Delayed-Input Non-Malleable Zero Knowledge and Multi-Party Coin Tossing in Four Rounds

Michele Ciampi
DIEM
Università di Salerno
ITALY
mciampi@unisa.it

Rafail Ostrovsky
UCLA
Los Angeles
rafail@cs.ucla.edu

Luisa Siniscalchi
DIEM
Università di Salerno
ITALY
lsiniscalchi@unisa.it

Ivan Visconti
DIEM
Università di Salerno
ITALY
visconti@unisa.it

Abstract

In this work 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 FOCS 2014 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 Eurocrypt 2016 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 an additional contribution of this work and is of independent interest.

1 Introduction

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 [PR05, PR08]. 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 [Pas04] 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 very recent work of Garg et al. [GMPP16b] 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 [GMPP16a, Pol16] are the one of Pass et al. [PPV08] based on non-falsifiable assumptions, or the one of Ciampi et al. [COSV16] based on sub-exponentially strong one-to-one one-way functions. The achieved round complexity (i.e., 4 rounds) is proven optimal in [GMPP16b] when simulation is black box and the number of bits in the output of the functionality is superlogarithmic.

A very recent result of Ananth et al. [ACJ17] constructs a 4-round MPC protocol for any functionality assuming DDH w.r.t. superpolynomial-time adversaries. The above state-of-the art leaves open the following question.

Open Question 3: is there a 4-round secure MPCT protocol under standard assumptions?

1.1 Our Contribution

In this paper 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.

1.2 MPCT from NMZK

A first main idea that allows us to bypass the strong requirements of the construction of [GMPP16b] is that we avoid robust/non-malleable commitments and instead focus on non-malleable zero knowledge. Since we

[1] An unpublished prior work of Goyal et al. [GKP+17] achieves the same result on MPCT using completely different techniques.

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 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. Moreover our NMZK is also crucially used in [COSV17b].

1.3 Technical Overview on 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 [COSV17a] 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.

[2] Indeed, even the construction of [GMPP16b] that makes use of a special non-malleable commitments requires also a delayed-input zero-knowledge argument.

In [COSV17a] 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 paper 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] (see Def. 9 for a formal definition of adaptive-input special soundness) 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 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 compound 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 $c^1, \ldots, c^{\mathsf{poly}(\lambda)}$ (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}

[3] See Section C.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.

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 $i = 1, \ldots, \mathsf{poly}(\lambda)$. 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 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 [COSV17a]. 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}. Following the proof approach provided in [COSV16], we need to prove that the MiM adversary does not do the same. More precisely we want to guarantee that there is no right session where the MiM commits to two signatures of two different messages. 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.

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 order to prove that also in the reductions to adaptive-input Special HVZK the MiM still does not commit to the trapdoor we can uses the same approach explained above. Note that in these reductions it

[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.

[7] The trapdoor for our protocol is represented by two signatures for a verification key chosen by the verifier.

[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}.

is crucial that the rewinds needed to extract the committed message in Π_{nm} do not disturb the challengers involved in the reductions.

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 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.

1.4 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 [GMPP16b] 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 straight-forward and discussed in Section 4.1.

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)[9]. 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.

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).

[9] Notice here how crucial is to delayed-input have synchronous many-many NMZK.

2 Definitions and Tools

Preliminaries. 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, $x \leftarrow Q$ sampling of x from Q with uniform distribution. We use the abbreviation PPT that stays for probabilistic polynomial time. We use $\mathsf{poly}(\cdot)$ to indicate a generic polynomial function. 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 $\mathsf{Rel}_{\mathsf{L}}$ the corresponding polynomial-time relation (that is, $\mathsf{Rel}_{\mathsf{L}}$ is such that $L = L_{\mathsf{Rel}_{\mathsf{L}}}$). We also use $\hat{L}$ to denotes the language that includes L and all well formed instances that are not in L. Let A and B be two interactive probabilistic algorithms. 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. 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 (resp. B) 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. In App. A and in App B we recall some useful definitions.

3 4-Round Delayed-Input NMZK from OWFs

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 [SCO+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 [COSV17a]) 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 [SCO+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 $\mathsf{Rel}_{\mathsf{L}}$. 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 $\tilde{x}_1 \ldots, \tilde{x}_{\mathsf{poly}(\lambda)}$[10] (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. $(x,w) \in \mathsf{Rel}_{\mathsf{L}}$, and sends them to $\mathcal{P}$[11].

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

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

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 (Delayed-input NMZK)**.** *A delayed-input argument system $\Pi = (\mathcal{P}, \mathcal{V})$ for an $\mathcal{NP}$-language L with witness relation 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 $S(1^\lambda, z)$.*
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. $(\tilde{x}_i, \tilde{w}_i) \in \mathsf{Rel}_\mathsf{L}$.*

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 paper 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 paper, 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 $\mathtt{id}$ the public key of a signature scheme signing the transcript. The MiM will have to use a different $\mathtt{id}$ or to break the signature scheme.

3.1 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 $\mathsf{NM} = (\mathcal{S}, \mathcal{R})$ (See App. A.3 for a formal definition).
3. Two instantiations of the adaptive-input special sound LS protocol described in App. C in order to construct a 4-round delayed-input public-coin proof system for the OR of compound statement $\Pi_{\mathsf{OR}} = (\mathcal{P}_{\mathsf{OR}}, \mathcal{V}_{\mathsf{OR}})$ as described in App. C.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 $\mathsf{Rel}_{\mathsf{L}_{\mathsf{nm}}}$ 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 $\mathsf{Rel}_{\mathsf{L_{OR}}} = \{((x_L, x_{\mathsf{nm}}), w) : ((x_L, w) \in \mathsf{Rel}_L)$ OR $((x_{\mathsf{nm}}, w) \in \mathsf{Rel}_{\mathsf{L_{nm}}})\}$ (see Theorem 10 in App. C.2 for more details).

Overview of our protocol. We now give an high-level description of our delayed-input NMZK of Fig. 1. For a formal description see Fig. 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 $(\mathsf{nm}_4, \mathtt{dec}_{\mathsf{nm}})$. $\mathcal{P}_{\mathsf{NMZK}}$ 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 $(\mathsf{ls}^3_L, \mathsf{ls}^4_L, \mathsf{nm}_4, s_1, \mathsf{ls}^3_{\mathsf{nm}}, \mathsf{ls}^4_{\mathsf{nm}}, x, x_{\mathsf{nm}})$.

