work, a transform that starting from a 3-round extractable one-one non-malleable commitment scheme outputs a 3-round concurrent non-malleable commitment scheme.

3.2.1 Informal Description

Our transform takes as input a 3-round extractable one-one NM commitment scheme Π_{wsyn}, a OWP f, a non-interactive perfectly binding commitment scheme NI, the 3-round delayed-input adaptive WI/PoK LS and outputs a 3-round fully concurrent (i.e., many-many) NM commitment scheme $\Pi_{\mathsf{NMCom}} = (\mathsf{NMSen}, \mathsf{NMRec})$.

Let m be the message that NMSen wants to commit. The high-level idea of our compiler is depicted in Fig. 3.1. The sender NMSen, on input the session-id $\mathtt{id}$ and the message m, computes the 1st round of the protocol by running LS and sending the 1st round of wsyn to commit to a random message s_0 using $\mathtt{id}$ as session-id. In the 2nd round the receiver NMRec sends the challenges of wsyn and LS, also sends a random value Y in the range of the OWP f[7]. In the last round NMSen commits to message m using NI, therefore obtaining $\mathtt{com}$, then computes the last round

[7]When sampling from the range of f corresponds to picking a random string, we have that our commitment scheme is public coin.


$\mathsf{NMSen}(m)$
NMRec
$\mathsf{a}_{\mathsf{wsyn}}(s_0), \mathsf{a}_{\mathsf{LS}}$
$\mathsf{c}_{\mathsf{wsyn}}(s_0), \mathsf{c}_{\mathsf{LS}}, Y$
$s_1, \mathsf{z}_{\mathsf{wsyn}}(s_0), \mathsf{z}_{\mathsf{LS}}, \mathtt{com}(m)$

- Y is an element taken from the range of the OWP f.
- $\mathtt{com}(m)$ is the perfectly binding commitment of m computed using NI.
- $(\mathsf{a}_{\mathsf{wsyn}}(s_0), \mathsf{c}_{\mathsf{wsyn}}(s_0), \mathsf{z}_{\mathsf{wsyn}}(s_0)) = \tau$ is the transcript of the execution of the NM commitment scheme Π_{wsyn} when the sender commits to s_0.
- $(\mathsf{a}_{\mathsf{LS}}, \mathsf{c}_{\mathsf{LS}}, \mathsf{z}_{\mathsf{LS}}) = \pi$ is the transcript of LS proving knowledge of either m and the randomness used to compute $\mathtt{com}$, or of $(s_0, \mathtt{dec})$, s.t. $f(s_0 \oplus s_1) = Y$ and $\mathtt{dec}$ is a valid decommitment of s_0 w.r.t. τ.

Figure 3.1 Informal description of our 3-round concurrent NM commitment scheme.

of wsyn, completes the transcript of LS, and finally sends a random string s_1. The protocol LS is used by NMSen to prove to NMRec that either she knows message m and the randomness used to compute $\mathtt{com}$, or she knows the values $(s_0, \mathtt{dec})$, such that $f(s_0 \oplus s_1) = Y$ and $\mathtt{dec}$ is a valid decommitment to s_0 w.r.t. the commitment computed using Π_{wsyn}. We observe that NMSen needs m only when computing the 3rd round, therefore our construction enjoys delayed-input correctness.

3.2.2 Our Compiler

Our compiler needs the following tools:

1. a OWP f that is secure against PPT adversaries and $\tilde{T}_f$-breakable;
2. a non interactive perfectly binding commitment scheme $\mathsf{NI} = (\mathsf{NISen}, \mathsf{NIRec})$ that is T_{NI}-hiding and $\tilde{T}_{\mathsf{NI}}$-breakable;
3. a 3-round extractable *one-one* NM commitment scheme $\Pi_{\mathsf{wsyn}} = (\mathsf{Sen}_{\mathsf{wsyn}}, \mathsf{Rec}_{\mathsf{wsyn}})$ that is T_{wsyn}-hiding/non-malleable, and $\tilde{T}_{\mathsf{wsyn}}$-breakable;
4. the LS proof system $\mathsf{LS} = (\mathcal{P}, \mathcal{V})$ for the language

$$\begin{aligned} L = \Big\{ \big((a, c, z), Y, s_1, \texttt{com}, \texttt{id}\big) : \exists\ (m, \sigma) \text{ s.t.}& \\ \texttt{com} = \mathsf{NISen}(m; \sigma) \texttt{ OR}& \\ \big(\exists(\rho, s_0)\text{s.t. } a = \mathrm{Sen}_{\mathsf{wsyn}}(\texttt{id}, s_0; \rho) \texttt{ AND}& \\ z = \mathrm{Sen}_{\mathsf{wsyn}}(\texttt{id}, c, s_0; \rho) \texttt{ AND } Y = f(s_0 \oplus s_1)\big)\Big\}& \end{aligned}$$

that is T_{LS}-WI for the corresponding relation $\mathsf{Rel_L}$.

Let λ be the security parameter of our scheme. We will use wlog λ also as security parameter for the hardness to invert f with respect to polynomial time adversaries. Then we consider the following hierarchy of security levels for the above tools: $T_f << T_{\mathsf{NI}} << \sqrt{T_{\mathsf{wsyn}}} << T_{\mathsf{wsyn}} << \sqrt{T_{\mathsf{LS}}} << T_{\mathsf{LS}}$ where by "$T << T'$" we mean that "$T \cdot \mathsf{poly}(\lambda) < T'$". We also require that:

- NI is T_{NI}-hiding, but is also $\tilde{T}_{\mathsf{NI}} = \sqrt{T_{\mathsf{wsyn}}}$-breakable;
- Π_{wsyn} is T_{wsyn} hiding/non-malleable, but the hiding is also $\tilde{T}_{\mathsf{wsyn}} = \sqrt{T_{\mathsf{LS}}}$-breakable.

Now we need to define different security parameters, one for each tool involved in the security proof to be consistent with the hierarchy of security levels defined above (a similar use of security parameters has been proposed in [PW10]). Given the security parameter λ of our scheme, we will make use of the following security parameters (all polynomially related to λ and such that the above hierarchy of security levels holds): λ for f, λ_{NI} for NI, λ_{wsyn} for Π_{wsyn}, λ_{LS} for LS.

We denote by Params the function that on input λ outputs $(\lambda_{\mathsf{NI}}, \lambda_{\mathsf{wsyn}}, \lambda_{\mathsf{LS}}, \ell)$ where ℓ is the size of the theorem to be proved using LS[8]. Our concurrent NM commitment scheme $\Pi_{\mathsf{NMCom}} = (\mathsf{NMSen}, \mathsf{NMRec})$ is fully described in Fig. 3.2 and an high-level overview is given in Fig. 3.1.

Theorem 23. Suppose there exist OWPs secure against subexponential-time adversaries, then Π_{NMCom} is a perfectly binding delayed-input commitment scheme.

Before start the proof of security, we recall that LS and NI can be constructed from OWPs secure against subexponential-time adversaries as well as Π_{wsyn} that can be constructed from OWPs secure against subexponential-time using the constructions of [GPR16] or the construction showed in Section 3.4.1.

Proof. **Correctness.** The delayed-input correctness of Π_{NMCom} follows by inspection considering the delayed-input completeness of LS, and the correctness of Π_{wsyn} and NI.

Binding. To prove the binding property we only observe that the message given in output in the decommitment phase of Π_{NMCom} is the message committed using NI. Moreover the decommitment phase of Π_{NMCom} coincides with the decommitment of NI and Π_{wsyn}. Since they are both perfectly binding we have that therefore Π_{NMCom} is perfectly binding too.

Hiding. The hiding property follows directly from the non-malleability property proved in Theorem 24. Indeed the proof of Theorem 24 does not rely on the hiding of Π_{NMCom}. □

In this section we prove our main theorem.

Theorem 24. Suppose there exist OWPs secure against subexponential-time adversaries, then Π_{NMCom} is concurrent (i.e., many-many) non-malleable.

Proof. Since we can use Proposition 1.2.7, we only need to prove that our commitment enjoys one-many non-malleability. More

[8]To compute 1st and 2nd round of LS only the length ℓ of the instance is required.

```
Common input: Security parameters: $\lambda$, $(\lambda_{\mathsf{NI}}, \lambda_{\mathsf{wsyn}}, \lambda_{\mathsf{LS}}, \ell) = \mathsf{Params}(\lambda)$.
$\mathsf{NMSen}$'s identity: $\mathtt{id} \in \{0,1\}^{\lambda}$.
Input to $\mathsf{NMSen}$: $m \in \{0,1\}^{\mathsf{poly}\{\lambda\}}$.

Commitment Phase:
    1. $\mathsf{NMSen} \rightarrow \mathsf{NMRec}$
        1. Pick $s_0 \in \{0,1\}^{\lambda}$.
        2. Pick a randomness $\rho \in \{0,1\}^{\lambda_{\mathsf{nm}}}$ and compute $\mathsf{a}_{\mathsf{wsyn}} = \mathrm{Sen}_{\mathsf{wsyn}}(\mathtt{id}, s_0; \rho)$.
        3. Pick a randomness $\alpha \in \{0,1\}^{\lambda_{\mathsf{LS}}}$ and compute $\mathsf{a}_{\mathsf{LS}} = \mathcal{P}(\ell; \alpha)$.
        4. Send $(\mathsf{a}_{\mathsf{wsyn}}, \mathsf{a}_{\mathsf{LS}})$ to $\mathsf{NMRec}$.
    5. $\mathsf{NMRec} \rightarrow \mathsf{NMSen}$
        1. Pick a randomness $\gamma$ and run $\mathsf{Rec}_{\mathsf{wsyn}}$ on input $(\mathtt{id}, \mathsf{a}_{\mathsf{wsyn}}; \gamma)$ to obtain $\mathsf{c}_{\mathsf{nm}}$.
        2. Pick a randomness $\beta$ and run $\mathcal{V}$ to obtain $\mathsf{c}_{\mathsf{LS}}$.
        3. Pick a random $y \in \{0,1\}^{\lambda}$ and compute $Y = f(y)$.
        4. Send $(\mathsf{c}_{\mathsf{wsyn}}, \mathsf{c}_{\mathsf{LS}}, Y)$ to $\mathsf{NMSen}$.
    5. $\mathsf{NMSen} \rightarrow \mathsf{NMRec}$
        1. Pick a randomness $\sigma \in \{0,1\}^{\lambda_{\mathsf{NI}}}$ and compute $(\mathtt{com}, \mathtt{dec}) = \mathsf{NISen}(m; \sigma)$.
        2. Pick $s_1 \leftarrow \{0,1\}^{\lambda}$.
        3. Compute $(\mathsf{z}_{\mathsf{wsyn}}, \mathtt{dec}_{\mathsf{wsyn}}) = \mathrm{Sen}_{\mathsf{wsyn}}(\mathtt{id}, \mathsf{c}_{\mathsf{wsyn}}, s_0; \rho)$.
        4. Set $x = \big((\mathsf{a}_{\mathsf{wsyn}}, \mathsf{c}_{\mathsf{wsyn}}, \mathsf{z}_{\mathsf{wsyn}}), Y, s_1, \mathtt{com}, \mathtt{id}\big)$ and $w = (m, \sigma, \perp, \perp)$ with $(|x| = \ell)$. Run $\mathsf{z}_{\mathsf{LS}} = \mathcal{P}(x, w, \mathsf{c}_{\mathsf{LS}}; \alpha)$ where $x$ is the theorem to be proven and $w$ is the witness.
        5. Send $(\mathsf{z}_{\mathsf{wsyn}}, \mathtt{com}, \mathsf{z}_{\mathsf{LS}}, s_1)$ to $\mathsf{NMRec}$.
    6. $\mathsf{NMRec}$: Set $x = \big((\mathsf{a}_{\mathsf{wsyn}}, \mathsf{c}_{\mathsf{wsyn}}, \mathsf{z}_{\mathsf{wsyn}}), Y, s_1, \mathtt{com}, \mathtt{id}\big)$ and abort iff $(\mathsf{a}_{\mathsf{LS}}, \mathsf{c}_{\mathsf{LS}}, \mathsf{z}_{\mathsf{LS}})$ is not accepting for $\mathcal{V}$ with respect to $x$.
Decommitment Phase:
    1. $\mathsf{NMSen} \rightarrow \mathsf{NMRec}$: Send $(\mathtt{dec}, m, \mathtt{dec}_{\mathsf{wsyn}}, s_0)$ to $\mathsf{NMRec}$.
    2. $\mathsf{NMRec}$: Accept $m$ as the committed message iff
        1. $\mathsf{NIRec}(\mathtt{dec}, \mathtt{com}, m) = 1$ and
        2. $\mathsf{Rec}_{\mathsf{wsyn}}$ on input $\gamma$, $(\mathsf{a}_{\mathsf{wsyn}}, \mathsf{c}_{\mathsf{wsyn}}, \mathsf{z}_{\mathsf{wsyn}}, \mathtt{id})$, $s_0$ and $\mathtt{dec}_{\mathsf{wsyn}}$ outputs 1.
```

Figure 3.2 Our 3-round concurrent NM commitment scheme.

formally with respect to a one-many adversary $\mathcal{A}$, we need to show that for all $m \in \{0,1\}^{\mathsf{poly}(\lambda)}$ it holds that:

$$\{\mathsf{mim}^{\mathcal{A},m}_{\Pi_{\mathsf{NMCom}}}(z)\}_{z\in\{0,1\}^\star} \approx \{\mathsf{sim}^{S}_{\Pi_{\mathsf{NMCom}}}(1^\lambda, z)\}_{z\in\{0,1\}^\star}$$

where S is the simulator depicted in Fig. 3.3.

This means that the real execution in which the sender runs NMSen to commit to a message m must be indistinguishable with respect to an execution in which a simulator Sim runs internally the MiM adversarial $\mathcal{A}$ sending a commitment of 0^λ, and then forwards the messages that $\mathcal{A}$ sends in the right sessions to receivers $\mathsf{NMRec}_1, \dots, \mathsf{NMRec}_{\mathsf{poly}(\lambda)}$.

We remark that in the security proof we denote by $\tilde{\delta}_i$ a value associated with the i-th right session (where the adversary $\mathcal{A}$ plays with a receiver NMRec_i with $i \in \{1, \dots, \mathsf{poly}(\lambda)\}$) where δ is the corresponding value in the left session. For example, the sender commits to v in the left session while $\mathcal{A}$ commits to $\tilde{v}_i$ in the i-th right session.

To prove the indistinguishability of the above two experiments we proceed by showing 3 hybrid experiments[9] $\mathcal{H}^m_i(z)$ with $i = 1, 2, 3$, where m is the message committed in the left session. Following [LP11b] we denote by $\{\mathsf{mim}^{\mathcal{A}}_{\mathcal{H}^m_i}(z)\}_{z\in\{0,1\}^\star}$ the random variable describing the view of the MiM $\mathcal{A}$ combined with the value it commits in the right interaction in hybrid $\mathcal{H}^m_i(z)$ (as usual, the committed value is replaced by $\perp$ if the right interaction does not correspond to a commitment that can be successfully opened or if $\mathcal{A}$ has copied the identity of the left interaction).

The first hybrid is the experiment in which in the left session NMSen commits to m, while in the right session we run $\mathsf{NMRec}_1, \dots, \mathsf{NMRec}_{\mathsf{poly}(\lambda)}$ for the rights sessions played by $\mathcal{A}$. We refer to this hybrid experiment as $\mathcal{H}^m_1(z)$, details follow below.

$\mathcal{H}^{\mathbf{m}}_{\mathbf{1}}(\mathbf{z})$.

Left session:

[9]We will describe the hybrid experiments in a succinct way focusing on the key steps (e.g., omitting sampling of randomness, generation of parameters $\lambda_{\mathsf{NI}}, \lambda_{\mathsf{wsyn}}, \lambda_{\mathsf{LS}}, \ell$).

1. First round.
   (a) Pick $s_0 \leftarrow \{0,1\}^\lambda$.
   (b) Compute $\mathsf{a}_{\mathsf{wsyn}} = \mathrm{Sen}_{\mathsf{wsyn}}(\mathtt{id}, s_0; \rho)$.
   (c) Compute $\mathsf{a}_{\mathsf{LS}} = \mathcal{P}(1^{\lambda_{\mathsf{LS}}}, \ell; \alpha)$.
   (d) Send $(\mathsf{a}_{\mathsf{wsyn}}, \mathsf{a}_{\mathsf{LS}})$ to $\mathcal{A}$.
2. Third round, upon receiving $(\mathsf{c}_{\mathsf{wsyn}}, \mathsf{c}_{\mathsf{LS}}, Y)$ from $\mathcal{A}$.
   (a) Compute $(\mathtt{com}, \mathtt{dec}) = \mathsf{NISen}(m; \sigma)$.
   (b) Pick $s_1 \leftarrow \{0,1\}^\lambda$.
   (c) Compute $(\mathsf{z}_{\mathsf{wsyn}}, \mathtt{dec}_{\mathsf{wsyn}}) = \mathrm{Sen}_{\mathsf{wsyn}}(\mathtt{id}, \mathsf{c}_{\mathsf{wsyn}}, s_0; \rho)$.
   (d) Set $x = \big((\mathsf{a}_{\mathsf{wsyn}}, \mathsf{c}_{\mathsf{wsyn}}, \mathsf{z}_{\mathsf{wsyn}}), Y, s_1, \mathtt{com}, \mathtt{id}\big)$ and $w = (m, \sigma, \perp, \perp)$ with $(|x| = \ell)$. Run $\mathsf{z}_{\mathsf{LS}} = \mathcal{P}(x, w, \mathsf{c}_{\mathsf{LS}}; \alpha)$.
   (e) Send $(\mathsf{z}_{\mathsf{wsyn}}, \mathtt{com}, \mathsf{z}_{\mathsf{LS}}, s_1)$ to $\mathcal{A}$.

Right sessions: act as a proxy between $\mathcal{A}$ and $\mathsf{NMRec}_1, \dots, \mathsf{NMRec}_{\mathsf{poly}(\lambda)}$.

We have that for all $m \in \{0,1\}^{\mathsf{poly}(\lambda)}$ $\{\mathsf{mim}^{\mathcal{A}}_{\mathcal{H}_1^m}(z)\}_{z\in\{0,1\}^\star}$ clearly corresponds to
$\{\mathsf{mim}^{\mathcal{A},m}_{\Pi_{\mathsf{NMCom}}}(z)\}_{z\in\{0,1\}^\star}$. Before we move on with the sequence of hybrid experiments we need to prove that, for all $i \in \{1, \dots, \mathsf{poly}(\lambda)\}$ $\mathcal{A}$ does not manage to invert any values $\tilde{Y_i}$ in the right sessions by sending a value $\tilde{s_1}_i$ such that $f(\tilde{s_0}_i \oplus \tilde{s_1}_i) = \tilde{Y_i}$ where $\tilde{s_0}_i$ is the message committed in the i-th right session through wsyn.

Lemma 3.2.1. *Let p_i be the probability that in the i-th right session, for $i \in \{1, \dots, \mathsf{poly}(\lambda)\}$, $\mathcal{A}$ sends a value $\tilde{s_1}_i$ such that $f(\tilde{s_1}_i \oplus \tilde{s_0}_i) = \tilde{Y_i}$ where $\tilde{s_0}_i$ is the value committed using* wsyn. *Then $p_i < \nu(\lambda)$ for some negligible function ν.*

Proof. Suppose by contradiction that for a right session i the claim does not hold. We can construct an adversary $\mathcal{A}_f$ that inverts the OWP f in polynomial time. Formally we consider a challenger $\mathcal{C}_f$ of f that chooses a random Y in the range of f and sends it to $\mathcal{A}_f$. $\mathcal{A}_f$ wins if it gives as output y such that $Y = f(y)$. Before describing the adversary we need to consider the augmented machine $\mathcal{M}$ that will be used by $\mathcal{A}_f$. $\mathcal{M}$ internally executes $\mathcal{A}$, and interacts with an external receiver $\mathsf{Rec}_{\mathsf{ext}}$ of the protocol Π_{wsyn} acting as the sender. Formally $\mathcal{M}$ acts as follows.

$\mathcal{M}(\mathbf{Y}, \varphi, \mathbf{z})$

1. Act in the left session with $\mathcal{A}$ (that runs using randomness φ) as in $\mathcal{H}_1^m(z)$.
2. For all $j \neq i \in \{1, \dots \mathsf{poly}(\lambda)\}$ run NMRec_j as in $\mathcal{H}_1^m(z)$. Instead run NMRec_i as described in steps 3, 4 and 5.
3. Upon receiving the 1st round of the i-th right session $(\tilde{\mathsf{a}}_{\mathsf{wsyn}_i}, \tilde{\mathsf{a}}_{\mathsf{LS}_i})$ from $\mathcal{A}$, send $\tilde{\mathsf{a}}_{\mathsf{wsyn}_i}$ to $\mathsf{Rec}_{\mathsf{ext}}$.
4. Upon receiving $\tilde{\mathsf{c}}_{\mathsf{nm_i}}$ from $\mathsf{Rec}_{\mathsf{ext}}$, run as follows:
   (a) Run $\mathcal{V}$ to obtain $\tilde{\mathsf{c}}_{\mathsf{LS_i}}$.
   (b) Set $\tilde{Y}_i = Y$.
   (c) Send $(\tilde{\mathsf{c}}_{\mathsf{wsyn_i}}, \tilde{\mathsf{c}}_{\mathsf{LS_i}}, \tilde{Y}_i)$ to $\mathcal{A}$.
5. Upon receiving the 3rd round of the
   i-th right session $(\tilde{\mathsf{z}}_{\mathsf{wsyn}_i}, \tilde{\mathtt{com}}_i, \tilde{\mathsf{z}}_{\mathsf{LS}_i}, \tilde{s_1}_i)$, set
   $\tilde{x} = \big((\tilde{\mathsf{a}}_{\mathsf{wsyn}_i}, \tilde{\mathsf{c}}_{\mathsf{wsyn}_i}, \tilde{\mathsf{z}}_{\mathsf{wsyn}_i}), \tilde{Y}, \tilde{s_1}_i, \tilde{\mathtt{com}}_i, \tilde{\mathtt{id}}\big)$ and abort iff
   $(\tilde{\mathsf{a}}_{\mathsf{LS}_i}, \tilde{\mathsf{c}}_{\mathsf{LS}_i}, \tilde{\mathsf{z}}_{\mathsf{LS}_i})$ is not accepting for $\mathcal{V}$ with respect to $\tilde{x}$.
6. Send $\tilde{\mathsf{z}}_{\mathsf{wsyn}_i}$ to $\mathsf{Rec}_{\mathsf{ext}}$.

Notice that the above execution of $\mathcal{M}$ is distributed identically to $\mathcal{H}_1^m(z)$ when $\mathsf{Rec}_{\mathsf{ext}}$ plays identically as honest receiver. Now we can conclude the proof of this lemma by describing how $\mathcal{A}_f$ works. $\mathcal{A}_f$ runs the extractor of Π_{wsyn} using $\mathcal{M}$ as sender (recall that an extractor of Π_{wsyn} plays only having access to a sender of Π_{wsyn}). We have that the extractor with non-negligible probability outputs the committed message of an execution that inverts f. By using the randomness φ, $\mathcal{A}_f$ can reconstruct the view of $\mathcal{A}$ and retrive the value $\tilde{s_1}_i$. Therefore $\mathcal{A}$ running in polynomial time[10] outputs with non-negligible probability the value $y = \tilde{s_0}_i \oplus \tilde{s_1}_i$ such that $f(y) = Y$. □

[10]The extractor is an expected polynomial-time algorithm while $\mathcal{A}_f$ must be a strict polynomial-time algorithm. Therefore $\mathcal{A}_f$ will run the extractor up to a given upperbounded number of steps that is higher than the expected running time of the extractor. Obviously with non-negligible probability the *truncated* extraction procedure will be completed successfully and this is sufficient for $\mathcal{A}_f$ to invert f. The same standard argument about truncating the execution of an expected polynomial-time algorithm will be needed later but for simplicity we will not repeat this discussion.

We now consider the second hybrid experiment $\mathcal{H}_2^m(z)$ where in the left session, after receiving Y from $\mathcal{A}$, the sender in time T_f finds a value y such that $Y = f(y)$. Then the sender sets and sends $s_1 = y \oplus s_0$, where s_0 is the value committed using Π_{wsyn}. The only difference between this hybrid experiment and $\mathcal{H}_1^m(z)$ is that $\mathcal{H}_2^m(z)$ runs in time sub-exponential in λ, and the value s_1 is equal to $y \oplus s_0$ where $Y = f(y)$. Formally $\mathcal{H}_2^m(z)$ is the following experiment.

$\boldsymbol{\mathcal{H}_2^m(z)}$.

Left session:

1. First round.
   (a) Pick $s_0 \leftarrow \{0,1\}^\lambda$.
   (b) Compute $\mathsf{a}_{\mathsf{wsyn}} = \mathrm{Sen}_{\mathsf{wsyn}}(\mathtt{id}, s_0; \rho)$.
   (c) Compute $\mathsf{a}_{\mathsf{LS}} = \mathcal{P}(1^{\lambda_{\mathsf{LS}}}, \ell; \alpha)$.
   (d) Send $(\mathsf{a}_{\mathsf{wsyn}}, \mathsf{a}_{\mathsf{LS}})$ to $\mathcal{A}$.
2. Third round. Upon receiving $(\mathsf{c}_{\mathsf{wsyn}}, \mathsf{c}_{\mathsf{LS}}, Y)$ from $\mathcal{A}$.
   (a) Compute $(\mathtt{com}, \mathtt{dec}) = \mathsf{NISen}(m; \sigma)$.
   (b) $\underline{\text{Run in time } T_f \text{ to compute } y \text{ such that } Y = f(y).}$
   (c) $\underline{\text{Set } s_1 = y \oplus s_0}$.
   (d) Compute $(\mathsf{z}_{\mathsf{wsyn}}, \mathtt{dec}_{\mathsf{wsyn}}) = \mathrm{Sen}_{\mathsf{wsyn}}(\mathtt{id}, \mathsf{c}_{\mathsf{wsyn}}, s_0; \rho)$.
   (e) Set $x = \big((\mathsf{a}_{\mathsf{wsyn}}, \mathsf{c}_{\mathsf{wsyn}}, \mathsf{z}_{\mathsf{wsyn}}), Y, s_1, \mathtt{com}, \mathtt{id}\big)$ and $w = (m, \sigma, \bot, \bot)$ with $(|x| = \ell)$. Run $\mathsf{z}_{\mathsf{LS}} = \mathcal{P}(x, w, \mathsf{c}_{\mathsf{LS}}; \alpha)$.
   (f) Send $(\mathsf{z}_{\mathsf{wsyn}}, \mathtt{com}, \mathsf{z}_{\mathsf{LS}}, s_1)$ to NMRec.

Right sessions: Act as a proxy between $\mathcal{A}$ and $\mathsf{NMRec}_1, \dots, \mathsf{NMRec}_{\mathsf{poly}(\lambda)}$.

When switching from $\mathcal{H}_1^m(z)$ to $\mathcal{H}_2^m(z)$ we will make sure that the following two properties hold.

1. For all message $m \in \{0,1\}^{\mathsf{poly}(\lambda)}$ it holds that $\mathsf{mim}^{\mathcal{A}}_{\mathcal{H}_1^m}(z) \approx \mathsf{mim}^{\mathcal{A}}_{\mathcal{H}_2^m}(z)$.[11]

[11]To simplify the notation here, and in the rest of the proof, we will omit that the indistinguishability between two distributions must hold for every auxiliary input z.

2. Let p_i be the probability that in the i-th right session of $\mathcal{H}_2$, for $i \in \{1, \ldots, \mathsf{poly}(\lambda)\}$, $\mathcal{A}$ sends a value $\tilde{s_1}_i$ such that $f(\tilde{s_1}_i \oplus \tilde{s_0}_i) = \tilde{Y}_i$ where $\tilde{s_0}_i$ is the value committed using wsyn. Then $p_i < \nu(\lambda)$ for some negligible function ν.

We now prove that the above two properties hold.

Lemma 3.2.2. *For all message $m \in \{0,1\}^{\mathsf{poly}(\lambda)}$ it holds that $\mathsf{mim}^{\mathcal{A}}_{\mathcal{H}_1^m}(z) \approx \mathsf{mim}^{\mathcal{A}}_{\mathcal{H}_2^m}(z)$.*

Proof. Suppose by contradiction that the distribution of $\mathsf{mim}^{\mathcal{A}}_{\mathcal{H}_1^m}(z)$ is distinguishable from $\mathsf{mim}^{\mathcal{A}}_{\mathcal{H}_2^m}(z)$; this means that there exists a distinguisher $\mathcal{D}$ that can tell apart such two distributions. We now use $\mathcal{D}$ and $\mathcal{A}$ to construct an adversary $\mathcal{A}_{\mathsf{Hiding}}$ that breaks the hiding of Π_{wsyn} in time $\mathsf{poly}(\lambda) \cdot T_{\mathsf{NI}}$ therefore reaching a contradiction[12]. Let $\mathcal{C}_{\mathsf{Hiding}}$ be the challenger of the hiding game, we consider two randomly chosen challenge messages (m_0, m_1) sent to $\mathcal{C}_{\mathsf{Hiding}}$. We now provide a formal description of the adversary $\mathcal{A}_{\mathsf{Hiding}}$.

$\mathcal{A}_{\mathsf{Hiding}}(\mathbf{m_0}, \mathbf{m_1}, \mathbf{z})$

1. Upon receiving the 1st round $\mathsf{a}_{\mathsf{wsyn}}$ from $\mathcal{C}_{\mathsf{Hiding}}$, run as follows:
   (a) Compute $\mathsf{a}_{\mathsf{LS}} = \mathcal{P}(1^{\lambda_{\mathsf{LS}}}, \ell; \alpha)$.
   (b) Send $(\mathsf{a}_{\mathsf{wsyn}}, \mathsf{a}_{\mathsf{LS}})$ to $\mathcal{A}$.
2. Upon receiving $(\mathsf{c}_{\mathsf{wsyn}}, \mathsf{c}_{\mathsf{LS}}, Y)$ from $\mathcal{A}$, send $\mathsf{c}_{\mathsf{wsyn}}$ to $\mathcal{C}_{\mathsf{wsyn}}$.
3. Upon receiving the 3rd round $\mathsf{z}_{\mathsf{wsyn}}$ from $\mathcal{C}_{\mathsf{Hiding}}$, run as follows:
   (a) Compute y such that $Y = f(y)$, set $s_1 = m_0 \oplus y$.
   (b) Compute $(\mathtt{com}, \mathtt{dec}) = \mathsf{NISen}(m; \sigma)$.
   (c) Set $x = \big((\mathsf{a}_{\mathsf{wsyn}}, \mathsf{c}_{\mathsf{wsyn}}, \mathsf{z}_{\mathsf{wsyn}}), Y, s_1, \mathtt{com}, \mathtt{id}\big)$ and $w = (m, \sigma, \bot, \bot)$ with $(|x| = \ell)$. Run $\mathsf{z}_{\mathsf{LS}} = \mathcal{P}(x, w, \mathsf{c}_{\mathsf{LS}}; \alpha)$.

[12]Recall that Π_{wsyn} is secure against adversaries running in time $\mathsf{poly}(\lambda) \cdot T_{\mathsf{NI}} < T_{\mathsf{wsyn}}$.

(d) Send $(\mathsf{z}_{\mathsf{wsyn}}, \mathtt{com}, \mathsf{z}_{\mathsf{LS}}, s_1)$ to $\mathcal{A}$.

4. Simulate $\mathsf{NMRec}_1, \dots, \mathsf{NMRec}_{\mathsf{poly}(\lambda)}$ with $\mathcal{A}$ when $\mathcal{A}$ plays as a sender.
5. Let M be an empty tuple. For all $i \in \{1, \dots, \mathsf{poly}(\lambda)\}$, consider $\tilde{\mathtt{com}}_i$, the non-interactive commitment received by NMRec_i, run in time T_{NI} to compute $\tilde{m}_i$ such that $\exists\ \tilde{\mathtt{dec}}$: $1 = \mathsf{NIRec}(\tilde{\mathtt{com}}_i, \tilde{\mathtt{dec}}, \tilde{m}_i)$ and add $\tilde{m}_i$ to M.
6. Give M and the view of $\mathcal{A}$ to the distinguisher $\mathcal{D}$ and output what $\mathcal{D}$ outputs.

The proof ends with the observation that if $\mathcal{C}_{\mathsf{Hiding}}$ has committed to m_0 then the xor of the committed value with s_1 is equal to y such that $f(y) = Y$, like in $\mathcal{H}_2^m(z)$. If instead $\mathcal{C}_{\mathsf{Hiding}}$ has committed to m_1 then the xor of the committed value and s_1 is equal to a random value, like in $\mathcal{H}_1^m(z)$. $\square$

Lemma 3.2.3. *Let p_i be the probability that in the i-th right session of $\mathcal{H}_2$, for $i \in \{1, \dots, \mathsf{poly}(\lambda)\}$, $\mathcal{A}$ sends a value $\tilde{s_1}_i$ such that $f(\tilde{s_1}_i \oplus \tilde{s_0}_i) = \tilde{Y}_i$ where $\tilde{s_0}_i$ is the value committed using* wsyn. *Then $p_i < \nu(\lambda)$ for some negligible function ν.*

Proof. Suppose by contradiction that for a right session i the claim does not hold. We can construct a distinguisher $\mathcal{D}_{\mathsf{wsyn}}$ and an adversary $\mathcal{A}_{\mathsf{wsyn}}$ that break the non-malleability of Π_{wsyn}. Let $\mathcal{C}_{\mathsf{wsyn}}$ be the challenger of the NM commitment and let (m_0, m_1) be two randomly chosen challenge messages given to $\mathcal{C}_{\mathsf{wsyn}}$.

$\mathcal{A}_{\mathsf{wsyn}}(m_0, m_1, z)$

Left session:

1. Act as $\mathcal{A}_{\mathsf{Hiding}}$ acts in the left session.

Right sessions:

1. For all $j \neq i \in \{1, \dots, \mathsf{poly}(\lambda)\}$ run NMRec_j as in $\mathcal{H}_2^m(z)$. Instead run NMRec_i as described in steps 1.1, 1.2 and 1.3.

