Constant-Round Concurrent NMWI and its relation to NMZK

Rafail Ostrovsky* Giuseppe Persiano† Ivan Visconti†

February 28, 2007

Abstract

One of the central questions in Cryptography is to design round-efficient protocols that are secure under man-in-the-middle attacks. In this paper we introduce and study the notion of non-malleable witness indistinguishability (NMWI) and examine its relation with the classic notion of non-malleable zero knowledge (NMZK). Indeed, despite tremendous applicability of witness indistinguishability, while a lot of attention has been given to NMZK, very little attention has been given to witness indistinguishability in case of man-in-the-middle attacks. We initiate this study, with several (perhaps somewhat surprising) results:

- We give the first definition of NMWI proof systems. Just like every NMZK proof is a zero-knowledge proof which aims to attain a very strong *proof independence* property, we require (and formalize) the notion that every NMWI proof is a witness indistinguishable proof system which enjoys a very strong *witness independence* property against any man-in-the-middle attack.
- We show the existence of a constant-round NMWI argument system for $\mathbb{NP}$ in the standard model (i.e. without any trusted or any other setup assumptions).
- It is known that *every* zero-knowledge (ZK) argument is also a *witness indistinguishable* (WI) argument, but not vice-versa, i.e. ZK $\subsetneq$ WI. Rather surprisingly, we show that NMWI and NMZK argument systems are *incomparable*. That is, we show that there exists a NMZK argument system that is not a NMWI argument system and we also show that there is a NMWI argument system that is not a NMZK argument system.
- We show that our constant-round NMWI argument system is also secure under a concurrent man-in-the-middle attack, i.e., it is a concurrent constant-round NMWI argument system. This is somewhat surprising since the question of a constant-round concurrent NMZK argument system is still open.
- We then turn our attention to Bare Public-Key (BPK) model. We show how to expand upon our concurrent NMWI result in the plain model to obtain a constant-round concurrent NMZK argument system for any $\mathbb{NP}$ language in the BPK model.

Keywords: zero knowledge, witness indistinguishability, non-malleability, concurrent composition.

The work of the authors has been supported in part by the European Commission through the IST program under Contract IST-2002-507932 ECRYPT. The work of the first author has also been supported in part by Intel equipment grant, NSF Cybertrust grant No. 0430254, Xerox Innovation group Award and IBM Faculty Award. The work of the last two authors has also been supported in part by the European Commission through the FP6 program under contract FP6-1596 AEOLUS.

*UCLA, Los Angeles, CA 90095, USA. rafail@cs.ucla.edu

†Dipartimento di Informatica ed Applicazioni, Università di Salerno, 84084 Fisciano (SA), Italy. {giuper,visconti}@dia.unisa.it

Contents

1 Introduction

Interactive proof systems play a central role in cryptography. Starting with the seminal paper of Goldwasser, Micali and Rackoff [23], the notion of zero knowledge and the simulation paradigm have been adopted in order to prove security of interactive proof systems. Indeed, Goldreich, Micali and Wigderson have shown that computational zero-knowledge proof systems exist for all of $\mathbb{NP}$ [21].

A related security notion for interactive proof systems is *witness indistinguishability* introduced by Feige and Shamir [18]. This notion is weaker than zero knowledge as it only requires that the adversarial verifier does not distinguish which of two given witnesses has been used by the prover. Witness indistinguishability is easily implied by zero knowledge, while not every witness indistinguishable proof is zero-knowledge one. Indeed, under plausible complexity assumptions, it is known how to construct two-round witness indistinguishable proof systems (i.e., "zaps" of Dwork and Naor [14]) for all non-trivial $\mathbb{NP}$ languages that are not zero knowledge, as implied by the works of Goldreich and Krawczyk [20] and Goldreich and Krawczyk [22]. The impact of the witness-indistinguishability notion has been proved central to many subsequent investigations on zero-knowledge proofs, including non-interactive zero knowledge [17, 9] non-black-box zero knowledge [1] and concurrent zero knowledge [15, 35, 27].

Security against man-in-the-middle attacks. Dolev, Dwork and Naor [12] proposed the notion of a non-malleable zero knowledge proof systems where security must be preserved even in case the adversary can play the role of a man-in-the-middle. This stronger attack allows the adversary to act as a prover in a proof and as a verifier in another proof with full control over the scheduling of the messages. Very informally, NMZK proofs defined in [12] are ZK proofs with an additional requirement: NMZK proofs are required to be resilient against "man-in-the-middle" attacks where the adversary tries to prove to a verifier a "related" theorem that he is getting from the prover. The notion of NMZK is proved to be extremely important in cryptography, since it captures the notion of *proof independence*, and led to multiple applications. Feasibility results for NMZK have been shown by using either black-box techniques and a super-constant number of rounds by Dolev, Dwork and Naor [13] or by using non-black-box techniques and obtaining computational soundness in a constant number of rounds by Barak [2] and Pass and Rosen [31].

Another aspect of Zero-Knowledge is that of *Concurrent* Zero-Knowledge introduced by Dwork, Naor and Sahai [15]. In this paper we consider a concurrent and non-malleable setting as well. In particular, we consider an adversary $\mathcal{A}$ mounting a *concurrent* man-in-the-middle attack in which $\mathcal{A}$ acts as a verifier interacting with a honest prover in polynomially many *left* proofs and acts as a prover interacting with honest verifiers in polynomially many *right* proofs. The issue of achieving protocols that combine concurrency and non-malleability has received a lot of attention, however many questions still remain open. In particular, constant-round concurrent NMZK proof systems have been shown by assuming the existence of trusted third parties or trusted common reference string [9, 3] or by using relaxed security notions [33, 6] or relaxed concurrency [25]. A construction for concurrent NMZK in the plain model has been given by Barak, Prabhakaran, and Sahai [5] with poly-logarithmic round complexity. The goal of constructing a constant round (necessarily non-black-box) concurrent non-malleable zero-knowledge proof system is open.

1.1 Our Results

Our work starts with the study of witness indistinguishability under a concurrent man-in-the-middle attack. Indeed, despite tremendous applicability of witness indistinguishable proofs, the notion of a concurrent non-malleable witness-indistinguishable (or NMWI) proof system was not addressed in the literature.

Concurrent non-malleable witness indistinguishability. We focus on a specific class of argument systems referred to as *commit-and-prove*[1] introduced in [26] and also considered in [8]. Informally, the transcript of a *commit-and-prove* argument system encodes through a commitment the witness used by the prover. We consider an adversary $\mathcal{A}$ mounting a *concurrent* man-in-the-middle attack. Our notion of non-malleable witness indistinguishability requires the *witnesses* encoded in the right proofs (in which $\mathcal{A}$ acts as a prover) to be independent from the witnesses used (by honest provers) in the left proofs.