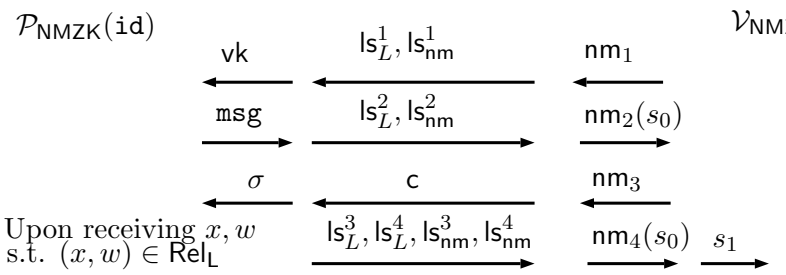
The verifier $\mathcal{V}_{\mathsf{NMZK}}$ accepts x iff the following conditions are satisfied:

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. $\mathcal{V}_{\mathsf{nm}}(x_{\mathsf{nm}}, \mathsf{ls}^1_{\mathsf{nm}}, \mathsf{ls}^2_{\mathsf{nm}}, \mathsf{ls}^3_{\mathsf{nm}}, \mathsf{ls}^4_{\mathsf{nm}}) = 1$.

The simulator extractor. Informally, the simulator $\mathsf{Sim}_{\mathsf{NMZK}}$ 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[12]. More precisely the simulator commits to a random string s_0, but computes s_1 s.t. $s_1 = (\sigma_1 || \sigma_2) \oplus s_0$[13]. 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 1.

[12] W.l.o.g. we assume that the signatures σ_1, σ_2 include the signed messages.

[13] For ease of exposition we will simply say that $\mathcal{A}_{\mathsf{NMZK}}$ commits to two signatures using NM.



- 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 1: Our 4-round delayed-input NMZK

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. 2.

Theorem 1. *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. Before that, we recall that $\mathsf{LS}_{\mathsf{nm}}$ and LS_L can be constructed from OWFs (see App. A) as well as Σ (using [Rom90]) and the 4-round public-coin synchronous honest-extractable non-malleable commitment scheme NM (see Sec. A.3).

(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 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

Common input: security parameter λ, identity $\mathtt{id} \in \{0,1\}^\lambda$ instances length: ℓ_L, ℓ_{nm}.
Input to $\mathcal{P}_{\mathsf{NMZK}}$: (x, w) s.t. $(x, w) \in \mathsf{Rel}_{\mathsf{L}}$, with (x, w) available only in the 4th round.
Commitment phase:

1. $\mathcal{V}_{\mathsf{NMZK}} \rightarrow \mathcal{P}_{\mathsf{NMZK}}$
    1. Run $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{Gen}(1^\lambda)$.
    2. Run $\mathsf{ls}_L^1 \leftarrow \mathcal{V}_L(1^\lambda, \ell_L)$.
    3. Run $\mathsf{ls}_{\mathsf{nm}}^1 \leftarrow \mathcal{V}_{\mathsf{nm}}(1^\lambda, \ell_{\mathsf{nm}})$.
    4. Run $\mathcal{R}$ on input 1^λ and $\mathtt{id}$ thus obtaining nm_1.
    5. Send $(\mathsf{vk}, \mathsf{ls}_L^1, \mathsf{ls}_{\mathsf{nm}}^1, \mathsf{nm}_1)$ to $\mathcal{P}_{\mathsf{NMZK}}$.
2. $\mathcal{P}_{\mathsf{NMZK}} \rightarrow \mathcal{V}_{\mathsf{NMZK}}$
    1. Run $\mathsf{ls}_L^2 \leftarrow \mathcal{P}_L(1^\lambda, \ell_L)$.
    2. Pick $\mathsf{ls}_{\mathsf{nm}}^3 \leftarrow \{0,1\}^\lambda$ run $\mathsf{ls}_{\mathsf{nm}}^2 \leftarrow S_{\mathsf{nm}}(1^\lambda, \mathsf{ls}_{\mathsf{nm}}^1, \mathsf{ls}_{\mathsf{nm}}^3, \ell_{\mathsf{nm}})$.
    3. Pick $s_0 \leftarrow \{0,1\}^\lambda$ and run $\mathcal{S}$ on input 1^λ, $\mathtt{id}$, nm_1, s_0 (in order to commit to the message s_0) thus obtaining nm_2.
    4. Pick a message $\mathtt{msg} \leftarrow \{0,1\}^\lambda$.
    5. Send $(\mathsf{ls}_L^2, \mathsf{ls}_{\mathsf{nm}}^2, \mathtt{msg}, \mathsf{nm}_2)$ 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 $\mathsf{Sign}(\mathsf{sk}, \mathtt{msg})$ to obtain a signature σ of $\mathtt{msg}$.
    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 $\mathsf{Ver}(\mathsf{vk}, \mathtt{msg}, \sigma) \neq 1$ then abort, continue as follows otherwise.
    2. Compute $\mathsf{ls}_L^3 = \mathsf{c} \oplus \mathsf{ls}_{\mathsf{nm}}^3$.
    3. Run $\mathsf{ls}_L^4 \leftarrow \mathcal{P}_L(x, w, \mathsf{ls}_L^3)$.
    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 $\mathsf{ls}_{\mathsf{nm}}^4 \leftarrow S_{\mathsf{nm}}(x_{\mathsf{nm}})$.
    6. Send $(\mathsf{ls}_L^3, \mathsf{ls}_L^4, \mathsf{nm}_4, s_1, \mathsf{ls}_{\mathsf{nm}}^3, \mathsf{ls}_{\mathsf{nm}}^4, x, x_{\mathsf{nm}})$ to $\mathcal{V}_{\mathsf{NMZK}}$.
5. $\mathcal{V}_{\mathsf{NMZK}}$: output 1 iff the following conditions are satisfied.
    1. c is equal to $\mathsf{ls}_L^3 \oplus \mathsf{ls}_{\mathsf{nm}}^3$.
    2. $\mathcal{V}_L(x, \mathsf{ls}_L^1, \mathsf{ls}_L^2, \mathsf{ls}_L^3, \mathsf{ls}_L^4) = 1$.
    3. $\mathcal{V}_{\mathsf{nm}}(x_{\mathsf{nm}}, \mathsf{ls}_{\mathsf{nm}}^1, \mathsf{ls}_{\mathsf{nm}}^2, \mathsf{ls}_{\mathsf{nm}}^3, \mathsf{ls}_{\mathsf{nm}}^4) = 1$.

Figure 2: Formal construction of our delayed-input NMZK.

computes the fourth round as follows[14].

$\mathsf{Sim}_{\mathsf{NMZK}}$ completes the commitment of s_0 by running $\mathcal{S}$ on input nm_3 thus obtaining $(\mathsf{nm}_4, \mathsf{dec}_{\mathsf{nm}})$ and sending nm_4. Furthermore $\mathsf{Sim}_{\mathsf{NMZK}}$ sets s_1 s.t. $s_1 = (\sigma_1||\sigma_2) \oplus s_0$, $x_{\mathsf{nm}} = (\mathsf{vk}, \mathtt{id}, \mathsf{nm}_1, \mathsf{nm}_2, \mathsf{nm}_3, \mathsf{nm}_4, s_1)$, $w_{\mathsf{nm}} = (\mathsf{dec}_{\mathsf{nm}}, s_0, \sigma_1, \mathtt{msg}_1, \sigma_2, \mathtt{msg}_2)$ 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 $\tilde{w}_1, \ldots, \tilde{w}_{\mathsf{poly}(\lambda)}$ 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 $i = 1, \ldots, \mathsf{poly}(\lambda)$).