(a) Forward $\tilde{\mathsf{a}}_{\mathsf{wsyn}_i}$ to $\mathsf{Rec}_{\mathsf{wsyn}}$.
(b) Upon receiving $\tilde{\mathsf{c}}_{\mathsf{wsyn}}$ from $\mathsf{Rec}_{\mathsf{wsyn}}$, pick a random $\tilde{\mathsf{c}}_{\mathsf{LS}_i}$, pick a random $\tilde{Y}_i$ and send $(\tilde{\mathsf{c}}_{\mathsf{wsyn}_i}, \tilde{\mathsf{c}}_{\mathsf{LS}_i}, \tilde{Y}_i)$ to $\mathcal{A}$.
(c) Upon receiving $\tilde{\mathsf{z}}_{\mathsf{wsyn}_i}$ from $\mathcal{A}$, send it to $\mathsf{Rec}_{\mathsf{wsyn}}$.

Let $\mathsf{mim}^{\mathcal{A}_{\mathsf{wsyn}}}(z)$ be the view of $\mathsf{mim}^{\mathcal{A}_{\mathsf{wsyn}}}(z)$ and the tuple of committed messages in the right session. The distinguisher $\mathcal{D}_{\mathsf{wsyn}}$ takes as input $\mathsf{mim}^{\mathcal{A}_{\mathsf{wsyn}}}(z)$ and acts as follows.

$\mathcal{D}_{\mathsf{wsyn}}(\mathsf{mim}^{\mathcal{A}_{\mathsf{wsyn}}}(z))$: Let $\tilde{s_{0i}}$ be the committed message sent in the i-right session by $\mathcal{A}_{\mathsf{wsyn}}$ to NMRec. Reconstruct the output messages of $\mathcal{A}$ (using the same randomness of $\mathsf{mim}^{\mathcal{A}_{\mathsf{wsyn}}}(z)$) to pick $\tilde{s_{1i}}$. If $f(\tilde{s_{1i}} \oplus \tilde{s_{0i}}) = \tilde{Y}_i$ output 1 and output 0 otherwise. The proof ends with the observation that if $\mathcal{C}_{\mathsf{wsyn}}$ has committed to m_0 then the xor of the committed value with s_{1i} is equal to y such that $f(y) = Y$ like in $\mathcal{H}_2^m$. If instead $\mathcal{C}_{\mathsf{Hiding}}$ has committed to m_1 then the xor of the committed value with s_{1i} is equal to a random string as in $\mathcal{H}_1^m$. □

The third hybrid experiment that we consider is equal to $\mathcal{H}_2^m(z)$ with the difference that the LS proof system is executed using s_0 and the randomness of the non-malleable commitment of s_0. Recall that $f(s_0 \oplus s_1) = Y$. We observe that in the left session of $\mathcal{H}_2^m(z)$ it already holds that $f(s_0 \oplus s_1) = Y$, therefore we can switch the witness used in LS and complete the execution of the proof system. Formally $\mathcal{H}_3^m(z)$ is the following experiment.

$\mathcal{H}_\mathbf{3}^\mathbf{m}(\mathbf{z})$.

Left sessions:

1. First round.
   (a) Pick $s_0 \leftarrow \{0,1\}^\lambda$.
   (b) Compute $\mathsf{a}_{\mathsf{wsyn}} = \mathrm{Sen}_{\mathsf{wsyn}}(\mathtt{id}, s_0; \rho)$.
   (c) Compute $\mathsf{a}_{\mathsf{LS}} = \mathcal{P}(1^{\lambda_{\mathsf{LS}}}, \ell; \alpha)$.
   (d) Send $(\mathsf{a}_{\mathsf{wsyn}}, \mathsf{a}_{\mathsf{LS}})$ to $\mathcal{A}$.
2. Third round. Upon receiving $(\mathsf{c}_{\mathsf{wsyn}}, \mathsf{c}_{\mathsf{LS}}, Y)$ from $\mathcal{A}$.

(a) Compute $(\texttt{com}, \texttt{dec}) = \mathsf{NISen}(m; \sigma)$.
(b) Run in time T_f to compute y such that $Y = f(y)$.
(c) Set $s_1 = s_0 \oplus y$.
(d) Compute $(\mathsf{z}_{\mathsf{wsyn}}, \texttt{dec}_{\mathsf{wsyn}}) = \mathrm{Sen}_{\mathsf{wsyn}}(\texttt{id}, \mathsf{c}_{\mathsf{wsyn}}, s_0; \rho)$.
(e) Compute $(\texttt{com}, \texttt{dec}) = \mathsf{NISen}(1^{\lambda_{\mathsf{NI}}}, m; \sigma)$.
(f) Set $x = \big((\mathsf{a}_{\mathsf{wsyn}}, \mathsf{c}_{\mathsf{wsyn}}, \mathsf{z}_{\mathsf{wsyn}}), Y, s_1, \texttt{com}, \texttt{id}\big)$ and $\underline{w = (\perp, \perp, s_0, \rho)}$ with $(|x| = \ell)$. Run $\mathsf{z}_{\mathsf{LS}} = \mathcal{P}(x, w, \mathsf{c}_{\mathsf{LS}}; \alpha)$.
(g) Send $(\mathsf{z}_{\mathsf{wsyn}}, \texttt{com}, \mathsf{z}_{\mathsf{LS}}, s_1)$ to $\mathcal{A}$.

Right sessions: Act as a proxy between $\mathcal{A}$ and $\mathsf{NMRec}_1, \dots, \mathsf{NMRec}_{\mathsf{poly}(\lambda)}$.

Even in this case we need to prove the following two properties.

1. For all message $m \in \{0,1\}^{\mathsf{poly}(\lambda)}$ it holds that $\mathsf{mim}^{\mathcal{A}}_{\mathcal{H}_2^m}(z) \approx \mathsf{mim}^{\mathcal{A}}_{\mathcal{H}_3^m}(z)$.
2. Let p_i be the probability that in the i-th right session of $\mathcal{H}_3$, for any $i \in \{1, \dots, \mathsf{poly}(\lambda)\}$, $\mathcal{A}$ sends a value $\tilde{s_1}_i$ such that $f(\tilde{s_1}_i \oplus \tilde{s_0}_i) = \tilde{Y}_i$ where $\tilde{s_0}_i$ is the value committed using wsyn. Then $p_i < \nu(\lambda)$ for some negligible function ν.

Lemma 3.2.4. *For any message $m \in \{0,1\}^{\mathsf{poly}(\lambda)}$ it holds that $\mathsf{mim}^{\mathcal{A}}_{\mathcal{H}_2^m}(z) \approx \mathsf{mim}^{\mathcal{A}}_{\mathcal{H}_3^m}(z)$.*

Proof. Suppose by contradiction that there exist a adversary $\mathcal{A}$ and a distinguisher $\mathcal{D}$ that can tell apart such two distributions. We can use this adversary and the associated distinguisher to construct an adversary $\mathcal{A}_{\mathsf{LS}}$ for the T_{LS}-witness-indistinguishable property of the LS protocol. Let $\mathcal{C}_{\mathsf{LS}}$ be the WI challenger, the adversary works as follows. $\mathcal{A}_{\mathsf{LS}}(\mathbf{z})$

1. Pick $s_0 \leftarrow \{0,1\}^\lambda$.
2. Compute $\mathsf{a}_{\mathsf{wsyn}} = \mathrm{Sen}_{\mathsf{wsyn}}(\texttt{id}, s_0; \rho)$.
3. Upon receiving a_{LS} from $\mathcal{C}_{\mathsf{LS}}$, send $(\mathsf{a}_{\mathsf{wsyn}}, \mathsf{a}_{\mathsf{LS}})$ to $\mathcal{A}$.
4. Upon receiving $(\mathsf{c}_{\mathsf{wsyn}}, \mathsf{c}_{\mathsf{LS}}, Y)$ from $\mathcal{A}$ run as follows:

   (a) Run in time T_f to compute y such that $Y = f(y)$.

   (b) Set $s_1 = s_0 \oplus y$.

(c) Compute $(\mathsf{z}_{\mathsf{wsyn}}, \mathtt{dec}_{\mathsf{wsyn}}) = \mathrm{Sen}_{\mathsf{wsyn}}(\mathtt{id}, \mathsf{c}_{\mathsf{wsyn}}, s_0; \rho)$.

(d) Compute $(\mathtt{com}, \mathtt{dec}) = \mathsf{NISen}(1^{\lambda_{\mathsf{NI}}}, m; \sigma)$.

(e) Set $x = \big((\mathsf{a}_{\mathsf{wsyn}}, \mathsf{c}_{\mathsf{wsyn}}, \mathsf{z}_{\mathsf{wsyn}}), Y, s_1, \mathtt{com}, \mathtt{id}\big)$, $w_0 = (\bot, \bot, s_0, \rho)$,
$w_1 = (m, \sigma, \bot, \bot)$ and send $(x, \mathsf{c}_{\mathsf{LS}}, w_0, w_1)$ to $\mathcal{C}_{\mathsf{LS}}$.

5. Upon receiving z_{LS} from $\mathcal{C}_{\mathsf{LS}}$, send $(\mathsf{z}_{\mathsf{wsyn}}, \mathtt{com}, \mathsf{z}_{\mathsf{LS}})$ to $\mathcal{A}$.
6. Simulate $\mathsf{NMRec}_1, \dots, \mathsf{NMRec}_{\mathsf{poly}(\lambda)}$ with $\mathcal{A}$, when $\mathcal{A}$ plays as a sender.
7. Let M be an empty tuple. For all $i \in \{1, \dots, \mathsf{poly}(\lambda)\}$, consider $\tilde{\mathtt{com}}_i$, the non-interactive commitment received by NMRec_i, and run in time $\tilde{T}_{\mathsf{NI}}$ to compute $\tilde{m}_i$ such that $\exists\ \tilde{\mathtt{dec}}$: $1 = \mathsf{NIRec}(\tilde{\mathtt{com}}_i, \tilde{\mathtt{dec}}, \tilde{m}_i)$ and add $\tilde{m}_i$ to M.
8. Give M and the view of $\mathcal{A}$ to the distinguisher $\mathcal{D}$.
9. Output what $\mathcal{D}$ outputs.

□

The proof ends with the observation that if $\mathcal{C}_{\mathsf{LS}}$ has has used as witness the randomness of the non-malleable commitment of the value s_0 such that $f(s_0 \oplus s_1) = Y$ then we are in the hybrid experiment $\mathcal{H}_3^m(z)$. If instead $\mathcal{C}_{\mathsf{LS}}$ has used as a witness the randomness used to compute the non-interactive commitment NI then we are in the hybrid experiment $\mathcal{H}_2^m(z)$.

Lemma 3.2.5. *Let p_i be the probability that in the i-th right session of $\mathcal{H}_3^m$, for $i \in \{1, \dots, \mathsf{poly}(\lambda)\}$, $\mathcal{A}$ sends a value $\tilde{s_1}_i$ such that $f(\tilde{s_1}_i \oplus \tilde{s_0}_i) = \tilde{Y}_i$ where $\tilde{s_0}_i$ is the value committed using* wsyn. *Then $p_i < \nu(\lambda)$ for some negligible function ν.*

Proof. Suppose by contradiction that for a right session i the claim does not hold, then we can construct an adversary $\mathcal{A}'_{\mathsf{LS}}$ for the T_{LS} witness-indistinguishable property of the LS protocol. Let $\mathcal{C}_{\mathsf{LS}}$ be the WI challenger, the adversary works as follows.

$\mathcal{A}'_{\mathsf{LS}}(\mathbf{z})$

1. Pick $s_0 \leftarrow \{0,1\}^\lambda$.
2. Compute $\mathsf{a}_{\mathsf{wsyn}} = \mathrm{Sen}_{\mathsf{wsyn}}(\mathtt{id}, s_0; \rho)$.
3. Upon receiving a_{LS} from $\mathcal{C}_{\mathsf{LS}}$, send $(\mathsf{a}_{\mathsf{wsyn}}, \mathsf{a}_{\mathsf{LS}})$ to $\mathcal{A}$.
4. Upon receiving $(\mathsf{c}_{\mathsf{wsyn}}, \mathsf{c}_{\mathsf{LS}}, Y)$ from $\mathcal{A}$, run as follow:

   (a) Run in time T_f to compute y such that $Y = f(y)$.

   (b) Set $s_1 = s_0 \oplus y$.

   (c) Compute $(\mathsf{z}_{\mathsf{wsyn}}, \mathtt{dec}_{\mathsf{wsyn}}) = \mathrm{Sen}_{\mathsf{wsyn}}(\mathtt{id}, \mathsf{c}_{\mathsf{wsyn}}, s_0; \rho)$.

   (d) Compute $(\mathtt{com}, \mathtt{dec}) = \mathsf{NISen}(1^{\lambda_{\mathsf{NI}}}, m; \sigma)$.

   (e) Set $x = \big((\mathsf{a}_{\mathsf{wsyn}}, \mathsf{c}_{\mathsf{wsyn}}, \mathsf{z}_{\mathsf{wsyn}}), Y, s_1, \mathtt{com}, \mathtt{id}\big)$, $w_0 = (\perp, \perp$
   $, s_0, \rho)$,
   $w_1 = (m, \sigma, \perp, \perp)$ and send $(x, \mathsf{c}_{\mathsf{LS}}, w_0, w_1)$ to $\mathcal{C}_{\mathsf{LS}}$.

5. Upon receiving z_{LS} from $\mathcal{C}_{\mathsf{LS}}$, send $(\mathsf{z}_{\mathsf{wsyn}}, \mathtt{com}, \mathsf{z}_{\mathsf{LS}})$ to $\mathcal{A}$.
6. Simulate $\mathsf{NMRec}_1, \dots, \mathsf{NMRec}_{\mathsf{poly}(\lambda)}$ with $\mathcal{A}$, when $\mathcal{A}$ plays as a sender.
7. Run in time $\tilde{T}_{\mathsf{wsyn}}$ to extract the value $\tilde{s_0}_i$ from the non-malleable commitment sent by $\mathcal{A}$ in the i-th session. Output 1 if $f(\tilde{s_0}_i \oplus \tilde{s_1}_i) = \tilde{Y}_i$ and output 0 otherwise.

The proof ends with the observation that if $\mathcal{C}_{\mathsf{LS}}$ has used $w_0 = (\perp, \perp, s_0, \rho)$ as a witness then $\mathcal{A}$ acts as in $\mathcal{H}_3^m(z)$, sending with non-negligible probability two shares such that the xor of them gives a puzzle solution. If $\mathcal{C}_{\mathsf{LS}}$ has used $w_1 = (m, \sigma, \perp, \perp)$ then the xor of the two shares is with overwhelming probability different from a puzzle solution as in $\mathcal{H}_2^m(z)$. □

The next hybrid experiment that we consider is $\mathcal{H}_3^0(z)$. The only differences between this hybrid experiment and $\mathcal{H}_3^m(z)$ is that the sender, using NI, commits to a message 0^λ instead of m. Formally the hybrid experiment is the following.

$\boldsymbol{\mathcal{H}_3^0(\mathbf{z})}$.

Left session:

1. First round.
   (a) Pick $s_0 \leftarrow \{0,1\}^\lambda$.
   (b) Compute $\mathsf{a}_{\mathsf{wsyn}} = \mathrm{Sen}_{\mathsf{wsyn}}(\mathtt{id}, s_0; \rho)$.
   (c) Compute $\mathsf{a}_{\mathsf{LS}} = \mathcal{P}(\ell; \alpha)$.
   (d) Send $(\mathsf{a}_{\mathsf{wsyn}}, \mathsf{a}_{\mathsf{LS}})$ to $\mathcal{A}$.
2. Third round. Upon receiving $(\mathsf{c}_{\mathsf{wsyn}}, \mathsf{c}_{\mathsf{LS}}, Y)$ from $\mathcal{A}$, run as follows:
   (a) Run in time T_f to compute y such that $Y = f(y)$.
   (b) Set $s_1 = s_0 \oplus y$.
   (c) Compute $(\mathsf{z}_{\mathsf{wsyn}}, \mathtt{dec}_{\mathsf{wsyn}}) = \mathrm{Sen}_{\mathsf{wsyn}}(\mathtt{id}, \mathsf{c}_{\mathsf{wsyn}}, s_0; \rho)$.
   (d) Compute $(\mathtt{com}, \mathtt{dec}) = \underline{\mathsf{NISen}(0^\lambda; \sigma)}$.
   (e) Set $x = \big((\mathsf{a}_{\mathsf{wsyn}}, \mathsf{c}_{\mathsf{wsyn}}, \mathsf{z}_{\mathsf{wsyn}}), Y, s_1, \mathtt{com}, \mathtt{id}\big)$ and $w = (\perp, \perp, s_0, \rho)$ with $(|x| = \ell)$. Run $\mathsf{z}_{\mathsf{LS}} = \mathcal{P}(x, w, \mathsf{c}_{\mathsf{LS}}; \alpha)$.
   (f) Send $(\mathsf{z}_{\mathsf{wsyn}}, \mathtt{com}, \mathsf{z}_{\mathsf{LS}}, s_1)$ to $\mathcal{A}$.

Right sessions: Act as a proxy between $\mathcal{A}$ and $\mathsf{NMRec}_1, \dots, \mathsf{NMRec}_{\mathsf{poly}(\lambda)}$.

We now prove the following properties.

1. Let p_i be the probability that in the i-th right session of $\mathcal{H}_3^0$, for any $i \in \{1, \dots, \mathsf{poly}(\lambda)\}$, $\mathcal{A}$ sends a value $\tilde{s_1}_i$ such that $f(\tilde{s_1}_i \oplus \tilde{s_0}_i) = \tilde{Y}_i$ where $\tilde{s_0}_i$ is the value committed using wsyn. Then $p_i < \nu(\lambda)$ for some negligible function ν.
2. For any message $m \in \{0,1\}^{\mathsf{poly}(\lambda)}$ it holds that $\mathsf{mim}^{\mathcal{A}}_{\mathcal{H}_3^m}(z) \approx \mathsf{mim}^{\mathcal{A}}_{\mathcal{H}_3^0}(z)$.

Lemma 3.2.6. *Let p_i be the probability that in the i-th right session of $\mathcal{H}_3^0$, for $i \in \{1, \dots, \mathsf{poly}(\lambda)\}$, $\mathcal{A}$ sends a value $\tilde{s_1}_i$ such that $f(\tilde{s_1}_i \oplus \tilde{s_0}_i) = \tilde{Y}_i$ where $\tilde{s_0}_i$ is the value committed using* wsyn. *Then $p_i < \nu(\lambda)$ for some negligible function ν.*

Proof. Suppose by contradiction that there exists a right session $i \in \{1, \dots, \mathsf{poly}(\lambda)\}$ in which $\mathcal{A}$ commit to a string $\tilde{s_0}$ such that $f(\tilde{s_0}_i \oplus \tilde{s_1}_i) = \tilde{Y}_i$ using Π_{wsyn}. Then we can construct an adversary $\mathcal{A}_{\mathsf{NI}}$ that breaks the hiding property of the non interactive commitment scheme NI. Let $\mathcal{C}_{\mathsf{NI}}$ be the challenger that on input $m_0 = 0^\lambda$ and $m_1 = m$, picks a random bit b, computes $(\mathtt{com}, \mathtt{dec}) = \mathsf{NISen}(1^{\lambda_{\mathsf{NI}}}, m_b; \sigma)$ and sends $\mathtt{com}$ to $\mathcal{A}_{\mathsf{NI}}$.

Before describing $\mathcal{A}_{\mathsf{NI}}$ we need to consider, as in the proof of Lemma 3.2.1, a machine $\mathcal{M}$ that internally executes $\mathcal{A}$, and interacts with a receiver $\mathsf{Rec}_{\mathsf{ext}}$ of the protocol Π_{wsyn} acting as the sender.

Formally $\mathcal{M}$ acts as follows.

$\mathcal{M}(\mathtt{com}, \varphi, \mathbf{z})$

Run $\mathcal{A}$ using randomness φ.

1. Pick $s_0 \leftarrow \{0,1\}^\lambda$.
2. Compute $\mathsf{a}_{\mathsf{wsyn}} = \mathrm{Sen}_{\mathsf{wsyn}}(\mathtt{id}, s_0; \rho)$.
3. Compute $\mathsf{a}_{\mathsf{LS}} = \mathcal{P}(1^{\lambda_{\mathsf{LS}}}, \ell; \alpha)$.
4. Send $(\mathsf{a}_{\mathsf{wsyn}}, \mathsf{a}_{\mathsf{LS}})$ to $\mathcal{A}$.
5. Upon receiving $(\mathsf{c}_{\mathsf{wsyn}}, \mathsf{c}_{\mathsf{LS}}, Y)$ from $\mathcal{A}$, run as follows:
   (a) Run in time T_f to compute y such that $Y = f(y)$.
   (b) Set $s_1 = s_0 \oplus y$.
   (c) Compute $(\mathsf{z}_{\mathsf{wsyn}}, \mathtt{dec}_{\mathsf{wsyn}}) = \mathrm{Sen}_{\mathsf{wsyn}}(\mathtt{id}, \mathsf{c}_{\mathsf{wsyn}}, s_0; \rho)$.
   (d) Set $x = \big((\mathsf{a}_{\mathsf{wsyn}}, \mathsf{c}_{\mathsf{wsyn}}, \mathsf{z}_{\mathsf{wsyn}}), Y, s_1, \mathtt{com}, \mathtt{id}\big)$ and $w = (\perp, \perp, s_0, \rho)$ with $(|x| = \ell)$. Run $\mathsf{z}_{\mathsf{LS}} = \mathcal{P}(x, w, \mathsf{c}_{\mathsf{LS}}; \alpha)$.
   (e) Send $(\mathsf{z}_{\mathsf{wsyn}}, \mathtt{com}, \mathsf{z}_{\mathsf{LS}}, s_1)$ to $\mathcal{A}$.
6. Let $i \in \{1, \dots, \mathsf{poly}(\lambda)\}$ be the right session that contradicts the claim. For all $j \neq i \in \{1, \dots \mathsf{poly}(\lambda)\}$ run NMRec_j as in $\mathcal{H}_4(m, z)$. Run NMRec_i as follows.
   (a) Upon receiving the 1rd round of the i-th right session $(\tilde{\mathsf{a}}_{\mathsf{wsyn}_i}, \tilde{\mathsf{a}}_{\mathsf{LS}_i})$ from $\mathcal{A}$, send $\tilde{\mathsf{a}}_{\mathsf{wsyn}_i}$ to the external receiver $\mathsf{Rec}_{\mathsf{ext}}$.
   (b) Upon receiving $\tilde{\mathsf{c}}_{\mathsf{nm}_\mathsf{i}}$ from $\mathsf{Rec}_{\mathsf{ext}}$, run as follows:
      i. Run $\mathcal{V}$ to obtain $\tilde{\mathsf{c}}_{\mathsf{LS}_\mathsf{i}}$.
      ii. Pick a random $\tilde{Y}_i$.
      iii. Send $(\tilde{\mathsf{c}}_{\mathsf{wsyn}_\mathsf{i}}, \tilde{\mathsf{c}}_{\mathsf{LS}_\mathsf{i}}, \tilde{Y}_i)$ to $\mathcal{A}$.

(c) Upon receiving the 3rd round of the i-th right session $(\tilde{\mathsf{z}}_{\mathsf{wsyn}_i}, \tilde{\mathtt{com}}_i, \tilde{\mathsf{z}}_{\mathsf{LS}_i}, \tilde{s_{1i}})$,
set $\tilde{x} = \big((\tilde{\mathsf{a}}_{\mathsf{wsyn}_i}, \tilde{\mathsf{c}}_{\mathsf{wsyn}_i}, \tilde{\mathsf{z}}_{\mathsf{wsyn}_i}), \tilde{Y}, \tilde{s_{1i}}, \tilde{\mathtt{com}}_i, \tilde{\mathtt{id}}\big)$ and abort iff $(\tilde{\mathsf{a}}_{\mathsf{LS}_i}, \tilde{\mathsf{c}}_{\mathsf{LS}_i}, \tilde{\mathsf{z}}_{\mathsf{LS}_i})$ is not accepted by $\mathcal{V}$ with respect to $\tilde{x}$.

(d) Send $\tilde{\mathsf{z}}_{\mathsf{wsyn}_i}$ to $\mathsf{Rec}_{\mathsf{ext}}$.

Now we can conclude the proof of this lemma by describing how $\mathcal{A}_{\mathsf{NI}}$ works. $\mathcal{A}_{\mathsf{NI}}$ runs the extractor of the protocol Π_{wsyn} using $\mathcal{M}$ as sender (recall that an extractor of Π_{wsyn} plays only having access to a sender of Π_{wsyn}). Since the extractor with non-negligible probability outputs the committed message we have that $\mathcal{A}_{\mathsf{NI}}$ retrives $\tilde{s_{0i}}$. Moreover $\mathcal{A}_{\mathsf{NI}}$ gets $\tilde{s_{1i}}$ by reconstructing the view of $\mathcal{A}$ using the randomness φ. Since by contradiction $\mathcal{A}$ contradicts the claim of this lemma, we have that $\mathcal{A}_{\mathsf{NI}}$ can break the hiding of NI because $f(\tilde{s_{0i}} \oplus \tilde{s_{1i}}) = \tilde{Y}$ with non-negligible probability in $\mathcal{H}_3^0(z)$ where $m_0 = 0^\lambda$ is committed in $\mathtt{com}$, while the same happens with negligible probability only in $\mathcal{H}_3^m(z)$ where $m_1 = m$. Therefore if this happens, $\mathcal{A}_{\mathsf{NI}}$ outputs 0, otherwise $\mathcal{A}_{\mathsf{NI}}$ outputs a random bit. □

Lemma 3.2.7. *For any message* $m \in \{0,1\}^{\mathsf{poly}(\lambda)}$ *it holds that* $\mathsf{mim}^{\mathcal{A}}_{\mathcal{H}_3^m}(z) \approx \mathsf{mim}^{\mathcal{A}}_{\mathcal{H}_3^0}(z)$.

Proof. Suppose by contradiction that there exists a distinguisher $\mathcal{D}$ and an adversary $\mathcal{A}$ such that $\mathsf{mim}^{\mathcal{A}}_{\mathcal{H}_3^m}(z)$ is distinguishable from $\mathsf{mim}^{\mathcal{A}}_{\mathcal{H}_3^0}(z)$ then we can construct an adversary $\mathcal{A}_{\mathsf{NI}}$ that breaks the hiding property of the non-interactive commitment scheme NI. Let $\mathcal{C}_{\mathsf{NI}}$ be the challenger that on input $m_0 = 0^\lambda$ and $m_1 = m$, picks a random bit b, computes $(\mathtt{com}, \mathtt{dec}) = \mathsf{NISen}(1^{\lambda_{\mathsf{NI}}}, m_b; \sigma)$ and sends $\mathtt{com}$ to $\mathcal{A}_{\mathsf{NI}}$. Before describing $\mathcal{A}_{\mathsf{NI}}$, we consider the following experiment $\mathcal{E}_{m_b}(\varphi, \mathtt{com}, z)$.

$\mathcal{E}_{\mathbf{m_b}}(\varphi, \mathtt{com}, \mathbf{z})$.
The randomness required from all next steps is take from φ.

Run $\mathcal{A}(z)$.

Left session:

1. First round.
   (a) Pick $s_0 \leftarrow \{0,1\}^\lambda$.
   (b) Compute $\mathsf{a}_{\mathsf{wsyn}} = \mathrm{Sen}_{\mathsf{wsyn}}(\mathtt{id}, s_0; \rho)$.
   (c) Compute $\mathsf{a}_{\mathsf{LS}} = \mathcal{P}(\ell; \alpha)$.
   (d) Send $(\mathsf{a}_{\mathsf{wsyn}}, \mathsf{a}_{\mathsf{LS}})$ to $\mathcal{A}$.
2. Third round. Upon receiving $(\mathsf{c}_{\mathsf{wsyn}}, \mathsf{c}_{\mathsf{LS}}, Y)$ from $\mathcal{A}$, run as follows:
   (a) Run in time T_f to compute y such that $Y = f(y)$.
   (b) Set $s_1 = s_0 \oplus y$.
   (c) Compute $(\mathsf{z}_{\mathsf{wsyn}}, \mathtt{dec}_{\mathsf{wsyn}}) = \mathrm{Sen}_{\mathsf{wsyn}}(\mathtt{id}, \mathsf{c}_{\mathsf{wsyn}}, s_0; \rho)$.
   (d) Set $x = \big((\mathsf{a}_{\mathsf{wsyn}}, \mathsf{c}_{\mathsf{wsyn}}, \mathsf{z}_{\mathsf{wsyn}}), Y, s_1, \mathtt{com}, \mathtt{id}\big)$ and $w = (\perp, \perp, s_0, \rho)$ with $(|x| = \ell)$. Run $\mathsf{z}_{\mathsf{LS}} = \mathcal{P}(x, w, \mathsf{c}_{\mathsf{LS}}; \alpha)$.
   (e) Send $(\mathsf{z}_{\mathsf{wsyn}}, \mathtt{com}, \mathsf{z}_{\mathsf{LS}}, s_1)$ to $\mathcal{A}$.

Right sessions: Act as a proxy between $\mathcal{A}$ and $\mathsf{NMRec}_1, \dots, \mathsf{NMRec}_{\mathsf{poly}(\lambda)}$.

Now we are ready to describe the adversary $\mathcal{A}_{\mathsf{NI}}$ for the hiding of NI. $\mathcal{A}_{\mathsf{NI}}$ executes the following steps.

1. Let M be an empty tuple. $\mathcal{A}_{\mathsf{NI}}$ runs $\mathcal{E}_{\mathbf{m_b}}(\varphi, \mathtt{com}, \mathbf{z})$.

2. For all $i \in \{1, \dots, \mathsf{poly}(\lambda)\}$, $\mathcal{A}_{\mathsf{NI}}$ runs the extractor of LS on the i-th right session of the execution of $\mathcal{E}_{\mathbf{m_b}}(\varphi, \mathtt{com}, \mathbf{z})$ obtaining $\tilde{m}_i$ and adds it to M.

3. Using the randomness φ, $\mathcal{A}_{\mathsf{NI}}$ reconstructs the view of $\mathcal{A}$ in the execution of $\mathcal{E}_{\mathbf{m_b}}(\varphi, \mathtt{com}, \mathbf{z})$. Use such view and M as input to $\mathcal{D}$.

4. Output what $\mathcal{D}$ outputs.

The proof ends with the observation that if $\mathcal{C}_{\mathsf{NI}}$ has committed to 0^λ then the view of $\mathcal{A}$ and the distribution of the committed messages coincide with $\mathcal{H}_3^0$, otherwise they coincide with $\mathcal{H}_3^m$. $\square$

```
**Common input:** Security parameters: $\lambda$, $(\lambda_{\mathsf{NI}}, \lambda_{\mathsf{wsyn}}, \lambda_{\mathsf{LS}}, \ell) = \mathsf{Params}(\lambda)$. Identity: $\mathtt{id} \in \{0,1\}^\lambda$.
**Internal simulation of the left session:**
    1. Pick $s_0 \leftarrow \{0,1\}^\lambda$.
    2. Pick a randomness $\rho$, and compute $(\mathtt{dec}_{\mathsf{wsyn}}, \mathsf{a}_{\mathsf{wsyn}}) = \mathrm{Sen}_{\mathsf{wsyn}}(\mathtt{id}, s_0; \rho)$.
    3. Pick a randomness $\alpha$ and compute $\mathsf{a}_{\mathsf{LS}} = \mathcal{P}(\ell; \alpha)$.
    4. Send $(\mathsf{a}_{\mathsf{wsyn}}, \mathsf{a}_{\mathsf{LS}})$ to $\mathcal{A}$.
    5. Upon receiving $(\mathsf{c}_{\mathsf{wsyn}}, \mathsf{c}_{\mathsf{LS}}, Y)$ from $\mathcal{A}$.
        1. Pick a randomness $\sigma$ and compute $(\mathtt{com}, \mathtt{dec}) = \mathsf{NISen}(1^{\lambda_{\mathsf{NI}}}, 0^\lambda; \sigma)$.
        2. Pick $s_1 \leftarrow \{0,1\}^\lambda$.
        3. Compute $\mathsf{z}_{\mathsf{wsyn}} = \mathrm{Sen}_{\mathsf{wsyn}}(\mathtt{id}, \mathsf{c}_{\mathsf{wsyn}}, s_0; \rho)$.
        4. Set $x = \big((\mathsf{a}_{\mathsf{wsyn}}, \mathsf{c}_{\mathsf{wsyn}}, \mathsf{z}_{\mathsf{wsyn}}), Y, s_1, \mathtt{com}, \mathtt{id}\big)$ and $w = (0^\lambda, \sigma, \perp, \perp)$ with $(|x| = \ell)$. Run $\mathsf{z}_{\mathsf{LS}} = \mathcal{P}(x, w, \mathsf{c}_{\mathsf{LS}}; \alpha)$ where $x$ is the theorem to be proven and $w$ is the witness.
        5. Send $(\mathsf{z}_{\mathsf{wsyn}}, \mathtt{com}, \mathsf{z}_{\mathsf{LS}}, s_1)$ to $\mathcal{A}$.
**Stand-alone commitment:**
    1. $\mathsf{Sim}$ acts as a proxy between $\mathcal{A}$ and $\mathsf{NMRec}_i$ for $i = 1, \ldots, \mathsf{poly}(\lambda)$.
```

Figure 3.3 The simulator Sim.