We present two different definitions of concurrent non-malleable witness indistinguishability. The first definition (see Def. 3.5) follows the original definition for stand-alone witness indistinguishability of [18] and requires that the distribution of the witnesses (and not simply the views as in [18]) encoded in the right proofs is independent of the witnesses used by the honest provers in the left proofs. We also present (see Def. 3.7) a *simulation-based* notion of witness indistinguishability which requires that, for each successful man-in-the-middle adversary $\mathcal{A}$, there exists a stand-alone prover S (i.e., S does not have access to the honest provers) that has access to $\mathcal{A}$ and succeeds in proving to honest verifiers the same statements proved by $\mathcal{A}$ during the concurrent man-in-the-middle attack. Moreover, the arguments proved by S to the honest verifiers encode the same witnesses encoded in the proofs given by $\mathcal{A}$. This notion actually combines the security requirements of both zero knowledge and witness indistinguishability under concurrent man-in-the-middle attacks and deserves further studies.

We construct a *constant-round* concurrent non-malleable witness indistinguishable (cNMWI, for short) argument of knowledge for all $\mathbb{NP}$ in the plain model. This construction relies upon the recent work by Pass and Rosen [30] where constant-round concurrent non-malleable commitments have been achieved. Thus we will formally state and prove the following result (see Theorem 4.4).

THEOREM I. (Informal): *Under standard complexity-theoretic assumptions, there exists a constant-round concurrent non-malleable witness indistinguishable argument of knowledge for all languages in* $\mathbb{NP}$.

It is well known that, if non-malleability is not an issue, zero knowledge implies witness indistinguishability, that is every zero-knowledge proof is also a witness-indistinguishable proof. In this paper we show that, quite surprisingly, there exists a non-malleable zero-knowledge argument that is not non-malleable witness indistinguishable. We also show that there exists a non-malleable witness indistinguishable argument that is not non-malleable zero-knowledge thus making the two security notions incomparable. Thus we will formally state and prove the following result (see Theorem 5.1 and Theorem 5.3). We also stress that this result holds even if we ignore all issues of concurrency and focus on non-malleability alone.

[1] This turns out to be the cleanest notion of the "witness used", namely the one defined in the commitment. Thus, we restrict our study to this class of commit-and-prove argument systems since: 1) they allow us to uniquely define what is the witness encoded in a proof; 2) they suffice for the constructions and separations that we give. There are generalizations of the notion, but these more general definitions make the presentation far more cumbersome.

THEOREM II. (Informal): *Under standard complexity-theoretic assumptions, non-malleable zero knowledge and non-malleable witness indistinguishability are incomparable.*

Concurrent non-malleable zero knowledge in the Bare Public Key Model. The Bare Public Key model (BPK model, in short) was introduced by Canetti, Goldreich, Goldwasser and Micali [7]). It is the model where each verifier registers some public information (called the *public key*) in a public file during a preprocessing stage. Each public key is associated with some secret information (called the *secret key*) that is known only to the owner of the public key. After the non-interactive preprocessing is completed, parties engage in the proof stage in which the actual arguments will be executed. We consider concurrent man-in-the-middle attacks in the BPK model that has the full power of a concurrent man-in-the-middle adversary during the proof stage and, in addition, complete control over the public file; that is, the adversary can remove entries from it and add new entries related to the ones owned by the honest parties.

The well known FLS paradigm of using witness-indistinguishable proofs allows one to obtain ZK from witness indistinguishability and has been used in several models. We show how to apply this paradigm using our Theorem I to construct in BPK model constant-round NMZK proof systems for all $\mathbb{NP}$. Thus we will formally state and prove the following result (see Theorem 6.4).

THEOREM III. (Informal): *Under standard complexity-theoretic assumptions in the* BPK *model there exists a constant-round concurrent non-malleable zero-knowledge argument of knowledge for all languages in* $\mathbb{NP}$.

Corruption model and adaptive inputs. In all our results we consider the static corruption model where the adversary has to choose the corrupted parties before the protocols start. Following the previous work in the area, we assume that the inputs (i.e., statements) for honest parties are fixed according to some predetermined distribution while the adversary can choose its inputs adaptively.

1.2 Related Work

Work related to witness indistinguishability. Very recently and independently from our work Micali, Pass and Rosen [28] presented an extension of the notion of witness indistinguishability for achieving a relaxed notion of secure computation that does not resort to the simulation paradigm. Their techniques are similar to ours but in this work, in contrast to [28], we achieve arguments of knowledge and focus on the use of these stronger notions of witness indistinguishability for achieving a notion of security based on simulation (i.e., concurrent NMZK). Moreover we show that the notions of non-malleable witness indistinguishability and NMZK are incomparable.

Work related to concurrent non-malleable zero knowledge in the plain model. We observe that in the plain model constant-round (non-concurrent) non-malleable zero knowledge has been recently obtained [2, 31] whereas obtaining constant-round concurrent zero knowledge in the plain model has been open for quite some time. The only constant-round concurrent zero-knowledge arguments known in the plain model impose a bound on the number of concurrent executions that the adversary can perform [1]. If we do not insist on constant-round protocols, non-malleability and security in a concurrent setting have been achieved by [5] which present protocol with poly-logarithmic round complexity.

Work related to concurrent non-malleable zero knowledge in the BPK model. On the other hand, constant-round concurrent zero knowledge has been obtained in the BPK model

in [7] (and in [10, 11] with a concurrent soundness guarantee). Given our results, the BPK model is, at the best of our knowledge, the weakest model in which *constant-round* concurrent non-malleable zero knowledge has been achieved. Previous results (some of which achieved stronger notions of security) required either the existence of trusted third parties (trusted key-registration functionalities [3], common reference strings [9, 8]), or achieved only quasi-security (simulation in super-polynomial time [33, 6]) or quasi-concurrency (timing assumptions [25]).

2 Preliminaries

A polynomial-time relation R is a relation for which it is possible to verify in time polynomial in $|x|$ whether $R(x, w) = 1$. We will consider $\mathbb{NP}$-languages L and denote by R_L the corresponding polynomial-time relation such that $x \in L$ if and only if there exists w such that $R_L(x, w) = 1$. We will call such a w a *valid witness for* $x \in L$ and denote by $W_L(x)$ the set of valid witnesses for $x \in L$. We will slightly abuse notation and, whenever L is clear from the context, we will simply write $W(x)$ instead of $W_L(x)$. Also for sequences $X = (x_1, \cdots, x_m)$ and $W = (w_1, \cdots, w_m)$, by the writing "$W \in W(X)$" we mean that $w_i \in W(x_i)$ for $i = 1, \cdots, m$.