Lemma 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 $i = 1, \ldots, \mathsf{poly}(\lambda)$. Note that $\{\mathsf{View}^{\mathcal{A}_{\mathsf{NMZK}}}_{\mathcal{H}_0}(1^\lambda, z)\}_{\lambda\in\mathbb{N}, z\in\{0,1\}^*} = \{\mathsf{View}^{\mathcal{A}}_{\mathsf{NMZK}}(1^\lambda, z)\}_{\lambda\in\mathbb{N}, z\in\{0,1\}^*}$.

  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

[14]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 $x_{\mathsf{nm}} \in L_{\mathsf{nm}}$.

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 $\{\mathsf{View}^{\mathcal{A}_{\mathsf{NMZK}}}_{\mathcal{H}_4}(1^\lambda, z)\}_{\lambda\in\mathbb{N}, z\in\{0,1\}^*} = \{S^1(1^\lambda, z)\}_{\lambda\in\mathbb{N}, z\in\{0,1\}^*}$.

  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 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 $(\tilde{x}_i, \tilde{w}_i) \in \mathsf{Rel}_{\mathsf{L}}$ for $i = 1, \ldots, \mathsf{poly}(\lambda)$.*

We now reconsider the hybrid experiments $\mathcal{H}_k$ for $k = 0, \ldots, 4$ described in the security proof of Lemma 1, and prove that they all enjoys an additional property. That is, in the right sessions $\mathcal{A}_{\mathsf{NMZK}}$ never commits, using NM, to a message $\tilde{s}_0$ and sends a value $\tilde{s}_1$ s.t. $\tilde{s}_0 \oplus \tilde{s}_1 = \tilde{\sigma}_1 || \tilde{\sigma}_2$ where $\tilde{\sigma}_1, \tilde{\sigma}_2$ are two signatures for to different messages. Since $\mathcal{A}_{\mathsf{NMZK}}$ does not commit to the signatures then the transcript computed using $\mathsf{LS}_{\mathsf{nm}}$ correspond to a false instance, therefore for the adaptive-input PoK property of Π_{OR}, $\mathcal{A}_{\mathsf{NMZK}}$ in the i-th right session chooses a statement $\tilde{x}_i$ and essentially completes the corresponding transcript of LS_L using the witness $\tilde{w}_i$ s.t. $(\tilde{x}_i, \tilde{w}_i) \in \mathsf{Rel}_{\mathsf{L}}$ for $i \in \{1, \ldots, \mathsf{poly}(\lambda)\}$. For the above chain of implications we are ensured that in all hybrids $\mathcal{A}_{\mathsf{NMZK}}$ uses the witnesses to complete the transcripts of Π^{OR} in the right sessions. Therefore also in the simulated experiment, that corresponds to the last hybrid experiment, the $\mathcal{A}_{\mathsf{NMZK}}$ behavior allows $\mathsf{Sim}_{\mathsf{NMZK}}$ to extract the witness used by $\mathcal{A}_{\mathsf{NMZK}}$ (that is internally executed by $\mathsf{Sim}_{\mathsf{NMZK}}$) using the extractor of Π^{OR} (that exists from the adaptive-PoK property enjoyed by Π^{OR}).

In order to prove that in $\mathcal{H}_0, \ldots, \mathcal{H}_4$ $\mathcal{A}_{\mathsf{NMZK}}$ does not commit to two signatures in any of the right sessions we rely on the "mild" non-malleability and the honest-extraction property enjoyed by NM. More precisely, in each hybrid experiment, we use the honest-extraction[15] property to extract the signatures from the right sessions (that by contradiction are committed using NM). During the proof we need to show that the rewinds made by the honest-extractor 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 extractor of NM do not rewind the challenger involved in the reduction. In particular, consider the hybrid $\mathcal{H}_3$ where

[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.

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 two hybrid experiments in order to prove that $\mathcal{A}_{\mathsf{NMZK}}$ does not commit to the signatures we rely on the adaptive-input Special HVZK and the rewinds 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. It is not clear how to do such a security proof by directly relying on the WI property of Π_{OR}. The formal proof for this lemma can be found in App. D. □

Theorem 2. *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 1. 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 1 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 1.

□

4 Multi-Party Coin-Tossing Protocol

4.1 4-Round Secure Multi-Party Coin Tossing: Π_{MPCT}

The high-level idea of our protocol Π_{MPCT} significantly differs from the one of [GMPP16b] (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 [GMPP16b] 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 $L = \{\mathtt{com} : \exists\ (\mathtt{dec}, m)$ 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 (Sec. C.2 for more details). Π_{OR} is a proof system for the $\mathcal{NP}$-language $L_{\mathtt{com}} = \{(\mathtt{com}_0, \mathtt{com}_1) : \exists\ (\mathtt{dec}, m)$ s.t. $\mathsf{Dec}(\mathtt{com}_0, \mathtt{dec}, m) = 1$ 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

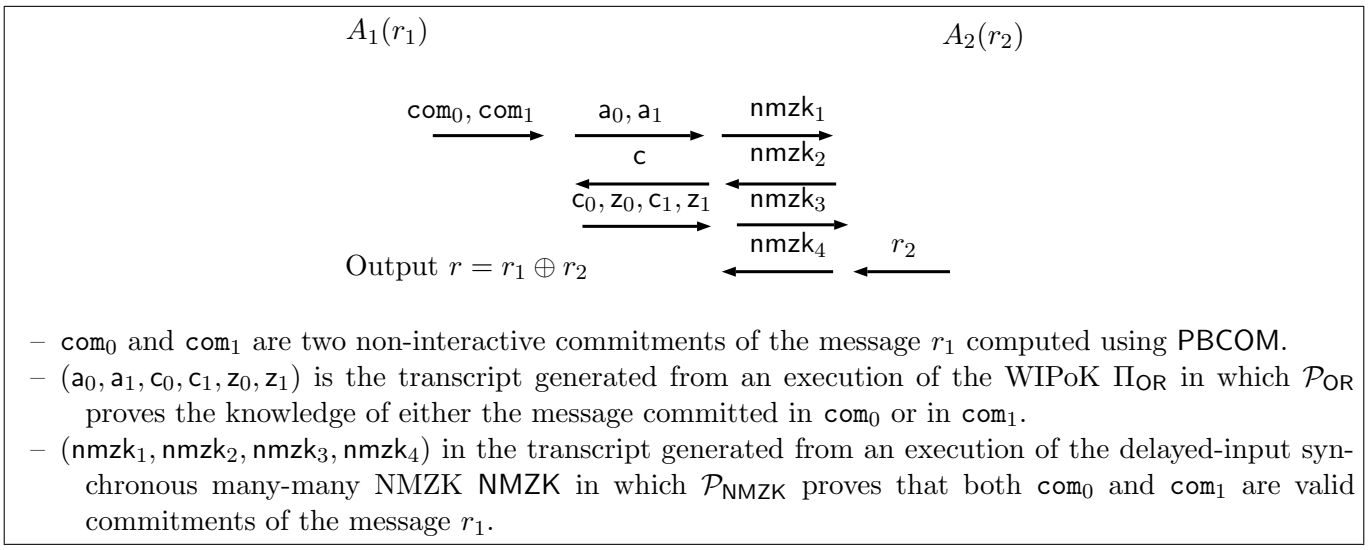
$$L_{\mathsf{NMZK}} = \{((\mathtt{com}_0, \mathtt{com}_1), m) :\ \forall i \in \{0,1\}\ \exists\ \mathtt{dec}_i \text{ s.t. } \mathsf{Dec}(\mathtt{com}_i, \mathtt{dec}_i, m) = 1\}.$$