The entire security proof now is almost over because we have proved that for all $m \in \{0,1\}^{\mathsf{poly}(\lambda)}$ the following relation holds:

$$
\begin{aligned}
\{\mathsf{mim}^{\mathcal{A},m}_{\Pi_{\mathsf{NMCom}}}(z)\}_{z\in\{0,1\}^\star} &= \{\mathsf{mim}^{\mathcal{A}}_{\mathcal{H}_1^m}(z)\}_{z\in\{0,1\}^\star} \\
\approx \{\mathsf{mim}^{\mathcal{A}}_{\mathcal{H}_2^m}(z)\}_{z\in\{0,1\}^\star} &\approx \{\mathsf{mim}^{\mathcal{A}}_{\mathcal{H}_3^m}(z)\}_{z\in\{0,1\}^\star} \\
\approx \{\mathsf{mim}^{\mathcal{A}}_{\mathcal{H}_3^0}(z)\}_{z\in\{0,1\}^\star} &\approx \{\mathsf{mim}^{\mathcal{A}}_{\mathcal{H}_2^0}(z)\}_{z\in\{0,1\}^\star} \\
\approx \{\mathsf{mim}^{\mathcal{A}}_{\mathcal{H}_1^0}(z)\}_{z\in\{0,1\}^\star} &= \{\mathsf{sim}^{S}_{\Pi_{\mathsf{NMCom}}}(1^\lambda, z)\}_{z\in\{0,1\}^\star}.
\end{aligned}
$$

We show in Figure 3.3 the simulator Sim.

We observe that in this proof we had to consider a delayed-input version of our commitment scheme. Indeed, the sender can choose the message m to be committed by sending the non-interactive commitment `com` of the message m in the third round. It is easy to see that the same security proof still works when the non-interactive commitment is sent in the 1st round, but then clearly the delayed-input property is lost. □

3.3 More 3-Round Protocols Against Concurrent MiM Attacks

In this section we show how to obtain some forms of 3-round arguments of knowledge and of 3-round identification schemes that are secure against concurrent MiM attacks.

3.3.1 NMWI Argument Systems

The definition of non-malleable witness indistinguishability (NMWI) given in [OPV08] requires that the witness *encoded in the proof* given by the MiM $\mathcal{A}$ be independent of the witness used by the honest prover in his proof. The concept of witness encoded in a proof becomes clear when considering *commit-and-prove* argument systems. In such arguments, the transcript includes a commitment that encodes in an unambiguous way the witness used by the prover. In a NMWI commit-and-prove argument system, the witness encoded in the proof produced by the $\mathcal{A}$ must be independent of the witness used (and thus encoded by the honest prover) in the proof in which $\mathcal{A}$ acts as a verifier. Similarly to the case of non-malleable commitments, one can give a definition with or without sessions ids. Here we use the one without sessions ids since it is more useful in the applications.

Let $\mathcal{A}$ be a MiM interacting in the left proof with $\mathcal{P}$ that is running on input x and witness w. In the right proof $\mathcal{A}$ interacts with $\mathcal{V}$ on common input $\tilde{x}$ chosen by $\mathcal{A}$. We denote by z the auxiliary information available to $\mathcal{A}$. NMWI is defined in terms of the random variable $\mathsf{wmim}^{\mathcal{A}}(x, w, z)$ that is the view of $\mathcal{A}$ that we denote by $\mathsf{View}^{\mathcal{P}}_{\mathcal{A}}(x, w, z)$ (i.e., the view of $\mathcal{A}$ when running with z

as auxiliary input and playing with $\mathcal{P}$ that runs on input (x, w)) and the witness encoded in the right proof given by $\mathcal{A}$. If the right proof is not accepting or the transcript is identical to the one of the left proof then the witness encoded is $\perp$; otherwise the string w committed by $\mathcal{A}$ in the right proof is returned. In other words, $\mathsf{wmim}^{\mathcal{A}}(x, w, z)$ consists of the view of $\mathcal{A}$ and the witness encoded in the right proof unless the proof is not accepting.

Definition 3.3.1 (NMWI argument system [OPV08])**.** *A commit-and-prove argument system* $\Pi = (\mathcal{P}, \mathcal{V})$ *for an* $\mathcal{NP}$*-language* L *and corresponding relation* $\mathsf{Rel_L}$ *is a* non-malleable witness indistinguishable *argument if, for all* PPT *man-in-the-middle adversaries* $\mathcal{A}$*, for all* $x \in L$ *and all* w, w' *such that* $\mathsf{Rel_L}(x, w)$*,* $\mathsf{Rel_L}(x, w')$ *for all auxiliary information* z *it holds that* $\{\mathsf{wmim}^{\mathcal{A}}(x, w, z)\} \approx \{\mathsf{wmim}^{\mathcal{A}}(x, w', z)\}$.

The above notion extends in a straight-forward way to the case of a concurrent MiM adversary trivially. Formally, the concurrent MiM $\mathcal{A}$ opens $\mathsf{poly}(\lambda)$ left and right proofs each with a common input of length $\mathsf{poly}(\lambda)$. $\mathcal{A}$ interacts in the i-th left proof with an instance of the honest prover $\mathcal{P}$ on common input "$x_i \in L$" and private input w_i such that $\mathsf{Rel_L}(x_i, w_i)$. In the j-th right proof $\mathcal{A}$ is interacting with the honest verifier V on common input $\tilde{x}_j$ of its choice.

Let X, W be respectively the sequence of instances and witnesses in input to $\mathcal{P}$ in the left proofs. Now the distribution $\mathsf{wmim}^{\mathcal{A}}(X, W, z)$ consists of the view $\mathsf{View}^{\mathcal{P}}_{\mathcal{A}}(X, W, z)$ of $\mathcal{A}$ along with a sequence $(\tilde{w}_1, \ldots, \tilde{w}_{\mathsf{poly}(\lambda)})$ and it holds that: if the j-th right proof is not accepting or the transcript is identical to the one of a left proof then $\tilde{w}_j = \perp$; otherwise, $\tilde{w}_j$ is the witness encoded in the j-th right proof.

Definition 3.3.2 (cNMWI argument [OPV08])**.** *A commit-and-prove argument system* $\Pi = (\mathcal{P}, \mathcal{V})$ *for an* $\mathcal{NP}$*-language* L *and corresponding relation* $\mathsf{Rel_L}$ *is* concurrent non-malleable witness indistinguishable *if, for all* PPT *concurrent man-in-the-middle adversaries* $\mathcal{A}$*, for all sequences* X *of* $\mathsf{poly}(\lambda)$ *elements of* L *of length*

$\mathsf{poly}(\lambda)$*, for all sequences* W *and* W' *of witnesses for* X*, and for all auxiliary information* z *it holds that* $\{\mathsf{wmim}^{\mathcal{A}}(X, W, z)\} \approx \{\mathsf{wmim}^{\mathcal{A}}(X, W', z)\}$.

3.3.2 Non-Malleable WI Arguments of Knowledge

Our concurrent NM commitment scheme when instantiated without sessions ids, can be used to obtain almost directly a *commit-and-prove* AoK. Recall that in our scheme there is a non-interactive commitment $\mathtt{com}$ of m and then rest of the protocol is an AoK. This AoK is used by the sender to claim that either he knows the message committed in $\mathtt{com}$, or he committed through Π_{wsyn} to a share s_0 that allows to compute the solution of the puzzle.

In order to be fully compliant with the notion of commit-and-prove AoK, we just need to make a trivial change to the statement of the LS subprotocol. Given an instance $x \in L$ and a witness w the prover of our commit-and-prove AoK uses the non-interactive commitment to commit to w, and uses the rest to prove that either he knows the committed message w that moreover is a witness for $x \in L$ or again, he committed through Π_{wsyn} to a share s_0 that allows to compute the solution of the puzzle.

More formally, we define a commit-and-prove AoK $\Pi_{\mathsf{CaP}} = (\mathcal{P}_{\mathsf{CaP}}, \mathcal{V}_{\mathsf{CaP}})$ that corresponds to our concurrent NM commitment scheme with some minimal changes. First, $\mathcal{P}_{\mathsf{CaP}}$ and $\mathcal{V}_{\mathsf{CaP}}$ have as a common input an instance $x \in L$, where L is an NP-language. Second, $\mathcal{P}_{\mathsf{CaP}}$ has as private input w such that $(x, w) \in \mathsf{Rel_L}$. Third, $\mathcal{P}_{\mathsf{CaP}}$ runs the sender NMSen having as input w, while $\mathcal{V}_{\mathsf{CaP}}$ runs the receiver NMRec with the exception of running the LS subprotocol LS for:

$$\begin{aligned}
&L_{\mathsf{CaP}} = \Big\{ \big(x, (a, c, z), Y, s_1, \mathtt{com}, \mathtt{id}\big) : \\
&(\exists\ (w, \sigma) \text{ s.t. } \mathtt{com} = \mathsf{NISen}(w; \sigma) \ \mathtt{AND}\ (x, w) \in \mathsf{Rel_L}) \\
&\mathtt{OR}\ \big(\exists (\rho, s_0) \text{ s.t. } a = \mathrm{Sen}_{\mathsf{wsyn}}(\mathtt{id}, s_0; \rho) \\
&\mathtt{AND} z = \mathrm{Sen}_{\mathsf{wsyn}}(\mathtt{id}, c, s_0; \rho)\ \mathtt{AND}\ Y = f(s_0 \oplus s_1)\big)\Big\}
\end{aligned}$$

that is WI for the relation

$$\begin{aligned}&\mathsf{Rel}_{\mathsf{L_{CaP}}} = \big\{\big(\big(x, (a, c, z), Y, s_1, \texttt{com}, \texttt{id}\big), (w, \sigma, s_0, \rho)\big) :\\ &(\texttt{com} = \mathsf{NISen}(w; \sigma)\ \texttt{AND}(x, w) \in \mathsf{Rel_L})\\ &\texttt{OR}\ \big(a = \mathrm{Sen}_{\mathsf{wsyn}}(\texttt{id}, s_0; \rho)\\ &\texttt{AND}z = \mathrm{Sen}_{\mathsf{wsyn}}(\texttt{id}, c, y; \rho)\ \texttt{AND}\ Y = f(s_0 \oplus s_1)\big)\big\}.\end{aligned}$$

We can now claim the following theorem.

Theorem 25. Suppose there exist OWPs secure against subexponential-time adversaries, then Π_{CaP} is a 3-round concurrent NMWI argument of knowledge.

Proof. The proof of this theorem is pretty straightforward given the previous proof for the concurrent non-malleability of our commitment scheme, therefore here we just point out the main intuition.

First of all, Π_{CaP} is clearly a commit-and-prove AoK. Indeed, there exists a commitment of the witness and there is an AoK proving that the committed message is a witness. In order to see this, notice that for any PPT malicious prover succeeding with non-negligible probability in proving a statement $x \in L$, the extractor of LS (of course this needs to be run against an augmented machine) would return (in expected polynomial time and with overwhelming probability) the committed witness since otherwise it would return a share s_0 that combined with s_1 allows to invert the OWP in polynomial time.

We can now focus on the concurrent NMWI property, and we can assume (by contradiction) that the adversary succeeds in encoding in the right sessions witnesses that are related to the witnesses encoded in the left sessions. Notice that the proof is almost identical to the one of Theorem 24. We can indeed prove the case of one prover and multiple verifiers (i.e., one-many), and then we can apply the fact that any one-many NMWIAoK is also a concurrent NMWIAoK. Indeed this was used in [OPV08] and follows similar arguments given in [PR05a, LPV08]. For the one-many case we can therefore follow the proof of Theorem 24 with the following trivial change. Instead of running hybrid experiments starting with a message m and ending with a message 0, in the proof of one-many concurrent NMWI we start with a witness w_0 and end with a witness w_1. Everything else remains untouched and all the reductions work directly. $\square$

We finally notice that Π_{CaP} can be instantiated to be public-coin and delayed-input, precisely as our concurrent non-malleable commitment scheme. While what we discussed above applies to arguments only, techniques to obtain proofs can be found in [CVZ11].

Instances with just one witness and non-transferability. Recall that the definition of NMWI considers two experiments that differ only on the witness used by the prover. Therefore it is unclear which security is given by a NMWIAoK when the instance has only one witness. In order to understand the security guaranteed by Π_{CaP} in such a case, consider the proof of concurrent NMWI, and thus, in turn, consider the proof of concurrent non-malleability of our commitment scheme. Notice that while the sequence of hybrids goes from an experiment where the committed message is m to an experiment where the committed message is 0, there is an experiment $\mathcal{H}_3(\cdot, z)$ in which the committed message is irrelevant. Indeed, the entire execution is based on inverting the OWP, in encrypting it through the shares s_0 and s_1 and in using this witness in the execution of LS. This experiment can be seen as the execution of a quasi-polynomial time simulator that breaks the puzzle[13] following the approach of [Pas03][14]. Therefore following the same observations of [Pas03, Pas04b] on the security offered by quasi-polynomial time simulation, our concurrent NMWIAoK even for instances with just one witness would not help the adversary in proving a statement whose witness is much harder to compute than breaking the puzzle.

[13]The puzzle can be implemented through a OWP that can be inverted in quasi-polynomial time.

[14]The work of Pass did not take into account MiM attacks.

The above discussion explains also the non-transferability flavor of Π_{CaP}. Indeed, at first sight, a MiM attack of an adversary $\mathcal{A}$ to an AoK should be an attempt of $\mathcal{A}$ to transfer the proof that it gets from the prover to a verifier. As such, an AoK that is secure against concurrent MiM attacks should provide some non-transferability guarantee. Since the success of $\mathcal{A}$ during a MiM attack can be replicated without a MiM attack by a quasi-polynomial time simulator, we have that Π_{CaP} guarantees non-transferability whenever computing the witnesses for the considered instances is assumed to be harder than breaking the puzzle.

Using NMWI for NMZK in the Bare Public-Key (BPK) model. In [OPV08] it is shown that a concurrent NMWIAoK Π gives directly a concurrent NMZKAoK in the BPK model. The construction is straightforward as it just consists of running Π twice, first from the verifier to the prover (proving knowledge of one out of two secrets) and then from the prover to the verifier (proving knowledge of either a witness for $x \in L$ or of one out of the two secrets of the verifier).

Our construction from Theorem 25 when combined with the construction of [OPV08] gives a candidate round-efficient concurrent NMZKAoK in the BPK model.

3.3.3 Identification Schemes

Identification schemes represent one of the most successful real-world applications of cryptographic protocols. We show here a 3-round identification scheme secure against concurrent MiM attacks following the concept of proving knowledge of a secret.

Identification schemes based on proving knowledge of a secret. The importance of this setting was for instance discussed in [COSV12] mentioning the following example. Consider a verifier $\mathcal{V}$ that provides a service to restricted group of provers $\mathcal{P}$. A malicious prover $\mathcal{P}^{\star}$ could give to another party B that is not part of the group, some partial information about his secret that is sufficient for B to obtain the service from $\mathcal{V}$, while still B does not know $\mathcal{P}^{\star}$'s secret. The paradigm of proving knowledge of a secret in an identification scheme allows to prevent attacks like the one just described. When the identification scheme consists in proving knowledge of a secret the sole fact that B convinces $\mathcal{V}$ is sufficient to claim that one can extract the whole secret from B. This implies that B obtained $\mathcal{P}^{\star}$'s secret corresponding to his identity, and thus B is actually $\mathcal{P}^{\star}$[15].

We now introduce a security definition that takes into account concurrent MiM attacks similarly to the definition CR2 (concurrent-reset on-line) of [BFGM01]. The definition of [BFGM01] also includes possible reset attacks in addition to allowing $\mathcal{A}$ to invoke multiple concurrent executions of the prover in the left sessions while $\mathcal{A}$ is interacting with the verifier. In the remaining part of this section we will ignore reset attacks since they are out of the purpose of our work. As described in [Kat02] in most network-based settings reset attacks are not an issue. Following the notation of [Kat02] we now give a formal security definitions for an identification scheme.

Definition 3.3.3. *Let* $\Pi = (\mathcal{K}, \mathcal{P}, \mathcal{V})$ *be a tuple of* PPT *algorithms. We say* Π *is an identification scheme secure against man-in-the-middle attacks if the following conditions hold:*

Correctness. *For all* $(\mathsf{pk}, \mathsf{sk})$ *output by* $\mathcal{K}(1^\lambda)$*, we have*

$$\text{Prob}\,[\ \langle \mathcal{P}(\mathsf{sk}), \mathcal{V}\rangle(\mathsf{pk}) = 1\ ] = 1.$$

Security. *For all* PPT *adversaries* $\mathcal{A}$ *there exists a negligible function* ν *such that*

$$\text{Prob}\left[\ (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathcal{K}(1^\lambda) : \langle \mathcal{A}^{\mathcal{P}(\mathsf{sk})}, \mathcal{V}\rangle(\mathsf{pk}) = 1 \ \ \texttt{AND} \ \ \tau \notin T\ \right] < \nu(\lambda),$$

where $\mathcal{A}$ *has oracle access to a stateful (i.e., non-resettable)* $\mathcal{P}(\mathsf{sk})$*,* T *is defined as the transcripts set of the interactions between* $\mathcal{P}(\mathsf{sk})$ *and* $\mathcal{A}$*, and* τ *is defined as the transcript of one of the interactions between* $\mathcal{A}$ *and* $\mathcal{V}$*. All interactions can be arbitrarily interleaved and* $\mathcal{A}$ *controls the scheduling of the messages.*

[15]This is instead not likely to happen in scenarios where the same secret key is used for other critical tasks such as signatures of any type of document.

Identification scheme from NMWI. Our construction $\Pi_{\mathsf{ID}} = (\mathcal{K}_{\mathsf{ID}}, \mathcal{P}_{\mathsf{ID}}, \mathcal{V}_{\mathsf{ID}})$ follows the approach of [OPV08, COSV12]. Let $f : \{0,1\}^\lambda \rightarrow \{0,1\}^\lambda$ be a one-way permutation, let λ be the security parameter. The public key of $\mathcal{P}_{\mathsf{ID}}$ is the pair $(\mathsf{pk}_0, \mathsf{pk}_1)$, the secret key is sk_b for a randomly chosen bit b, such that $\mathsf{pk_b} = f(\mathsf{sk}_b)$. Therefore the algorithm $\mathcal{K}_{\mathsf{ID}}$ takes as input the security parameter and outputs $((\mathsf{pk}_0, \mathsf{pk}_1), \mathsf{sk}_b)$ as described above. The protocol simply consists in $\mathcal{P}_{\mathsf{ID}}$ running our 3-round concurrent NMWIAoK $\mathsf{\Pi_{CaP}}$ with $\mathcal{V}_{\mathsf{ID}}$ to prove that it *knows* the pre-image of either pk_0 or pk_1. Formally, let $L_{\mathtt{id}}$ be the following language $L_{\mathtt{id}} = \{(y_0, y_1) : \exists\ x \ \in \{0,1\}^\lambda$ such that $y_0 = f(x)$ OR $y_1 = f(x)\}$, then the identification scheme consists of $\mathcal{P}_{\mathsf{ID}}$ proving the statement $(\mathsf{pk}_0, \mathsf{pk}_1) \in L_{\mathtt{id}}$ using $\mathsf{\Pi_{CaP}}$. Fig. 3.4 summarizes our identification scheme. Now we can claim the following theorem.

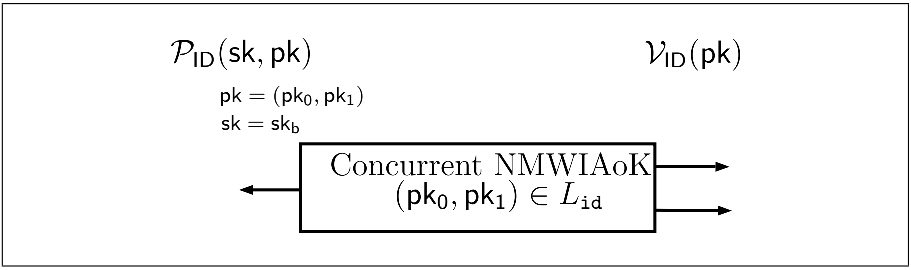

$\mathcal{P}_{\mathsf{ID}}(\mathsf{sk}, \mathsf{pk})$
$\mathcal{V}_{\mathsf{ID}}(\mathsf{pk})$
$\mathsf{pk} = (\mathsf{pk}_0, \mathsf{pk}_1)$
$\mathsf{sk} = \mathsf{sk_b}$
Concurrent NMWIAoK
$(\mathsf{pk}_0, \mathsf{pk}_1) \in L_{\mathtt{id}}$

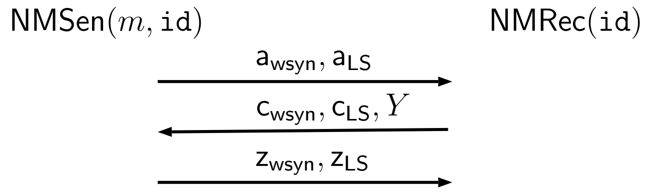
Figure 3.4 Our 3-round identification scheme Π_{ID} from our 3-round concurrent NMWIAoK.

Theorem 26. Assume the existence of OWPs secure against subexponential-time adversaries then Π_{ID} is an identification scheme secure against concurrent MiM attacks.

The proof is again straight-forward. If a PPT $\mathcal{A}$ succeeds then concurrent NMWI of $\mathsf{\Pi_{CaP}}$ guarantees that the witness that he encoded in the proof is independent of the one encoded in the proofs given by $\mathcal{P}$. Therefore by using the AoK property of $\mathsf{\Pi_{CaP}}$ we can invert f with non-negligible probability.

3.4 3-Round One-One NM Commitments

In this section we show our one-one non-malleable commitment scheme. In particular we construct a compiler that, on input a 3-round synchronous weak one-one NM commitment scheme, gives as output a 3-round extractable one-one NM commitment scheme assuming OWPs secure against subexponential-time adversaries.


$\mathsf{NMSen}(m, \mathtt{id})$
$\mathsf{NMRec}(\mathtt{id})$
$\mathsf{a}_{\mathsf{wsyn}}, \mathsf{a}_{\mathsf{LS}}$
$\mathsf{c}_{\mathsf{wsyn}}, \mathsf{c}_{\mathsf{LS}}, Y$
$\mathsf{z}_{\mathsf{wsyn}}, \mathsf{z}_{\mathsf{LS}}$

- Y is an element taken from the range of the OWP f.
- $\tau = (\mathsf{a}_{\mathsf{wsyn}}, \mathsf{c}_{\mathsf{wsyn}}, \mathsf{z}_{\mathsf{wsyn}})$ is the transcript of $\langle \mathsf{Sen}_{\mathsf{wsyn}}(m), \mathsf{Rec}_{\mathsf{wsyn}} \rangle(\mathtt{id})$.
- $(\mathsf{a}_{\mathsf{LS}}, \mathsf{c}_{\mathsf{LS}}, \mathsf{z}_{\mathsf{LS}})$ is the transcript of LS for proving knowledge of either the decommitment of τ to a message $\neq \perp$ or of the preimage of Y.

Figure 3.5 Informal description of our 3-round NM commitment scheme $\Pi_{1-1\mathsf{NMCom}}$.

3.4.1 Our 3-Round One-One NM Commitment

Our construction is based on a compiler that takes as input a 3-round synchronous weak one-one NM commitment scheme $\Pi_{\mathsf{wsyn}} = (\mathsf{Sen}_{\mathsf{wsyn}}, \mathsf{Rec}_{\mathsf{wsyn}})$, a OWP f, a WI adaptive PoK for $\mathcal{NP}$ LS, and outputs a 3-round extractable one-one NM commitment scheme $\Pi_{1-1\mathsf{NMCom}} = (\mathsf{NMSen}, \mathsf{NMRec})$.

In order to construct our compiler we consider the following tools:

Common input: security parameters: λ, $(\lambda_{\mathsf{wsyn}}, \lambda_{\mathsf{LS}}, \ell) = \mathsf{Params}(\lambda)$, $\mathtt{id} \in \{0,1\}^{\lambda}$.
Input to NMSen**:** $m \in \{0,1\}^{\mathsf{poly}\{\lambda\}}$.
Commitment phase:

1. $\mathsf{NMSen} \rightarrow \mathsf{NMRec}$
    1. Run $\mathsf{Sen}_{\mathsf{wsyn}}$ on input $1^{\lambda_{\mathsf{wsyn}}}$, $\mathtt{id}$ and m thus obtaining the 1st round $\mathsf{a}_{\mathsf{wsyn}}$ of Π_{wsyn}.
    2. Run $\mathcal{P}$ on input $1^{\lambda_{\mathsf{LS}}}$ and ℓ thus obtaining the 1st round a_{LS} of LS.
    3. Send $(\mathsf{a}_{\mathsf{wsyn}}, \mathsf{a}_{\mathsf{LS}})$ to NMRec.
2. $\mathsf{NMRec} \rightarrow \mathsf{NMSen}$
    1. Run $\mathsf{Rec}_{\mathsf{wsyn}}$ on input $\mathtt{id}$ and $\mathsf{a}_{\mathsf{wsyn}}$ thus obtaining the 2nd round $\mathsf{c}_{\mathsf{wsyn}}$ of Π_{wsyn}.
    2. Run $\mathcal{V}$ on input a_{LS} thus obtaining the 2nd round c_{LS} of LS.
    3. Pick a random $Y \in \{0,1\}^{\lambda}$.
    4. Send $(\mathsf{c}_{\mathsf{wsyn}}, \mathsf{c}_{\mathsf{LS}}, Y)$ to NMSen.
3. $\mathsf{NMSen} \rightarrow \mathsf{NMRec}$
    1. Run $\mathsf{Sen}_{\mathsf{wsyn}}$ on input $\mathsf{c}_{\mathsf{wsyn}}$ thus obtaining the 3rd round $\mathsf{z}_{\mathsf{wsyn}}$ of Π_{wsyn} and the decommitment information $\mathtt{dec}_{\mathsf{wsyn}}$.
    2. Set $x = (\mathsf{a}_{\mathsf{wsyn}}, \mathsf{c}_{\mathsf{wsyn}}, \mathsf{z}_{\mathsf{wsyn}}, Y, \mathtt{id})$ and $w = (m, \mathtt{dec}_{\mathsf{wsyn}}, \perp)$ with $|x| = \ell$. Run $\mathcal{P}$ on input x, w, and c_{LS} thus obtaining the 3rd round z_{LS} of LS.
    3. Send $(\mathsf{z}_{\mathsf{wsyn}}, \mathsf{z}_{\mathsf{LS}})$ to NMRec.
4. NMRec: Set $x = (\mathsf{a}_{\mathsf{wsyn}}, \mathsf{c}_{\mathsf{wsyn}}, \mathsf{z}_{\mathsf{wsyn}}, Y, \mathtt{id})$ and abort iff $(\mathsf{a}_{\mathsf{LS}}, \mathsf{c}_{\mathsf{LS}}, \mathsf{z}_{\mathsf{LS}})$ is not accepted by $\mathcal{V}$ for $x \in L$.

Decommitment phase:

1. $\mathsf{NMSen} \rightarrow \mathsf{NMRec}$: Send $(\mathtt{dec}_{\mathsf{wsyn}}, m)$ to NMRec.
2. NMRec: accept m as the committed message if and only if $\mathsf{Rec}_{\mathsf{wsyn}}$ on input $(m, \mathtt{dec}_{\mathsf{wsyn}})$ accepts m as a committed message of $(\mathsf{a}_{\mathsf{wsyn}}, \mathsf{c}_{\mathsf{wsyn}}, \mathsf{z}_{\mathsf{wsyn}}, \mathtt{id})$.

Figure 3.6 Our 3-round NM commitment scheme $\Pi_{\mathsf{1-1NMCom}}$.

1. a OWP f that is secure against PPT adversaries and that is $\tilde{T}_f$-breakable;
2. a 3-round one-one synchronous weak NM commitment scheme $\Pi_{\mathsf{wsyn}} = (\mathsf{Sen}_{\mathsf{wsyn}}, \mathsf{Rec}_{\mathsf{wsyn}})$ that is T_{wsyn}-hiding/NM, and $\tilde{T}_{\mathsf{wsyn}}$-breakable;
3. the LS PoK $\mathsf{LS} = (\mathcal{P}, \mathcal{V})$ for the language

$$\begin{aligned} L = \big\{(a, c, z, Y, \mathtt{id}) : \exists\ (m, \mathtt{dec}, y) \text{ s.t.} \\ \big(\mathsf{Rec}_{\mathsf{wsyn}} \text{ on input } (a, c, z, m, \mathtt{dec}, \mathtt{id}) \\ \text{accepts } m \neq \perp \text{ as a decommitment of } (a, c, z, \mathtt{id}) \\ \mathtt{OR}\ Y = f(y)\big)\big\} \end{aligned}$$

that is T_{LS}-WI for the corresponding relation $\mathsf{Rel}_{\mathsf{L}}$.

Let λ be the security parameter of our scheme. We use w.l.o.g. λ also as security parameter for the one-wayness of f with respect to polynomial-time adversaries. We consider the following hierarchy of security levels: $\tilde{T}_f << T_{\mathsf{wsyn}} << \tilde{T}_{\mathsf{wsyn}} = \sqrt{T_{\mathsf{LS}}} << T_{\mathsf{LS}}$ where by "$T << T'$" we mean that "$T \cdot \mathsf{poly}(\lambda) < T'$".

Now, similarly to [PW10, COSV16], we define different security parameters, one for each tool involved in the security proof to be consistent with the hierarchy of security levels defined above. Given the security parameter λ of our scheme, we will make use of the following security parameters: 1) λ for the OWP f; 2) λ_{wsyn} for the synchronous weak one-one NM commitment scheme; 3) λ_{LS} for LS.

All of them are polynomially related to λ and they are such that the above hierarchy of security levels holds. In the construction we assume for simplicity to have a function Params that on input λ outputs $(\lambda_{\mathsf{wsyn}}, \lambda_{\mathsf{LS}}, \ell)$ where ℓ is the length of the theorem to be proved using LS.[16] The detailed scheme is described in Fig. 3.6 and a compact version is depicted in Fig. 3.5.

Roughly speaking our compiler works as follows. Let m be the message that NMSen wants to commit. The sender NMSen, on input the session-id $\mathtt{id}$ and the message m, computes the 1st

[16]To compute 1st and 2nd round of LS only the length ℓ of the instance is required.

round of the protocol by sending the 1st round a_{LS} of LS and the 1st round $\mathsf{a}_{\mathsf{wsyn}}$ of Π_{wsyn} (to commit to the message m using $\mathtt{id}$ as session-id). In the 2nd round the receiver NMRec sends challenges $\mathsf{c}_{\mathsf{wsyn}}$ and c_{LS} of Π_{wsyn} and LS, also picks and sends an element Y in the range of f. In the 3rd round NMSen computes the 3rd round of Π_{wsyn} and completes the transcript for LS by sending $\mathsf{z}_{\mathsf{wsyn}}$ and z_{LS}. Let $\tau = (\mathsf{a}_{\mathsf{wsyn}}, \mathsf{c}_{\mathsf{wsyn}}, \mathsf{z}_{\mathsf{wsyn}})$ be the transcript of the execution of Π_{wsyn}. LS is used by NMSen to prove knowledge of either a decommitment of τ to a message $\neq\perp$ or of a preimage of Y.

Theorem 27. Suppose there exists a synchronous weak one-one NM commitment scheme and OWPs, both secure against subexponential-time adversaries, then $\Pi_{1-1\mathsf{NMCom}}$ is a one-one NM commitment scheme.

We are now ready to start the proof, that is divided in two parts. First we prove that $\Pi_{1-1\mathsf{NMCom}}$ is a commitment scheme. Then we prove that $\Pi_{1-1\mathsf{NMCom}}$ is a NM commitment scheme. Before that, we recall that LS can be constructed from OWPs secure against subexponential-time adversaries as well as Π_{wsyn} that can be constructed from OWPs secure against subexponential-time using the constructions of [GPR16, GRRV14].

Lemma 3.4.1. $\Pi_{1-1\mathsf{NMCom}}$ *is a statistically-binding computationally-hiding commitment scheme.*

Proof. **Correctness.** The correctness of $\Pi_{1-1\mathsf{NMCom}}$ follows immediately from the delayed-input completeness of LS, and the correctness of Π_{wsyn}.

Perfect Binding. Observe that the message given in output in the decommitment phase of $\Pi_{1-1\mathsf{NMCom}}$ is the message committed using Π_{wsyn}. Moreover the decommitment phase of $\Pi_{1-1\mathsf{NMCom}}$ coincides with the decommitment phase of Π_{wsyn}. Since Π_{wsyn} is perfectly binding we have that the same holds for $\Pi_{1-1\mathsf{NMCom}}$.

Hiding. Following Def. 1.2.1 to prove the hiding of $\Pi_{1-1\mathsf{NMCom}}$ we have to show that the experiment $\mathsf{ExpHiding}^{0}_{\mathcal{A},\Pi_{1-1\mathsf{NMCom}}}(\lambda)$ in which NMSen commits to a message m_0 is computationally indistinguishable from the experiment $\mathsf{ExpHiding}^{1}_{\mathcal{A},\Pi_{1-1\mathsf{NMCom}}}(\lambda)$ in which

NMSen commits to a message m_1. In order to prove this indistinguishability we consider the following hybrid experiments.

- The 1st hybrid experiment $\mathcal{H}^0(\lambda)$ is equal to the real game experiment $\mathsf{ExpHiding}^0_{\mathcal{A},\Pi_{1-1\mathsf{NMCom}}}(\lambda)$, with the difference that a value y s.t. $Y = f(y)$ is computed and used as a witness for LS. Observe that in order to compute y the commitment phase takes time $\tilde{T}_f$. The indistinguishability between $\mathcal{H}^0(\lambda)$ and $\mathsf{ExpHiding}^0_{\mathcal{A},\Pi_{1-1\mathsf{NMCom}}}(\lambda)$ comes from the adaptive-input WI of LS, that holds against adversaries with running time bounded by $T_{\mathsf{LS}} >> \tilde{T}_f$.

- The 2nd hybrid $\mathcal{H}^1(\lambda)$ differs from $\mathcal{H}^0(\lambda)$ in the message committed by the adversary using Π_{wsyn}. More precisely, Π_{wsyn} is used by NMSen to commit to the message m_1 instead of m_0. The indistinguishability between $\mathcal{H}^0(\lambda)$ and $\mathcal{H}^1(\lambda)$ comes from the hiding of Π_{wsyn} and noticing that the hiding of Π_{wsyn} still holds against adversaries with running time bounded by $T_{\mathsf{wsyn}} >> \tilde{T}_f$.

The proof ends with the observation that $\mathcal{H}^1(\lambda) \approx \mathsf{ExpHiding}^1_{\mathcal{A},\Pi_{1-1\mathsf{NMCom}}}(\lambda)$. The indistinguishability between $\mathcal{H}^1(\lambda)$ and $\mathsf{ExpHiding}^1_{\mathcal{A},\Pi_{1-1\mathsf{NMCom}}}(\lambda)$ comes from the adaptive-WI property of LS and from the observation that, as before, the adaptive-input WI of LS still holds against adversaries with running time bounded by $T_{\mathsf{LS}} >> \tilde{T}_f$.