For a language L we will denote by L_n^m the set of sequences of m elements of L each of length at most n. A *negligible* function $\nu(k)$ is a function such that for any constant $c < 0$ and for all sufficiently large k, $\nu(k) < k^c$.

Indistinguishability. Let $\mathcal{S}$ be a set of strings. An *ensemble* of random variables $X = \{X_s\}_{s\in\mathcal{S}}$ is a sequence of random variables indexed by elements of $\mathcal{S}$.

Definition 2.1 *Two ensembles of random variables $X = \{X_s\}_{s\in\mathcal{S}}$ and $Y = \{Y_i\}_{s\in\mathcal{S}}$ are* computationally indistinguishable *if for every probabilistic polynomial-time algorithm D there exists a negligible function ν such that for any $s \in \mathcal{S}$*

$$|\text{Prob}[\ \alpha \leftarrow X_s : D(s, \alpha) = 1\ ] - \text{Prob}[\ \alpha \leftarrow Y_s : D(s, \alpha) = 1\ ]| < \nu(|s|).$$

Definition 2.2 *Two ensembles of random variables $X = \{X_s\}_{s\in\mathcal{S}}$ and $Y = \{Y_s\}_{s\in\mathcal{S}}$ are* statistically indistinguishable *if there exists a negligible function ν such that for all $s \in \mathcal{S}$*

$$\sum_{\alpha} |\text{Prob}[\ X_s = \alpha\ ] - \text{Prob}[\ Y_s = \alpha\ ]| < \nu(|s|).$$

Definition 2.3 *Two ensembles of random variables $X = \{X_s\}_{s\in\mathcal{S}}$ and $Y = \{Y_s\}_{s\in\mathcal{S}}$ are* perfectly indistinguishable *if for all $s \in \mathcal{S}$*

$$\sum_{\alpha} |\text{Prob}[\ X_s = \alpha\ ] - \text{Prob}[\ Y_s = \alpha\ ]| = 0.$$

One-way functions. Before describing our construction we review the classical notion of a one-way function.

Definition 2.4 *A polynomial-time computable function $f : \{0,1\}^\star \to \{0,1\}^\star$ is called* one-way *if for every probabilistic polynomial time algorithm A there exists a negligible function ν such that*

$$\text{Prob}[\ x \leftarrow \{0,1\}^n; y \leftarrow A(f(x), 1^n) : f(y) = f(x)\ ] < \nu(n).$$

Interactive argument/proof systems. An *interactive proof (resp., argument) system* [23] for a language L is a pair of interactive Turing machines $\langle P, V\rangle$, satisfying the requirements of *completeness* and *soundness*. Informally, completeness requires that for any $x \in L$, at the end of the interaction between P and V, where P has on input a valid witness for $x \in L$, V rejects with negligible probability. Soundness requires that for any $x \notin L$, for any computationally unbounded (resp., probabilistic polynomial-time for arguments) $P^\star$, at the end of the interaction between $P^\star$ and V, V accepts with negligible probability. We denote by $\langle P, V\rangle(x)$ the output of the verifier V when interacting on common input x with prover P. Also, sometimes we will use the notation $\langle P(w), V\rangle(x)$ to stress that prover P receives as additional input witness w for $x \in L$.

Formally, we have the following definition.

Definition 2.5 *A pair of interactive Turing machines $\langle P, V\rangle$ is an* interactive proof system *for the language L, if V is probabilistic polynomial-time and*

1. *Completeness: There exists a negligible function $\nu(\cdot)$ such that for every $x \in L$ and for every $w \in W(x)$*
$$\text{Prob}[\ \langle P(w), V\rangle(x) = 1\ ] \geq 1 - \nu(|x|).$$
2. *Soundness: For every $x \notin L$ and for every interactive Turing machines $P^\star$ there exists a negligible function $\nu(\cdot)$ such that*
$$\text{Prob}[\ \langle P^\star, V\rangle(x) = 1\ ] < \nu(|x|).$$

If the soundness condition holds only with respect to probabilistic polynomial-time interactive Turing machines $P^\star$ then $\langle P, V\rangle$ is called an argument.

Since all protocols we give are actually argument (rather than proof) systems, we will now focus on argument systems only. Also from now on we assume that all interactive Turing machines are probabilistic polynomial-time.

Zero knowledge. The classical notion of zero knowledge has been introduced in [23]. In a zero-knowledge argument system a prover can prove the validity of a statement to a verifier without releasing any additional information. This concept is formalized by requiring the existence of an expected polynomial-time algorithm, called the *simulator*, whose output is indistinguishable from the view of the verifier.

We start by defining the concept of a view of an interactive Turing machine. Let A and B be two interactive Turing machines that run on common input x and assume that A and B have additional information z_A and z_B. We denote by $\mathsf{View}_B^A(x, z_A, z_B)$ the random variable describing the *view of* B; that is, B's random coin tosses, internal state sequence, and messages received by B during its interaction with A.

We are now ready to present the notion of a zero-knowledge argument.

Definition 2.6 *An interactive argument system $\langle P, V\rangle$ for a language L is* zero-knowledge *if for all polynomial-time verifiers $V^\star$, there exists an expected polynomial-time algorithm S running in expected polynomial time such that the ensembles*

$$\{\mathsf{View}_{V^\star}^P(x, w, z)\}_{x\in L, w\in W(x), z\in\{0,1\}^\star} \text{ and } \{S(x, z)\}_{x\in L, z\in\{0,1\}^\star}$$

are computationally indistinguishable.

If the two ensembles are statistically/perfectly *indistinguishable then $\langle P, V\rangle$ is* statistical/perfect zero-knowledge.

2.1 Non-Malleable Argument Systems

The notion of non-malleability has been first considered in [12]. Non-malleability is concerned with an adversary $\mathcal{A}$ that mounts a so-called *man-in-the-middle attack* on two concurrent executions of a protocol Π. Even though in this paper we will consider non-malleability only in relation to argument systems, non-malleability can be considered with respect to any protocol.