[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}.

Informally, by running NMZK, one can prove that 2 commitments contain the same message m.

4.2 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. 3. For a formal description of Π_{MPCT} we refer the reader to Sec. 4.3. In Fig. 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.



- com_0 and 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 com_0 or in 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 com_0 and com_1 are valid commitments of the message r_1.

Figure 3: Π_{MPCT}: Informal description of the execution from A_1 to A_2. The execution from A_2 to A_1 is symmetric.

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 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 Formal Description

Let $P = \{P_1, \ldots, P_n\}$ be the set of parties. Furthermore, denote by $(\mathtt{id}_1, \ldots, \mathtt{id}_n)$[17] the unique identities of parties $\{P_1, \ldots, P_n\}$, respectively. Let us denote by $\mathsf{F}_{\mathsf{MPCT}} : (1^\lambda)^n \to \{0,1\}^\lambda$ the function $\mathsf{F}_{\mathsf{MPCT}}(r_1, \ldots, r_n) = r_1 \oplus \cdots \oplus r_n$. The protocol starts with each party P_i choosing a random string r_i for $i = 1, \ldots, n$. 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 [GMPP16b] 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 $l = 1, \ldots, 4$.

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 $\tilde{m}_1 = (\tilde{\mathtt{com}}_0, \tilde{\mathtt{com}}_1, \tilde{\mathsf{a}}_0, \tilde{\mathsf{a}}_1, \tilde{\mathsf{nmzk}}_1)$.
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 $\tilde{m}_2 = (\tilde{\mathsf{c}}, \tilde{\mathsf{nmzk}}_2)$.
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 $\tilde{m}_3 = (\tilde{\mathsf{nmzk}}_3, \tilde{\mathsf{c}}_0, \tilde{\mathsf{c}}_1, \tilde{\mathsf{z}}_0, \tilde{\mathsf{z}}_1)$.
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.

[17] As discuss in the Definition 1 the use of the identifiers can be avoid, we use them, to uniformity of notation.

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 $\tilde{m}_4 = (r_1, \tilde{x}_{\mathsf{NMZK}}, \tilde{\mathsf{nmzk}}_4)$.

Output computation of Π_{MPCT}. Check, for each party, if $(\mathsf{nmzk}_1^i, \mathsf{nmzk}_2^i, \mathsf{nmzk}_3^i, \mathsf{nmzk}_4^i)$ is accepting for $\mathcal{V}_{\mathsf{NMZK}}$ with respect to the instance x_{NMZK}^i $(i = 1, \ldots, n)$ 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 3. *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.*

The formal security proof can be found in App. E.

5 Acknowledgments

We thank Giuseppe Persiano and Alessandra Scafuro for several discussions on delayed-input protocols.

Research supported in part by "GNCS - INdAM", EU COST Action IC1306, NSF grant 1619348, DARPA, US-Israel BSF grant 2012366, OKAWA Foundation Research Award, IBM Faculty Research Award, Xerox Faculty Research Award, B. John Garrick Foundation Award, Teradata Research Award, and Lockheed-Martin Corporation Research Award. The views expressed are those of the authors and do not reflect position of the Department of Defense or the U.S. Government.

The work of 1st, 3rd and 4th authors has been done in part while visiting UCLA.

References

[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.

[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.

[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.

[Blu86] Manuel Blum. How to prove a theorem so no one else can claim it. In *In Proceedings of the International Congress of Mathematicians*, pages 1444–1454, 1986.

[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.

[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.

[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. 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.

[COSV17b] 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.

[CPS13] Kai-Min Chung, Rafael Pass, and Karn Seth. Non-black-box simulation from one-way functions and applications to resettable security. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 231–240. ACM, 2013.

[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.

[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.

[GKP+17] Vipul Goyal, Ashutosh Kumar, Sunoo Park, Silas Richelson, and Akshayaram Srinivasan. New constructions of non-malleable commitments and applications. Private communication, 2017.

[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.

[GMPP16a] Sanjam Garg, Pratyay Mukherjee, Omkant Pandey, and Antigoni Polychroniadou. Personal communication, August 2016.

[GMPP16b] 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.

[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.

[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.

[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.

[Lin10] Yehuda Lindell. Foundations of cryptography 89-856. http://u.cs.biu.ac.il/~lindell/89-856/complete-89-856.pdf, 2010.

[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.

[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 non-malleability. 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.

[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.

[OV12] Rafail Ostrovsky and Ivan Visconti. Simultaneous resettability from collision resistance. *Electronic Colloquium on Computational Complexity (ECCC)*, 19:164, 2012.

[Pas04] 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.

[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.

[PR05] 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.

[PR08] Rafael Pass and Alon Rosen. New and improved constructions of nonmalleable cryptographic protocols. *SIAM J. Comput.*, 38(2):702–752, 2008.

[PW09] Rafael Pass and Hoeteck Wee. Black-box constructions of two-party protocols from one-way functions. In *Theory of Cryptography, 6th Theory of Cryptography Conference, TCC 2009, San Francisco, CA, USA, March 15-17, 2009. Proceedings*, pages 403–418, 2009.

[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.

[SCO+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.

A Standard Definitions

Definition 2 (One-way function (OWF))**.** *A function $f : \{0,1\}^\star \to \{0,1\}^\star$ is called one way if the following two conditions hold:*

- *there exists 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 1-to-1 OWF *if $f(x) \neq f(y)\ \forall (x,y) \in \{0,1\}^\star$.*

Definition 3 (Following the notation of [CPS13])**.** *A triple of* PPT *algorithms* $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ *is called a* signature scheme *if it satisfies the following properties.*

Validity: *For every pair $(s,v) \leftarrow \mathsf{Gen}(1^\lambda)$, and every $m \in \{0,1\}^\lambda$, we have that*

$$\mathsf{Ver}(v, m, \mathsf{Sign}(s, m)) = 1.$$

Security: *For every* PPT *$\mathcal{A}$, there exists a negligible function ν, such that for all auxiliary input $z \in \{0,1\}^\star$ it holds that:*

$$\Pr[(s,v) \leftarrow \mathsf{Gen}(1^\lambda); (m,\sigma) \leftarrow \mathcal{A}^{\mathsf{Sign}(s,\cdot)}(z,v) \wedge \mathsf{Ver}(v,m,\sigma) = 1 \wedge m \notin Q] < \nu(\lambda)$$

where Q denotes the set of messages whose signatures were requested by $\mathcal{A}$ to the oracle $\mathsf{Sign}(s,\cdot)$.