□

Lemma 3.4.2. $\Pi_{1-1\mathsf{NMCom}}$ *is a one-one NM commitment scheme.*

The proof of security is divided in two cases, in the first case we consider an adversarial MiM $\mathcal{A}_{\mathsf{NMCom}}$ that acts in a synchronized way, while in the second case $\mathcal{A}_{\mathsf{NMCom}}$ is non-synchronized. In both cases we want to show that the committed value (and the view) of $\mathcal{A}_{\mathsf{NMCom}}$ when interacting with a prover NMSen that commits to a message m is indistinguishable from the committed value (and the view) of a simulator. The proof for the synchronous case goes through a series of hybrid experiments listed below.

- We consider the real game experiment $\mathcal{H}_1^m(z)$ in which in the left session NMSen commits to m, while in the right session NMRec interacts with $\mathcal{A}_{\mathsf{NMCom}}$. Now we prove that in the right session the MiM adversary $\mathcal{A}_{\mathsf{NMCom}}$ does not commit to a message $\tilde{m} =\perp$. By contradiction if $\mathcal{A}_{\mathsf{NMCom}}$ commits to $\tilde{m} =\perp$ then the witness used to complete an accepting transcript for LS is a value $\tilde{y}$ s.t. $f(\tilde{Y}) = \tilde{y}$. Then, by using the adaptive-input PoK property of LS we can reach a contradiction by inverting f in polynomial time.

- The 2nd hybrid is $\mathcal{H}_2^m(z)$ and it differs from $\mathcal{H}_1^m(z)$ only in the witness used to compute the LS transcript. The adversary $\mathcal{A}_{\mathsf{NMCom}}$, running in sub-exponential time, computes a value y s.t. $f(y) = Y$, and uses it as witness for the execution of LS. From the adaptive-input WI (that is stronger than inverting the OWP and of breaking Π_{wsyn}) of LS, the view and the committed message of $\mathcal{A}_{\mathsf{NMCom}}$ do not change between $\mathcal{H}_2^m(z)$ and $\mathcal{H}_1^m(z)$.

- We now consider the hybrid experiment $\mathcal{H}_1^0(z)$ that differs from the first hybrid experiment that we have considered $\mathcal{H}_1^m(z)$ in the committed message. Indeed in this case, the message committed in the left session is 0^λ. We observe that $\mathcal{H}_1^0(z)$ actually is the simulated game. As for the hybrid experiment $\mathcal{H}_1^m(z)$ we need to prove that in the right session the MiM adversary $\mathcal{A}_{\mathsf{NMCom}}$ does not commit to a message $\tilde{m} =\perp$. By contradiction if $\mathcal{A}_{\mathsf{NMCom}}$ commits to $\tilde{m} =\perp$ then the witness used to complete an accepting transcript for LS is a value $\tilde{y}$ s.t. $f(\tilde{Y}) = \tilde{y}$. Then, by using the adaptive-input PoK property of LS we can reach a contradiction by inverting f in polynomial time.

- The last hybrid experiment that we consider is $\mathcal{H}_2^0(z)$ and it differs from $\mathcal{H}_1^0(z)$ only in the witness used to compute the LS transcript. In more details the adversary $\mathcal{A}_{\mathsf{NMCom}}$, running in sub-exponential time, computes a value y s.t. $f(y) = Y$, and uses it as witness for the execution of LS. From the adaptive-input WI (that is stronger than inverting the OWP and of breaking Π_{wsyn}) of LS, the view and the committed message of $\mathcal{A}_{\mathsf{NMCom}}$ do not change between $\mathcal{H}_2^0(z)$ and $\mathcal{H}_1^0(z)$.

To conclude this proof we show that the view and the committed message of $\mathcal{A}_{\mathsf{NMCom}}$ acting in $\mathcal{H}_1^m(z)$ are indistinguishable from the view and the committed message of $\mathcal{A}_{\mathsf{NMCom}}$ acting in $\mathcal{H}_1^0(z)$. For what has been argued above, it remains to show that the view and the committed message of $\mathcal{H}_2^m(z)$ are indistinguishable from the view and the committed message of $\mathcal{H}_2^0(z)$. This is ensured by the synchronous weak one-one non-malleability of Π_{wsyn}. Here we need only to use a *weak* synchronous one-one NM commitment since we are guaranteed, from the above arguments, that whenever $\mathcal{A}_{\mathsf{NMCom}}$ completes a commitment in a right session the underlying commitment computed through Π_{wsyn} corresponds to $\perp$ with negligible probability only both in $\mathcal{H}_2^m(z)$ and in $\mathcal{H}_2^0(z)$.

The proof for the asynchronous case is much simpler and relies on the hiding of $\Pi_{\mathsf{1-1NMCom}}$. More precisely we observe that in case of asynchronous scheduling it is possible to rewind the adversary $\mathcal{A}_{\mathsf{NMCom}}$ without rewinding the sender in the left session. This allows us to extract (in polynomial time) the witness used by the adversary in the execution of LS, that with overwhelming probability corresponds to the committed message. Therefore we contradict the hiding of $\Pi_{\mathsf{1-1NMCom}}$. Note that the only two possible schedules that are possible for a 3-round NM commitment scheme is the synchronous one or a one that allow the extraction of the message from the transcript of LS without rewinding the sender (i.e. asynchronous schedule). This observation concludes the proof.

Chapter 4

Delayed-Input Non-Malleable Zero Knowledge

4.1 Overview of the Chapter

Non-malleable zero-knowledge (NMZK) and secure multi-party computation (MPC) are fundamental primitives in Cryptography. In this work we will study these two primitives and for the case of MPC we will focus on the coin-tossing functionality that is among the most studied functionalities.

NMZK. The first construction of NMZK was given by Dolev at at. in [DDN91]. Later on, Barak in [Bar02] showed the first constant-round construction. An improved construction was then given by Pass and Rosen in [PR05b, PR08b]. The work of Goyal et al. [GRRV14] obtained the first round-optimal construction requiring only 4 rounds and one-way functions (OWFs). Their construction requires the instance and the witness to be known already when the prover plays his first round. Their definition is the standard one-one definition where the adversary opens two sessions, one with a prover and one with a verifier.

The fact that the instance and the witness need to be known already at the second round is an important limitation when NMZK is used as subprotocol to prove statements about another subprotocol played in parallel. Moreover the one-one security is an important limitation when NMZK is used in a multi-party scenario where several of such argument systems are played in parallel.

The above two limitations clearly raise the following natural and interesting open questions:

Open Question 1: is there a 4-round delayed-input NMZK argument system?

Open Question 2: is there a 4-round many-many synchronous NMZK argument system?

Multi-party coin-flipping (MPCT). In [KOS03], Katz et al. obtained a constant-round secure MPC protocol using sub-exponential hardness assumptions. This results was then improved by Pass in [Pas04b] that showed how to get bounded-concurrent secure MPC for any functionality with standard assumptions. Further results of Goyal [Goy11] and Goyal et al. [GLOV12] relied on better assumptions but with a round complexity still far from optimal.

A recent work of Garg et al. [GMPP16] makes a long jump ahead towards fully understanding the round complexity of secure MPCT. They show that the existence of a 3-round 3-robust parallel non-malleable commitment scheme implies a 4-round protocol for secure MPCT for polynomially many coins with black-box simulation. Some candidate instantiations of such special commitment scheme [GMPP16, Pol16] are the one of Pass et al. [PPV08] based on non-falsifiable assumptions, or the one described in this thesis based on sub-exponentially OWPs. The achieved round complexity (i.e., 4 rounds) is proven optimal in [GMPP16] when simulation is black box and the number of bits in the output of the functionality is superlogarithmic.

A very recent line of works [ACJ17, BHP17, HHPV17, BGJ$^+$17] improves the round complexity of MPC for any functionality, constructing a 4-round MPC protocol that relays on number theoretic assumptions or LWE. Furthermore the work of [BL17] construct a 5-round MPC protocol for any functionality from any 5-round delayed-semi-malicious oblivious transfer (OT) protocol.

The above state-of-the art leaves open the following question.

Open Question 3: is there a 4-round secure MPCT protocol under standard generic assumptions, rather than specific number-theoretic assumptions?

4.1.1 Our Contribution

In this work we solve the above 3 open problems. More precisely we present the following results:

1. a *delayed-input* 4-round one-*many* NMZK argument Π_{NMZK} from OWFs, therefore solving Open Question 1; moreover Π_{NMZK} is also a *delayed-input* many-many *synchronous* NMZK argument, therefore solving Open Question 2;
2. a 4-round MPCT protocol Π_{MPCT} from one-to-one OWFs, therefore solving Open Question 3[1].

The two constructions are not uncorrelated. Indeed Π_{MPCT} uses Π_{NMZK} as subprotocol and exploits the special properties (e.g., delayed input, many-many synchronous) of Π_{NMZK}. Moreover both Π_{NMZK} and Π_{MPCT} make use of a special proof of knowledge that offers additional security guarantees when played in parallel with other protocols. Designing such a proof of knowledge is an additional contribution of this work and is of independent interest.

Interestingly, several years after the 4-round zero knowledge argument system from OWFs of [BJY97], the same optimal round complexity and optimal complexity assumptions have been shown sufficient in this work for delayed-input NMZK and in [COP+14] for resettably sound zero knowledge.

More details on our two new constructions follow below.

MPCT from NMZK. A first main idea that allows us to bypass the strong requirements of the construction of [GMPP16] is that we avoid robust/non-malleable commitments and instead focus on non-malleable zero knowledge. Since we want a 4-round MPCT protocol, we need to rely on 4-round NMZK. The only known construction is the one of [GRRV14]. Unfortunately their

[1]An unpublished prior work of Goyal et al. [GKP+17] achieves the same result on MPCT using completely different techniques.

NMZK argument system seems to be problematic to use in our design of a 4-round MPCT protocol. There are two main reasons. The first reason is that the construction of [GRRV14] uses the technique of secure computation in the head and therefore requires the instance already in the second round. This is often a problem when the NMZK argument is played in parallel with other subprotocols as in our construction. Indeed these additional subprotocols end in the 3rd or 4th round and typically[2] need to be strengthened by a zero-knowledge proof of correctness. The second reason is that in the setting of 4-round MPCT the adversary can play as a many-many synchronous man-in-the-middle (MiM), while the construction of [GRRV14] is proved one-one non-malleable only.

We therefore improve the state-of-the-art on NMZK constructing a delayed-input NMZK argument system. Our construction only needs one-way functions and is secure even when a) there are polynomially many verifiers (i.e., it is a one-many NMZK argument), and b) there are polynomially many provers and they are in parallel. We will crucially use both the delayed-input property and security with parallelized many provers and verifiers in our secure MPCT construction.

Other applications. Interestingly our one-many (many-many synchronous) delayed-input NMZK argument has been used to develop important cryptographic protocols. In particular, in [COSV17c] the authors construct a 4-round (round optimal) two-party computation protocol with simultaneous message exchange channel that crucially relays on our NMZK. Furthermore, the work of [BL17] in order to construct a 5-round MPC protocol also uses our NMZK. In both constructions the delayed-input property and the retained security under parallel composition are crucial.

[2]Indeed, even the construction of [GMPP16] that makes use of a special non-malleable commitments requires also a delayed-input zero-knowledge argument.

4.1.2 Technical Overview of Our NMZK

Issues in natural constructions of NMZK. A natural construction of a NMZK argument from OWFs consists of having: 1) a 3-round sub-protocol useful to extract a trapdoor from the verifier of NMZK; 2) a 4-round non-malleable commitment of the witness for the statement to be proved; 3) a 4-round witness-indistinguishable proof of knowledge (WIPoK) to prove that either the committed message is a witness or the trapdoor is known. By combining instantiations from OWFs of the above 3 tools in parallel we could obtain 4-round NMZK from OWFs. The simulator-extractor for such a scheme would 1) extract the trapdoor from the verifier; 2) commit to 0 in the non-malleable commitment; 3) use the trapdoor as witness in the WIPoK; 4) extract the witness from the arguments given by the MiM by extracting from the WIPoK or from the non-malleable commitment.

Unfortunately it is not clear how to prove the security of this scheme when all sub-protocols are squeezed into 4 rounds. The problem arises from the interactive nature of the involved primitives. Indeed notice that the 4-round non-malleable commitment is executed in parallel with the 4-round WIPoK. When in a hybrid of the security proof the trapdoor is used as witness in the 4-round WIPoK played on the left, the MiM could do the same and also commits to the message 0 in the non-malleable commitment. To detect this behavior, in order to break the WI, the reduction should extract the message committed in the non-malleable commitment by rewinding the MiM. This implies that also the 4-round WIPoK involved in the reduction must be rewound (we recall that these two sub-protocols are executed in parallel). It is important to observe that if in some hybrid we allow the MiM to commit to the message 0 when the witness of the WIPoK given on the left is switched to the trapdoor, then the simulator-extractor (that corresponds to the final hybrid) will have no way to extract a witness from the MiM (and this is required by the definition of NMZK). Indeed from a successful MiM that commits to 0 the extraction from the WIPoK can only give in output the trapdoor. Therefore the simulator-extractor would fail.

A special delayed-input WIPoK Π^{OR}. In order to overcome the above problem we follow a recent idea proposed in [COSV17b] where non-interactive primitives instead of 3-rounds WIPoKs are used in order to construct a concurrent non-malleable commitment in four rounds. In this way, in every security reduction to such primitives, it will be always possible to extract the message committed in the non-malleable commitment without interfering with the challenger involved in the reduction.

In [COSV17b] the authors propose an ad-hoc technique that avoids such a *rewinding issue* by using a combination of instance-dependent trapdoor commitments (IDTCom) and special honest-verifier zero knowledge (Special HVZK) proofs of knowledge. In this work we propose a generic approach to construct a special delayed-input WIPoK Π^{OR} that can be nicely composed with other protocols in parallel. We construct Π^{OR} in two steps.

In 1st step we consider the construction of 3-round WIPoK for $\mathcal{NP}$ of Lapidot and Shamir (LS) [LS90][3] that enjoys adaptive-input Special HVZK[4] and observe that LS does not enjoy adaptive-input special soundness. That is, given and accepting transcript $(a, 0, z_0)$ for the statement x_0 and an accepting transcript $(a, 1, z_1)$ for the statement x_1, then only the witness x_1 can be efficiently extracted. More precisely, only the witness for the statement where the challenge-bit was equal to 1[5] can be extracted. Therefore we propose a compiler that using $\mathsf{LS} = (\mathcal{P}, \mathcal{V})$ in a black-box way outputs a 3-round protocol $\mathsf{LS}' = (\mathcal{P}', \mathcal{V}')$ that maintains the adaptive-input Special HVZK and moreover enjoys adaptive-input special soundness.

In the second step we show how to combine the OR composition of statements proposed in [CDS94] with LS' in oder to obtain

[3]See Section 4.4.1 for a detailed description of [LS90].

[4]By *adaptive-input* we mean that the security of the cryptographic primitive holds even when the statement to be proved is adversarially chosen in the last round.

[5]For ease of exposition be consider LS with one-bit challenge, but our result hold for an arbitrarily chosen challenge length.

a WIPoK Π^{OR} such that: a) a reduction can be successfully completed even when there are rewinds due to another protocol played in parallel; b) the statement (and the corresponding witness) are required to be known only in the last round. Both properties are extremely helpful when a WIPoK is played with other protocols in parallel.

We now give more details about the two steps mentioned above.

- *First step:* $\mathsf{LS}' = (\mathcal{P}', \mathcal{V}')$. Our construction of LS' works as follows. The prover $\mathcal{P}'$ runs two times $\mathcal{P}$ using different randomnesses thus obtaining two first rounds of LS a_0 and a_1. Upon receiving the challenge-bit b from the verifier $\mathcal{V}$, the statement x to be proved and the corresponding witness w, $\mathcal{P}'$ runs $\mathcal{P}$ in order to compute the answer z_0 with respect to the challenge b for a_0 and the answer z_1 with respect to the challenge $1 - b$ for a_1. $\mathcal{V}'$ accepts if both (a_0, b, z_0, x) and $(a_1, 1-b, z_1, x)$ are accepting for $\mathcal{V}$. We now observe that every accepting transcript for LS' contains a sub-transcript that is accepting for $\mathcal{V}$ where the bit 1 has been used as a challenge. From what we have discussed above, it is easy to see that LS' enjoys adaptive-input special soundness.

- *Second step: adaptive-input PoK for the OR of statements.* We combine together two executions of LS' by using the trick for composing two Σ-protocols Σ_0, Σ_1 to construct a Σ-protocol for the $\mathcal{NP}$-language L_0 OR L_1 [CDS94]. Let the compound statement to be proved (x_0, x_1), with $x_0 \in L_0$ and $x_1 \in L_1$, and let w_b be the witness for x_b. The protocol Π^{OR} proposed in [CDS94] considers two Σ-protocols Σ_0 and Σ_1 (respectively for L_0 and L_1) executed in parallel, but after receiving the challenge c form the verifier, the prover can use as challenges for Σ_0 and Σ_1 every pair (c_0, c_1) s.t. $c_0 \oplus c_1 = c$. Therefore the prover could choose in advance one of the challenge to be used, (e.g., c_{1-b}), and compute the other one by setting $c_b = c \oplus c_{1-b}$. In this way the transcript for Σ_{1-b} can be computed using the Special HVZK simulator while the transcript for Σ_b is computed using the witness w_b. Thus the prover has the "freedom" of picking one out of two of the challenge before seeing c, but still being able to complete the execution of both Σ_0 and Σ_1 for every c. We will show that this "freedom" is sufficient to switch between using w_0 and w_1 (in order to prove WI) even when it is required to answer to additional (and different) challenges $c^1, \dots, c^{\mathsf{poly}(\lambda)}$ (i.e., when some rewinds occur). Indeed it is possible to change the witness used (from w_0 to w_1) in two steps relying first on the Special HVZK of Σ_1, and then on the Special HVZK of Σ_0. More precisely we consider the hybrid experiment H^{w_0} as the experiment where in Π^{OR} the witness w_0 is used (analogously we define H^{w_1}). We now consider H^{w_0,w_1} that differs from H^{w_0} because both the witnesses w_0 and w_1 are used. We prove that H^{w_0} and H^{w_0,w_1} are indistinguishable due to the Special HVZK of Σ_1 even tough Π^{OR} is rewound polynomially many times. The reduction works as follows. A challenge c_1 is chosen before the protocol Π^{OR} starts and the Special HVZK challenger is invoked thus obtaining (a_1, z_1). The transcript for Σ_0 is computed by the reduction using the witness w_0 in order to answer to the challenge $c_0^i = c^i \oplus c_1$ for $i = 1, \dots, \mathsf{poly}(\lambda)$. We recall the we are in a setting where Π^{OR} could be rewound, and therefore the reduction needs to answer to multiple challenges. We observe that the reduction to the Special HVZK is not disturbed by these rewinds because c_1 can be kept fixed. The same arguments can be used to prove that H^{w_0,w_1} is computationally indistinguishable from H^{w_1}.

We then show that Π^{OR} preserves the special-soundness of the input Σ-protocols, as well as preserves the adaptive-input special soundness when instead of two Σ-protocols, two instantiations of LS' are used. Moreover the above reductions to Special HVZK can be done relying on adaptive-input Special HVZK. Finally Π^{OR} can be upgrade from adaptive-input special soundness to adaptive-input PoK using a theorem of [CPS+16b].

Our NMZK argument system NMZK. We run Π^{OR} in parallel with a 4-round public-coin one-one honest-extractable synchronous non-malleable commitment scheme Π_{nm}[6]. A construction for such a scheme in 4 rounds was given by [GPR16]. The prover of the NMZK argument runs Π^{OR} in order to prove either the

[6]All such properties are pretty standard except honest extractability. Informally, this property means that there is a successful extractor that gives in output the committed message having black-box access to a honest sender.

validity of some $\mathcal{NP}$-statement, or that the non-malleable commitment computed using Π_{nm} contains a trapdoor. The simulator for NMZK works by extracting the trapdoor, committing to it using the non-malleable commitment, and using knowledge of both the trapdoor and the opening information used to compute the non-malleable commitment as a witness for Π^{OR}. The 3-round subprotocol from OWFs for the trapdoor extraction follows the one of [COSV17b]. More precisely the trapdoor is represented by the knowledge of two signatures under a verification key sent by the verifier in the 1st round. In order to allow the extraction of the trapdoor, the verifier of NMZK sends a signature of a message randomly chosen in the 3rd round by the prover.

The security proof of one-many NMZK. The simulator of NMZK extracts the trapdoor[7], and commits to it using Π_{nm}. In each hybrid experiments of the security proof we want to guarantee that in each right sessions the MiM still uses a witness for the statement proved and does not use the trapdoor. In this way we are ensured that in the simulated experiment (which corresponds to the last hybrid experiments) we continue to extract the witnesses for the statements proved by MiM in the $\mathsf{poly}(\lambda)$ right sessions.

The first observation is that in the real game experiments the MiM does not use the trapdoor in all right sessions, otherwise it is possible to made a reduction to the security of the signature scheme. Then, in the first hybrid we switch to the commitment of two signatures of two different messages in Π_{nm} and we want to ensure that the MiM does not do the same. For this proof we follow the same approach provided in [COSV16]. The reduction to the non-malleability of the underlying commitment scheme isolates one right session guessing that the MiM has committed there to the trapdoor. The distinguisher for the non-malleable commitment takes as input the committed message an checks if it corresponds to two signatures of two different messages for a given signature key.

[7]The trapdoor for our protocol is represented by two signatures for a verification key chosen by the verifier.

The above proof approach works only with synchronous sessions (i.e., for synchronous one-many NMZK). Indeed Π_{nm} is secure only in the synchronous case. In order to deal with the asynchronous case we rely on the *honest-extractability* of Π_{nm}. We recall that Π^{OR} is run in parallel with Π_{nm} in order to ensure that either the witness for an $\mathcal{NP}$-statement x is known or the trapdoor has been *correctly* committed using Π_{nm}. For our propose we only need to ensure that the MiM never commits to the trapdoor. If this is not the case than there exists a right session where the MiM is committing correctly to the trapdoor using Π_{nm} with non-negligible probability. This means that we can extract the message committed by the MiM by just relying on the honest-extractability of Π_{nm}. Therefore we can make a reduction to the hiding of Π_{nm}[8].

In the remained hybrids we switch the witness used to compute the transcript of Π^{OR}: we start to prove using Π^{OR} that the message committed in Π_{nm} are two signatures [9]. In order to prove that also the MiM continues to not use the trapdoor in all right sessions the trapdoor we will relay on the adaptive-input Special HVZK property of Π^{OR}. In more details, in the right sessions we can extracted a witness w_{OR} from the transcript of Π^{OR}, and then check that w_{OR} does not correspond to the trapdoor and to the opening information of the commitment computed using Π_{nm}. In these reductions it is crucial that the rewinds made by the extractor of Π^{OR} do not disturb the challengers involved in the reductions.

To demonstrate that the view of the MiM in this hybrid experiments remains indistinguishable from the view of the MiM in the real game we will relay on the hiding of Π_{nm} and on the adaptive-input Special HVZK of Π^{OR}.

From one-many NMZK to synchronous many-many NMZK. Our one-many NMZK is also synchronous many-many NMZK. Indeed, the simulator can extract (simultaneously) the

[8]A rewind made in an asynchronous session does not interfere with (i.e., does not rewind) the challenger of the hiding of Π_{nm}.

[9]For this part of the proof we use the approach that we explained in the previous paragraph.

trapdoor from the right sessions, playing as described above. The only substantial difference is that we need to use a many-one non-malleable commitment with all the properties listed above. Following the approach proposed in the security proof of Proposition 1 provided in [LPV08], it is possible to claim that a synchronous (one-one) non-malleable commitment is also synchronous many-one non-malleable.

4.1.3 4-Round Secure Multi-Party Coin Tossing

Our MPCT protocol will critically make use of our delayed-input synchronous many-many NMZK from OWFs, and of an additional instantiation of Π^{OR}. Similarly to [GMPP16] our protocol consists of each party committing to a random string r, that is then sent in the clear in the last round. Moreover there will be a simulatable proof of correctness of the above commitment w.r.t. r, that is given to all parties independently. The output consists of the $\bigoplus$ of all opened strings. We now discuss in more details the messages exchanged by a pair of parties P_1 and P_2 in our multi-party coin tossing protocol Π_{MPCT}. The generalization to n players is straightforward and discussed in Section 4.3.1.

Informal description of the protocol. P_1, using a perfectly binding computationally hiding commitment scheme, commits in the first round to a random string r_1 two times thus obtaining $\mathtt{com}_0, \mathtt{com}_1$. Moreover P_1 runs Π^{OR} in order to prove knowledge of either the message committed in $\mathtt{com}_0$ or the message committed in $\mathtt{com}_1$. In the last (fourth) round P_1 sends r_1. In parallel, an execution of a NMZK ensures that both $\mathtt{com}_0$ and $\mathtt{com}_1$ contain the same message r_1 (that is sent in the fourth round)[10]. When P_1 receives the last round that contains r_2, P_1 computes and outputs $r_1 \oplus r_2$. P_2 symmetrically executes the same steps using as input r_2.

[10]Notice here how crucial is to delayed-input have synchronous many-many NMZK.

The simulator for Π_{MPCT} runs the simulator of NMZK and extracts the input $r^\star$ from the malicious party using the PoK extractor of Π^{OR}. At this point the simulator invokes the functionality thus obtaining r and plays in the last round $r_s = r \oplus r^\star$. Note that the values that the simulator commits in $\mathtt{com}_0$ and $\mathtt{com}_1$ are unrelated to r_s and this is possible because the NMZK is simulated. The extraction of the input from the adversary made by the simulator needs more attention. Indeed the security of NMZK will ensure that, even though the simulator cheats (he commits to a random string in both $\mathtt{com}_0$ and $\mathtt{com}_1$) the adversary can not do the same. Therefore the only way he can complete an execution of Π_{MPCT} consists of committing two times to $r^\star$ in the first round, and send the same value in the fourth round. This means that the value extracted (in the third round) from the PoK extractor of Π^{OR} is the input of the malicious party. Our security proof consists of showing the indistinguishability of hybrid experiments. The first hybrid experiment differs from the real game by using the simulator of NMZK. The simulator, in order to extract the trapdoor from the adversary, rewinds from the third to the second round, thus rewinding also Π^{OR}. Indeed the adversary, for every different second round of the NMZK could sent a different second round for Π^{OR}. This becomes a problem when we consider the hybrid experiment H_i where the witness for Π^{OR} changes. Due to the rewinds made by the simulator of the NMZK it is not clear how to rely on the security of the WI property of Π^{OR} (the challenger of WI would be rewound). This is the reason why, also in this case, we need to consider an intermediate hybrid experiment H^{w_0,w_1} where both witnesses of Π^{OR} can be used. Then we can prove the indistinguishability between H^{w_0,w_1} and H_i still relying on the Special HVZK of the sub-protocol used in Π^{OR} (Blum's protocol suffices in this case).

4.2 4-Round Delayed-Input NMZK from OWFs

4.2.1 Our Protocol: NMZK.

For our construction of a 4-round delayed-input non-malleable zero knowledge $\mathsf{NMZK} = (\mathcal{P}_{\mathsf{NMZK}}, \mathcal{V}_{\mathsf{NMZK}})$ for the $\mathcal{NP}$-language L we use the following tools.

1. A signature scheme $\Sigma = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$;
2. A 4-round public-coin synchronous honest-extractable non-malleable commitment scheme $\mathsf{NM} = (\mathcal{S}, \mathcal{R})$.
3. Two instantiations of the adaptive-input special sound LS protocol in order to construct a 4-round delayed-input public-coin proof system for the OR of statements $\Pi_{\mathsf{OR}} = (\mathcal{P}_{\mathsf{OR}}, \mathcal{V}_{\mathsf{OR}})$ as described in Section. 2.2. More in details we use the following proof systems.
    1. A 4-round delayed-input public coin $\mathsf{LS}_L = (\mathcal{P}_L, \mathcal{V}_L)$ for the $\mathcal{NP}$-language L with adaptive-input Special HVZK simulator S_L. $\mathsf{LS}_L = (\mathcal{P}_L, \mathcal{V}_L)$ is adaptive-input special sound for the corresponding relation Rel_L with instance length ℓ_L.
    2. A 4-round delayed-input public coin $\mathsf{LS}_{\mathsf{nm}} = (\mathcal{P}_{\mathsf{nm}}, \mathcal{V}_{\mathsf{nm}})$ with adaptive-input Special HVZK simulator S_{nm}. $\mathsf{LS}_{\mathsf{nm}} = (\mathcal{P}_{\mathsf{nm}}, \mathcal{V}_{\mathsf{nm}})$ is adaptive-input special sound for the $\mathcal{NP}$-relation $\mathsf{Rel}_{\mathsf{L_{nm}}}$ where

$$\begin{aligned}
&L_{\mathsf{nm}} = \{(\mathsf{vk}, \tau = (\mathtt{id}, \mathsf{nm}_1, \mathsf{nm}_2, \mathsf{nm}_3, \mathsf{nm}_4), s_1 :\\
&\exists(\mathtt{dec}_{\mathsf{nm}}, s_0, \sigma_1, \mathtt{msg}_1, \sigma_2, \mathtt{msg}_2) \text{ s.t.}\\
&\mathsf{Ver}(\mathsf{vk}, \mathtt{msg}_1, \sigma_1) = 1 \text{ AND} \mathsf{Ver}(\mathsf{vk}, \mathtt{msg}_2, \sigma_2) = 1\\
&\text{AND } \mathtt{msg}_1 \neq \mathtt{msg}_2 \text{ AND}\\
&\mathcal{R} \text{ accepts } (\mathtt{id}, s_1, \mathtt{dec}_{\mathsf{nm}}) \text{ as a valid decommitment of } \tau\\
&\text{AND } s_0 \oplus s_1 = \sigma_1 || \sigma_2\}.
\end{aligned}$$

    We denote with ℓ_{nm} the dimension of the instances belonging to L_{nm}. Informally by running $\mathsf{LS}_{\mathsf{nm}}$ one can prove that the message committed using a non-malleable commitment XORed with the value s_1 represents two signatures for two different messages w.r.t. the verification key vk.

Moreover Π^{OR} is also adaptive-input PoK for the relation $\mathsf{Rel}_{\mathsf{L_{OR}}} = \{((x_L, x_{\mathsf{nm}}), w) : ((x_L, w) \in \mathsf{Rel}_L) \text{ OR } ((x_{\mathsf{nm}}, w) \in \mathsf{Rel}_{\mathsf{L_{nm}}})\}$ (see Theorem 35 in Section. 4.4.1 for more details).

Overview of our protocol: NMZK. We now give an high-level description of our delayed-input NMZK of Fig. 4.1. For a formal description see Fig. 4.2.

In the **first round** $\mathcal{V}_{\mathsf{NMZK}}$ computes a pair of signature-verification keys $(\mathsf{sk}, \mathsf{vk})$ sending vk to $\mathcal{P}_{\mathsf{NMZK}}$. Also $\mathcal{V}_{\mathsf{NMZK}}$ computes the (public coin) first rounds nm_1 of NM, $\mathsf{ls}^1_L \leftarrow \mathcal{V}_L(1^\lambda, \ell_L)$ and $\mathsf{ls}^1_{\mathsf{nm}} \leftarrow \mathcal{V}_L(1^\lambda, \ell_{\mathsf{nm}})$. $\mathcal{V}_{\mathsf{NMZK}}$ completes the first round by sending $(\mathsf{vk}, \mathsf{ls}^1_L, \mathsf{ls}^1_{\mathsf{nm}}, \mathsf{nm}^1)$ to $\mathcal{P}_{\mathsf{NMZK}}$.

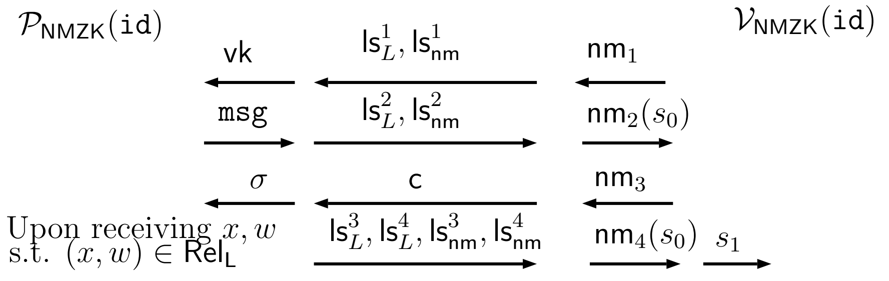
In the **second round** $\mathcal{P}_{\mathsf{NMZK}}$ computes $\mathsf{ls}^2_L \leftarrow \mathcal{P}_L(1^\lambda, \mathsf{ls}^1_L, \ell_L)$ and sends ls^2_L. Furthermore picks $\mathsf{ls}^3_{\mathsf{nm}} \leftarrow \{0,1\}^\lambda$ and runs $\mathsf{ls}^2_{\mathsf{nm}} \leftarrow S_{\mathsf{nm}}(1^\lambda, \mathsf{ls}^1_{\mathsf{nm}}, \mathsf{ls}^3_{\mathsf{nm}}, \ell_{\mathsf{nm}})$ in order to send $\mathsf{ls}^2_{\mathsf{nm}}$. $\mathcal{P}_{\mathsf{NMZK}}$ now commits to a random message s_0 using the non-malleable commitment NM by running $\mathcal{S}$ on input $1^\lambda, s_0, \mathsf{nm}_1$ and the identity $\mathtt{id}$ thus obtaining and sending nm_2. Also $\mathcal{P}_{\mathsf{NMZK}}$ sends a random message $\mathtt{msg}$.

In the **third round** of the protocol, upon receiving $\mathtt{msg}$, $\mathcal{V}_{\mathsf{NMZK}}$ computes and sends a signature σ of $\mathtt{msg}$ by running $\mathsf{Sign}(\mathsf{sk}, \mathtt{msg})$. $\mathcal{V}_{\mathsf{NMZK}}$ picks and sends $\mathsf{c} \leftarrow \{0,1\}^\lambda$. Also he computes and sends the (public coin) third rounds nm_3 of NM.