Let $\Pi = \langle P, V \rangle$ be an argument system for the language L. A *man-in-the-middle adversary* $\mathcal{A}$ for Π acts as a verifier in one proof (called the *left* proof) and verifies the validity of a statement "$x \in L$" being proved by a honest party running P; and acts as a prover in another proof (called the *right* proof) in which $\mathcal{A}$ tries to convince a honest party running V of the validity of a statement $\tilde{x} \in L$ of his choice. It is assumed that $\mathcal{A}$ has complete control of the communication channel and therefore decides the scheduling of the messages. Very informally, Π is non-malleable if, whenever $x \neq \tilde{x}$, the left proof does not help $\mathcal{A}$ in the right proof.

Let us proceed more formally. For a man-in-the-middle adversary $\mathcal{A}$, we consider two executions: the *man-in-the-middle* execution and the *stand-alone* execution.

In the man-in-the-middle execution we have three parties: a honest prover P, a honest verifier V and man-in-the-middle adversary $\mathcal{A}$. In the left proof P and $\mathcal{A}$ (acting as a verifier) interact on common input $x \in L$; P receives $w \in W(x)$ as private input and $\mathcal{A}$ receives auxiliary information $z \in \{0,1\}^\star$. In the right proof $\mathcal{A}$ (acting as a prover) and V interact on common input $\tilde{x}$ chosen by $\mathcal{A}$. We denote by $\mathsf{mim}_V^{\mathcal{A}}(x, w, z)$ the random variable describing the output of V in this scenario which is V's decision and the right input $\tilde{x}$ chosen by $\mathcal{A}$. If $x = \tilde{x}$ then $\mathsf{mim}_V^{\mathcal{A}}(x, w, z)$ is the random variable that assigns positive probability only to $\perp$.

In the stand-alone execution we have only two parties: a machine S (the simulator) and a verifier V. S with access to $\mathcal{A}$ and auxiliary information $z \in \{0,1\}^\star$, interacts with V on common input x. We denote by $\mathsf{sta}_V^S(x, z)$ the random variable describing the output of V in this interaction.

Definition 2.7 (non-malleable argument system) *An argument system $\Pi = \langle P, V \rangle$ for a language L is* non-malleable *if for every probabilistic polynomial-time man-in-the-middle adversary $\mathcal{A}$, there exists a probabilistic algorithm S running in expected polynomial time and a negligible function ν such that, for every $x \in L$, $w \in W(x)$, and for every $z \in \{0,1\}^\star$*

$$|\text{Prob}[\ \mathsf{mim}_V^{\mathcal{A}}(x, w, z) = 1\ ] - \text{Prob}[\ \mathsf{sta}_V^S(x, z) = 1\ ]| < \nu(|x|).$$

Tag-based non-malleability. The above definition does not say anything about the case in which $\mathcal{A}$ proves in the right proof the same theorem P proved in the left proof (that is, $\tilde{x} = x$). Actually, there is no way of preventing $\mathcal{A}$ from relaying messages from the left proof to the right proof and vice versa. The next definition requires that if, $x = \tilde{x}$, then $\mathcal{A}$'s proof must be somehow different from P's.

Consider a family $\{\langle P_{\mathtt{tag}}, V_{\mathtt{tag}} \rangle\}_{\mathtt{tag}}$ of argument systems indexed by a string $\mathtt{tag}$. As before, we will consider the man-in-the-middle execution and the stand-alone execution. More specifically, in the man-in-the-middle execution we consider $\mathcal{A}$ that, on input x and auxiliary information z, interacts in the left proof with the prover $P_{\mathtt{tag}}$ on input (x, w) and in the right proof with verifier $V_{\tilde{\mathtt{tag}}}$ on input $\tilde{x}$. The tag $\tilde{\mathtt{tag}}$ of the right proof as well the input $\tilde{x}$ of the right proof are chosen adaptively by $\mathcal{A}$. We denote by $\mathsf{mim}_V^{\mathcal{A}}(\mathtt{tag}, x, w, z)$ the random variable describing the output of V in this scenario (it is V's decision, the tag $\tilde{\mathtt{tag}}$ and the statement $\tilde{x} \in L$). Similarly $\mathsf{sta}_V^S(\mathtt{tag}, x, z)$ is defined as the output of the verifier while interacting with

S. Similarly to the previous case, if the right proof contains the same tag used in the left proof, then $\mathsf{mim}_V^{\mathcal{A}}(\mathtt{tag}, x, w, z)$ gives positive probability only to the string $\perp$.

Definition 2.8 (tag-based non-malleable argument) *A family of argument systems* $\Pi = \{\langle P_{\mathtt{tag}}, V_{\mathtt{tag}}\rangle\}_{\mathtt{tag}}$ *for a language* L *is a* tag-based non-malleable argument with tags of length ℓ *if for every probabilistic polynomial-time man-in-the-middle adversary* $\mathcal{A}$*, a probabilistic algorithm* S *running in expected polynomial time and a negligible function* ν *such that for every* $x \in L$*,* $w \in W(x)$*, for every* $\mathtt{tag} \in \{0,1\}^\ell$*, and for every* $z \in \{0,1\}^\star$

$$|\text{Prob}[\ \mathsf{mim}_V^{\mathcal{A}}(\mathtt{tag}, x, w, z) = 1\ ] - \text{Prob}[\ \mathsf{sta}_V^{S}(\mathtt{tag}, x, z) = 1\ ]| < \nu(|x|).$$

Non-malleable zero knowledge. Consider an argument system $\Pi = \langle P, V\rangle$ for an $\mathbb{NP}$-language L. Let $\mathcal{A}$ be a man-in-the-middle adversary attacking Π. Then, with a slight abuse of notation, by $\mathsf{View}_{\mathcal{A}}^{P}(x, w, z)$ we denote the random variable describing the view obtained by $\mathcal{A}$ in the left and the right proof (including the sequence of its internal states and messages sent and received by $\mathcal{A}$) when given auxiliary information z. In the left proof $\mathcal{A}$ is interacting with a honest prover P on common input "$x \in L$" and P receives a valid witness w for x as private input. In the right proof $\mathcal{A}$ interacts with the honest verifier on input $\tilde{x}$ chosen by $\mathcal{A}$.

Definition 2.9 (NMZK argument system) *A non-malleable argument system* $\Pi = \langle P, V\rangle$ *for a language* L *is* non-malleable zero-knowledge *(in short* NMZK*) if for any probabilistic polynomial-time man-in-the-middle adversary* $\mathcal{A}$*, there exists a probabilistic algorithm* S *running in expected polynomial time such that, the ensembles*

$$\{\mathsf{View}_{\mathcal{A}}^{P}(x, w, z)\}_{x\in L, w\in W(x), z\in\{0,1\}^\star} \text{ and } \{S(x, z)\}_{x\in L, z\in\{0,1\}^\star}$$

are computationally indistinguishable.