Definition 4 (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 exists a negligible function ν such that for sufficiently large $\lambda \in \mathbb{N}$,*

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

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 paper we also use the definition of *Statistical Indistinguishability*. This definition is the same as Definition 4 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.

Definition 5 (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:*

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

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.

Definition 6 (Proof/argument system)**.** *A pair of* PPT *interactive algorithms $\Pi = (\mathcal{P}, \mathcal{V})$ constitute 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 $(x, w) \in \mathsf{Rel}_{\mathsf{L}}$, it holds that:*

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

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:*

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

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 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]. An interactive protocol $\Pi = (\mathcal{P}, \mathcal{V})$ is *public coin* if, at every round, $\mathcal{V}$ simply tosses a predetermined number of coins (i.e. a random challenge) and sends the outcome to the prover. Moreover we say that the transcript τ of an execution $b = \langle \mathcal{P}(z), \mathcal{V}\rangle(x)$ is *accepting* if $b = 1$.

Definition 7 (Proof of Knowledge [LP11b])**.** *A protocol $\Pi = (\mathcal{P}, \mathcal{V})$ that enjoys completeness is a* proof of knowledge *(PoK) for the relation $\mathsf{Rel}_{\mathsf{L}}$ if there exists a probabilistic expected polynomial-time machine E, called the extractor, such that for every algorithm $\mathcal{P}^\star$, there exists a negligible function ν, every statement $x \in \{0,1\}^\lambda$, every randomness $r \in \{0,1\}^\star$ and every auxiliary input $z \in \{0,1\}^\star$,*

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

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 our security proofs we make use of the following observation. An interactive protocol Π that enjoys the property of completeness and PoK (AoK) is a proof (an argument) system. Indeed suppose by contradiction that is not. By the definition of PoK (AoK) it is possible to extract the witness for every theorem $x \in \{0,1\}^\lambda$ proved by $\mathcal{P}^\star_r$ with probability greater than $\text{Prob}\,[\,\langle \mathcal{P}^\star_r(z), \mathcal{V}\rangle(x) = 1\,]$; contradiction.

In this paper we also consider the *adaptive-input* PoK/AoK property for all the protocols that enjoy delayed-input completeness. Adaptive-input PoK/AoK ensures that the PoK/AoK property still holds when a malicious prover can choose the statement adaptively at the last round.

A *3-round protocol* $\Pi = (\mathcal{P}, \mathcal{V})$ for a relation $\mathsf{Rel}_{\mathsf{L}}$ is an interactive protocol played between a prover $\mathcal{P}$ and a verifier $\mathcal{V}$ on common input x and private input w of $\mathcal{P}$ s.t. $(x, w) \in \mathsf{Rel}_{\mathsf{L}}$. In a 3-round protocol the first message a and the third message z are sent by $\mathcal{P}$ and the second messages c is played by $\mathcal{V}$. At the end of the protocol $\mathcal{V}$ decides to accept or reject based on the data that he has seen, i.e. $x, \mathsf{a}, \mathsf{c}, \mathsf{z}$.

We usually denote the message c sent by $\mathcal{V}$ as a *challenge*, and as *challenge length* the number of bit of c.

Definition 8 (Σ-Protocol)**.** *A 3-round public-coin protocol $\Pi = (\mathcal{P}, \mathcal{V})$ for a relation $\mathsf{Rel}_{\mathsf{L}}$ is a Σ-Protocol if the following properties hold:*

- *Completeness: if $(\mathcal{P}, \mathcal{V})$ follow the protocol on input x and private input w to $\mathcal{P}$ s.t. $(x, w) \in \mathsf{Rel}_{\mathsf{L}}$, $\mathcal{V}$ always accepts.*
- *Special soundness: if there exists a polynomial time algorithm such that, for any pair of accepting transcripts on input x, $(\mathsf{a}, \mathsf{c}_1, \mathsf{z}_1), (\mathsf{a}, \mathsf{c}_2, \mathsf{z}_2)$ where $\mathsf{c}_1 \neq \mathsf{c}_2$, outputs witness w such that $(x, w) \in \mathsf{Rel}_{\mathsf{L}}$.*
- *Special Honest Verifier Zero-knowledge (Special HVZK): there exists a PPT simulator algorithm Sim that for any $x \in L$, security parameter λ and any challenge c works as follow: $(\mathsf{a}, \mathsf{z}) \leftarrow \mathsf{Sim}(1^\lambda, x, \mathsf{c})$. Furthermore, the distribution of the output of 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 such that $(x, w) \in \mathsf{Rel}_{\mathsf{L}}$*[18]*.*

[18]Note that we require that the two transcripts are computationally indistinguishable as in [GMY06], instead of following [CDS94] that requires the perfect indistinguishability between the two transcripts.

Definition 9. *A delayed-input 3-round protocol* $\Pi = (\mathcal{P}, \mathcal{V})$ *for relation* $\mathsf{Rel}_{\mathsf{L}}$ *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* $(x_1, w_1) \in \mathsf{Rel}_{\mathsf{L}}$ *and* $(x_2, w_2) \in \mathsf{Rel}_{\mathsf{L}}$.

Definition 10. *A delayed-input 3-round protocol* $\Pi = (\mathcal{P}, \mathcal{V})$ *for relation* $\mathsf{Rel}_{\mathsf{L}}$ *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. $\mathsf{z} \leftarrow \mathsf{Sim}(x, \rho)$[19]*, 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* $(x, w) \in \mathsf{Rel}_{\mathsf{L}}$.

A.1 Commitment Schemes

Definition 11 (Commitment Scheme)**.** *Given a security parameter* 1^λ*, a commitment scheme* $\mathtt{CS} = (\mathsf{Sen}, \mathsf{Rec})$ *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* $\mathtt{com}$*, and the private output* $\mathtt{d}$ *of* Sen.

In the decommitment phase, Sen *sends to* Rec *a decommitment information* $(m, \mathtt{d})$ *such that* Rec *accepts* m *as the decommitment 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 *given as output 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*[20]*.* Rec *on input* m *and* $\mathtt{d}$ *accepts* m *as decommitment of* $\mathtt{com}$.