In the **fourth round** $\mathcal{P}_{\mathsf{NMZK}}$ checks whether or not σ is a valid signature for $\mathtt{msg}$ w.r.t. the verification key vk. In the negative case $\mathcal{P}_{\mathsf{NMZK}}$ aborts, otherwise he continues with the following steps. $\mathcal{P}_{\mathsf{NMZK}}$ computes $\mathsf{ls}^3_L = \mathsf{ls}^3_{\mathsf{nm}} \oplus \mathsf{c}$. Upon receiving the instance x to be proved and the witness w s.t. $(x, w) \in \mathsf{Rel}_L$, $\mathcal{P}_{\mathsf{NMZK}}$ completes the transcript for LS_L running $\mathsf{ls}^4_L \leftarrow \mathcal{P}_L(x, w, \mathsf{ls}^3_L)$. At this point $\mathcal{P}_{\mathsf{NMZK}}$ completes the commitment of s_0 by running $\mathcal{S}$ on input nm_3 thus obtaining $(\mathsf{nm}_4, \mathsf{dec}_{\mathsf{nm}})$. $\mathcal{P}_{\mathsf{NMZK}}$ picks a random string s_1, sets $x_{\mathsf{nm}} = (\mathsf{vk}, \mathtt{id}, \mathsf{nm}_1, \mathsf{nm}_2, \mathsf{nm}_3, \mathsf{nm}_4, s_1)$ and runs $\mathsf{ls}^4_{\mathsf{nm}} \leftarrow S_{\mathsf{nm}}(x_{\mathsf{nm}})$. $\mathcal{P}_{\mathsf{NMZK}}$ completes the fourth round by sending $(\mathsf{ls}^3_L, \mathsf{ls}^4_L, \mathsf{nm}_4, s_1, \mathsf{ls}^3_{\mathsf{nm}}, \mathsf{ls}^4_{\mathsf{nm}}, x, x_{\mathsf{nm}})$.

The verifier $\mathcal{V}_{\mathsf{NMZK}}$ accepts x iff the following conditions are satisfied:

1. c is equal to $\mathsf{ls}^3_L \oplus \mathsf{ls}^3_{\mathsf{nm}}$;
2. $\mathcal{V}_L(x, \mathsf{ls}^1_L, \mathsf{ls}^2_L, \mathsf{ls}^3_L, \mathsf{ls}^4_L) = 1$ and
3. $\mathcal{V}_{\mathsf{nm}}(x_{\mathsf{nm}}, \mathsf{ls}^1_{\mathsf{nm}}, \mathsf{ls}^2_{\mathsf{nm}}, \mathsf{ls}^3_{\mathsf{nm}}, \mathsf{ls}^4_{\mathsf{nm}}) = 1$.


$\mathcal{P}_{\mathsf{NMZK}}(\mathtt{id})$
$\mathcal{V}_{\mathsf{NMZK}}(\mathtt{id})$
vk
$\mathsf{ls}^1_L, \mathsf{ls}^1_{\mathsf{nm}}$
nm_1
msg
$\mathsf{ls}^2_L, \mathsf{ls}^2_{\mathsf{nm}}$
$\mathsf{nm}_2(s_0)$
σ
c
nm_3
Upon receiving x, w s.t. $(x, w) \in \mathsf{Rel_L}$
$\mathsf{ls}^3_L, \mathsf{ls}^4_L, \mathsf{ls}^3_{\mathsf{nm}}, \mathsf{ls}^4_{\mathsf{nm}}$
$\mathsf{nm}_4(s_0)$
s_1

- vk is a a verification key of a signature scheme and σ is a valid signature of the message $\mathtt{msg}$.
- s_0 and s_1 are two random strings.
- $\tau = (\mathtt{id}, \mathsf{nm}_1, \mathsf{nm}_2, \mathsf{nm}_3, \mathsf{nm}_4)$ represents the transcript of $\langle \mathcal{S}(s_0), \mathcal{R}\rangle(\mathtt{id})$ that is, a commitment of the message s_0 computed using the synchronous honest-extractable non-malleable commitment scheme NM.
- $((\mathsf{ls}^1_L, \mathsf{ls}^1_{\mathsf{nm}}), (\mathsf{ls}^2_L, \mathsf{ls}^2_{\mathsf{nm}}), \mathsf{c}, (\mathsf{ls}^3_L, \mathsf{ls}^4_L, \mathsf{ls}^3_{\mathsf{nm}}, \mathsf{ls}^4_{\mathsf{nm}}))$ is the transcript generated from an execution of Π_{OR}, in more details:
  - c is equal to $\mathsf{ls}^3_{\mathsf{nm}} \oplus \mathsf{ls}^3_L$.
  - $(\mathsf{ls}^1_L, \mathsf{ls}^2_L, \mathsf{ls}^3_L, \mathsf{ls}^4_L)$ is the transcript output from the honest prover procedure of LS_L proving the knowledge of the witness for $x \in L$.
  - $(\mathsf{ls}^1_{\mathsf{nm}}, \mathsf{ls}^2_{\mathsf{nm}}, \mathsf{ls}^3_{\mathsf{nm}}, \mathsf{ls}^4_{\mathsf{nm}})$ is the transcript output of a adaptive-input Special HVZK simulator of $\mathsf{LS}_{\mathsf{nm}}$ proving knowledge of a decommitment of τ to the message s_0 s.t. $s_0 \oplus s_1 = \sigma_1 || \sigma_2$ where σ_1, σ_2 are two signatures of two different messages w.r.t vk.

Figure 4.1 Our 4-round delayed-input NMZK

The simulator extractor. Informally, the simulator $\mathsf{Sim}_{\mathsf{NMZK}}$

Common input: security parameter λ, identity $\texttt{id} \in \{0,1\}^\lambda$ instances length: ℓ_L, ℓ_{nm}.
Input to $\mathcal{P}_{\mathsf{NMZK}}$: (x, w) s.t. $(x, w) \in \mathsf{Rel_L}$, with (x, w) available only in the 4th round.

1. $\mathcal{V}_{\mathsf{NMZK}} \rightarrow \mathcal{P}_{\mathsf{NMZK}}$
    1. Run $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{Gen}(1^\lambda)$.
    2. Run $\mathsf{ls}^1_L \leftarrow \mathcal{V}_L(1^\lambda, \ell_L)$.
    3. Run $\mathsf{ls}^1_{\mathsf{nm}} \leftarrow \mathcal{V}_{\mathsf{nm}}(1^\lambda, \ell_{\mathsf{nm}})$.
    4. Run $\mathcal{R}$ on input 1^λ and $\texttt{id}$ thus obtaining nm_1.
    5. Send $(\mathsf{vk}, \mathsf{ls}^1_L, \mathsf{ls}^1_{\mathsf{nm}}, \mathsf{nm}_1)$ to $\mathcal{P}_{\mathsf{NMZK}}$.
2. $\mathcal{P}_{\mathsf{NMZK}} \rightarrow \mathcal{V}_{\mathsf{NMZK}}$
    1. Run $\mathsf{ls}^2_L \leftarrow \mathcal{P}_L(1^\lambda, \ell_L)$.
    2. Pick $\mathsf{ls}^3_{\mathsf{nm}} \leftarrow \{0,1\}^\lambda$ run $\mathsf{ls}^2_{\mathsf{nm}} \leftarrow S_{\mathsf{nm}}(1^\lambda, \mathsf{ls}^1_{\mathsf{nm}}, \mathsf{ls}^3_{\mathsf{nm}}, \ell_{\mathsf{nm}})$.
    3. Pick $s_0 \leftarrow \{0,1\}^\lambda$ and run $\mathcal{S}$ on input 1^λ, $\texttt{id}$, nm_1, s_0 (in order to commit to the message s_0) thus obtaining nm_2.
    4. Pick a message $\texttt{msg} \leftarrow \{0,1\}^\lambda$.
    5. Send $(\mathsf{ls}^2_L, \mathsf{ls}^2_{\mathsf{nm}}, \texttt{msg}, \mathsf{nm}_2)$ to $\mathcal{V}_{\mathsf{NMZK}}$.
3. $\mathcal{V}_{\mathsf{NMZK}} \rightarrow \mathcal{P}_{\mathsf{NMZK}}$
    1. Pick $\mathsf{c} \leftarrow \{0,1\}^\lambda$.
    2. Run $\mathcal{R}$ on input nm_2 thus obtaining nm_3.
    3. Run $\mathsf{Sign}(\mathsf{sk}, \texttt{msg})$ to obtain a signature σ of $\texttt{msg}$.
    4. Send $(\mathsf{c}, \mathsf{nm}_3, \sigma)$ to $\mathcal{P}_{\mathsf{NMZK}}$.
4. $\mathcal{P}_{\mathsf{NMZK}} \rightarrow \mathcal{V}_{\mathsf{NMZK}}$
    1. If $\mathsf{Ver}(\mathsf{vk}, \texttt{msg}, \sigma) \neq 1$ then abort, continue as follows otherwise.
    2. Compute $\mathsf{ls}^3_L = \mathsf{c} \oplus \mathsf{ls}^3_{\mathsf{nm}}$.
    3. Run $\mathsf{ls}^4_L \leftarrow \mathcal{P}_L(x, w, \mathsf{ls}^3_L)$.
    4. Run $\mathcal{S}$ on input nm_3 thus obtaining $(\mathsf{nm}_4, \mathsf{dec}_{\mathsf{nm}})$.
    5. Pick $s_1 \leftarrow \{0,1\}^\lambda$, set $x_{\mathsf{nm}} = (\mathsf{vk}, \mathsf{nm}_1, \mathsf{nm}_2, \mathsf{nm}_3, \mathsf{nm}_4, s_1)$ and run $\mathsf{ls}^4_{\mathsf{nm}} \leftarrow S_{\mathsf{nm}}(x_{\mathsf{nm}})$.
    6. Send $(\mathsf{ls}^3_L, \mathsf{ls}^4_L, \mathsf{nm}_4, s_1, \mathsf{ls}^3_{\mathsf{nm}}, \mathsf{ls}^4_{\mathsf{nm}}, x, x_{\mathsf{nm}})$ to $\mathcal{V}_{\mathsf{NMZK}}$.
5. $\mathcal{V}_{\mathsf{NMZK}}$: output 1 iff the following conditions are satisfied.
    1. c is equal to $\mathsf{ls}^3_L \oplus \mathsf{ls}^3_{\mathsf{nm}}$.
    2. $\mathcal{V}_L(x, \mathsf{ls}^1_L, \mathsf{ls}^2_L, \mathsf{ls}^3_L, \mathsf{ls}^4_L) = 1$.
    3. $\mathcal{V}_{\mathsf{nm}}(x_{\mathsf{nm}}, \mathsf{ls}^1_{\mathsf{nm}}, \mathsf{ls}^2_{\mathsf{nm}}, \mathsf{ls}^3_{\mathsf{nm}}, \mathsf{ls}^4_{\mathsf{nm}}) = 1$.

Figure 4.2 Formal construction of our delayed-input NMZK.

of our protocol interacts with the adversary $\mathcal{A}_{\mathsf{NMZK}}$ emulating both the prover in the left session and polynomially many verifiers in the right sessions. In the right sessions $\mathsf{Sim}_{\mathsf{NMZK}}$ interacts with $\mathcal{A}_{\mathsf{NMZK}}$ as the honest verifiers do. While, in the left session for an instance $x \in L$ chosen adaptively by $\mathcal{A}_{\mathsf{NMZK}}$, $\mathsf{Sim}_{\mathsf{NMZK}}$ proves, using Π_{OR}, that the message committed in NM contains two signatures of two different messages w.r.t. the verification key vk. In more details $\mathsf{Sim}_{\mathsf{NMZK}}$ runs the adaptive-input Special HVZK simulator of LS_L to complete the transcript for LS_L w.r.t. the instance x. In order to use the honest prover procedure to compute the transcript of $\mathsf{LS}_{\mathsf{nm}}$, $\mathsf{Sim}_{\mathsf{NMZK}}$ extracts two signatures for two different messages by rewinding $\mathcal{A}_{\mathsf{NMZK}}$ from the third to the second round and by committing to them using NM[11]. More precisely the simulator commits to a random string s_0, but computes s_1 s.t. $s_1 = (\sigma_1||\sigma_2) \oplus s_0$[12]. Therefore the execution of Π_{OR} can be completed by using the knowledge of the two signatures committed using NM. We use the xor trick originally provided in [COSV16] in order to avoid any additional requirement w.r.t. the underlying non-malleable commitment scheme NM. Indeed if the sender of NM could decide the message to commit in the last round, then $\mathsf{Sim}_{\mathsf{NMZK}}$ can simply compute the first round of NM, extract the signature, and compute the last round of NM by committing to $\sigma_1||\sigma_2$. It is important to observe that even though the non-malleable commitment scheme of [GPR16] fixes the message to be committed in the third round, there is in general no guarantee that such a scheme is secure against an adversary that adaptively chooses the challenge messages in the last round of the non-malleability security game. Therefore, even though the completeness of our scheme would work without using the trick of [COSV16], it would be unclear, in general, how to prove the security of our final scheme. A formal description of $\mathsf{Sim}_{\mathsf{NMZK}}$ can be found in the proof of Theorem 28.

[11]W.l.o.g. we assume that the signatures σ_1, σ_2 include the signed messages.

[12]For ease of exposition we will simply say that $\mathcal{A}_{\mathsf{NMZK}}$ commits to two signatures using NM.

4.2.2 Formal Description of Our Delayed-Input NMZK and Security Proof

The formal construction of our delayed-input NMZK $\mathsf{NMZK} = (\mathcal{P}_{\mathsf{NMZK}}, \mathcal{V}_{\mathsf{NMZK}})$ for the $\mathcal{NP}$-language L can be found in Fig. 4.2.

Theorem 28. If OWFs exist, then NMZK is a 4-round delayed-input NMZK AoK for $\mathcal{NP}$.

Proof. We divide the security proof in two parts, proving that NMZK enjoys delayed-input completeness and NMZK. The proof of NMZK is divided also in two lemmas, one for each of the two properties of Def. 1.3.1. Before that, we recall that $\mathsf{LS}_{\mathsf{nm}}$ and LS_L can be constructed from OWFs (see Sec. 1.1) as well as Σ (using [Rom90]) and the 4-round public-coin synchronous honest-extractable non-malleable commitment scheme NM (see Theorem 1 of [GPR16]).

(Delayed-Input) Completeness. The completeness follows directly from the delayed-input completeness of $\mathsf{LS}_{\mathsf{nm}}$ and LS_L, the correctness of NM and the validity of Σ. We observe that, due to the delayed-input property of LS_L, the statement x (and the respective witness w) are used by $\mathcal{P}_{\mathsf{NMZK}}$ only to compute the last round. Therefore also NMZK enjoys delayed-input completeness.

(Delayed-Input) NMZK. Following Definition 1.3.1 we start by describing how the simulator $\mathsf{Sim}_{\mathsf{NMZK}}$ for NMZK works. In the left session $\mathsf{Sim}_{\mathsf{NMZK}}$ interacts with the MiM adversary $\mathcal{A}_{\mathsf{NMZK}}$ in the following way. Upon receiving the first round, vk, ls^1_L, $\mathsf{ls}^1_{\mathsf{nm}}$, nm_1, from $\mathcal{A}_{\mathsf{NMZK}}$, $\mathsf{Sim}_{\mathsf{NMZK}}$ on input $\mathsf{ls}^1_{\mathsf{nm}}$ computes $\mathsf{ls}^2_{\mathsf{nm}}$ by running $\mathcal{P}_{\mathsf{nm}}$. $\mathsf{Sim}_{\mathsf{NMZK}}$ picks $\mathsf{ls}^3_L \leftarrow \{0,1\}^\lambda$ and runs S_L on input 1^λ, ℓ_L, ls^1_L, ls^3_L thus obtaining ls^2_L. $\mathsf{Sim}_{\mathsf{NMZK}}$, in order to commit to a random message s_0 runs $\mathcal{S}$ on input nm_1, the identity $\mathtt{id}$ and s_0 thus obtaining nm_2. $\mathsf{Sim}_{\mathsf{NMZK}}$ sends $\mathsf{ls}^2_L, \mathsf{ls}^2_{\mathsf{nm}}, \mathsf{nm}_2$ and a random message $\mathtt{msg}_1$ to $\mathcal{A}_{\mathsf{NMZK}}$. Upon receiving the third round, c, nm_3, σ_1, and instance x to be proved from $\mathcal{A}_{\mathsf{NMZK}}$, the simulator checks whether or not σ_1 is a valid signature for $\mathtt{msg}_1$ w.r.t. the verification key vk. In the negative case $\mathsf{Sim}_{\mathsf{NMZK}}$ aborts, otherwise $\mathsf{Sim}_{\mathsf{NMZK}}$ rewinds $\mathcal{A}_{\mathsf{NMZK}}$ from the third to the second round in order to obtain a second signature σ_2 for a different message $\mathtt{msg}_2$. After the extraction of the signatures $\mathsf{Sim}_{\mathsf{NMZK}}$ returns to the main thread and computes the fourth round as follows[13].

$\mathsf{Sim}_{\mathsf{NMZK}}$ completes the commitment of s_0 by running $\mathcal{S}$ on input nm_3 thus obtaining $(\mathsf{nm}_4, \mathtt{dec}_{\mathsf{nm}})$ and sending nm_4. Furthermore $\mathsf{Sim}_{\mathsf{NMZK}}$ sets s_1 s.t. $s_1 = (\sigma_1 || \sigma_2) \oplus s_0$,
$x_{\mathsf{nm}} = (\mathsf{vk}, \mathtt{id}, \mathsf{nm}_1, \mathsf{nm}_2, \mathsf{nm}_3, \mathsf{nm}_4, s_1)$, $w_{\mathsf{nm}} = (\mathtt{dec}_{\mathsf{nm}}, s_0, \sigma_1, \mathtt{msg}_1, \sigma_2, \mathtt{msg}_2)$ and completes the transcript for $\mathsf{LS}_{\mathsf{nm}}$ obtaining $\mathsf{ls}^4_{\mathsf{nm}}$ by running the prover procedure $\mathcal{P}_{\mathsf{nm}}$ on input x_{nm}, w_{nm} and $\mathsf{ls}^3_L \oplus \mathsf{c}$. At this point $\mathsf{Sim}_{\mathsf{NMZK}}$ runs the adaptive-input Special HVZK simulator S_L on input x thus obtaining ls^4_L. Then the values
$(\mathsf{ls}^3_L, \mathsf{ls}^4_L, \mathsf{nm}_4, s_1, \mathsf{ls}^3_{\mathsf{nm}}, \mathsf{ls}^4_{\mathsf{nm}}, x, x_{\mathsf{nm}})$ are sent to $\mathcal{A}_{\mathsf{NMZK}}$. At the end of the execution $\mathsf{Sim}_{\mathsf{NMZK}}$ outputs $\mathcal{A}_{\mathsf{NMZK}}$'s view in the main thread. Furthermore, he uses the extractor of LS_L to extract and output, from the $\mathsf{poly}(\lambda)$ right sessions, the witnesses $\tilde{w}_1, \dots, \tilde{w}_{\mathsf{poly}(\lambda)}$ used by $\mathcal{A}_{\mathsf{NMZK}}$ to compute the transcript of Π^{OR} (the witnesses correspond to statements $\tilde{x}_i$ proved by $\mathcal{A}_{\mathsf{NMZK}}$ in the i-th right session, for $i = 1, \dots, \mathsf{poly}(\lambda)$).

Lemma 4.2.1. $\{\mathsf{Sim}_{\mathsf{NMZK}}{}^1(1^\lambda, z)\}_{\lambda \in \mathbb{N}, z \in \{0,1\}^*} \approx \{\mathsf{View}^{\mathcal{A}_{\mathsf{NMZK}}}(1^\lambda, z)\}_{\lambda \in \mathbb{N}, z \in \{0,1\}^*}$, *where* $\mathsf{Sim}_{\mathsf{NMZK}}{}^1(1^\lambda, z)$ *denotes the 1st output of* $\mathsf{Sim}_{\mathsf{NMZK}}$.

In order to prove the above lemma we consider the series of hybrid experiments described below. In the proof we denote with $\{\mathsf{View}^{\mathcal{A}_{\mathsf{NMZK}}}_{\mathcal{H}_i}(1^\lambda, z)\}_{\lambda \in \mathbb{N}, z \in \{0,1\}^*}$ the random variable that describes the view of $\mathcal{A}_{\mathsf{NMZK}}$ in the hybrid $\mathcal{H}_i(1^\lambda, z)$. Let p the probability that in the real execution $\mathcal{A}_{\mathsf{NMZK}}$ completes the left session.

- We start considering the hybrid experiment $\mathcal{H}_0(1^\lambda, z)$ in which in the left session $\mathcal{P}_{\mathsf{NMZK}}$ interacts with $\mathcal{A}_{\mathsf{NMZK}}$ and in the i-th right session $\mathcal{V}_{\mathsf{NMZK}i}$ interacts with $\mathcal{A}_{\mathsf{NMZK}}$, for $i = 1, \dots, \mathsf{poly}(\lambda)$. Note that $\{\mathsf{View}^{\mathcal{A}_{\mathsf{NMZK}}}_{\mathcal{H}_0}(1^\lambda, z)\}_{\lambda \in \mathbb{N}, z \in \{0,1\}^*} = \{\mathsf{View}^{\mathcal{A}}_{\mathsf{NMZK}}(1^\lambda, z)\}_{\lambda \in \mathbb{N}, z \in \{0,1\}^*}$.

[13]Note that it is possible to complete the main thread, due to the delayed-input completeness of $\mathsf{LS}_{\mathsf{nm}}$, and to the fact that we do not need to change the second round of NM (that is, we do not need to change the committed message s_0) in order to have $x_{\mathsf{nm}} \in L_{\mathsf{nm}}$.

The hybrid experiment $\mathcal{H}_1(1^\lambda, z)$ differs from $\mathcal{H}_0(1^\lambda, z)$ only in the fact that in the left session of $\mathcal{H}_1(1^\lambda, z)$ $\mathcal{A}_{\mathsf{NMZK}}$ is rewound from the third to the second round, in order to extract two signatures σ_1, σ_2 for two distinct messages $(\mathtt{msg}_1, \mathtt{msg}_2)$ w.r.t. a verification key vk. Note that after p rewinds the probability of not obtaining a valid new signature is less than $1/2$. Therefore the probability that $\mathcal{A}_{\mathsf{NMZK}}$ does not give a second valid signature for a randomly chosen message after λ/p rewinds is negligible in λ. For the above reason the procedure of extraction of signatures for different messages in $\mathcal{H}_1(1^\lambda, z)$ succeeds except with negligible probability. Observe that the above deviation increases the abort probability of the experiment only by a negligible amount, therefore $\{\mathsf{View}^{\mathcal{A}_{\mathsf{NMZK}}}_{\mathcal{H}_0}(1^\lambda, z)\}_{\lambda\in\mathbb{N}, z\in\{0,1\}^*} \equiv_s \{\mathsf{View}^{\mathcal{A}_{\mathsf{NMZK}}}_{\mathcal{H}_1}(1^\lambda, z)\}_{\lambda\in\mathbb{N}, z\in\{0,1\}^*}$.

- The hybrid experiment $\mathcal{H}_2(1^\lambda, z)$ differs from $\mathcal{H}_1(1^\lambda, z)$ only in the message committed using NM. Indeed $\mathcal{P}_{\mathsf{NMZK}}$ commits using NM to two signatures σ_1, σ_2 of two distinct messages $(\mathtt{msg}_1, \mathtt{msg}_2)$ instead of a random message. In more details, $\mathcal{P}_{\mathsf{NMZK}}$ commits to a random string s_0 using NM and in 4th round sets and sends $s_1 = (\sigma_1||\sigma_2) \oplus s_0$, instead of sending s_1 as a random string. Observe that the procedure of extraction of the signatures succeeds in $\mathcal{H}_2(1^\lambda, z)$ with non-negligible probability, because the first three rounds are played exactly as in $\mathcal{H}_1(1^\lambda, z)$. Now we can claim that $\{\mathsf{View}^{\mathcal{A}_{\mathsf{NMZK}}}_{\mathcal{H}_2}(1^\lambda, z)\}_{\lambda\in\mathbb{N}, z\in\{0,1\}^*}$ and $\{\mathsf{View}^{\mathcal{A}_{\mathsf{NMZK}}}_{\mathcal{H}_1}(1^\lambda, z)\}_{\lambda\in\mathbb{N}, z\in\{0,1\}^*}$ are computationally indistinguishable by using the computationally-hiding property of NM. Suppose by contradiction that there exist an adversary $\mathcal{A}_{\mathsf{NMZK}}$ and a distinguisher $\mathcal{D}_{\mathsf{NMZK}}$ such that $\mathcal{D}_{\mathsf{NMZK}}$ distinguishes $\{\mathsf{View}^{\mathcal{A}_{\mathsf{NMZK}}}_{\mathcal{H}_1}(1^\lambda, z)\}_{\lambda\in\mathbb{N}, z\in\{0,1\}^*}$ from $\{\mathsf{View}^{\mathcal{A}_{\mathsf{NMZK}}}_{\mathcal{H}_2}(1^\lambda, z)\}_{\lambda\in\mathbb{N}, z\in\{0,1\}^*}$. Then we can construct an adversary $\mathcal{A}_{\mathsf{Hiding}}$ that breaks the computationally hiding of NM in the following way. $\mathcal{A}_{\mathsf{Hiding}}$ sends to the challenger of the hiding game $\mathcal{C}_{\mathsf{Hiding}}$ two random messages (m_0, m_1). Then, in the left session $\mathcal{A}_{\mathsf{Hiding}}$ acts as $\mathcal{P}_{\mathsf{NMZK}}$ except for messages of NM for which he acts as proxy between $\mathcal{C}_{\mathsf{Hiding}}$ and $\mathcal{A}_{\mathsf{NMZK}}$. When $\mathcal{A}_{\mathsf{Hiding}}$ computes the last round of the left session $\mathcal{A}_{\mathsf{Hiding}}$ sets and sends $s_1 = \sigma_1||\sigma_2 \oplus m_0$. In the right sessions $\mathcal{A}_{\mathsf{Hiding}}$ interacts with $\mathcal{A}_{\mathsf{ZK}}$ acting as $\mathcal{V}_{\mathsf{NMZK}}$ does. At the end of the execution $\mathcal{A}_{\mathsf{Hiding}}$ runs $\mathcal{D}_{\mathsf{NMZK}}$ and outputs what $\mathcal{D}_{\mathsf{NMZK}}$ outputs. It is easy to see that if $\mathcal{C}_{\mathsf{Hiding}}$ commits to m_1 then, $\mathcal{A}_{\mathsf{ZK}}$ acts as in $\mathcal{H}_1(1^\lambda, z)$, otherwise he acts as in $\mathcal{H}_2(1^\lambda, z)$. Note that the reduction to the hiding property of NM is possible because the rewinds to extract a second signature do not affect the execution with the challenger of NM that remains straight-line.

- The hybrid experiment $\mathcal{H}_3(1^\lambda, z)$ differs from $\mathcal{H}_2(1^\lambda, z)$ in the way the transcript of $\mathsf{LS}_{\mathsf{nm}}$ is computed. More precisely, the prover $\mathcal{P}_{\mathsf{nm}}$ of $\mathsf{LS}_{\mathsf{nm}}$ is used to compute the messages $\mathsf{ls}^2_{\mathsf{nm}}$ and $\mathsf{ls}^4_{\mathsf{nm}}$ instead of using the adaptive-input Special HVZK simulator. Note that due to the delayed-input property of $\mathsf{LS}_{\mathsf{nm}}$ the statement $x_{\mathsf{nm}} = (\mathsf{vk}, \mathsf{nm}_1, \mathsf{nm}_2, \mathsf{nm}_3, \mathsf{nm}_4, s_1)$ and the witness $w_{\mathsf{nm}} = (\mathtt{dec}_{\mathsf{nm}}, s_0, \sigma_1, \mathtt{msg}_1, \sigma_2, \mathtt{msg}_2)$ are required by $\mathcal{P}_{\mathsf{nm}}$ only to compute $\mathsf{ls}^4_{\mathsf{nm}}$ and are not needed to compute $\mathsf{ls}^2_{\mathsf{nm}}$. Observe that the procedure of extraction of the signatures succeeds in $\mathcal{H}_3(1^\lambda, z)$ with non-negligible probability due to the adaptive-input Special HVZK of $\mathsf{LS}_{\mathsf{nm}}$. From the adaptive-input Special HVZK of $\mathsf{LS}_{\mathsf{nm}}$ it follows that $\{\mathsf{View}^{\mathcal{A}_{\mathsf{NMZK}}}_{\mathcal{H}_2}(1^\lambda, z)\}_{\lambda\in\mathbb{N}, z\in\{0,1\}^*}$ and $\{\mathsf{View}^{\mathcal{A}_{\mathsf{NMZK}}}_{\mathcal{H}_3}(1^\lambda, z)\}_{\lambda\in\mathbb{N}, z\in\{0,1\}^*}$ are computationally indistinguishable.

- The hybrid $\mathcal{H}_4(1^\lambda, z)$ differs from $\mathcal{H}_3(1^\lambda, z)$ in the way the transcript of LS_L is computed. More precisely, the adaptive-input Special HVZK simulator of LS_L is used to compute the messages ls^2_L and ls^4_L using as input ls^1_L received by $\mathcal{A}_{\mathsf{NMZK}}$, the statement x and a random string ls^3_L chosen by the hybrid experiment. We observe that in order to complete the execution of Π_{OR} the honest prover procedure $\mathcal{P}_{\mathsf{nm}}$ can be used on input x_{nm}, w_{nm} and $\mathsf{ls}^3_{\mathsf{nm}} = \mathsf{ls}^3_L \oplus \mathsf{c}$. Moreover adaptive-input Special HVZK of LS_L ensures that the extraction procedure of the signatures succeeds in $\mathcal{H}_4(1^\lambda, z)$ with non-negligible probability and that $\{\mathsf{View}^{\mathcal{A}_{\mathsf{NMZK}}}_{\mathcal{H}_4}(1^\lambda, z)\}_{\lambda\in\mathbb{N}, z\in\{0,1\}^*} \approx \{\mathsf{View}^{\mathcal{A}_{\mathsf{NMZK}}}_{\mathcal{H}_3}(1^\lambda, z)\}_{\lambda\in\mathbb{N}, z\in\{0,1\}^*}$. Note that $\mathcal{H}_4(1^\lambda, z)$ corresponds to the simulated experiment, that is the experiment where $\mathsf{Sim}_{\mathsf{NMZK}}$ interacts with the adversary $\mathcal{A}_{\mathsf{NMZK}}$ emulating both a prover in the left session and polynomially many verifiers in the right sessions. This implies that $\{\mathsf{View}^{\mathcal{A}_{\mathsf{NMZK}}}_{\mathcal{H}_4}(1^\lambda, z)\}_{\lambda\in\mathbb{N}, z\in\{0,1\}^*} = \{S^1(1^\lambda, z)\}_{\lambda\in\mathbb{N}, z\in\{0,1\}^\star}$.

The proof ends with the observation that for all $\lambda \in \mathbb{N}, z \in \{0,1\}^*$ it holds that: $\{\mathsf{View}^{\mathcal{A}}_{\mathsf{NMZK}}(1^\lambda, z)\}_{\lambda,z} = \{\mathsf{View}^{\mathcal{A}_{\mathsf{NMZK}}}_{\mathcal{H}_0}(1^\lambda, z)\}_{\lambda,z} \approx$

$$\cdots \approx \{\mathsf{View}^{\mathcal{A}_{\mathsf{NMZK}}}_{\mathcal{H}_4}(1^\lambda, z)\}_{\lambda,z} = \{S^1(1^\lambda, z)\}_{\lambda,z}$$

Lemma 4.2.2. *Let $\tilde{x}_1, \ldots, \tilde{x}_{\mathsf{poly}(\lambda)}$ be the right-session statements appearing in $\mathsf{View} = \mathsf{Sim}_{\mathsf{NMZK}}{}^1(1^\lambda, z)$ and let $\mathtt{id}$ be the identity of the left session and $\tilde{\mathtt{id}}_1, \ldots, \tilde{\mathtt{id}}_{\mathsf{poly}(\lambda)}$ be the identities of right sessions appearing in View. If the i-th right session is accepting and $\mathtt{id} \neq \tilde{\mathtt{id}}_i$ for $i = 1, \ldots, \mathsf{poly}(\lambda)$, then except with negligible probability, the second output of $\mathsf{Sim}_{\mathsf{NMZK}}(1^\lambda, z)$ is $\tilde{w}_i$ such that $(\tilde{x}_i, \tilde{w}_i) \in \mathsf{Rel_L}$ for $i = 1, \ldots, \mathsf{poly}(\lambda)$.*

Overview of the proof of Lemma 4.2.2.. We now reconsider the hybrid experiments $\mathcal{H}_k$ for $k = 0, \ldots, 4$ described in the security proof of Lemma 4.2.1, and prove that in all hybrids $\mathcal{A}_{\mathsf{NMZK}}$ chooses a statement $\tilde{x}_i \in L$ and uses the witnesses for it to complete the transcripts of Π^{OR} in the i-th the right session, for $i = 1, \ldots, \mathsf{poly}(\lambda)$.

In the hybrids $\mathcal{H}_0, \ldots, \mathcal{H}_2$ we will relay on the following chain of implications. If in the right sessions $\mathcal{A}_{\mathsf{NMZK}}$ never commits to the signatures[14] then the transcript computed using $\mathsf{LS_{nm}}$ corresponds to a false instance. Since Π^{OR} enjoys adaptive-input PoK property we can argue that the transcript for Π^{OR} is computed by $\mathcal{A}_{\mathsf{NMZK}}$ using a witness $\tilde{w}_i$ for the statement $\tilde{x}_i$. This means that $\mathcal{A}_{\mathsf{NMZK}}$ completes transcript corresponding to LS_L using the witness $\tilde{w}_i$ s.t. $(\tilde{x}_i, \tilde{w}_i) \in \mathsf{Rel_L}$ for $i \in \{1, \ldots, \mathsf{poly}(\lambda)\}$.