If the two ensembles are statistically/perfectly indistinguishable then Π *is said to be* non-malleable statistical/perfect zero-knowledge*.*

NMZK arguments of knowledge. The notion of non-malleable zero knowledge argument of *knowledge* is obtained by requiring that the simulator also outputs the witness encoded in the right proof (in which the man-in-the-middle adversary $\mathcal{A}$ plays as a prover). This notion was introduced by [9] for non-interactive NMZK and clearly implies non-malleability.

Definition 2.10 (NMZK arguments of knowledge) *An argument system* $\Pi = \langle P, V\rangle$ *for a language* L *is a* non-malleable zero-knowledge argument of knowledge *if for every probabilistic polynomial-time man-in-the-middle adversary* $\mathcal{A}$*, there exists a probabilistic algorithm* S *(called the simulator-extractor) running in expected polynomial time such that by denoting as* $S(x, z) = (S_0(x, z), S_1(x, z))$*, the output of* $S(x, z)$*, we have that:*

1. $\{S_0(x, z)\}_{x\in L, z\in\{0,1\}^\star}$ *is computationally indistinguishable from* $\{\mathsf{View}_{\mathcal{A}}^{P}(x, w, z)\}_{x\in L, w\in W(x), z\in\{0,1\}^\star}$*;*
2. $S_1(x, z) = \tilde{w}$ *and if the right proof is accepting with common input* $\tilde{x} \neq x$ *we have that, except with negligible probability,* $\tilde{w} \in W(\tilde{x})$*.*

The notion of tag-based NMZK argument of knowledge is obtained by requiring that the extraction procedure is successful if the right proof has a tag different from the one of the left proof. We also stress that we allow the adversary $\mathcal{A}$ to pick the theorem and the tag to be used in the right proof.

2.2 Concurrent Non-Malleable Zero-Knowledge Arguments of Knowledge

In a more powerful man-in-the-middle attack, the adversary $\mathcal{A}$ is not restricted to one proof on the left and one proof on the right but instead $\mathcal{A}$ is allowed to concurrently play polynomially many left and right proofs. We call such an adversary a *concurrent man-in-the-middle adversary*. Specifically, let k be the security parameter. Consider a vector $X = (x_1, \ldots, x_m)$ of $m = \mathsf{poly}(k)$ inputs each of length $n = \mathsf{poly}(k)$ and a vector $W = (w_1, \ldots, w_m)$, such that $w_1 \in W(x_1), \ldots, w_m \in W(x_m)$. In the man-in-the-middle execution, the i-th left proof, for $1 \leq i \leq m$, is played by (an instance of) the honest prover P on input (x_i, w_i) and by the adversary $\mathcal{A}$ on input (x_i, z), for some auxiliary information z; the j-th right proof, for $1 \leq j \leq l$, is played by $\mathcal{A}$ on input $\tilde{x}_j$ chosen by $\mathcal{A}$ and auxiliary information z and by (an instance of) the verifier V on input $\tilde{x}_j$. We assume that $\mathcal{A}$ has complete control over the network and thus decides when each message of each proof is delivered.

cNMZK arguments of knowledge. The next definition extends the notion of a NMZK Argument of Knowledge (as defined in Definition 2.10) to the concurrent scenario.

Definition 2.11 (cNMZK arguments of knowledge) *An argument system $\Pi = \langle P, V \rangle$ for the language L is a* concurrent non-malleable zero-knowledge argument of knowledge *(a* cNMZK argument of knowledge*) if for every probabilistic polynomial-time concurrent man-in-the-middle adversary $\mathcal{A}$ there exists a probabilistic algorithm S (called the simulator-extractor) running in expected polynomial time such that for all $m = \mathsf{poly}(k)$ and $n = \mathsf{poly}(k)$, by denoting with $S(X, z) = (S_0(X, z), S_1(X, z))$ the output of S on input (X, z) , we have that:*

1. *$\{S_0(X, z)\}_{X \in L_n^m, z \in \{0,1\}^\star}$ and $\{\mathsf{View}_{\mathcal{A}}^P(X, W, z)\}_{X \in L_n^m, W \in W(X), z \in \{0,1\}^\star}$ are computationally indistinguishable;*
2. *$S_1(X, z) = (\tilde{w}_1, \ldots, \tilde{w}_m)$ where, except with negligible probability, $\tilde{w}_j \in W(\tilde{x}_j)$ for $1 \leq j \leq m$ and $\tilde{x}_j \notin X$ is the common input of the j-th accepting right proof.*

To define the notion of tag-based cNMZK argument of knowledge we define the view $\mathsf{View}_{\mathcal{A}}^P(T, X, W, z)$ of a tag-based man-in-the-middle adversary $\mathcal{A}$ when T is the sequence of tags, X is the sequence of inputs and W is the sequence of witnesses used in the left proofs as all messages receives by $\mathcal{A}$ in left and right proofs along with $\mathcal{A}$'s internal coin tosses.

Definition 2.12 (tag-based cNMZK arguments of knowledge) *A family $\Pi = \{\langle P_{\mathtt{tag}}, V_{\mathtt{tag}} \rangle\}_{\mathtt{tag}}$ of argument systems for the language L is a* tag-based concurrent non-malleable zero-knowledge argument of knowledge with tags of length ℓ *(a* cNMZK argument of knowledge*) if for every probabilistic polynomial-time tag-based concurrent man-in-the-middle adversary $\mathcal{A}$ there exists a probabilistic algorithm S (called the simulator-extractor) running in expected polynomial time such that for all $m = \mathsf{poly}(k)$ and $n = \mathsf{poly}(k)$, and for all sequences T of m tags of length ℓ by denoting with $S(T, X, z) = (S_0(T, X, z), S_1(T, X, z))$ the output of S on input (T, X, z), we have that:*

1. *$\{S_0(T, X, z)\}_{T \in \{0,1\}^{ml}, X \in L_n^m, z \in \{0,1\}^\star}$ and $\{\mathsf{View}_{\mathcal{A}}^P(T, X, W, z)\}_{T \in \{0,1\}^{ml}, X \in L_n^m, W \in W(X), z \in \{0,1\}^\star}$ are computationally indistinguishable;*
2. *$S_1(T, X, z) = (\tilde{w}_1, \ldots, \tilde{w}_m)$ and for all accepting right proofs j with tag $\tilde{\mathtt{tag}}_j \notin T$ we have that, except with negligible probability, $\tilde{w}_j \in W(\tilde{x}_j)$.*

One-left many-right cNMZK arguments of knowledge. Weaker notions of cNMZK can be obtained by restricting the power of the concurrent man-in-the-middle adversary $\mathcal{A}$. If we

allow the adversary to be active in only one left proof, then we obtain the notion of a *one-left many-right cNMZK argument of knowledge.* The following theorem is from [31, 30, 32][2].