Statistical (resp. Computational) Hiding([Lin10]): *for any adversary (resp.* 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 adversary (resp. PPT *adversary)* $\mathcal{A}$*, there exist a negligible function* ν*, s.t.:*

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

Statistical (resp. Computational) Binding: *for every commitment* $\mathtt{com}$ *generated during the commitment phase by a possibly malicious unbounded (resp. malicious* PPT*) sender* $\mathsf{Sen}^\star$ *there exists a negligible function* ν *such that* $\mathsf{Sen}^\star$*, with probability at most* $\nu(\lambda)$*, outputs two decommitments* $(m_0, \mathtt{d}_0)$ *and* $(m_1, \mathtt{d}_1)$*, with* $m_0 \neq m_1$*, such that* Rec *accepts both decommitments.*

We also say that a commitment scheme is perfectly binding *iff* $\nu(\lambda) = 0$.

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

[20] In this paper we consider a non-interactive decommitment phase only.

When a commitment scheme $(\mathsf{Com}, \mathsf{Dec})$ *is non-interactive, to not overburden the notation, we use the following notation.*

- *Commitment phase.* $(\mathtt{com}, \mathtt{dec}) \leftarrow \mathsf{Com}(m)$ *denotes that* $\mathtt{com}$ *is the commitment of the message* m *and* $\mathtt{dec}$ *represents the corresponding decommitment information.*
- *Decommitment phase.* $\mathsf{Dec}(\mathtt{com}, \mathtt{dec}, m) = 1$.

A.2 3-Round Honest-Extractable Commitment Schemes

Informally, a 3-round commitment scheme is honest-extractable if there exists an efficient extractor that having black-box access to any efficient honest sender that successfully performs the commitment phase, outputs the only committed string that can be successfully decommitted. We give now a definition that follows the one of [PW09].

Definition 12 (Honest-Extractable Commitment Scheme)**.** *A perfectly (resp. statistically) binding commitment scheme* $\mathtt{ExCS} = (\text{ExSen}, \text{ExRec})$ *is an* honest-extractable commitment scheme *if there exists an expected* PPT *extractor* $\mathtt{ExtCom}$ *that given oracle access to any honest sender* ExSen*, outputs a pair* (τ, m) *such that the following two 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).*

A.3 Non-Malleable Commitments

A commitment scheme involves two players: sender and receiver. Informally, it consists of two phases, a commitment phase and a decommitment phase. In the commitment phase the sender, with a secret input m, interacts with the receiver. In the end of this interaction we say that a *commitment* of the message m has been computed. Moreover the receiver still does not know what m is (i.e. m is hidden) and at the same time the sender can subsequently (i.e., during the decommitment phase) open this commitment only to m (see Def. 11 for a formal definition of commitment scheme).

In order to define a non-malleable commitment we follow [LPV08, LPV09]. Let $\Pi = (\mathsf{Sen}, \mathsf{Rec})$ be a statistically binding commitment scheme. And let λ be the security parameter. Consider a MiM adversary $\mathcal{A}$ that, on auxiliary input z participates in a left and a right session. In the left sessions the MiM adversary $\mathcal{A}$ interacts with Sen receiving commitment to value m using an identity $\mathtt{id}$ of its choice. In the right session $\mathcal{A}$ interacts with Rec attempting to commit to a related value $\tilde{m}$ again using identity of its choice $\tilde{\mathtt{id}}$. If the right commitment is invalid, or undefined, its value is set to $\perp$. Furthermore, if $\tilde{\mathtt{id}} = \mathtt{id}$ then $\tilde{m}$ is also set to $\perp$ (i.e., a commitment where the adversary uses the same identity of the honest senders is considered invalid). Let $\mathsf{mim}_{\Pi}^{\mathcal{A},m}(z)$ denote a random variable that describes the values $\tilde{m}$ and the view of $\mathcal{A}$ in the above experiment.

Definition 13. *[Non-malleable commitment scheme [LPV08, LPV09]] A commitment scheme is* non-malleable with respect to commitment *if, for every* PPT *MiM adversary $\mathcal{A}$, for every $m_0 \in \{0,1\}^{\mathsf{poly}(\lambda)}$ and $m_1 \in \{0,1\}^{\mathsf{poly}(\lambda)}$ the following holds*

$$\{\mathsf{mim}_{\Pi}^{\mathcal{A},m_0}(z)\}_{z\in\{0,1\}^\star} \approx \{\mathsf{mim}_{\Pi}^{\mathcal{A},m_1}(z)\}_{z\in\{0,1\}^\star}.$$

We say that a commitment is valid or well formed if it admits a decommitment to a message $m \neq \perp$.

For our propose we use a 4-round synchronous honest-extractable non-malleable commitment. That is, a commitment 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 one-to-one OWFs).

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 we do not require any form of extractability against malicious senders.

B Definition of Secure Computation

Here we recall some useful definitions for our application. Our Multi-Party Computation (MPC) protocol for coin tossing is secure in the same model used in [GMPP16b], therefore some definitions are taken almost verbatim from [GMPP16b]. 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]).

B.1 Two-Party Computation

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 $F : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^* \times \{0,1\}^*$ 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 behaviour. Loosely speaking, the aim of a secure two-party protocol is to protect an honest party against dishonest behaviour by the other party. In this paper, 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[21] where the communication is open and delivery of messages is not guaranteed. For simplicity, 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

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

$\perp$. 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 14 (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:*

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

C Special WIPoK

C.1 Improving the Soundness of LS

In this section we consider the 3-round WIPoK for the $\mathcal{NP}$-complete language of graph Hamiltonicity (HC), provided in [LS90], and we will refer to this construction as the *LS protocol*. An interesting property of this WIPoK is that only the size of the statement need to be known before the last round by both the prover and the verifier. 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 (see Definition 9).

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 $\mathsf{ls}^1_0 \leftarrow \mathcal{P}(1^\lambda, \kappa; \rho_0)$, $\mathsf{ls}^1_1 \leftarrow \mathcal{P}(1^\lambda, \kappa; \rho_1)$ 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 $\mathsf{ls}^3_0 \leftarrow \mathcal{P}(G, w, b, \rho_0)$, $\mathsf{ls}^3_1 \leftarrow \mathcal{P}(G, w, 1-b, \rho_1)$ and sends $(\mathsf{ls}^3_0, \mathsf{ls}^3_1)$.
4. $\mathcal{V}^{\mathsf{imp}}$ accepts iff $\mathcal{V}(G, \mathsf{ls}^1_0, b, \mathsf{ls}^3_0) = 1$ and $\mathcal{V}(G, \mathsf{ls}^1_1, 1-b, \mathsf{ls}^3_1) = 1$.