In order to prove that $\mathcal{A}_{\mathsf{NMZK}}$ does not commit to two signatures in any of the right sessions of $\mathcal{H}_0, \ldots, \mathcal{H}_2$, we rely on the "mild" non-malleability and on the honest-extraction property enjoyed by NM. More precisely, in these hybrid experiments, we use the honest-extraction[15] property to extract the signatures from

[14]We recall that $\mathcal{A}_{\mathsf{NMZK}}$ to commit to the signatures follows the following procedure. $\mathcal{A}_{\mathsf{NMZK}}$ commits using NM to a message $\tilde{s}_0$ and sends a string $\tilde{s}_1$ s.t. $\tilde{s}_0 \oplus \tilde{s}_1 = \tilde{\sigma}_1 || \tilde{\sigma}_2$ and $\tilde{\sigma}_1, \tilde{\sigma}_2$ are two signatures for two different messages w.r.t. the verification key $\tilde{\mathsf{vk}}$ sent in the first round of the i-th right session

[15]Observe that in our case is sufficient that the extraction holds against honest sender, because for our security proof we only need to be sure that the commitment computed using NM is not a commitment of signatures.

the right sessions (that by contradiction are committed using NM by $\mathcal{A}_{\mathsf{NMZK}}$).

Since in $\mathcal{H}_2$ we are guaranteed that in all the right sessions $\mathcal{A}_{\mathsf{NMZK}}$ chooses a statement $\tilde{x}_i \in L$ and uses the witnesses for it to run Π^{OR}, we can claim that $\mathcal{A}_{\mathsf{NMZK}}$ still does the same in $\mathcal{H}_3$. Otherwise, we can construct a reduction that breaks the adaptive-input Special HVZK of $\mathsf{LS}_{\mathsf{nm}}$. Indeed, it is possible to check which witness is using $\mathcal{A}_{\mathsf{NMZK}}$ to compute Π^{OR} in the right sessions of $\mathcal{H}_3$ by running the extractor of Π^{OR}. Therefore, if in a right session, let us say the i-th, we do not extract a witness for statement $\tilde{x}_i$ (chosen by $\mathcal{A}_{\mathsf{NMZK}}$) from Π^{OR}, then it is possible to make a reduction to the adaptive-input Special HVZK of $\mathsf{LS}_{\mathsf{nm}}$. Similar arguments can be used to prove that also in $\mathcal{H}_4$ (which corresponds to the simulated experiment) we continue to extract the witnesses for $\tilde{x}_i \in L$ for $i = 1, \dots, \mathsf{poly}(\lambda)$. During the proof we need to show that the rewinds made by the honest-extractor and by the extractor of Π^{OR} do not interfere with the various reductions. Roughly speaking, our security proof works because only non-interactive primitives are used, therefore the rewinds made by the extractors of NM and of Π^{OR} do not rewind the challenger involved in the reductions. In particular, let us consider the hybrid $\mathcal{H}_3$ where we switch from the adaptive-input Special HVZK simulator of $\mathsf{LS}_{\mathsf{nm}}$ to the honest prover procedure and $\mathcal{H}_4$ where we start to use adaptive-input Special HVZK simulator of LS_L. In this reductions the rewinds made by the extractor of Π^{OR} do not affect the reduction. Indeed, when we rely on adaptive-input Special HVZK of LS_L ($\mathsf{LS}_{\mathsf{nm}}$) the honest prover procedure of $\mathsf{LS}_{\mathsf{nm}}$ (LS_L) can be used in order to complete the execution of Π^{OR}. In this way the third round ls^3_L ($\mathsf{ls}^3_{\mathsf{nm}}$) can be kept fixed thus computing $\mathsf{ls}^3_{\mathsf{nm}} = \mathsf{c}^i \oplus \mathsf{ls}^3_L$ ($\mathsf{ls}^3_L = \mathsf{c}^i \oplus \mathsf{ls}^3_{\mathsf{nm}}$) for every c^i that could be sent by $\mathcal{A}_{\mathsf{NMZK}}$ during the rewinds. We observe that it would be not clear how to do such a security proof by directly relying on the WI property of Π^{OR}. It follows the formal proof of this lemma. $\square$

Formal proof of Lemma 4.2.2. In order to simplify the security proof, here we actually consider the notions of *multi-SHVZK* and *multi-hiding* instead of adaptive-input Special HVZK and hiding. The only differences with the classical definition of adaptive-input Special HVZK is the following. Let $(\mathsf{ls}^1, \mathsf{ls}^3, x)$ be a challenge. The challenger of multi-SHVZK picks a random bit b and compute an accepting transcript $t = (\mathsf{ls}^1, \mathsf{ls}^2, \mathsf{ls}^3, \mathsf{ls}^4)$ for x. If $b = 0$ then t has been computed by using the honest prover procedure $\mathcal{P}$, otherwise has been computed using the adaptive-input Special HVZK simulator. The adversary, upon receiving t, either outputs his guess $b' \in \{0, 1\}$, or asks to receive another transcript t according to a new possibly challenge $(\mathsf{ls}^{1'}, \mathsf{ls}^{3\prime}, x')$. Note that the adversary can ask a polynomial number of transcripts according to different challenges before he outputs b'. The adversary is successful if $\text{Prob}\left[\, b = b' \,\right] - 1/2$ is non-negligible in the security parameter. It is easy to see that a protocol is adaptive-input Special HVZK iff it is multi-SHVZK.

The only differences with the classical definition of hiding is the following. Let m_0 and m_1 be the challenge messages. The challenger of multi-hiding picks a random bit b and compute the commitment of m_b. The adversary, upon receiving the commitment, either outputs his guess $b' \in \{0, 1\}$, or asks to receive another commitment of m_b (the latter step can be executed a polynomial number of times). The adversary is successful if $\text{Prob}\left[\, b = b' \,\right] - 1/2$ is non-negligible in the security parameter. It is easy to see that a commitment scheme is hiding iff it is multi-hiding.

We now reconsider the hybrid experiments $\mathcal{H}_k$ for $k = 0, \dots, 4$ described in the security proof of Lemma 4.2.1, and we want to show that in each of these hybrid experiments in every right sessions $\mathcal{A}_{\mathsf{NMZK}}$ does not cheat. In other words we want to ensure, that for every i-th accepting right session of $\mathcal{H}_k(1^\lambda, z)$ $\mathcal{A}_{\mathsf{NMZK}}$ chooses a statement $\tilde{x}_i$ and completes the corresponding transcript of Π^{OR} using the witness $\tilde{w}_i$ s.t. $(\tilde{x}_i, \tilde{w}_i) \in \mathsf{Rel}_{\mathsf{L}}$, and $\mathtt{id} \neq \tilde{\mathtt{id}}_i$, for all $i = 1, \dots, \mathsf{poly}(\lambda)$ and all $k = 0, \dots, 4$. In more details we prove that the following claim holds in all the hybrid experiments.

Claim 4.2.3. *Let $\tilde{x}_1, \dots, \tilde{x}_{\mathsf{poly}(\lambda)}$ be the right-session statements appearing in view of $\mathcal{A}_{\mathsf{NMZK}}$ and let $\mathtt{id}$ be the identity of the left session and $\tilde{\mathtt{id}}_1, \dots, \tilde{\mathtt{id}}_{\mathsf{poly}(\lambda)}$ be the identities of right sessions ap-*

pearing in this view. If the i-th right session is accepting and $\mathtt{id} \neq \tilde{\mathtt{id}}_i$ *for* $i = 1, \ldots, \mathsf{poly}(\lambda)$*, then except with negligible probability,* $\mathcal{A}_{\mathsf{NMZK}}$ *computes the transcript of* Π^{OR} *using a witness* w_i *s.t.* $(\tilde{x}_i, \tilde{w}_i) \in \mathsf{Rel}_L$.

We start by reconsidering the hybrid experiments $\mathcal{H}_0(1^\lambda, z)$ and demonstrate that the following claim holds in $\mathcal{H}_0(1^\lambda, z)$.

Claim 4.2.4. *The probability that* $\mathcal{A}_{\mathsf{NMZK}}$ *commits using* NM *to a message* $\tilde{s}_0$ *and sends a string* $\tilde{s}_1$ *s.t.* $\tilde{s}_0 \oplus \tilde{s}_1 = \tilde{\sigma}_1 || \tilde{\sigma}_2$ *and* $\tilde{\sigma}_1, \tilde{\sigma}_2$ *are two signatures for two different messages w.r.t. the verification key* $\tilde{\mathsf{vk}}$ *sent in the first round of the i-th right session for some* $i \in \{1, \ldots, \mathsf{poly}(\lambda)\}$ *is negligible.*

Suppose by contradiction $\mathcal{A}_{\mathsf{NMZK}}$ commits to the signatures in the i-th right sessions the that we can construct an adversary $\mathcal{A}_\Sigma$ that breaks the security of the signature scheme Σ, for some $i \in \{1, \ldots, \mathsf{poly}(\lambda)\}$. Let $\tilde{\mathsf{vk}}$ be the challenge verification key. The adversary $\mathcal{A}_\Sigma$ interacts against the MiM adversary $\mathcal{A}_{\mathsf{NMZK}}$ in the left session as a honest prover does. In the rights sessions he acts as a honest verifier does except for a i-th right session, for which he acts in the following way. In the i-th right session $\mathcal{A}_{\mathsf{NMZK}}$ uses $\tilde{\mathsf{vk}}$ to compute the first round and the oracle $\mathsf{Sign}(\tilde{\mathsf{sk}}, \cdot)$ to compute a signature $\tilde{\sigma}_1$ of a message $\tilde{\mathtt{msg}}_1$ sent by $\mathcal{A}_{\mathsf{NMZK}}$ in the second round. At the end of the execution $\mathcal{A}_\Sigma$ extracts from the commitment $\tilde{\tau} = (\tilde{\mathtt{id}}, \tilde{\mathsf{nm}}_1, \tilde{\mathsf{nm}}_2, \tilde{\mathsf{nm}}_3, \tilde{\mathsf{nm}}_4)$ computed using NM two signatures $\tilde{\sigma}_1, \tilde{\sigma}_2$ for two different messages $\tilde{\mathtt{msg}}_1, \tilde{\mathtt{msg}}_2$ w.r.t. $\tilde{\mathsf{vk}}$. $\mathcal{A}_{\mathsf{NMZK}}$ outputs $\tilde{\sigma}_2, \tilde{\mathtt{msg}}_2$. Observe that the extraction succeeds with non-negligible probability, because by contradiction we are assuming that $\mathcal{A}_{\mathsf{NMZK}}$ commits (correctly) to two signatures in $\tilde{\tau}$. The proof ends with the observation that $\mathsf{Sign}(\tilde{\mathsf{sk}}, \cdot)$ is called only once.

Note that from Claim 4.2.4 follows that $\mathcal{A}_{\mathsf{NMZK}}$ does not commit to two signatures (except with negligible probability) which implies that in $\mathcal{H}_0(1^\lambda, z)$ the transcript computed using $\mathsf{LS}_{\mathsf{nm}}$ corresponds to a false instance. Since Π_{OR} enjoys adaptive-input PoK property we can argue that $\mathcal{A}_{\mathsf{NMZK}}$ in the i-th right sessions chooses a true statement $\tilde{x}_i$ and computes the transcript of LS_L (therefore the transcript of Π^{OR}) using a witness w_i s.t. $(\tilde{x}_i, \tilde{w}_i) \in \mathsf{Rel}_L$, for all $i = 1, \dots, \mathsf{poly}(\lambda)$. For the above chain of implications we can conclude that also the Claim 4.2.3 holds in $\mathcal{H}_0(1^\lambda, z)$.

The next hybrid that we reconsider is $\mathcal{H}_1(1^\lambda, z)$. It is already showed in the proof of Lemma 4.2.1, that the view of $\mathcal{A}_{\mathsf{NMZK}}$ in $\mathcal{H}_0(1^\lambda, z)$ is statistically close to the view of $\mathcal{A}_{\mathsf{NMZK}}$ in $\mathcal{H}_1(1^\lambda, z)$, this implies that the Claim 4.2.3 and Claim 4.2.4 still hold in in $\mathcal{H}_1(1^\lambda, z)$.

The next hybrid that we reconsider is $\mathcal{H}_2(1^\lambda, z)$. To prove that the Claim 4.2.4 still holds in $\mathcal{H}_2(1^\lambda, z)$ we use two different properties of NM. We cannot rely only on the non-malleability of NM because this property holds only against a synchronous MiM. Therefore for the asynchronous case we need relay on the multi-hiding of NM.

We demonstrate this claim arguing separately that a) a synchronous $\mathcal{A}_{\mathsf{NMZK}}$, except with negligible probability, does not commit to the pair of signatures in any of the synchronous right sessions; b) an asynchronous $\mathcal{A}_{\mathsf{NMZK}}$ does not commit to the pair of signatures in any of the asynchronous right sessions. In more details.

(a) Suppose by contradiction that the right session where the synchronous $\mathcal{A}_{\mathsf{NMZK}}$ commits to the signatures with non-negligible probability in the i-th right session (with $i \in \{1, \dots, \mathsf{poly}(\lambda)\}$). This means that when $\mathcal{P}_{\mathsf{NMZK}}$ commits to the signatures in the left session $\mathcal{A}_{\mathsf{NMZK}}$ starts to commit to the signatures with non-negligible probability in at least one synchronous right sessions. Based on this observation we can construct a distinguisher $\mathcal{D}_{\mathsf{nm}}$ and an adversary $\mathcal{A}_{\mathsf{nm}}$ that breaks the synchronous non-malleability of NM. Let $\mathcal{C}_{\mathsf{nm}}$ be the challenger of NM and let (m_0, m_1) be the two random challenge messages.

In the left session $\mathcal{A}_{\mathsf{nm}}$ acts as $\mathcal{P}_{\mathsf{NMZK}}$ does with $\mathcal{A}_{\mathsf{NMZK}}$ according to both $\mathcal{H}_2(1^\lambda, z)$ and $\mathcal{H}_1(1^\lambda, z)$ with the following differences: 1) $\mathcal{A}_{\mathsf{nm}}$ plays as proxy between $\mathcal{C}_{\mathsf{nm}}$ and $\mathcal{A}_{\mathsf{NMZK}}$ w.r.t. messages of NM; 2) two signatures σ_1, σ_2 are extracted from the left session through rewinds; 3) during rewinds of the left a random third round $\tilde{\mathsf{nm}}_3$ is played to simulate the receiver of NM in the right sessions; 4) $\mathcal{A}_{\mathsf{nm}}$ in the last round of the left session sends s_1 s.t. $s_1 = m_0 \oplus \sigma_1||\sigma_2$.

In the right sessions $\mathcal{A}_{\mathsf{NMZK}}$ acts as $\mathcal{V}_{\mathsf{NMZK}}$ does according to both $\mathcal{H}_2(1^\lambda, z)$ and $\mathcal{H}_1(1^\lambda, z)$ except for the i-th right session. In this $\mathcal{A}_{\mathsf{NMZK}}$ acts as $\mathcal{V}_{\mathsf{NMZK}}$ does except for the messages of NM for which he acts as a proxy between $\mathcal{C}_{\mathsf{nm}}$ and $\mathcal{A}_{\mathsf{NMZK}}$. Then $\mathcal{D}_{\mathsf{nm}}$, on input the message $\tilde{m}$ committed in the i-th right session by $\mathcal{A}_{\mathsf{nm}}$ and his randomness, reconstructs the view of $\mathcal{A}_{\mathsf{NMZK}}$ and recovers the messages $\tilde{s}_1$ sent by $\mathcal{A}_{\mathsf{NMZK}}$ in the last round of the i-th right session. If $\tilde{s}_1 \oplus \tilde{m} = \sigma_1||\sigma_2$ then $\mathcal{D}_{\mathsf{nm}}$ outputs a random bit, and 0 otherwise. Since by contradiction $\mathcal{A}_{\mathsf{NMZK}}$ commits to the signatures with overwhelming probability in at least one right session only when $\mathcal{P}_{\mathsf{NMZK}}$ commits to the signatures, then $\mathcal{D}_{\mathsf{nm}}$ can tell apart which message has been committed by the MiM adversary $\mathcal{A}_{\mathsf{nm}}$. We notice that the reduction in the i-th right session queries the receiver of NM involved in the reduction only once. This because during the extraction of the signatures, from the left session, all the messages $\tilde{\mathsf{nm}}_3$ can be simulated by the reduction due to the public-coin property of NM.

(b) Suppose by contradiction that the right session where the asynchronous $\mathcal{A}_{\mathsf{NMZK}}$ commits with non-negligible probability to the signatures is the i-th right session (with $i \in \{1, \dots, \mathsf{poly}(\lambda)\}$), then we construct an adversary $\mathcal{A}_{\mathsf{Hiding}}$ that break the multi-hiding of NM. Let $\mathcal{C}_{\mathsf{nm}}$ be the challenger of NM. The adversary $\mathcal{A}_{\mathsf{Hiding}}$ that we construct interacts with $\mathcal{A}_{\mathsf{NMZK}}$ in the left and the right sessions according to both $\mathcal{H}_2(1^\lambda, z)$ and $\mathcal{H}_1(1^\lambda, z)$ for all messages except for the messages of NM. For these messages $\mathcal{A}_{\mathsf{Hiding}}$ acts as a proxy between $\mathcal{A}_{\mathsf{NMZK}}$ and the challenger $\mathcal{C}_{\mathsf{Hiding}}$ in the left session. More formally, against the challenger of multi-hiding $\mathcal{C}_{\mathsf{Hiding}}$, $\mathcal{A}_{\mathsf{Hiding}}$ works as following.

1. Upon receiving the 1st round from $\mathcal{A}_{\mathsf{NMZK}}$, $\mathcal{A}_{\mathsf{Hiding}}$ sends

two random messages m_0, m_1 as the challenge message together with nm_1 received from $\mathcal{A}_{\mathsf{NMZK}}$ to $\mathcal{C}_{\mathsf{Hiding}}$.

2. Upon receiving nm_2 from $\mathcal{C}_{\mathsf{Hiding}}$, $\mathcal{A}_{\mathsf{Hiding}}$ uses it to compute and send the 2nd round of NMZK to $\mathcal{A}_{\mathsf{NMZK}}$ on the left.

3. Upon receiving the 3rd round from $\mathcal{A}_{\mathsf{NMZK}}$, $\mathcal{A}_{\mathsf{NMZK}}$ extracts two valid signatures σ_1, σ_2 for two different messages from the left session and sends nm_3 received from $\mathcal{A}_{\mathsf{NMZK}}$ to $\mathcal{C}_{\mathsf{Hiding}}$.

4. Upon receiving nm_4 from $\mathcal{C}_{\mathsf{Hiding}}$, $\mathcal{A}_{\mathsf{Hiding}}$ uses nm_4 to complete the left session against $\mathcal{A}_{\mathsf{NMZK}}$ sending s_1 s.t. $s_1 = m_0 \oplus \sigma_1 || \sigma_2$.

5. Consider the i-th right session. If $\mathcal{A}_{\mathsf{Hiding}}$ extracts from the commitment $\tilde{\tau} = (\tilde{\mathtt{id}}, \tilde{\mathsf{nm}}_1, \tilde{\mathsf{nm}}_2, \tilde{\mathsf{nm}}_3, \tilde{\mathsf{nm}}_4)$ two signatures $\tilde{\sigma}_1, \tilde{\sigma}_2$ for two different messages then he outputs 1, otherwise he outputs a random bit.

It easy to see that if $\mathcal{C}_{\mathsf{Hiding}}$ commits to m_1 then, $\mathcal{A}_{\mathsf{ZK}}$ acts as in $\mathcal{H}_1(1^\lambda, z)$, otherwise $\mathcal{A}_{\mathsf{ZK}}$ acts as in $\mathcal{H}_2(1^\lambda, z)$.

Observe that when $\mathcal{A}_{\mathsf{Hiding}}$ rewinds the right sessions it could happen that also the left session is rewound. This does not cause any problem, because if $\mathcal{A}_{\mathsf{Hding}}$ has to play again the second round of the left session he starts a new interaction against the challenger of multi-hiding executing all steps described above starting from step 1. We also observe that all the sessions where the extraction on the right rewinds $\mathcal{C}_{\mathsf{Hiding}}$ are actually synchronized. Therefore, in that case we can rely on the non-malleability of NM (following the part (a) of the proof). Finally note that in $\mathcal{H}_2(1^\lambda, z)$ $\mathcal{A}_{\mathsf{SHVZK}}$ extracts two signatures from the left session with non-negligible probability, for the same arguments provided in the proof of Lemma 4.2.1.

Note that the Claim 4.2.4 still holds in $\mathcal{H}_2(1^\lambda, z)$. Therefore for the same chain of implications described in $\mathcal{H}_0(1^\lambda, z)$ we can conclude that also in $\mathcal{H}_2(1^\lambda, z)$ $\mathcal{A}_{\mathsf{NMZK}}$ in the i-th accepting right session chooses a statement $\tilde{x}_i$ and computes the transcript of Π^{OR} using a witness w_i s.t. $(\tilde{x}_i, \tilde{w}_i) \in \mathsf{Rel}_L$, for all $i = 1, \dots, \mathsf{poly}(\lambda)$, e.g. the Claim 4.2.3 holds also $\mathcal{H}_2(1^\lambda, z)$.

The next hybrid that we reconsider is $\mathcal{H}_3(1^\lambda, z)$. We show that Claim 4.2.3 holds in $\mathcal{H}_3(1^\lambda, z)$, otherwise we can break the multi-SHVZK of $\mathsf{LS}_{\mathsf{nm}}$. Roughly speaking, we demonstrate that $\mathcal{A}_{\mathsf{NMZK}}$ in $\mathcal{H}_2(1^\lambda, z)$ does not commit to the signatures (expect with negligible probability) in all the right sessions. Therefore the only witnesses that is possible to extract (using the extractor of Π^{OR}), in the i-th accepting right session, from the transcript of Π^{OR} is a w_i s.t. $(\tilde{x}_i, \tilde{w}_i) \in \mathsf{Rel}_L$, for all $i = 1, \dots, \mathsf{poly}(\lambda)$. In $\mathcal{H}_3(1^\lambda, z)$ we can check which witnesses is extracted from the transcript of Π^{OR} in the right sessions, and made a reduction to multi-SHVZK of $\mathsf{LS}_{\mathsf{nm}}$ if we do not extracted the witnesses for the statements $\tilde{x}_1, \dots, \tilde{x}_{\mathsf{poly}(\lambda)}$. More details follow.

Suppose by contradiction that the Claim 4.2.3 does not hold, then we can construct an adversary $\mathcal{A}_{\mathsf{SHVZK}}$ against the multi-SHVZK of $\mathsf{LS}_{\mathsf{nm}}$. Let $\mathcal{C}_{\mathsf{SHVZK}}$ be the challenger for the security game of multi-SHVZK. $\mathcal{A}_{\mathsf{SHVZK}}$ works as following.

1. $\mathcal{A}_{\mathsf{SHVZK}}$ interacts with $\mathcal{A}_{\mathsf{NMZK}}$ in order to receive the first round and sends $\mathsf{ls}^1_{\mathsf{nm}}$ to $\mathcal{C}_{\mathsf{SHVZK}}$.

2. Upon receiving $\mathsf{ls}^2_{\mathsf{nm}}$ from $\mathcal{C}_{\mathsf{SHVZK}}$ uses it to compute and send to $\mathcal{A}_{\mathsf{NMZK}}$ the second round according to both $\mathcal{H}_3(1^\lambda, z)$ and $\mathcal{H}_2(1^\lambda, z)$.

3. Upon receiving the third round from $\mathcal{A}_{\mathsf{NMZK}}$, $\mathcal{A}_{\mathsf{SHVZK}}$ extracts the signatures σ_1, σ_2 from the left session and computes the fourth round nm_4 of NM and sets $s_1 = \sigma_1 || \sigma_2 \oplus s_0, x_{\mathsf{nm}} = (\mathsf{vk}, \mathtt{id}, \mathsf{nm}_1, \mathsf{nm}_2, \mathsf{nm}_3, \mathsf{nm}_4, s_1)$,
$w_{\mathsf{nm}} = (\mathtt{dec}_{\mathsf{nm}}, s_0, \sigma_1, \mathtt{msg}_1, \sigma_2, \mathtt{msg}_2)$. He sends to the challenger of the SHVZK $\mathcal{C}_{\mathsf{SHVZK}}$ the statement x_{nm} the witness w_{nm} and the round $\mathsf{ls}^3_{\mathsf{nm}}$ received from $\mathcal{A}_{\mathsf{NMZK}}$.

4. Upon receiving $\mathsf{ls}^4_{\mathsf{nm}}$ from $\mathcal{C}_{\mathsf{SHVZK}}$ he uses it to compute the last round of NMZK.

5. Consider the i-th right session. $\mathcal{A}_{\mathsf{SHVZK}}$ using the extractor of Π^{OR} (that exists for the adaptive-PoK property enjoyed by Π^{OR}) extracts from the transcript of Π^{OR} a witness $\tilde{w}_{\mathsf{OR},i}$ for the statement ($\tilde{x}_i$ OR $\tilde{x}_{\mathsf{nm},i}$). Then if $(\tilde{x}_i, \tilde{w}_{\mathsf{OR},i}) \in \mathsf{Rel}_L$ $\mathcal{A}_{\mathsf{SHVZK}}$ outputs a random bit, otherwise he outputs 1.

It easy to see that if $\mathcal{C}_{\mathsf{SHVZK}}$ sends $\mathsf{ls}^2_{\mathsf{nm}}, \mathsf{ls}^4_{\mathsf{nm}}$ that are computed using the honest prover procedure of $\mathsf{LS}_{\mathsf{nm}}$ then, $\mathcal{A}_{\mathsf{ZK}}$ acts as in $\mathcal{H}_3(1^\lambda, z)$, otherwise he acts as in $\mathcal{H}_2(1^\lambda, z)$. Observe that when $\mathcal{A}_{\mathsf{SHVZK}}$ rewinds the right session it could happen that also the left session is rewound. This does not cause any problem because $\mathcal{A}_{\mathsf{SHVZK}}$ can keep fixed $\mathsf{ls}^3_{\mathsf{nm}}$ during the rewinds in order to complete an accepting transcript for Π_{OR} even tough different third rounds of Π_{OR} are sent during the rewinds by $\mathcal{A}_{\mathsf{NMZK}}$. More precisely when multiple third rounds $\mathsf{c}^1, \mathsf{c}^2, \dots, \mathsf{c}^{\mathsf{poly}(\lambda)}$ are received, $\mathcal{A}_{\mathsf{SHVZK}}$ just computes $\mathsf{ls}^{3\prime}_L = \mathsf{ls}^3_{\mathsf{nm}} \oplus \mathsf{c}^i$ for $i = 1, \dots, \mathsf{poly}(\lambda)$ and runs the honest prover procedure $\mathcal{P}_L$ on input statement x, the witness w and the challenge $\mathsf{ls}^{3\prime}_L$ thus obtaining $\mathsf{ls}^{4\prime}_L$. In this way $\mathcal{A}_{\mathsf{SHVZK}}$ can complete the execution against $\mathcal{A}_{\mathsf{NMZK}}$ by sending in the fourth round $(\mathsf{ls}^{4\prime}_L, \mathsf{nm}_4, s_1, \mathsf{ls}^4_{\mathsf{nm}}, x, x_{\mathsf{nm}})$ without rewinding $\mathcal{C}_{\mathsf{SHVZK}}$. We observe that a rewind made on the right could rewind the entire left session. In this case the challenger needs to be invoked multiple times in order to receive multiple transcripts w.r.t. Π_L. The multi-SHVZK allow to do such interaction against $\mathcal{C}_{\mathsf{SHVZK}}$. Finally note that in $\mathcal{H}_3(1^\lambda, z)$ $\mathcal{A}_{\mathsf{SHVZK}}$ extracts two signatures from the left session with non-negligible probability, for the same arguments provided in the proof of Lemma 4.2.1.

The next hybrid that we reconsider is $\mathcal{H}_4(1^\lambda, z)$. Also, in this hybrid we show that Claim 4.2.3 still holds, otherwise we can break the multi-SHVZK of LS_L. The security proof is almost equal to the security proof showed to demonstrate that the Claim 4.2.3 holds in $\mathcal{H}_4(1^\lambda, z)$.

Note that $\mathcal{H}_4(1^\lambda, z)$ corresponds to the simulated experiment, that is the experiment where $\mathsf{Sim}_{\mathsf{ZK}}$ interacts with the adversary $\mathcal{A}_{\mathsf{NMZK}}$ emulating both a prover in the left session and polynomially many verifiers in the right sessions. Furthermore the Claim 4.2.3 holds in $\mathcal{H}_4(1^\lambda, z)$, which means that we are ensured (with high probability) that in the simulated experiment $\mathcal{A}_{\mathsf{NMZK}}$ uses the witnesses to complete the transcripts of LS_L in all the right sessions. Therefore the $\mathcal{A}_{\mathsf{NMZK}}$ behavior allows $\mathsf{Sim}_{\mathsf{NMZK}}$ to extract the witnesses used by $\mathcal{A}_{\mathsf{NMZK}}$ (that is internally executed by $\mathsf{Sim}_{\mathsf{NMZK}}$) using the extractor of Π^{OR} (that exists for the adaptive-PoK property enjoyed by Π^{OR}).

This observations conclude the proof.

Theorem 29. If OWFs exists, then NMZK is a delayed-input synchronous many-many NMZK AoK for $\mathcal{NP}$.

Proof. The proof proceeds very similarly to the one showed for Theorem 28. The main difference between these two proofs is that we now have to consider also polynomially many synchronous left sessions played in parallel. Therefore the only difference between this proof and the one of Theorem 28 is that in the reductions we need to rely on the security of a many-one non-malleable commitment scheme and on the adaptive-input SHVZK that is closed under parallel composition. Therefore, when we make a reduction on the adaptive-input SHVZK, we can simply use the parallel version of the primitives. Regarding a many-one non-malleable commitment, we notice that using the same arguments of the security proof of Proposition 1 provided in [LPV08], it is possible to claim that a synchronous (one-one) non-malleable commitment is also synchronous many-one non-malleable. Therefore no additional assumptions are required in order to prove that NMZK is also delayed-input synchronous many-many NMZK. Note also that, the simulator needs to extract the trapdoor (the signatures of two different messages) in all the left (synchronous) sessions completed in the main thread. We can show that the extraction succeeds except with negligible probability using the same arguments used in the security proof of Theorem 28.

$\square$

4.3 Multi-Party Coin-Tossing Protocol

4.3.1 4-Round Secure Multi-Party Coin Tossing: Π_{MPCT}

The high-level idea of our protocol Π_{MPCT} significantly differs from the one of [GMPP16] (e.g., we use our 4-round delayed-input synchronous many-many NMZK instead of 3-round 3-robust parallel non-malleable commitment scheme). Similarly to [GMPP16] our protocol simply consists of each party committing to a random string r, which is opened in the last round along with a simulatable proof of correct opening given to all parties independently. The output consists of the $\oplus$ of all opened strings. Let's see in more details how our Π_{MPCT} works. For our construction we use the following tools.

1. A non-interactive perfectly binding computationally hiding commitment scheme $\mathsf{PBCOM} = (\mathsf{Com}, \mathsf{Dec})$.
2. A Σ-protocol $\mathsf{BL}_L = (\mathcal{P}_L, \mathcal{V}_L)$ for the $\mathcal{NP}$-language $L = \{\mathtt{com} : \exists\ (\mathtt{dec}, m) \text{ s.t. } \mathsf{Dec}(\mathtt{com}, \mathtt{dec}, m) = 1\}$ with Special HVZK simulator Sim_L. We uses two instantiations of BL_L in order to construct the protocol for the OR of two statements Π_{OR} as described earlier. Π_{OR} is a proof system for the $\mathcal{NP}$-language $L_{\mathtt{com}} = \{(\mathtt{com}_0, \mathtt{com}_1) : \exists\ (\mathtt{dec}, m) \text{ s.t. } \mathsf{Dec}(\mathtt{com}_0, \mathtt{dec}, m) = 1 \text{ OR } \mathsf{Dec}(\mathtt{com}_1, \mathtt{dec}, m) = 1\}$ [16]. Informally, by running Π_{OR}, one can prove the knowledge of the message committed in $\mathtt{com}_0$ or in $\mathtt{com}_1$.
3. A 4-round delayed-input synchronous many-many NMZK $\mathsf{NMZK} = (\mathcal{P}_{\mathsf{NMZK}}, \mathcal{V}_{\mathsf{NMZK}})$ for the following $\mathcal{NP}$-language

$$L_{\mathsf{NMZK}} = \{((\mathtt{com}_0, \mathtt{com}_1), m) : \ \forall i \in \{0,1\}\ \exists\ \mathtt{dec}_i \\ \text{s.t. } \mathsf{Dec}(\mathtt{com}_i, \mathtt{dec}_i, m) = 1\}.$$

   Informally, by running NMZK, one can prove that 2 commitments contain the same message m.

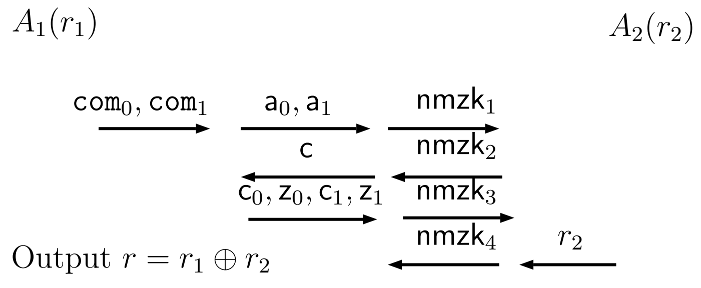
[16]We use Π_{OR} in a non-black box way, but for ease of exposition sometimes we will refer to entire protocol Π_{OR} in order to invoke the proof of knowledge property enjoyed by Π_{OR}.