Theorem 2.13 ([31, 30, 32]) *Assume that there exists a family of claw-free permutations. Then for any* $\mathbb{NP}$ *language* L *there exists a constant-round tag-based one-left many-right cNMZK arguments of knowledge* $\Pi = \{\langle P_{\mathtt{tag}}, V_{\mathtt{tag}}\rangle\}_{\mathtt{tag}}$ *that is perfect zero-knowledge for all* $\mathbb{NP}$*.*

According to the above definition, Theorem 2.13 above says that for any efficient concurrent man-in-the-middle adversary $\mathcal{A}$ there exists an efficient simulator S that guarantees:

1. the view of both the left proofs and the right proofs given in output by S are perfectly indistinguishable from the interaction of $\mathcal{A}$ with honest provers and honest verifiers;
2. the extraction succeeds for all accepting right proofs in which concurrent man-in-the-middle adversary has used a tag not appearing in any left proof;
3. S also outputs the witnesses for the accepting right proofs given by A; this means that the capability of any man-in-the-middle adversary A in proving statements in right proofs is owned by S (that is stand-alone) for computationally indistinguishable statements.

The last property is based on a technique referred to as simulation-extraction that combines non-black-box simulation with black-box extraction. Indeed, while the simulator is simulating a proof to the adversary, the extractor rewinds the adversary (and thus the simulation itself) and still extracts a valid witness from the proof given by the adversary.

3 Non-Malleable Witness Indistinguishability

In this section we discuss the notion of a witness indistinguishable argument and introduce the notion of a *non-malleable witness indistinguishable* argument system.

3.1 Witness Indistinguishability

The notion of a witness indistinguishable argument was introduced in [18] and requires the *view* of the (adversarial) verifier when interacting with a honest prover to be independent of the witness used by the prover. This notion therefore concerns $\mathbb{NP}$ statements for which there exists more than one witness. Even though witness indistinguishability yields weaker security guarantees than zero knowledge, in several cases witness indistinguishability is sufficient for the specific task at hand and it gives very efficient protocols. Furthermore, the celebrated FLS technique [17] can be used for obtaining zero knowledge from witness indistinguishability.

Let us now proceed more formally. Let $\Pi = \langle P, V\rangle$ be an argument system for language L. A *witness indistinguishability adversary* V' for Π receives as input $x \in L$, $w^0, w^1 \in W(x)$ and auxiliary information z. V' interacts with machine $P^\star$ that has a bit $b \in \{0,1\}$ wired-in. $P^\star$ receives as input (x, w^0, w^1) and executes the code of the honest prover P on input (x, w^b). For $b \in \{0,1\}$, we denote by $\mathsf{WIExpt}^b_{P,V'}(x, w^0, w^1, z)$ the random variable describing the output of V' when interacting on input (x, w^0, w^1, z) with prover $P^\star$ running on input (x, w^0, w^1) and b is the wired-in bit of $P^\star$.

Definition 3.1 *Argument system* $\Pi = \langle P, V\rangle$ *for the language* L *is* witness indistinguishable *if for all probabilistic polynomial-time witness indistinguishability adversaries* V' *there exists a*

[2]The use of claw-free permutations and the perfect zero-knowledge property are in particular discussed in [32].

negligible function ν such that for all $x \in L$, all witnesses $w^0, w^1 \in W(x)$ and all $z \in \{0,1\}^\star$

$$|\text{Prob}[\ \mathsf{WIExpt}^0_{P,V'}(x, w^0, w^1, z) = 1\ ] - \text{Prob}[\ \mathsf{WIExpt}^1_{P,V'}(x, w^0, w^1, z) = 1\ ]| < \nu(|x|).$$

We stress that witness indistinguishability holds with respect to adversaries that know both witnesses.

A stronger notion can be obtained if we consider adversaries that can concurrently execute several proofs. More precisely, a *concurrent witness indistinguishability adversary* V' for argument system $\Pi = \langle P, V\rangle$ for language L receives as input security parameter 1^k, sequence $X = (x_1, \ldots, x_m)$ of $m = \mathsf{poly}(k)$ elements of L each of length $n = \mathsf{poly}(k)$, two sequences $W^0 = (w^0_1, \cdots, w^0_m)$ and $W^1 = (w^1_1, \cdots, w^1_m)$ such that $w^0_i, w^1_i \in W(x_i)$ and auxiliary information z. V' interacts with m copies of machine $P^\star$ (one copy for each x_i). All copies of machine $P^\star$ have the same random bit $b \in \{0,1\}$ wired-in and the i-th copy of $P^\star$ receives as input (x_i, w^0_i, w^1_i) and executes the code of the honest prover P on input (x_i, w^b_i). V' has control of the network and decides in which order messages from different executions are delivered. For $b \in \{0,1\}$, we define the random variable $\mathsf{WIExpt}^b_{P,V'}(X, W^0, W^1, z)$ as the output of V' when interacting with m copies of machine $P^\star$ with bit b wired-in. We have the following definition.

Definition 3.2 *An argument system $\Pi = \langle P, V\rangle$ for the language L is* concurrent witness indistinguishable *(*cWI argument system*) if for all efficient non-uniform adversaries V', for all k, for all $m = \mathsf{poly}(k)$ and $n = \mathsf{poly}(k)$, there exists a negligible function ν such that for all sequences X of m elements of L of length n, for all sequences $W^0, W^1 \in W(X)$ and for all $z \in \{0,1\}^*$ it holds that*

$$|\text{Prob}[\ \mathsf{WIExpt}^0_{P,V'}(X, W^0, W^1, z) = 1\ ] - \text{Prob}[\ \mathsf{WIExpt}^1_{P,V'}(X, W^0, W^1, z) = 1\ ]| < \nu(k).$$

It is known (see [18]) that witness indistinguishability is closed under concurrent composition. Moreover the constructions of zaps [14, 24] imply that, under additional complexity/number-theoretic assumptions, there exists witness indistinguishable arguments for any non-trivial $\mathbb{NP}$ language that is not zero-knowledge. Finally we stress that the FLS paradigm [17] that allows one to obtain zero knowledge from witness indistinguishability is the most used technique for designing zero knowledge protocols.

3.2 Witness Indistinguishability under Man-In-The-Middle Attacks

In this section we present the notion of a non-malleable witness indistinguishable argument.