Theorem 4. *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 $\mathsf{LS}^{\mathsf{imp}}$: $\big((\mathsf{ls}_0, \mathsf{ls}_1), 0, (\mathsf{ls}^3_0, \mathsf{ls}^3_1)\big)$ for the statement G and $\big((\mathsf{ls}_0, \mathsf{ls}_1), 1, (\mathsf{ls}^{3'}_1, \mathsf{ls}^{3'}_1)\big)$ for the statement G'. We can isolate the sub-transcripts $(\mathsf{ls}_0, 0, \mathsf{ls}^3_0)$ and $(\mathsf{ls}_0, 1, \mathsf{ls}^{3'}_0)$ and observe that $\mathcal{V}(G, \mathsf{ls}^1_0, 0, \mathsf{ls}^3_0) = 1 = \mathcal{V}(G' \mathsf{ls}^1_0, 1, \mathsf{ls}^{3'}_0)$.

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's 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 10). 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 $\mathsf{ls}_b^1 \leftarrow S(1^\lambda, b, \kappa; \rho_b)$, $\mathsf{ls}_{1-b}^1 \leftarrow S(1^\lambda, 1-b, \kappa; \rho_{1-b})$ and outputs $(\mathsf{ls}_b^1, \mathsf{ls}_{1-b}^1)$.
2. S^{imp}, on input the graph G, ρ_0 and ρ_1 computes $\mathsf{ls}_b^3 \leftarrow S(G, \rho_b)$, $\mathsf{ls}_{1-b}^3 \leftarrow S(G, \rho_{1-b})$ and outputs $(\mathsf{ls}_b^3, \mathsf{ls}_{1-b}^3)$.

The transcript $\big((\mathsf{ls}_b^1, \mathsf{ls}_{1-b}^1), b, (\mathsf{ls}_b^3, \mathsf{ls}_{1-b}^3)\big)$ 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 5. *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 6. *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 paper we will omit the $\mathcal{NP}$ reduction therefore assuming that the above scheme works directly on a given $\mathcal{NP}$-language L.

C.2 Combining (adaptive-input) Special HVZK PoK Through [CDS94]

In our paper we use the well known technique for composing two Σ-protocols to compute the OR for compound statement [CDS94, GMY06]. In more details, let $\Pi_0 = (\mathcal{P}_0, \mathcal{V}_0)$ and $\Pi_1 = (\mathcal{P}_1, \mathcal{V}_1)$ be Σ-protocols for the respective $\mathcal{NP}$-relation $\mathsf{Rel}_{\mathsf{L}_0}$ (with Special HVZK simulator Sim_0) and $\mathsf{Rel}_{\mathsf{L}_1}$ (with Special

HVZK simulator Sim_1). Then it is possible to use Π_0 and Π_1 to construct $\Pi^{\mathsf{OR}} = (\mathcal{P}_{\mathsf{OR}}, \mathcal{V}_{\mathsf{OR}})$ for relation $\mathsf{Rel}_{\mathsf{OR}} = \{((x_0, x_1), w) : ((x_0, w) \in \mathsf{Rel}_{\mathsf{L}_0}) \text{ OR } ((x_1, w) \in \mathsf{Rel}_{\mathsf{L}_1})\}$ that works as follows.

Protocol $\Pi^{\mathsf{OR}} = (\mathcal{P}_{\mathsf{OR}}, \mathcal{V}_{\mathsf{OR}})$: Let w_b with $b \in \{0,1\}$ be s.t. $(x_b, w_b) \in \mathsf{Rel}_{\mathsf{L}_b}$. $\mathcal{P}_{\mathsf{OR}}$ and $\mathcal{V}_{\mathsf{OR}}$ on common input (x_0, x_1) and private input w_b compute the following steps.

- $\mathcal{P}_{\mathsf{OR}}$ computes $\mathsf{a}_b \leftarrow \mathcal{P}_b(1^\lambda, x_b, w_b)$. Furthermore he picks $\mathsf{c}_{1-b} \leftarrow \{0,1\}^\lambda$ and computes $(\mathsf{a}_{1-b}, \mathsf{z}_{1-b}) \leftarrow \mathsf{Sim}_{1-b}(1^\lambda, x_{1-b}, \mathsf{c}_{1-b})$. $\mathcal{P}_{\mathsf{OR}}$ sends $\mathsf{a}_0, \mathsf{a}_1$ to $\mathcal{V}_{\mathsf{OR}}$.
- $\mathcal{V}_{\mathsf{OR}}$, upon receiving $\mathsf{a}_0, \mathsf{a}_1$ picks $\mathsf{c} \leftarrow \{0,1\}^\lambda$ and sends c to $\mathcal{P}_{\mathsf{OR}}$.
- $\mathcal{P}_{\mathsf{OR}}$, upon receiving c computes $\mathsf{c}_b = \mathsf{c}_{1-b} \oplus \mathsf{c}$ and computes $\mathsf{z}_b \leftarrow \mathcal{P}_b(\mathsf{c}_b)$. $\mathcal{P}_{\mathsf{OR}}$ sends $\mathsf{c}_0, \mathsf{c}_1, \mathsf{z}_0\ \mathsf{z}_1$ to $\mathcal{V}_{\mathsf{OR}}$.
- $\mathcal{V}_{\mathsf{OR}}$ checks that the following conditions holds: $\mathsf{c} = \mathsf{c}_0 \oplus \mathsf{c}_1$, $\mathcal{V}_0(x_0, \mathsf{a}_0, \mathsf{c}_0, \mathsf{z}_0) = 1$ and $\mathcal{V}_1(x_1, \mathsf{a}_1, \mathsf{c}_1, \mathsf{z}_1) = 1$. If all the checks succeed then outputs 1, otherwise outputs 0.

Theorem 7. *([CDS94]) Let Σ_0 and Σ_1 be two Σ-protocols, then $\Pi^{\mathsf{OR}} = (\mathcal{P}_{\mathsf{OR}}, \mathcal{V}_{\mathsf{OR}})$ is a Σ-protocol for $\mathsf{Rel}_{\mathsf{L}_{\mathsf{OR}}}$.*

Theorem 8. *([Dam10]) Let $\Pi = (\mathcal{P}, \mathcal{V})$ be a Σ-protocol for relation $\mathsf{Rel}_{\mathsf{L}}$ with negligible soundness error[22], then Π is a proof of knowledge for $\mathsf{Rel}_{\mathsf{L}}$.*

In our work we instantiate Π^{OR} using as Π_0 and Π_1 the Blum's protocol [Blu86] for the $\mathcal{NP}$-complete language for graph Hamiltonicity (that also is a Σ-Protocol). Therefore Th. 7 (and Th. 8) 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 9. *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 $\mathsf{Rel}_{\mathsf{L}_{\mathsf{OR}}}$.*

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 $\big((\pi_0, \pi_1), \pi^{2\prime}, (\pi_0^{2\prime}, \pi_0^{3\prime}, \pi_1^{2\prime} \pi_1^{3\prime})\big)$ for the statement (x_0', x_1'), where $\pi^2 \neq \pi^{2\prime}$. We observe that since $\pi^2 \neq \pi^{2\prime}$, $\pi^2 = \pi_0^2 \oplus \pi_1^2$ and $\pi^{2\prime} = \pi_0^{2\prime} \oplus \pi_1^{2\prime}$ it holds that either $\pi_0^2 \neq \pi_0^{2\prime}$ or $\pi_1^2 \neq \pi_1^{2\prime}$. Suppose w.l.o.g. that $\pi_0^2 \neq \pi_0^{2\prime}$. 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 $(\pi_0, \pi_0^{2\prime}, \pi_0^{3\prime})$ 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 $\pi_1^2 \neq \pi_1^{2\prime}$. □

Using a result of [CPS+16b] we can claim the following theorem.