4.3.2 Π_{MPCT}: Informal Description and Security Intuition

The high level description of our protocol between just two parties (A_1, A_2) is given in Fig. 4.3. For a formal description of Π_{MPCT} we refer the reader to Sec. 4.3.3. In Fig. 4.3 we consider an execution of Π_{MPCT} that goes from A_1 to A_2 (the execution from A_2 to A_1 is symmetric). We recall that the protocol is executed simultaneously by both A_1 and A_2. The main idea is the following. Each party commits to his input using two instantiations of a non-interactive commitment. More precisely we have that A_1 computes two non-interactive commitments $\mathtt{com}_0$ and $\mathtt{com}_1$ (along with their decommitment information $\mathtt{dec}_0$ and $\mathtt{dec}_1$) of the message r_1. Each party also runs Π_{OR} for the $\mathcal{NP}$-language $L_{\mathtt{com}}$, from the first to the third round, in order to prove knowledge of the message committed in $\mathtt{com}_0$ or in $\mathtt{com}_1$. In the last round each party sends his own input (i.e. r_1 for A_1 and r_2 for A_2) and proves, using a delayed-input synchronous many-many non-malleable ZK for the $\mathcal{NP}$-language L_{NMZK}, that messages committed using PBCOM were actually equal to that input (i.e. r_1 for A_1 and r_2 for A_2). That is, A_1 sends r_1 and proves that $\mathtt{com}_0$ and $\mathtt{com}_1$ are valid commitments of the message r_1.

Intuition about the security of Π_{MPCT}. Let A_1^* be the corrupted party.

Informally the simulator Sim works as follows. Sim starts an interaction against A_1^* using as input a random string y until the third round of Π_{MPCT} is received by A_1^*. More precisely, in the first round he computes two commitments $\mathtt{com}_0$ and $\mathtt{com}_1$ (along with their decommitment information $\mathtt{dec}_0$ and $\mathtt{dec}_1$) of y, and runs $\mathcal{P}_{\mathsf{OR}}$ using as a witness $(\mathtt{dec}_1, y)$. After the 3rd round Sim extracts the input r_1^* of the corrupted party A_1^* using the extractor E_{OR} of Π_{OR} (that exists from the PoK property of Π_{OR}) and sends r_1^* to the ideal world functionality. At this point Sim receives r from the ideal-world functionality, and completes the execution of the 4th round by sending $r_2 = r \oplus r_1^*$. We observe that Sim, in order to send a string r_2 that differs from y in the 4th round, has to


$A_1(r_1)$
$A_2(r_2)$
$\mathtt{com}_0, \mathtt{com}_1$
$\mathsf{a}_0, \mathsf{a}_1$
nmzk_1
c
nmzk_2
$\mathsf{c}_0, \mathsf{z}_0, \mathsf{c}_1, \mathsf{z}_1$
nmzk_3
nmzk_4
r_2
Output $r = r_1 \oplus r_2$

- $\mathtt{com}_0$ and $\mathtt{com}_1$ are two non-interactive commitments of the message r_1 computed using PBCOM.
- $(\mathsf{a}_0, \mathsf{a}_1, \mathsf{c}_0, \mathsf{c}_1, \mathsf{z}_0, \mathsf{z}_1)$ is the transcript generated from an execution of the WIPoK Π_{OR} in which $\mathcal{P}_{\mathsf{OR}}$ proves the knowledge of either the message committed in $\mathtt{com}_0$ or in $\mathtt{com}_1$.
- $(\mathsf{nmzk}_1, \mathsf{nmzk}_2, \mathsf{nmzk}_3, \mathsf{nmzk}_4)$ in the transcript generated from an execution of the delayed-input synchronous many-many NMZK NMZK in which $\mathcal{P}_{\mathsf{NMZK}}$ proves that both $\mathtt{com}_0$ and $\mathtt{com}_1$ are valid commitments of the message r_1.

Figure 4.3 Π_{MPCT}: Informal description of the execution from A_1 to A_2. The execution from A_2 to A_1 is symmetric.

cheat in NMZK. This is done by simply running the simulator of NMZK. To prove the security of our scheme we will go through a sequence of hybrid experiments in order to show that the output view of the adversary in the real world can be simulated in the ideal world by Sim. The security proof strongly relies on the non-malleable zero knowledge property of NMZK. Indeed the aim of NMZK is to ensure that the adversary does not maul the messages received from Sim. That is, the behavior of A_1^* allows to extract, in every hybrid experiments that we will consider, the correct input of A_1^*. This holds even in case the commitments sent by Sim to A_1^* are commitments of a random string y, and the value sent in the 4th round is inconsistent with the value committed in the first round.

4.3.3 Π_{MPCT}: Formal Description

Let $P = \{P_1, \dots, P_n\}$ be the set of parties. Furthermore, denote by $(\mathtt{id}_1, \dots, \mathtt{id}_n)$[17] the unique identities of parties $\{P_1, \dots, P_n\}$, respectively. Let us denote by $\mathsf{F}_{\mathsf{MPCT}} : (1^\lambda)^n \to \{0,1\}^\lambda$ the function $\mathsf{F}_{\mathsf{MPCT}}(r_1, \dots, r_n) = r_1 \oplus \cdots \oplus r_n$. The protocol starts with each party P_i choosing a random string r_i for $i = 1, \dots, n$. It consists of four rounds, i.e., all parties send messages in each round and the messages of all executions are seen by every party. Following [GMPP16] we describe the protocol between two parties (A_1, A_2) observing that the real protocol actually consists of n simultaneous executions of a two-party coin-tossing protocol $\Pi_{\mathsf{MPCT}} = (A_1, A_2)$ between parties (P_i, P_j) where P_i acts as A_1 with input r_i and P_j acts as A_2 with input r_j (both are symmetric). Let the input of A_1 be r_1, and the input of A_2 be r_2. The set of messages enabling A_1 to learn the output are denoted by (m_1, m_2, m_3, m_4) where (m_1, m_3) are sent by A_1 and (m_2, m_4) are sent by A_2. Likewise, the set of messages enabling A_2 to learn the output are denoted by $(\tilde{m}_1, \tilde{m}_2, \tilde{m}_3, \tilde{m}_4)$ where $(\tilde{m}_1, \tilde{m}_3)$ are sent by A_2 and $(\tilde{m}_2, \tilde{m}_4)$ are sent by A_1. Therefore, messages $(m_l, \tilde{m}_l)$ are simultaneously exchanged in the l-th round for $l = 1, \dots, 4$.

Protocol Π_{MPCT}. *Common input:* security parameter λ, instances length: ℓ_{NMZK}, $\ell_{\mathtt{com}}$.
Round 1. We first describe how A_1 constructs m_1.

1. Compute $(\mathtt{com}_0, \mathtt{dec}_0) \leftarrow \mathsf{Com}(r_1)$ and $(\mathtt{com}_1, \mathtt{dec}_1) \leftarrow \mathsf{Com}(r_1)$.
2. Compute $\mathsf{a}_0 \leftarrow \mathcal{P}_L(1^\lambda, \mathtt{com}_0, (\mathtt{dec}_0, r_1))$.
3. Pick $\mathsf{c}_1 \leftarrow \{0,1\}^\lambda$ and compute $(\mathsf{a}_1, \mathsf{z}_1) \leftarrow \mathsf{Sim}_L(1^\lambda, \mathtt{com}_1, \mathsf{c}_1)$.
4. Run $\mathcal{V}_{\mathsf{NMZK}}$ on input 1^λ and ℓ_{NMZK} thus obtaining the 1st round nmzk_1 of NMZK.
5. Message m_1 is defined to be $(\mathtt{com}_0, \mathtt{com}_1, \mathsf{a}_0, \mathsf{a}_1, \mathsf{nmzk}_1)$.

Likewise, A_2 performs the same action as A_1 in order to construct $\tilde{m}_1 = (\tilde{\mathtt{com}}_0, \tilde{\mathtt{com}}_1, \tilde{\mathsf{a}}_0, \tilde{\mathsf{a}}_1, \widetilde{\mathsf{nmzk}}_1)$.

[17]As discuss in the Definition 1.3.1 the use of the identifiers can be avoid, we use them, to uniformity of notation.

Round 2. In this round A_2 sends message m_2 and A_1 sends $\tilde{m}_2$. We first describe how A_2 constructs m_2.

1. Run $\mathcal{P}_{\mathsf{NMZK}}$ on input 1^λ, $\mathtt{id}_2$, ℓ_{NMZK} and nmzk_1 thus obtaining the 2nd round nmzk_2 of NMZK.
2. Pick $\mathsf{c} \leftarrow \{0,1\}^\lambda$.
3. Define message $m_2 = (\mathsf{c}, \mathsf{nmzk}_2)$.

Likewise, A_1 performs the same actions as A_2 in the previous step to construct the message $\tilde{m}_2 = (\tilde{\mathsf{c}}, \widetilde{\mathsf{nmzk}}_2)$.

Round 3. In this round A_1 sends message m_3 and A_2 sends $\tilde{m}_3$. A_1 prepares m_3 as follows.

1. Compute $\mathsf{c}_0 = \mathsf{c} \oplus \mathsf{c}_1$ and $\mathsf{z}_0 \leftarrow \mathcal{P}_L(\mathsf{c}_0)$.
2. Run $\mathcal{V}_{\mathsf{NMZK}}$ on input nmzk_2 thus obtaining the 3rd round nmzk_3 of NMZK.
3. Define $m_3 = \big(\mathsf{nmzk}_3, \mathsf{c}_0, \mathsf{c}_1, \mathsf{z}_0, \mathsf{z}_1\big)$.

Likewise, A_2 performs the same actions as A_1 in the previous step to construct the message $\tilde{m}_3 = (\widetilde{\mathsf{nmzk}}_3, \tilde{\mathsf{c}}_0, \tilde{\mathsf{c}}_1, \tilde{\mathsf{z}}_0, \tilde{\mathsf{z}}_1)$.

Round 4. In this round A_2 sends message m_4 and A_1 sends $\tilde{m}_4$. A_2 prepares m_4 as follows.

1. Check that the following conditions are satisfied: a) $\mathsf{c} = \mathsf{c}_0 \oplus \mathsf{c}_1$; b) the transcript $\mathsf{a}_0, \mathsf{c}_0, \mathsf{z}_0$ is accepting w.r.t. the instance $\mathtt{com}_0$; c) the transcript $\mathsf{a}_1, \mathsf{c}_1, \mathsf{z}_1$ is accepting w.r.t. the instance $\mathtt{com}_1$. If one of the check fails then output $\perp$, otherwise continue with the following steps.
2. Set $x_{\mathsf{NMZK}} = (\tilde{\mathtt{com}}_0, \tilde{\mathtt{com}}_1, r_2)$ and $w_{\mathsf{NMZK}} = (\tilde{\mathtt{dec}}_0, \tilde{\mathtt{dec}}_1)$.
3. Run $\mathcal{P}_{\mathsf{NMZK}}$ on input nmzk_3, the statement to be proved x_{NMZK} and the witness w_{NMZK} s.t. $(x_{\mathsf{NMZK}}, w_{\mathsf{NMZK}}) \in \mathsf{Rel}_{L_{\mathsf{NMZK}}}$, thus obtaining the 4th round nmzk_4 of NMZK.
4. Define $m_4 = (r_2, x_{\mathsf{NMZK}}, \mathsf{nmzk}_4)$.

Likewise, A_1 performs the same actions as A_2 in the previous step to construct the message $\tilde{m}_4 = (r_1, \tilde{x}_{\mathsf{NMZK}}, \widetilde{\mathsf{nmzk}}_4)$.

Output computation of Π_{MPCT}. Check, for each party, if $(\mathsf{nmzk}^i_1, \mathsf{nmzk}^i_2, \mathsf{nmzk}^i_3, \mathsf{nmzk}^i_4)$ is accepting for $\mathcal{V}_{\mathsf{NMZK}}$ with respect to the instance x^i_{NMZK} $(i = 1, \ldots, n)$ and that all pairs of parties used the same inputs $(r_1, \ldots, r_n)$. If so, output $r = r_1 \oplus \cdots \oplus r_n$.

Theorem 30. If one-to-one OWFs exist, then the multi-party protocol Π_{MPCT} securely computes the multi-party coin-tossing functionality with black-box simulation.

Proof. Let $P = \{P_1, \ldots, P_n\}$ be the set of parties participating in the execution of Π_{MPCT}. Also let $P^* \subseteq P$ be the set of parties corrupted by the adversary $\mathcal{A}$. The simulator Sim only generates messages on behalf of parties $P \setminus P^*$. In particular, we show that for every adversary $\mathcal{A}$ there exists an "ideal" world adversary Sim such that

$$\mathsf{REAL}_{\Pi_{\mathsf{MPCT}},\mathcal{A}(z)}(1^\lambda) \approx \mathsf{IDEAL}_{\mathsf{F}_{\mathsf{MPCT}},\mathsf{Sim}(z)}(1^\lambda).$$

We prove this claim by considering hybrid experiments $\mathcal{H}_1, \ldots, \mathcal{H}_7$ as described below. Without loss of generality we will assume that party P_1 is the only honest party since our protocol is secure against $n-1$ corruptions. We denote the output of the parties in the hybrid experiment $\mathcal{H}_i$ with $\{\mathsf{OUT}_{\mathcal{H}_i,\mathcal{A}(z)}(1^\lambda)\}$.

- The 1st hybrid experiment $\mathcal{H}_1$ is identical to the real execution. More specifically, $\mathcal{H}_1$ starts $\mathcal{A}$ with fresh randomness and interacts with it as P_1 would do using uniform randomness r_1 as input. The output of $\mathcal{H}_1$ consists of $\mathcal{A}$'s view. We observe that, by construction, the output of $\mathcal{A}$ in the real execution is identically distributed to $\mathcal{H}_1$. Moreover, all the messages generated on the behalf of P^* are honestly computed with overwhelming probability due to the soundness of NMZK.
- The 2nd hybrid experiment $\mathcal{H}_2$ is identical to $\mathcal{H}_1$ except that this hybrid experiment also extracts the P^*'s inputs $r_2^*, \ldots, r_n^*$. In order to obtain $r_2^*, \ldots, r_n^*$, $\mathcal{H}_2$ runs the extractor E_{OR} of Π_{OR} on each execution of Π_{OR} made by a malicious party. Note that the existence of E_{OR} is guaranteed from the adaptive-input PoK property of Π_{OR}. If the extractor fails, then $\mathcal{H}_2$ aborts. At this point $\mathcal{H}_2$ completes the 4th round and prepares the output exactly as $\mathcal{H}_1$[18].

[18]Also in this case we are considering an adversary that completes the execution of Π_{MPCT} against Sim with non-negligible probability. In the case that the abort probability of the adversary is overwhelming then the security proof is already over.

$\{\mathsf{OUT}_{\mathcal{H}_1,\mathcal{A}(z)}(1^\lambda)\}$ and $\{\mathsf{OUT}_{\mathcal{H}_2,\mathcal{A}(z)}(1^\lambda)\}$ are statistically close, and the extraction is successful in expected polynomial time, both claims follow from the adaptive-input PoK property of Π_{OR}. Observe that we are guaranteed that what E_{OR} outputs correspond to the input of the the malicious party, from the fact that with non-negligible probability $\mathcal{A}$ correctly computes all the steps of Π_{MPCT}. More precisely the soundness of NMZK ensures that the extracted values correspond to the $r_2^*, \ldots, r_n^*$ received in the last round.

- The 3rd hybrid experiment $\mathcal{H}_3$ differs from $\mathcal{H}_2$ in the way the transcript for the delayed-input synchronous many-many NMZK NMZK is computed. More precisely in this hybrid experiment the simulator $\mathsf{Sim}^{\mathsf{NMZK}}$ for NMZK is used. Following [GMPP16, ACJ17] the extraction of NMZK's trapdoor and the extraction of P^*'s input are performed during the same steps. Observe that these two extraction procedures do not interfere with each other, indeed they just rewind from the third to the second round by sending a freshly generated second round.

  The first property of $\mathsf{Sim}^{\mathsf{NMZK}}$ (see Definition 1.3.1) ensures that $\{\mathsf{OUT}_{\mathcal{H}_2,\mathcal{A}(z)}(1^\lambda)\}$ is computationally indistinguishable from $\{\mathsf{OUT}_{\mathcal{H}_3,\mathcal{A}(z)}(1^\lambda)\}$. Moreover the second property enjoyed by $\mathsf{Sim}^{\mathsf{NMZK}}$ (simulation-extraction) ensures that in $\mathcal{H}_3$ the witnesses can be extracted from $\mathcal{A}$ (one witness for every execution of NMZK made by every malicious P_i^*), therefore we are guaranteed that $\mathcal{A}$ correctly computes all the steps of Π_{MPCT}. That is, the value $r_2^*, \ldots, r_n^*$ sent by the malicious party in the last round are actually committed in the second round sent by $\mathcal{A}$. It is important to observe that in this hybrid experiment the probability that $\mathcal{A}$ completes the third round is negligible close to the probability of completing the third round in $\mathcal{H}_2$ (otherwise the output of the two experiments would be distinguishable). Therefore the probability that E_{OR} works correctly in this experiment is negligibly close to the probability that E_{OR} works in $\mathcal{H}_2$. This holds because, following the Definition 1.1.5, the probability of E_{OR} to given in output a valid witness for the instance $(\mathtt{com}_0, \mathtt{com}_1)$ is negligible close to the probability that $\mathcal{A}$ completes an accepting third round.

- The 4th hybrid experiment $\mathcal{H}_4$ differs from $\mathcal{H}_3$ in the way $\mathtt{com}_1$ is computed. More precisely, instead of committing to r_1 in $\mathtt{com}_1$ a commitment of a random string y is made. We claim that $\{\mathsf{OUT}_{\mathcal{H}_3,\mathcal{A}(z)}(1^\lambda)\}$ and $\{\mathsf{OUT}_{\mathcal{H}_4,\mathcal{A}(z)}(1^\lambda)\}$ are computationally indistinguishable due to the computationally hiding of PBCOM. We claim also that in $\mathcal{H}_4$ $\mathcal{A}$ still behaves correctly, indeed we can use the simulator extractor $\mathsf{Sim}^{\mathsf{NMZK}}$ in order to check whether the theorem proved by every party controlled by $\mathcal{A}$ using NMZK are still true. If it is not the case, then we can make a reduction to the hiding of $\mathtt{com}_1$[19].
- The 5th hybrid experiment $\mathcal{H}_5$ follows the same steps of $\mathcal{H}_4$ except that the honest prover procedure $(\mathcal{P}_L)$, instead of the Special HVZK simulator (Sim_L), is used to compute the prover's messages $\mathsf{a}_1, \mathsf{z}_1$ of the transcript $\tau_1 = (\mathsf{a}_1, \mathsf{c}_1, \mathsf{z}_1)$ w.r.t. the instance $\mathtt{com}_1$.

  Suppose now by contradiction that the output distributions of the hybrid experiments are distinguishable, then we can show a malicious verifier $\mathcal{V}^\star$ that distinguishes between a transcript $\tau_1 = (\mathsf{a}_1, \mathsf{c}_1, \mathsf{z}_1)$ computed using Sim_L and one computed using the honest prover procedure. In more details, let $\mathcal{C}_{\mathsf{SHVZK}}$ be the challenger of the Special HVZK. $\mathcal{V}^\star$ picks $\mathsf{c}_1 \leftarrow \{0,1\}^\lambda$ and sends c_1 to $\mathcal{C}_{\mathsf{SHVZK}}$. Upon receiving $\mathsf{a}_1, \mathsf{z}_1$ from $\mathcal{C}_{\mathsf{SHVZK}}$ $\mathcal{V}^\star$ plays all the messages of Π_{MPCT} as in $\mathcal{H}_4$ ($\mathcal{H}_5$) except for the messages of τ_1 where he $\mathcal{V}^\star$ acts as a proxy between $\mathcal{C}_{\mathsf{SHVZK}}$ and $P^\star$. At the end of the execution $\mathcal{V}^\star$ runs the distinguisher D that distinguishes the output distribution of $\mathcal{H}_4$ from the output distribution of $\mathcal{H}_5$ and outputs what D outputs. We observe that if $\mathcal{C}_{\mathsf{SHVZK}}$ sends a simulated transcript then $P_2^\star$ acts as in $\mathcal{H}_4$ otherwise he acts as in $\mathcal{H}_5$.

[19]In order to extract the witnesses for the theorems proved by every party controlled by $\mathcal{A}$, $\mathsf{Sim}^{\mathsf{NMZK}}$ needs to rewind also from the 4th to the 3rd round, but this does not affect the reduction.

There is a subtlety in the above reduction $\mathcal{V}^\star$ runs the $\mathsf{Sim}^{\mathsf{NMZK}}$ that rewinds from the third to the second round. This means that $\mathcal{V}^\star$ has to be able to complete during the rewinds the third round while receiving different challenges $\mathsf{c}^1, \dots, \mathsf{c}^{\mathsf{poly}(\lambda)}$ w.r.t. Π_{OR}. Since we are splitting the challenge c, $\mathcal{V}^\star$ can just keep fixed the value c_1 reusing the same z_1 (sent by $\mathcal{C}_{\mathsf{SHVZK}}$) and computing an answer to a_0 using the knowledge of the decommitment information of $\mathtt{com}_0$. To argue that $\mathcal{A}$ correctly computes all the steps of Π_{MPCT}, also in this hybrid experiment we can use the simulator-extractor $\mathsf{Sim}^{\mathsf{NMZK}}$ to check whether the theorem proved by $\mathcal{A}$ is still true. If it is not the case we can construct a reduction to the Special HVZK property of BL_L. Note that the rewinds of $\mathsf{Sim}^{\mathsf{NMZK}}$ from the fourth to the third round do not affect the reduction. Moreover, the fact that $\mathsf{Sim}^{\mathsf{NMZK}}$ extracts the witnesses for the theorems proved by every party controlled by $\mathcal{A}$ still ensures that $\mathcal{A}$ behaves honestly.

- $\mathcal{H}_6$ proceeds exactly as $\mathcal{H}_5$ except that the Special HVZK simulator (Sim_L), instead of honest procedure ($\mathcal{P}_L$), is used to compute the prover's messages $\mathsf{a}_0, \mathsf{z}_0$ of the transcript $\tau_0 = (\mathsf{a}_0, \mathsf{c}_0, \mathsf{z}_0)$ w.r.t. the instance $\mathtt{com}_0$.

  We claim that $\{\mathsf{OUT}_{\mathcal{H}_5,\mathcal{A}(z)}(1^\lambda)\}$ and $\{\mathsf{OUT}_{\mathcal{H}_6,\mathcal{A}(z)}(1^\lambda)\}$ are computationally indistinguishable due the same arguments used to prove that $\{\mathsf{OUT}_{\mathcal{H}_4,\mathcal{A}(z)}(1^\lambda)\} \approx \{\mathsf{OUT}_{\mathcal{H}_5,\mathcal{A}(z)}(1^\lambda)\}$. Furthermore we claim that $\mathcal{A}$ still behaves honestly for the same arguments given in $\mathcal{H}_5$.
- The 7th hybrid experiment $\mathcal{H}_7$ differs from $\mathcal{H}_6$ in the way $\mathtt{com}_0$ is computed. More precisely, instead of committing to r_1 in $\mathtt{com}_0$, a commitment of a random string y is computed. For the same arguments used to prove that $\{\mathsf{OUT}_{\mathcal{H}_3,\mathcal{A}(z)}(1^\lambda)\} \approx \{\mathsf{OUT}_{\mathcal{H}_4,\mathcal{A}(z)}(1^\lambda)\}$, we claim that $\{\mathsf{OUT}_{\mathcal{H}_6,\mathcal{A}(z)}(1^\lambda)\} \approx \{\mathsf{OUT}_{\mathcal{H}_7,\mathcal{A}(z)}(1^\lambda)\}$ and that $\mathcal{A}$ still behaves honestly. We observe that r_1 appears only in the 4th round. More precisely there is no relation between r_1 and the values committed in $\mathcal{H}_1$. Therefore the security proof is almost over. Indeed our simulator Sim proceeds as $\mathcal{H}_7$ until the 3rd round, then invokes the func-

tionality thus obtaining a value r and completes the 4th round of $\mathcal{H}_7$ setting $r_1 = r \oplus \cdots \oplus r_n^*$.

□

4.4 Special WIPoK

4.4.1 Improving the Soundness of LS

In this section we show that the LS protocol does not enjoys special soundness when the statement to be proved is adaptively chosen by the prover in the last round. That is, if two accepting transcripts (that share the first round) are provided w.r.t. to two different instances x_0 and x_1, then only the witness w for x_b is extracted (with $b \in \{0, 1\}$). More precisely, given the accepting transcript $(\mathsf{ls}^1, \mathsf{ls}^2_0, \mathsf{ls}^3_0)$ for the statement x_0 and $(\mathsf{ls}^1, \mathsf{ls}^2_1, \mathsf{ls}^3_1)$ for the statement x_1 (with $\mathsf{ls}^2_0 \neq \mathsf{ls}^2_1$) then it could be that only w_b can be extracted. We provide a construction that overcomes this issue, allowing the extraction of the witnesses for both x_0 and x_1 thus obtaining a Σ-protocol where the special soundness holds even when the two accepting transcripts refer to different theorems adaptively chosen in the last round. Following [CPS$^+$16b] we refer to this property as adaptive-input special soundness.

Before showing why LS is not already adaptive-input special sound and how our construction works, we briefly describe the LS protocol with one-bit challenge following [OV12].

Let $\mathcal{P}$ be prover and $\mathcal{V}$ the verifier. The common input of $\mathcal{P}$ and $\mathcal{V}$ is κ, that represents the number of vertexes of the instance G to be proved. The graph G is represented by a $\kappa \times \kappa$ adjacency matrix MG where $\mathsf{MG}[i][j] = 1$ if there exists an edge between vertexes i and j in G. A non-edge position i,j is a pair of vertexes that are not connected in G and for which $\mathsf{MG}[i][j] = 0$.

- $\mathcal{P}$ picks a random κ-vertex cycle graph C and commits bit-by-bit to the corresponding adjacency matrix using a statistically binding commitment scheme.
- $\mathcal{V}$ responds with a randomly chosen bit b.

- $\mathcal{P}$ on input the graph G and the Hamiltonian cycle w executes the following steps. If $b = 0$, $\mathcal{P}$ opens all the commitments, showing that the matrix committed in the first round is actually an κ-vertex cycle. If $b = 1$, $\mathcal{P}$ sends a permutation π mapping the vertex of C in G. Then it opens the commitment of the adjacency matrix of C corresponding to the non-edges of the graph G.
- $\mathcal{V}$ accepts (outputs 1) if what he receives in the third round is consistent with the bit b that he was sent in the second round.

Getting the answer for both $b = 0$ and $b = 1$ (w.r.t. to the same graph G) allows the extraction of the cycle for G. The reason is the following. For $b = 0$ one gets the random cycle C. Then for $b = 1$ one gets the permutation mapping the random cycle in the actual cycle that is given to $\mathcal{P}$ before the last message of the protocol.

We now observe that a malicious prover $\mathcal{P}^\star$ could gives the answer for $b = 0$ w.r.t. to the graph G_0 and the answer for $b = 1$ w.r.t. the graph G_1 (due to the delayed-input nature of LS). This means that even knowing two accepting transcripts that share the first round, the permutation that maps the vertexes of C in G_0 it is not known. Therefore an efficient algorithm can only compute the cycle w_1 of G_1 and gets no information about the Hamiltonian cycle of G_0. Summing up, given the accepting transcripts $(\mathsf{ls}^1, 0, \mathsf{ls}^3_0)$ for the graph G_0 and $(\mathsf{ls}^1, 1, \mathsf{ls}^3_1)$ for the graph G_1, only the Hamiltonian cycle for G_1 can be computed. That is, only the cycle for the graph proved by $\mathcal{P}^\star$ to be Hamiltonian using as a second round the challenge 1 can be efficiently computed. Starting from this observation, in order to allow an efficient algorithm to compute cycles for both G_0 and G_1, we construct an improved version of LS that we denoted with $\mathsf{LS}^{\mathsf{imp}} = (\mathcal{P}^{\mathsf{imp}}, \mathcal{V}^{\mathsf{imp}})$. $\mathsf{LS}^{\mathsf{imp}}$ uses LS in a black-box way. For ease of exposition we use the following notation. $\mathsf{ls}^1 \leftarrow \mathcal{P}(1^\lambda, \kappa; \rho)$ denotes that $\mathcal{P}$ is executed on input the security parameter (in unary) 1^λ, κ and the randomness ρ and gives in output the first round of LS ls^1. $\mathsf{ls}^3 \leftarrow \mathcal{P}(G, w, \mathsf{ls}^2, \rho)$ denotes that $\mathcal{P}$ has computed the third round of LS by running on input the graph G, the cycle w for the graph G, the bit ls^2 and the randomness used to compute ls^1. $\mathcal{V}(\mathsf{ls}^1, \mathsf{ls}^2, \mathsf{ls}^3, G)$ denotes the output of $\mathcal{V}$ on input $\mathsf{ls}^1, \mathsf{ls}^2, \mathsf{ls}^3$ and the graph G. Let κ be the number of vertexes of the graph G to be proved, our $\mathsf{LS}^{\mathsf{imp}} = (\mathcal{P}^{\mathsf{imp}}, \mathcal{V}^{\mathsf{imp}})$ works as follows.

1. $\mathcal{P}^{\mathsf{imp}}$ on input the security parameter λ, κ and the randomness $\rho_0 || \rho_1$ computes $\mathsf{ls}_0^1 \leftarrow \mathcal{P}(1^\lambda, \kappa; \rho_0)$, $\mathsf{ls}_1^1 \leftarrow \mathcal{P}(1^\lambda, \kappa; \rho_1)$ and sends $(\mathsf{ls}_1^0, \mathsf{ls}_1^1)$ to $\mathcal{V}^{\mathsf{imp}}$.
2. $\mathcal{V}^{\mathsf{imp}}$ picks and sends a random bit b.
3. $\mathcal{P}^{\mathsf{imp}}$, upon receiving b, on input the graph G and the Hamiltonian cycle w for G computes $\mathsf{ls}_0^3 \leftarrow \mathcal{P}(G, w, b, \rho_0)$, $\mathsf{ls}_1^3 \leftarrow \mathcal{P}(G, w, 1-b, \rho_1)$ and sends $(\mathsf{ls}_0^3, \mathsf{ls}_1^3)$.
4. $\mathcal{V}^{\mathsf{imp}}$ accepts iff $\mathcal{V}(G, \mathsf{ls}_0^1, b, \mathsf{ls}_0^3) = 1$ and $\mathcal{V}(G, \mathsf{ls}_1^1, 1-b, \mathsf{ls}_1^3) = 1$.

Theorem 31. Assuming one-to-one OWFs, $\mathsf{LS}^{\mathsf{imp}}$ is a Σ-protocol with adaptive-input Special HVZK simulator and adaptive-input special soundness. Moreover $\mathsf{LS}^{\mathsf{imp}}$ is Zero Knowledge.

Proof. **(Delayed-input) Completeness.** The (delayed-input) completeness of $\mathsf{LS}^{\mathsf{imp}}$ comes from the (delayed-input) completeness of LS.

Adaptive-input special soundness. Let us consider two accepting transcripts that share the first round for $\mathsf{LS}^{\mathsf{imp}}$: $\big((\mathsf{ls}_0, \mathsf{ls}_1), 0, (\mathsf{ls}_0^3, \mathsf{ls}_1^3)\big)$ for the statement G and $\big((\mathsf{ls}_0, \mathsf{ls}_1), 1, (\mathsf{ls}_1^{3'}, \mathsf{ls}_1^{3'})\big)$ for the statement G'. We can isolate the sub-transcripts $(\mathsf{ls}_0, 0, \mathsf{ls}_0^3)$ and $(\mathsf{ls}_0, 1, \mathsf{ls}_0^{3'})$ and observe that $\mathcal{V}(G, \mathsf{ls}_0^1, 0, \mathsf{ls}_0^3) = 1 = \mathcal{V}(G'\mathsf{ls}_0^1, 1, \mathsf{ls}_0^{3'})$. From what we discuss before about LS we know that in this case the witness w for G' can be extracted. Also let us now consider the two sub-transcripts $(\mathsf{ls}_1, 1, \mathsf{ls}_1^3)$ and $(\mathsf{ls}_1, 0, \mathsf{ls}_1^{3'})$. Also in this case, by observing that $\mathcal{V}(G, \mathsf{ls}_1, 1, \mathsf{ls}_1^3) = 1 = \mathcal{V}(G', \mathsf{ls}_1, 0, \mathsf{ls}_1^{3'})$, the cycle for G can be efficiently computed.

Adaptive-input Special HVZK. Following [MV16], we consider an adaptive-input Special HVZK simulator S associated to the LS protocol. This is equal to a Special HVZK simulator with the additional property that the first round can be simulated without knowing the instance to be proved (see Definition 1.1.12). In more details S works in two phases. In the first phase just 1^λ, the challenge ls^2, the number of vertexes κ is used to output the first round ls^1. We denote this phase using: $\mathsf{ls}^1 \leftarrow S(1^\lambda, \mathsf{ls}^2, \kappa)$. In the second phase S takes as input the instance and output the third round ls^3. We denote this phase using $\mathsf{ls}^3 \leftarrow S(G)$. The adaptive-input Special HVZK simulator S^{imp} for $\mathsf{LS}^{\mathsf{imp}}$ just internally runs S two times, once using b and once using $1-b$ as a challenge. In more details the two phase of S^{imp} are the following.

1. S^{imp}, on input 1^λ, the challenge b, κ and the randomness $\rho_b||\rho_{1-b}$, computes $\mathsf{ls}^1_b \leftarrow S(1^\lambda, b, \kappa; \rho_b)$, $\mathsf{ls}^1_{1-b} \leftarrow S(1^\lambda, 1-b, \kappa; \rho_{1-b})$ and outputs $(\mathsf{ls}^1_b, \mathsf{ls}^1_{1-b})$.
2. S^{imp}, on input the graph G, ρ_0 and ρ_1 computes $\mathsf{ls}^3_b \leftarrow S(G, \rho_b)$, $\mathsf{ls}^3_{1-b} \leftarrow S(G, \rho_{1-b})$ and outputs $(\mathsf{ls}^3_b, \mathsf{ls}^3_{1-b})$.