Motivation. The naive approach would be to define non-malleable witness indistinguishability by extending Definition 3.1 of witness indistinguishability to a man-in-the-middle adversary in much the same way in which Definition 3.1 was extended to handle concurrent adversaries in Definition 3.2. We call this notion *naive non-malleable witness indistinguishability.* (Indeed, such notion is not sufficient for our purposes). In fact, if we follow this approach then we could prove that any witness indistinguishable argument system is also a naive NMWI argument system. Notice, though that this notion does not capture a stated goal of non-malleability as it only puts restrictions on the output of the adversary (that is, on its own view). In our definition of non-malleable witness indistinguishability instead we shall require that the witness *encoded in the proof* [3] given by the man-in-the-middle adversary $\mathcal{A}$ is independent from the witness used

[3] Indeed, our restriction of commit-and-proof functionality aims at streamlining this definition. More general definitions are also possible, but do not seem to be necessary for our purposes.

by the honest prover in the left proof. Notice that $\mathcal{A}$ might be unaware of the witness it has used in the right proof and thus this possibility is not ruled-out by naive non-malleable witness indistinguishability.

More specifically, we focus on a specific class of argument systems referred to as *commit-and-prove* argument systems (previously considered in [8, 26]). Informally, the transcript of a commit-and-prove argument encodes in an unambiguous way the witness used by the prover (even though it might not be efficiently extracted from the transcript). In a non-malleable witness indistinguishable commit-and-prove argument we require the witness encoded in the proof produced by the man-in-the-middle adversary to be independent of the witness used (by the honest prover) in the proof in which the adversary acts as a verifier.

For general argument systems it is not clear whether the notion of witness encoded is well defined as there could be more than one. Therefore, we focus on commit-and-prove argument systems for which the notion of the witness encoded is well defined and commit-and-prove arguments actually suffice for proving our main result.

Commit-and-prove argument systems. A commit-and-prove argument system $\Pi = \langle P, V \rangle$ for a language L is a two-stage protocol. On input x, in the first stage the prover and the verifier execute a commitment protocol by which the prover commits to a string w. In the second stage, the prover proves to the verifier that the committed string w is a valid witness for "$x \in L$". We study commit-and-prove argument systems in which the commitment scheme used in the first stage is non-interactive and statistically binding, therefore the notion of *witness encoded in the proof* is well defined and it corresponds to the string committed to by the *first* prover-to-verifier message. If the proof is not accepted by the verifier, we consider the witness to be encoded in the proof to be the string $\perp$.

We shall require that in a non-malleable witness indistinguishable commit-and-prove argument system the man-in-the-middle adversary encodes in the right proof a witness that is independent from the one that the honest prover has used in the left proof.

NMWI commit-and-prove arguments. Let $\mathcal{A}$ be a man-in-the-middle adversary interacting in the left proof with the honest prover P that is running on input x and witness w. In the right proof $\mathcal{A}$ is interacting with the honest verifier V on common input $\tilde{x}$ chosen by $\mathcal{A}$. We denote by z the auxiliary information available to $\mathcal{A}$. The notion of non-malleable witness indistinguishability is defined in terms of the random variable $\mathsf{wmim}^{\mathcal{A}}(x, w, z)$ that is the distribution of the output of the following process: a transcript `trans` of an interaction of $\mathcal{A}$, including the left and the right proof, is picked according to distribution $\mathsf{View}^{P}_{\mathcal{A}}(x, w, z)$ of the view of $\mathcal{A}$ and the output of procedure wit applied to `trans` is returned. If the right proof of `trans` is not accepting or it has x has common input then wit returns the string $\perp$; otherwise a (possibly non-efficient) extraction procedure is applied on `trans` and the string w committed to by the first message of $\mathcal{A}$ in the right proof is returned. In other words, $\mathsf{wmim}^{A}(x, w, z)$ is the distribution of the witness encoded in the right proof unless the proof is non-accepting or has the same common input as the left proof. We are now ready to define non-malleable witness indistinguishability.

Definition 3.3 (NMWI) *A commit-and-prove argument system $\Pi = \langle P, V \rangle$ for an $\mathbb{NP}$-language L is* non-malleable witness indistinguishable *(in short,* NMWI*) if, for all probabilistic polynomial-time man-in-the-middle adversaries $\mathcal{A}$, for all probabilistic polynomial-time algorithms D, there exists a negligible function ν such that for all $x \in L$ and all $w, w' \in W(x)$, for all auxiliary information z it holds that*

$$|\text{Prob}[\ D(x, w, w', \mathsf{wmim}^{\mathcal{A}}(x, w, z), z) = 1\ ] - \text{Prob}[\ D(x, w, w', \mathsf{wmim}^{\mathcal{A}}(x, w', z), z) = 1\ ]| < \nu(|x|).$$

Similarly to NMZK, we can obtain a tag-based definition on non-malleable witness indistinguishability. We consider a man-in-the-middle adversary $\mathcal{A}$ interacting in the left proof with tag $\mathtt{tag}$ with the honest prover P that is running on input instance x and witness w. In the right proof, $\mathcal{A}$ is interacting with the honest prover V on common input $\tilde{x}$ and tag $\tilde{\mathtt{tag}}$ of its choice. We denote by z the auxiliary information available to $\mathcal{A}$. With a slight abuse of notation, we define the random variable $\mathsf{wmim}^{\mathcal{A}}(\mathtt{tag}, x, w, z)$ similarly to $\mathsf{wmim}^{\mathcal{A}}(x, w, z)$ with the only difference that a modified procedure wit is used. The modified wit procedure returns $\perp$ if the right proof is not accepting or $\mathtt{tag}$ is the tag of the right proof. Otherwise it returns the witness encoded in the right proof.