Theorem 10. *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 paper 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 paper 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).

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

In the rest of the paper, 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].

D Formal Proof of Lemma 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 $\text{Prob}[\, b = b' \,] - 1/2$ 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 $\text{Prob}[\, b = b' \,] - 1/2$ 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 $k = 0, \ldots, 4$ described in the security proof of Lemma 1, and we define an event SIGNM_k foe each of these hybrids, details follow. For $k = 0, \ldots, 4$ SIGNM_k is the event that the i-th right session of $\mathcal{H}_k(1^\lambda, z)$ $\mathcal{A}_{\mathsf{NMZK}}$ successfully 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 $i \in \{1, \ldots, \mathsf{poly}(\lambda)\}$. Furthermore the i-th right session is accepting and $\mathtt{id} \neq \tilde{\mathtt{id}}_i$ for some $i \in \{1, \ldots, \mathsf{poly}(\lambda)\}$. We will show that the probability of SIGNM_k is negligible.

We start by reconsidering the hybrid experiment $\mathcal{H}_0(1^\lambda, z)$.

Claim 1. $\text{Prob}[\, \mathsf{SIGNM}_0 \,] < \nu(\lambda)$ *for some negligible function* ν.

Suppose by contradiction that the right session where $\mathcal{A}_{\mathsf{NMZK}}$ commits to the signatures is the i-th right session (with $i \in \{1, \ldots, \mathsf{poly}(\lambda)\}$), then we can construct an adversary $\mathcal{A}_{\mathbf{\Sigma}}$ that breaks the security of the signature scheme $\mathbf{\Sigma}$. Let $\tilde{\mathsf{vk}}$ be the challenge verification key. The adversary $\mathcal{A}_{\mathbf{\Sigma}}$ 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 $\mathcal{A}_{\mathbf{\Sigma}}$ 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.

The next hybrid that we reconsider is $\mathcal{H}_1(1^\lambda, z)$. We know from the proof of Lemma 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 $\text{Prob}[\, \mathsf{SIGNM}_1 \,] < \nu(\lambda)$ for some negligible function ν.

The next hybrid that we reconsider is $\mathcal{H}_2(1^\lambda, z)$. To prove that the probability of the event SIGNM_2 is negligible 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.

Claim 2. $\text{Prob}[\,\mathsf{SIGNM}_2\,] < \nu(\lambda)$ *for some negligible function* ν.

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 $i \in \{1, \ldots, \mathsf{poly}(\lambda)\}$). 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 $i \in \{1, \ldots, \mathsf{poly}(\lambda)\}$), 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 of Claim 2). 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 1.

The next hybrid that we reconsider is $\mathcal{H}_3(1^\lambda, z)$. Also, in this hybrid we show that the probability of the event SIGNM_3 is negligible, otherwise we can break the multi-SHVZK of $\mathsf{LS}_{\mathsf{nm}}$. In more details we prove the following claim.

Claim 3. $\mathrm{Prob}[\ \mathsf{SIGNM}_3\ ] < \nu(\lambda)$ *for some negligible function* ν.

Suppose by contradiction that the right session where $\mathcal{A}_{\mathsf{NMZK}}$ commits to the signatures is the i-th right session (with $i \in \{1, \ldots, \mathsf{poly}(\lambda)\}$), 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. If $\mathcal{A}_{\mathsf{SHVZK}}$ 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 $\mathcal{C}_{\mathsf{SHVZK}}$ 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 $\mathsf{c}^1, \mathsf{c}^2, \ldots, \mathsf{c}^{\mathsf{poly}(\lambda)}$ are received, $\mathcal{A}_{\mathsf{SHVZK}}$ just computes $\mathsf{ls}^{3\,\prime}_L = \mathsf{ls}^3_{\mathsf{nm}} \oplus \mathsf{c}^i$ for $i = 1, \ldots, \mathsf{poly}(\lambda)$ 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 1.

The next hybrid that we reconsider is $\mathcal{H}_4(1^\lambda, z)$. Also, in this hybrid we show that the probability of the event SIGNM_4 is negligible, otherwise we can break the multi-SHVZK of LS_L. In more details we prove the following claim.

Claim 4. $\text{Prob}[\,\mathsf{SIGNM}_4\,] < \nu(\lambda)$ *for some negligible function* ν.

The security proof is almost equal to the security proof of Claim 3.

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. From Claim 4 follows that, in the right sessions, $\mathcal{A}_{\mathsf{NMZK}}$ never commits (except with negligible probability), using NM, to a message $\tilde{s}_0$ and sends a value $\tilde{s}_1$ s.t. $\tilde{s}_0 \oplus \tilde{s}_1 = \tilde{\sigma}_1 || \tilde{\sigma}_2$ where $\tilde{\sigma}_1, \tilde{\sigma}_2$ are two signatures for to different messages. Since $\mathcal{A}_{\mathsf{NMZK}}$ does not commit to the signatures then the transcript computed using $\mathsf{LS}_{\mathsf{nm}}$ corresponds to a false instance, therefore for the adaptive-input PoK property of Π_{OR}, $\mathcal{A}_{\mathsf{NMZK}}$ in the i-th right session chooses a statement $\tilde{x}_i$ and essentially completes the corresponding transcript of LS_L using the witness $\tilde{w}_i$ s.t. $(\tilde{x}_i, \tilde{w}_i) \in \mathsf{Rel}_{\mathsf{L}}$ for $i = 1, \ldots, \mathsf{poly}(\lambda)$. For the above sequence of implications we are ensured that in the simulated experiment $\mathcal{A}_{\mathsf{NMZK}}$ uses the witnesses to complete the transcripts of LS_L in 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 LS_L (that exists for the adaptive-PoK property enjoyed by LS_L).

This observations conclude the proof.

E Formal Proof of Theorem 3

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 $\mathcal{H}_1$[23].

[23] 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 [GMPP16b, 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) 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 7, 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 a 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 $\mathtt{com}_1$[24].
- 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 $\mathcal{H}_5$.

  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 $\mathsf{c}^1, \ldots, \mathsf{c}^{\mathsf{poly}(\lambda)}$ 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

[24] 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.

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 functionality thus obtaining a value r and completes the 4th round of $\mathcal{H}_7$ setting $r_1 = r \oplus \cdots \oplus r_n^*$.

□