The transcript $\big((\mathsf{ls}^1_b, \mathsf{ls}^1_{1-b}), b, (\mathsf{ls}^3_b, \mathsf{ls}^3_{1-b})\big)$ output by S^{imp} is is computationally indistinguishable from a transcript computed by $\mathcal{P}^{\mathsf{imp}}$ (that uses as input an Hamiltonian cycle w of G) due to the security of the underlying adaptive-input Special HVZK simulator S.

Zero-Knowledge. The ZK simulator of $\mathsf{LS}^{\mathsf{imp}}$ just needs to guess the bit b chosen by the adversarial verifier and runs the adaptive-input Special HVZK simulator.

□

It is easy to see that (as for LS) if we consider λ parallel executions of $\mathsf{LS}^{\mathsf{imp}}$ then we obtain a protocol LS^λ that still enjoys adaptive-input completeness, adaptive-input special soundness, adaptive-input Special HVZK. Moreover LS^λ is WI. Formally, we can claim the following theorems.

Theorem 32. Assuming one-to-one OWFs, LS^λ is a Σ-protocol with adaptive-input Special HVZK, adaptive-input special soundness and WI.

Proof. Completeness, adaptive-input special soundness and adaptive-input Special HVZK come immediately from the adaptive-input special soundness and adaptive-input Special HVZK of $\mathsf{LS}^{\mathsf{imp}}$. The WI comes from the observation that $\mathsf{LS}^{\mathsf{imp}}$ is WI (due to the zero knowledge property), and that WI is preserved under parallel (and concurrent) composition. □

Theorem 33. Assuming OWFs, LS^λ is a 4-round public-coin interactive protocol with adaptive-input Special HVZK, adaptive-input special soundness and WI.

Proof. The proof of this theorem just relies on the observation that in order to instantiate a statistically binding commitment scheme using OWFs an additional round is required to compute the first round of Naor's commitment scheme [Nao91]. □

Observe that since Hamiltonicity is an $\mathcal{NP}$-complete language, the above constructions work for any $\mathcal{NP}$ language through $\mathcal{NP}$ reductions. For simplicity in the rest of the work we will omit the $\mathcal{NP}$ reduction therefore assuming that the above scheme works directly on a given $\mathcal{NP}$-language L.

Combining (adaptive-input) Special HVZK PoK Through [CDS94] In our work we use the well known technique for composing two Σ-protocols to compute the OR of statements [CDS94, GMY06], described in Section 2.2. In our work we instantiate Π^{OR} using as Π_0 and Π_1 the Blum's protocol [Blu86a] for the $\mathcal{NP}$-complete language for graph Hamiltonicity (that also is a Σ-Protocol). Therefore Th. 4 (and Th. 2) can be applied.

We also consider an instantiation of Π^{OR} using as $\Pi = (\mathcal{P}, \mathcal{V})$ our LS^λ. If we instantiate Π^{OR} using LS^λ and the corresponding adaptive-input Special HVZK simulator LS^λ, then Π^{OR} is adaptive-input special soundness. More formally we can claim the following theorem.

Theorem 34. If Π^{OR} is instantiated using LS^λ (and the corresponding adaptive-input Special HVZK simulator S^λ), then Π^{OR} enjoys the delayed-input completeness and adaptive-input special sound for the $\mathcal{NP}$-relation $\mathsf{Rel}_{\mathsf{L_{OR}}}$.

Proof. The delayed-input completeness follows from the delayed-input completeness of LS^λ.

Adaptive-input special soundness. Let us consider two accepting transcripts that share the first round for Π^{OR}: $\big((\pi_0, \pi_1), \pi^2, (\pi_0^2, \pi_0^3, \pi_1^2, \pi_1^3)\big)$ for the statement (x_0, x_1) and

$\big((\pi_0, \pi_1), \pi^{2'}, (\pi_0^{2'}, \pi_0^{3'}, \pi_1^{2'}\pi_1^{3'})\big)$ for the statement (x'_0, x'_1), where $\pi^2 \neq \pi^{2'}$. We observe that since $\pi^2 \neq \pi^{2'}$, $\pi^2 = \pi_0^2 \oplus \pi_1^2$ and $\pi^{2'} = \pi_0^{2'} \oplus \pi_1^{2'}$ it holds that either $\pi_0^2 \neq \pi_0^{2'}$ or $\pi_1^2 \neq \pi_1^{2'}$. Suppose w.l.o.g. that $\pi_0^2 \neq \pi_0^{2'}$. Then we are guaranteed from the adaptive-input special soundness of LS^λ that using the transcripts $(\pi_0, \pi_0^2, \pi_0^3)$ and $(\pi_0, \pi_0^{2'}, \pi_0^{3'})$ the values (w_a, w_b) s.t. $(x_0, w_a) \in \mathsf{Rel}_{\mathsf{L}_0}$ and $(x'_0, w_b) \in \mathsf{Rel}_{\mathsf{L}_0}$ can be extracted in polynomial-time. The same arguments can be used when $\pi_1^2 \neq \pi_1^{2'}$. □

Using a result of [CPS+16b] we can claim the following theorem.

Theorem 35. Π^{OR} instantiated using LS^λ is adaptive-input PoK for the $\mathcal{NP}$-relation $\mathsf{Rel}_{\mathsf{L}_{\mathsf{OR}}}$.

It would be easy to prove that Π^{OR} is also WI, however in this work we are not going to rely directly on the WI property of Π^{OR}, in order to deal with the rewinding issue that we have described earlier. More precisely, in the two main contributions of this work we will use Π^{OR} (the one instantiated from Blum's protocol and the one instantiated using LS^λ) in a non-black box way in order to prove the security of our protocols. It will be crucial for our reduction to rely on the (adaptive-input) Special HVZK of Π_0 and Π_1 instead of using directly the WI property of Π^{OR}. The intuitively reason is that it is often easier in a reduction to rely on the security of a non-interactive primitive (like Special HVZK is) instead of an interactive primitive (like WI). This is the reason why we use the OR composition of [CDS94, GMY06] combined with the Blum's protocol (or the LS protocol) instead of relying on the (adaptive-input) WI provided by a Blum's protocol (LS protocol).

In the rest of the work, in order to rely on OWFs only, we sometimes use a four round version of Blum's and LS protocols. In this case there is an additional initial round that goes from the verifier to the prover and corresponds to the first round of Naor's commitment scheme [Nao91].

Conclusion

In this thesis we developed new cryptographic building blocks that can be used to improve the overall round complexity of the protocols they are used in. We have considered both constructions of theoretical interest and practical constructions. We have also shown round-efficient larger protocols that make use of our building blocks and improve the state of the art.

Some interesting and important open questions related to our work are still open.

Our OR-composition technique relaxes the requirement of the CDS-OR technique consisting in specifying all instances already at the beginning of the protocol. However still our result does not match the power of LS where no theorem is required for the protocol to start. An immediate open question is whether one can improve our OR transform so that none of the two theorems is required when the protocol starts. A first step to answer this question is made in [CPS$^+$16b] that realizes an OR-composition where no theorem is needed for the protocol to start, however this construction relays on DDH assumption. It is interesting to obtain the same result without requiring computational assumptions.

Regarding non-malleable commitment schemes various natural and fascinating questions remain open after our work. Examples of open questions about concurrent NM commitments are the following: 1) the existence of 3-round schemes based on standard generic hardness assumptions (e.g., trapdoor permutations) w.r.t. polynomial-time adversaries only; 2) the existence of 3-round schemes with black-box use of primitives.

Other interesting questions for future work are the existence of a 4-round multi-party computation protocol for any functionality under standard generic hardness assumptions (e.g., trapdoor permutations) w.r.t. polynomial-time adversaries only, rather than specific number-theoretic assumptions.

Acknowledgments

I would like to express my deepest gratitude to my advisor Ivan Visconti who didn't ask me to pay for his advices. If he had asked that, I believe a whole life of work wouldn't be enough to repay my debt. His valuable lessons and his constant support allowed me to grow up during these 3 years.

I have a debt of gratitude to Carlo Blundo: he planted the first seed of this PhD during my master thesis. I am very grateful for his support.

I am thankful to Giuseppe Persiano for his valuable advices.

I would like to thank Rafail Ostrovsky who gave me the opportunity to spend more then a year at UCLA. It was a great experience that taught me a lot, not only from a research point of view.

I really enjoyed the three months that I spent with the people of the crypto group of Aarhus University, thanks to Claudio Orlandi and Ivan Daamgard for giving me this opportunity.

I would like to thank Yuval Ishai and Sanjam Garg for reviewing my PhD thesis.

Thanks to my co-author, colleague and very good friend Michele Ciampi, I am happy to have shared these 3 years with him. A big "thank you" also goes to all other my friends, especially to Alessandro who is always there for me.

Last but not least, I would like to thank all my family. In particular, my parents Maria Rosaria and Francesco, my results are the sum of their life lessons and their love. Thanks to Anna for being an amazing sister. Finally thanks to you, the one who allows my smile to shine.

Bibliography

[ACJ17] Prabhanjan Ananth, Arka Rai Choudhuri, and Abhishek Jain. A new approach to round-optimal secure multiparty computation. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology - CRYPTO 2017 - 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part I*, volume 10401 of *Lecture Notes in Computer Science*, pages 468–499. Springer, 2017.

[AOS13] Masayuki Abe, Tatsuaki Okamoto, and Koutarou Suzuki. Message recovery signature schemes from sigma-protocols. *IEICE Transactions*, 96-A(1):92–100, 2013.

[Bar02] Boaz Barak. Constant-round coin-tossing with a man in the middle or realizing the shared random string model. In *43rd Symposium on Foundations of Computer Science (FOCS 2002), 16-19 November 2002, Vancouver, BC, Canada, Proceedings*, pages 345–355, 2002.

[BFGM01] Mihir Bellare, Marc Fischlin, Shafi Goldwasser, and Silvio Micali. Identification protocols secure against reset attacks. In *Advances in Cryptology - EUROCRYPT 2001, International Conference on the Theory and Application of Cryptographic Techniques, Innsbruck, Austria, May 6-10, 2001, Proceeding*, pages 495–511, 2001.

[BGJ$^+$17] Saikrishna Badrinarayanan, Vipul Goyal, Abhishek Jain, Yael Tauman Kalai, Dakshita Khurana, and Amit Sahai. Promise zero knowledge and its applications to round optimal mpc. Cryptology ePrint Archive, Report 2017/1088, 2017. `https://eprint.iacr.org/2017/1088`.

[BGR$^+$15] Hai Brenner, Vipul Goyal, Silas Richelson, Alon Rosen, and Margarita Vald. Fast non-malleable commitments. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, CO, USA, October 12-6, 2015*, pages 1048–1057, 2015.

[BHP17] Zvika Brakerski, Shai Halevi, and Antigoni Polychroniadou. Four round secure computation without setup. In Yael Kalai and Leonid Reyzin, editors, *Theory of Cryptography - 15th International Conference, TCC 2017, Baltimore, MD, USA, November 12-15, 2017, Proceedings, Part I*, volume 10677 of *Lecture Notes in Computer Science*, pages 645–677. Springer, 2017.

[BJY97] Mihir Bellare, Markus Jakobsson, and Moti Yung. Round-optimal zero-knowledge arguments based on any one-way function. In *Advances in Cryptology - EUROCRYPT '97, International Conference on the Theory and Application of Cryptographic Techniques, Konstanz, Germany, May 11-15, 1997, Proceeding*, volume 1233 of *Lecture Notes in Computer Science*, pages 280–305. Springer, 1997.

[BKZZ16] Foteini Baldimtsi, Aggelos Kiayias, Thomas Zacharias, and Bingsheng Zhang. Indistinguishable proofs of work or knowledge. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *Advances in Cryptology - ASIACRYPT 2016 - 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part II*, volume 10032 of *Lecture Notes in Computer Science*, pages 902–933, 2016.

[BL17] Fabrice Benhamouda and Huijia Lin. k-round mpc from k-round ot via garbled interactive circuits. Cryptology ePrint Archive, Report 2017/1125, 2017. `https://eprint.iacr.org/2017/1125`.

[Blu86a] Manuel Blum. How to prove a theorem so no one else can claim it. In *In Proceedings of the International Congress of Mathematicians*, 1986.

[Blu86b] Manuel Blum. How to prove a theorem so no one else can claim it. In *In Proceedings of the International Congress of Mathematicians*, 1986.

[BP15] Nir Bitansky and Omer Paneth. Zaps and non-interactive witness indistinguishability from indistinguishability obfuscation. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *Theory of Cryptography - 12th Theory of Cryptography Conference, TCC 2015, Warsaw, Poland, March 23-25, 2015, Proceedings, Part II*, volume 9015 of *Lecture Notes in Computer Science*, pages 401–427. Springer, 2015.

[BPS06] Boaz Barak, Manoj Prabhakaran, and Amit Sahai. Concurrent non-malleable zero knowledge. In *47th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2006), 21-24 October 2006, Berkeley, California, USA, Proceedings*, pages 345–354, 2006.

[BPSV08] Carlo Blundo, Giuseppe Persiano, Ahmad-Reza Sadeghi, and Ivan Visconti. Improved security notions and protocols for non-transferable identification. In *Computer Security - ESORICS 2008, 13th European Symposium on Research in Computer Security, Málaga, Spain, October 6-8, 2008. Proceedings*, volume 5283 of *Lecture Notes in Computer Science*, pages 364–378. Springer, 2008.

[BR93] Mihir Bellare and Phillip Rogaway. Entity authentication and key distribution. In Douglas R. Stinson, editor, *Advances in Cryptology - CRYPTO '93, 13th Annual International Cryptology Conference, Santa Barbara, California, USA, August 22-26, 1993, Proceedings*, volume 773 of *Lecture Notes in Computer Science*, pages 232–249. Springer, 1993.

[CD98] Ronald Cramer and Ivan Damgård. Zero-knowledge proofs for finite field arithmetic; or: Can zero-knowledge be for free? In Hugo Krawczyk, editor, *Advances in Cryptology - CRYPTO '98, 18th Annual International Cryptology Conference, Santa Barbara, California, USA, August 23-27, 1998, Proceedings*, volume 1462 of *Lecture Notes in Computer Science*, pages 424–441. Springer, 1998.

[CD08] Ran Canetti and Ronny Ramzi Dakdouk. Extractable perfectly one-way functions. In *Automata, Languages and Programming, 35th International Colloquium, ICALP 2008, Reykjavik, Iceland, July 7-11, 2008, Proceedings, Part II - Track B: Logic, Semantics, and Theory of Programming & Track C: Security and Cryptography Foundations*, pages 449–460, 2008.

[CDS94] Ronald Cramer, Ivan Damgård, and Berry Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In YvoG. Desmedt, editor, *Advances in Cryptology — CRYPTO '94*, volume 839 of *Lecture Notes in Computer Science*, pages 174–187. Springer Berlin Heidelberg, 1994.

[CGGM00] Ran Canetti, Oded Goldreich, Shafi Goldwasser, and Silvio Micali. Resettable zero-knowledge (extended abstract). In F. Frances Yao and Eugene M. Luks, editors, *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing, May 21-23, 2000, Portland, OR, USA*, pages 235–244. ACM, 2000.

[CKPR01] Ran Canetti, Joe Kilian, Erez Petrank, and Alon Rosen. Black-box concurrent zero-knowledge requires omega˜(log n) rounds. In *Proceedings on 33rd Annual ACM Symposium on Theory of Computing, July 6-8, 2001, Heraklion, Crete, Greece*, pages 570–579, 2001.

[COP$^+$14] Kai-Min Chung, Rafail Ostrovsky, Rafael Pass, Muthuramakrishnan Venkitasubramaniam, and Ivan Visconti. 4-round resettably-sound zero knowledge. In Yehuda Lindell, editor, *Theory of Cryptography - 11th Theory of Cryptography Conference, TCC 2014, San Diego, CA, USA, February 24-26, 2014. Proceedings*, volume 8349 of *Lecture Notes in Computer Science*, pages 192–216. Springer, 2014.

[COSV12] Chongwon Cho, Rafail Ostrovsky, Alessandra Scafuro, and Ivan Visconti. Simultaneously resettable arguments of knowledge. In *Theory of Cryptography - 9th Theory of Cryptography Conference, TCC 2012, Taormina, Sicily, Italy, March 19-21, 2012. Proceedings*, pages 530–547, 2012.

[COSV16] Michele Ciampi, Rafail Ostrovsky, Luisa Siniscalchi, and Ivan Visconti. Concurrent non-malleable commitments (and more) in 3 rounds. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part III*, volume 9816 of *Lecture Notes in Computer Science*, pages 270–299. Springer, 2016. Full version `https://eprint.iacr.org/2016/566`.

[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, *Theory of Cryptography - 15th International Conference, TCC 2017, Baltimore, MD, USA, November 12-15, 2017, Proceedings, Part I*, volume 10677 of *Lecture Notes in Computer Science*, pages 711–742. Springer, 2017. Full version `https://eprint.iacr.org/2017/931`.

[COSV17b] Michele Ciampi, Rafail Ostrovsky, Luisa Siniscalchi, and Ivan Visconti. Four-round concurrent non-malleable commitments from one-way functions. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology - CRYPTO 2017 - 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part II*, volume 10402 of *Lecture Notes in Computer Science*, pages 127–157. Springer, 2017. Full version `https://eprint.iacr.org/2016/621`.

[COSV17c] Michele Ciampi, Rafail Ostrovsky, Luisa Siniscalchi, and Ivan Visconti. Round-optimal secure two-party computation from trapdoor permutations. In *Theory of Cryptography, Fifteenth Theory of Cryptography Conference, TCC 2017, Baltimore, USA, November 12-15, 2017, Proceedings*, Lecture Notes in Computer Science. Springer, 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, *Theory of Cryptography - 13th International Conference, TCC 2016-A, Tel Aviv, Israel, January 10-13, 2016, Proceedings, Part II*, volume 9563 of *Lecture Notes in Computer Science*, pages 112–141. Springer, 2016. Full version `http://eprint.iacr.org/2015/810`.

[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, *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part II*, volume 9666 of *Lecture Notes in Computer Science*, pages 63–92. Springer, 2016. Full version `https://eprint.iacr.org/2016/175`.

[Cra96] Ronald Cramer. *Modular design of secure yet practical cryptographic protocols*. PhD thesis, University of Amsterdam, 1996.

[CVZ11] Zhenfu Cao, Ivan Visconti, and Zongyang Zhang. On constant-round concurrent non-malleable proof systems. *Inf. Process. Lett.*, 111(18):883–890, 2011.

[Dam10] Ivan Damgård. On Σ-protocol. `http://www.cs.au.dk/~ivan/Sigma.pdf`, 2010.

[DDN91] Danny Dolev, Cynthia Dwork, and Moni Naor. Non-malleable cryptography (extended abstract). In *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing, May 5-8, 1991, New Orleans, Louisiana, USA*, pages 542–552, 1991.

[DDO$^+$01] Alfredo De Santis, Giovanni Di Crescenzo, Rafail Ostrovsky, Giuseppe Persiano, and Amit Sahai. Robust non-interactive zero knowledge. In Joe Kilian, editor, *Advances in Cryptology - CRYPTO 2001, 21st Annual International Cryptology Conference, Santa Barbara, California, USA, August 19-23, 2001, Proceedings*, volume 2139 of *Lecture Notes in Computer Science*, pages 566–598. Springer, 2001.

[DG03] Ivan Damgård and Jens Groth. Non-interactive and reusable non-malleable commitment schemes. In *Proceedings of the 35th Annual ACM Symposium on Theory of Computing, June 9-11, 2003, San Diego, CA, USA*, pages 426–437, 2003.

[DN00] Cynthia Dwork and Moni Naor. Zaps and their applications. In *41st Annual Symposium on Foundations of Computer Science, FOCS 2000, 12-14 November 2000, Redondo Beach, California, USA*, pages 283–293, 2000.

[DPV04] Giovanni Di Crescenzo, Giuseppe Persiano, and Ivan Visconti. Improved setup assumptions for 3-round resettable zero knowledge. In *Advances in Cryptology - ASIACRYPT 2004, 10th International Conference on the Theory and Application of Cryptology and Information Security, Jeju Island, Korea, December 5-9, 2004, Proceedings*, volume 3329 of *Lecture Notes in Computer Science*, pages 530–544. Springer, 2004.

[FFS87] Uriel Feige, Amos Fiat, and Adi Shamir. Zero knowledge proofs of identity. In *Proceedings of the 19th Annual ACM Symposium on Theory of Computing, 1987, New York, New York, USA*, pages 210–217, 1987.

[FLS90] Uriel Feige, Dror Lapidot, and Adi Shamir. Multiple non-interactive zero knowledge proofs based on a single random string (extended abstract). In *31st Annual Symposium on Foundations of Computer Science, St. Louis, Missouri, USA, October 22-24, 1990, Volume I*, pages 308–317. IEEE Computer Society, 1990.

[GK15] Jens Groth and Markulf Kohlweiss. One-out-of-many proofs: Or how to leak a secret and spend a coin. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part II*, volume 9057 of *Lecture Notes in Computer Science*, pages 253–280. Springer, 2015.

[GKP$^+$17] Vipul Goyal, Ashutosh Kumar, Sunoo Park, Silas Richelson, and Akshayaram Srinivasan. New constructions of non-malleable commitments and applications. Private communication, 2017.

[GL89] Oded Goldreich and Leonid A. Levin. A hard-core predicate for all one-way functions. In *Proceedings of the 21st Annual ACM Symposium on Theory of Computing, May 14-17, 1989, Seattle, Washigton, USA*, pages 25–32, 1989.

[GLOV12] Vipul Goyal, Chen-Kuei Lee, Rafail Ostrovsky, and Ivan Visconti. Constructing non-malleable commitments: A black-box approach. In *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012*, pages 51–60, 2012.

[GMPP16] Sanjam Garg, Pratyay Mukherjee, Omkant Pandey, and Antigoni Polychroniadou. The exact round complexity of secure computation. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part II*, volume 9666 of *Lecture Notes in Computer Science*, pages 448–476. Springer, 2016.

[GMY06] Juan A. Garay, Philip MacKenzie, and Ke Yang. Strengthening zero-knowledge protocols using signatures. *Journal of Cryptology*, 19(2):169–209, 2006.

[Gol09] Oded Goldreich. *Foundations of cryptography: volume 2, basic applications.* Cambridge university press, 2009.

[GOS06] Jens Groth, Rafail Ostrovsky, and Amit Sahai. Perfect non-interactive zero knowledge for NP. In Serge Vaudenay, editor, *Advances in Cryptology - EUROCRYPT 2006, 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28 - June 1, 2006, Proceedings*, volume 4004 of *Lecture Notes in Computer Science*, pages 339–358. Springer, 2006.

[Goy11] Vipul Goyal. Constant round non-malleable protocols using one way functions. In *Proceedings of the 43rd ACM Symposium on Theory of Computing, STOC 2011, San Jose, CA, USA, 6-8 June 2011*, pages 695–704, 2011.

[GPR16] Vipul Goyal, Omkant Pandey, and Silas Richelson. Textbook non-malleable commitments. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 1128–1141, 2016. Full version: Cryptology ePrint Archive, Report 2015/1178.

[GQ88] Louis C. Guillou and Jean-Jacques Quisquater. A practical zero-knowledge protocol fitted to security microprocessor minimizing both transmission and memory. In Christoph G. Günther, editor, *Advances in Cryptology - EUROCRYPT '88, Workshop on the Theory and Application of of Cryptographic Techniques, Davos, Switzerland, May 25-27, 1988, Proceedings*, volume 330 of *Lecture Notes in Computer Science*, pages 123–128. Springer, 1988.

[GRRV14] Vipul Goyal, Silas Richelson, Alon Rosen, and Margarita Vald. An algebraic approach to non-malleability. In *55th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2014, Philadelphia, PA, USA, October 18-21, 2014*, pages 41–50, 2014. An updated full version is available at `http://eprint.iacr.org/2014/586`.

[HHPV17] Shai Halevi, Carmit Hazay, Antigoni Polychroniadou, and Muthuramakrishnan Venkitasubramaniam. Round-optimal secure multi-party computation. Cryptology ePrint Archive, Report 2017/1056, 2017. `https://eprint.iacr.org/2017/1056`.

[HKR$^+$14] Feng Hao, Matthew Nicolas Kreeger, Brian Randell, Dylan Clarke, Siamak Fayyaz Shahandashti, and Peter Hyun-Jeen Lee. Every vote counts: Ensuring integrity in large-scale electronic voting. In *2014 Electronic Voting Technology Workshop/Workshop on Trustworthy Elections, EVT/WOTE '14, San Diego, CA, USA, August 18-19, 2014*. USENIX Association, 2014.

[HM96] Shai Halevi and Silvio Micali. Practical and provably-secure commitment schemes from collision-free hashing. In *Advances in Cryptology - CRYPTO '96, 16th Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 1996, Proceedings*, pages 201–215, 1996.

[Kat02] Jonathan Katz. *Efficient Cryptographic Protocols Preventing "Man-in-the-Middle" Attacks.* PhD thesis, Columbia University, 2002.

[Khu17] Dakshita Khurana. Round optimal concurrent non-malleability from polynomial hardness. In Yael Kalai and Leonid Reyzin, editors, *Theory of Cryptography - 15th International Conference, TCC 2017, Baltimore, MD, USA, November 12-15, 2017, Proceedings, Part II*, volume 10678 of *Lecture Notes in Computer Science*, pages 139–171. Springer, 2017.

[KO04] Jonathan Katz and Rafail Ostrovsky. Round-optimal secure two-party computation. In *Advances in Cryptology - CRYPTO 2004, 24th Annual International Cryptology-Conference, Santa Barbara, California, USA, August 15-19, 2004, Proceedings*, pages 335–354, 2004.

[KOS03] Jonathan Katz, Rafail Ostrovsky, and Adam D. Smith. Round efficiency of multi-party computation with a dishonest majority. In Eli Biham, editor, *Advances in Cryptology - EUROCRYPT 2003, International Conference on the Theory and Applications of Cryptographic Techniques, Warsaw, Poland, May 4-8, 2003, Proceedings*, volume 2656 of *Lecture Notes in Computer Science*, pages 578–595. Springer, 2003.

[KS17] Dakshita Khurana and Amit Sahai. How to achieve non-malleability in one or two rounds. In Chris Umans, editor, *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 564–575. IEEE Computer Society, 2017.

[Lin15] Yehuda Lindell. An efficient transform from Sigma protocols to NIZK with a CRS and non-programmable random oracle. In *Theory of Cryptography - 12th Theory of Cryptography Conference, TCC 2015, Warsaw, Poland, March 23-25, 2015, Proceedings, Part I*, pages 93–109, 2015.

[LP11a] Huijia Lin and Rafael Pass. Concurrent non-malleable zero knowledge with adaptive inputs. In Yuval Ishai, editor, *Theory of Cryptography - 8th Theory of Cryptography Conference, TCC 2011, Providence, RI, USA, March 28-30, 2011. Proceedings*, volume 6597 of *Lecture Notes in Computer Science*, pages 274–292. Springer, 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, *Proceedings of the 43rd ACM Symposium on Theory of Computing, STOC 2011, San Jose, CA, USA, 6-8 June 2011*, pages 705–714. ACM, 2011.

[LP15] Huijia Lin and Rafael Pass. Constant-round nonmalleable commitments from any one-way function. *J. ACM*, 62(1):5:1–5:30, 2015.

[LPS17] Huijia Lin, Rafael Pass, and Pratik Soni. Two-round and non-interactive concurrent non-malleable commitments from time-lock puzzles. In Chris Umans, editor, *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 576–587. IEEE Computer Society, 2017.

[LPV08] Huijia Lin, Rafael Pass, and Muthuramakrishnan Venkitasubramaniam. Concurrent non-malleable commitments from any one-way function. In Ran Canetti, editor, *Theory of Cryptography, Fifth Theory of Cryptography Conference, TCC 2008, New York, USA, March 19-21, 2008.*, volume 4948 of *Lecture Notes in Computer Science*, pages 571–588. Springer, 2008.

[LPV09] Huijia Lin, Rafael Pass, and Muthuramakrishnan Venkitasubramaniam. A unified framework for concurrent security: universal composability from stand-alone nonmalleability. In *Proceedings of the 41st Annual ACM Symposium on Theory of Computing,STOC 2009, Bethesda, MD, USA, May 31 - June 2, 2009*, pages 179–188, 2009.

[LS90] Dror Lapidot and Adi Shamir. Publicly verifiable non-interactive zero-knowledge proofs. In *Advances in Cryptology - CRYPTO*, 1990.

[Mau09] Ueli M. Maurer. Unifying zero-knowledge proofs of knowledge. In Bart Preneel, editor, *Progress in Cryptology - AFRICACRYPT 2009, Second International Conference on Cryptology in Africa, Gammarth, Tunisia, June 21-25, 2009. Proceedings*, volume 5580 of *Lecture Notes in Computer Science*, pages 272–286. Springer, 2009.

[Mau15] Ueli Maurer. Zero-knowledge proofs of knowledge for group homomorphisms. *Designs, Codes and Cryptography*, pages 1–14, 2015.

[MR01] Silvio Micali and Leonid Reyzin. Soundness in the public-key model. In *Advances in Cryptology - CRYPTO 2001, 21st Annual International Cryptology Conference, Santa Barbara, California, USA, August 19-23, 2001, Proceedings*, pages 542–565, 2001.

[MV16] Arno Mittelbach and Daniele Venturi. Fiat-shamir for highly sound protocols is instantiable. In Vassilis Zikas and Roberto De Prisco, editors, *Security and Cryptography for Networks - 10th International Conference, SCN 2016, Amalfi, Italy, August 31 - September 2, 2016, Proceedings*, volume 9841 of *Lecture Notes in Computer Science*, pages 198–215. Springer, 2016.

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

[NY89] Moni Naor and Moti Yung. Universal one-way hash functions and their cryptographic applications. In *Proceedings of the 21st Annual ACM Symposium on Theory of Computing, May 14-17, 1989, Seattle, Washigton, USA*, pages 33–43, 1989.

[OPV08] Rafail Ostrovsky, Giuseppe Persiano, and Ivan Visconti. Constant-round concurrent non-malleable zero knowledge in the bare public-key model. In *Automata, Languages and Programming, 35th International Colloquium, ICALP 2008, Reykjavik, Iceland, July 7-11, 2008, Proceedings, Part II - Track B: Logic, Semantics, and Theory of Programming & Track C: Security and Cryptography Foundations*, pages 548–559, 2008.

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

[OV12] Rafail Ostrovsky and Ivan Visconti. Simultaneous resettability from collision resistance. *Electronic Colloquium on Computational Complexity (ECCC)*, 19:164, 2012.

[Pas03] Rafael Pass. Simulation in quasi-polynomial time, and its application to protocol composition. In Eli Biham, editor, *Advances in Cryptology - EUROCRYPT 2003, International Conference on the Theory and Applications of Cryptographic Techniques, Warsaw, Poland, May 4-8, 2003, Proceedings*, volume 2656 of *Lecture Notes in Computer Science*, pages 160–176. Springer, 2003.

[Pas04a] Rafael Pass. Alternative variants of zero-knowledge proofs. Master's thesis, Kungliga Tekniska Högskolan, 2004. Licentiate Thesis Stockholm, Sweden.

[Pas04b] Rafael Pass. Bounded-concurrent secure multi-party computation with a dishonest majority. In László Babai, editor, *Proceedings of the 36th Annual ACM Symposium on Theory of Computing, Chicago, IL, USA, June 13-16, 2004*, pages 232–241. ACM, 2004.

[Pas13] Rafael Pass. Unprovable security of perfect NIZK and non-interactive non-malleable commitments. In *TCC*, pages 334–354, 2013.

[Ped91] Torben P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In *Advances in Cryptology - CRYPTO '91, 11th Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15, 1991, Proceedings*, pages 129–140, 1991.

[Pol16] Antigoni Polychroniadou. *On the Communication and Round Complexity of Secure Computation.* PhD thesis, Aarhus University, December 2016.

[PPV08] Omkant Pandey, Rafael Pass, and Vinod Vaikuntanathan. Adaptive one-way functions and applications. In *Advances in Cryptology - CRYPTO 2008, 28th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2008. Proceedings*, pages 57–74, 2008.

[PR05a] Rafael Pass and Alon Rosen. Concurrent non-malleable commitments. In *46th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2005), 23-25 October 2005, Pittsburgh, PA, USA, Proceedings*, pages 563–572, 2005.

[PR05b] Rafael Pass and Alon Rosen. New and improved constructions of non-malleable cryptographic protocols. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing, Baltimore, MD, USA, May 22-24, 2005*, pages 533–542, 2005.

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

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

[PS96] David Pointcheval and Jacques Stern. Security proofs for signature schemes. In Ueli M. Maurer, editor, *Advances in Cryptology - EUROCRYPT '96, International Conference on the Theory and Application of Cryptographic Techniques, Saragossa, Spain, May 12-16, 1996, Proceeding*, volume 1070 of *Lecture Notes in Computer Science*, pages 387–398. Springer, 1996.

[PW10] Rafael Pass and Hoeteck Wee. Constant-round non-malleable commitments from sub-exponential one-way functions. In *Advances in Cryptology - EUROCRYPT 2010, 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, French Riviera, May 30 - June 3, 2010. Proceedings*, pages 638–655, 2010.

[Rom90] John Rompel. One-way functions are necessary and sufficient for secure signatures. In *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing, May 13-17, 1990, Baltimore, Maryland, USA*, pages 387–394, 1990.

[Sch89] C.P. Schnorr. Efficient identification and signatures for smart cards. In Gilles Brassard, editor, *Advances in Cryptology — CRYPTO' 89 Proceedings*, volume 435 of *Lecture Notes in Computer Science*, pages 239–252. Springer New York, 1989.

[SV12] Alessandra Scafuro and Ivan Visconti. On round-optimal zero knowledge in the bare public-key model. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology - EUROCRYPT 2012 - 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK, April 15-19, 2012. Proceedings*, volume 7237 of *Lecture Notes in Computer Science*, pages 153–171. Springer, 2012.

[YZ07] Moti Yung and Yunlei Zhao. Generic and practical resettable zero-knowledge in the bare public-key model. In Moni Naor, editor, *Advances in Cryptology - EUROCRYPT 2007, 26th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Barcelona, Spain, May 20-24, 2007, Proceedings*, volume 4515 of *Lecture Notes in Computer Science*, pages 129–147. Springer, 2007.