Definition 3.4 (tag-based NMWI) *A family of commit-and-prove argument systems* $\Pi = \{\langle P_{\mathtt{tag}}, V_{\mathtt{tag}}\rangle\}_{\mathtt{tag}}$ *for an* $\mathbb{NP}$*-language* L *is a* tag-based non-malleable witness indistinguishable argument with tags of length ℓ *(in short, a tag-based* NMWI*) if, for all probabilistic polynomial-time man-in-the-middle adversaries* $\mathcal{A}$*, for all probabilistic polynomial-time algorithms* D*, there exists a negligible function* ν *such that for all* $x \in L$*, for all tags* $\mathtt{tag} \in \{0,1\}^{\ell}$*, for all pairs* (w, w') *of witnesses for* x*, and for all auxiliary information* z *it holds that*

$$|\text{Prob}[\ D(x, w, w', \mathsf{wmim}^{\mathcal{A}}(\mathtt{tag}, x, w, z), z) = 1\ ] - \text{Prob}[\ D(x, w, w', \mathsf{wmim}^{\mathcal{A}}(\mathtt{tag}, x, w', z), z) = 1\ ]| < \nu(|x|).$$

3.3 Concurrent Non-Malleable Witness Indistinguishability

In this section we extend the notion of non-malleable witness indistinguishability to the concurrent setting by considering a concurrent man-in-the-middle adversary $\mathcal{A}$ that opens $m = \mathsf{poly}(k)$ left and right proofs each with a common input of length $n = \mathsf{poly}(k)$. Here k refers to the security parameter. $\mathcal{A}$ interacts in the i-th left proof with an instance of the honest prover P on common input "$x_i \in L$" and private prover's input $w_i \in W(x_i)$. In the j-th right proof $\mathcal{A}$ is interacting with the honest verifier V on common input $\tilde{x}_j$ of its choice.

To define concurrent non-malleable witness indistinguishability, we extend $\mathsf{wmim}^{\mathcal{A}}(X, W, z)$ to sequences of inputs and witnesses in the following way. The distribution $\mathsf{wmim}^{\mathcal{A}}(X, W, z)$ is the distribution of the output of the following procedure. First a transcript $\mathtt{trans}$ is sampled according to the view $\mathsf{View}_P^{\mathcal{A}}(X, W, z)$ of $\mathcal{A}$. Then the output of the following extension of the procedure wit applied to $\mathtt{trans}$ is returned. Procedure wit returns a sequence $(\tilde{w}_1, \cdots, \tilde{w}_m)$ where m is the number of right proofs and it holds that: if the j-th right proof is non-accepting or has the same common input as one of the left proofs then $\tilde{w}_j = \perp$; otherwise, $\tilde{w}_j$ is the witness encoded in the j-th right proof.

Definition 3.5 (cNMWI) *A commit-and-prove argument system* $\Pi = \langle P, V\rangle$ *for an* $\mathbb{NP}$*-language* L *is* concurrent non-malleable witness indistinguishable *(in short,* cNMWI*) if, for all probabilistic polynomial-time concurrent man-in-the-middle adversaries* $\mathcal{A}$*, for all* $m = \mathsf{poly}(k)$*, for all* $n = \mathsf{poly}(k)$*, for all probabilistic polynomial-time algorithms* D*, there exists a negligible function* ν *such that for all* k*, for all sequences* X *of* m *elements of* L *of length* n*, for all sequences* W *and* W' *of witnesses for* X*, and for all auxiliary information* z *it holds that*

$$|\text{Prob}[\ D(X, W, W', \mathsf{wmim}^{\mathcal{A}}(X, W, z), z) = 1\ ] - \text{Prob}[\ D(X, W, W', \mathsf{wmim}^{\mathcal{A}}(X, W', z), z) = 1\ ]| < \nu(k).$$

As done for non-malleable witness indistinguishability, we can obtain a tag-based definition of concurrent non-malleable witness indistinguishability and we define $\mathsf{wmim}^{A}(T, X, W, z)$ so to take into account the tags and not the inputs of the right proofs. We stress again that $\mathcal{A}$ is allowed to choose the inputs and the tags for the right proofs.

Definition 3.6 (tag-based cNMWI) *A family* $\Pi = \{\langle P_{\mathtt{tag}}, V_{\mathtt{tag}}\rangle\}_{\mathtt{tag}}$ *of commit-and-prove argument systems for the language* L *is a* tag-based concurrent non-malleable witness indistinguishable argument *(a* tag-based cNMWI*) with tags of length* ℓ *if, for all probabilistic polynomial-time concurrent man-in-the-middle adversaries* $\mathcal{A}$*, for all* $m = \mathsf{poly}(k)$*, for all* $n = \mathsf{poly}(k)$ *and for all probabilistic polynomial-time algorithms* D*, there exists a negligible function* ν *such that for all* k*, for all sequences* X *of* m *elements of* L *of length* n*, for all sequences* T *of tags of length* ℓ*, for all sequences* W *and* W' *of witnesses for* X*, and for all auxiliary information* z *it holds that*

$$|\text{Prob}[\ D(X, W, W', \mathsf{wmim}^{\mathcal{A}}(T, X, W, z), z) = 1\ ] - \text{Prob}[\ D(X, W, W', \mathsf{wmim}^{\mathcal{A}}(T, X, W', z), z) = 1\ ]| < \nu(k).$$

We will also consider a relaxed notion of concurrent non-malleable witness indistinguishability where the adversary is allowed to open only one left proof. We denote this restricted notion of concurrent non-malleable witness indistinguishability as *one-left many-right* concurrent non-malleable witness indistinguishability.

Comparison with NMZK. We stress here that NMZK requires the existence of a simulator while NM witness indistinguishability does not. Instead, NM witness indistinguishability crucially considers the possible witnesses that are encoded in the proofs given by the man-in-the-middle while NMZK requirements are satisfied when a valid witness is given in output by the simulator.

Comparison with non-malleable commitments. The notion of non-malleable witness indistinguishability is similar to the notion of non-malleable commitment with respect to commitment [12, 31]. Indeed, both notions concern the security of a primitive against man-in-the-middle attacks by considering a string that is encoded in the output of the adversary. This string is a committed message in case of non-malleable commitments while it is an encoded witness in case of non-malleable witness indistinguishability.

3.4 Simulation-Based cNMWI arguments

In this section we give a simulation-based definition of non-malleable witness indistinguishability. We consider only the tag-based case. Let $\mathcal{A}$ be a concurrent man-in-the-middle adversary and consider the following two executions. The first execution is the *man-in-the-middle* execution where the concurrent man-in-the-middle adversary $\mathcal{A}$ interacts with several copies of the honest prover in the left proofs and with several copies of the honest verifier in the right proofs. For this execution we define distribution $\mathsf{wmim}^{\mathcal{A}}(T, X, W, z)$ as done in the previous section. Also, we stress that $\mathcal{A}$ can choose the inputs for the right proofs as well as the tags. In the second execution, called the *stand-alone* execution, we consider a simulator S that, without receiving any witness for the inputs X of the left instances and without interacting with a honest prover, manages to output the transcripts of the left and the right proofs. We denote by $\mathsf{wsta}^{S}(T, X, z)$ the random variable that describes output of the following procedure. First a transcript $\mathtt{trans}$ is sampled according to the distribution of the output of $S(T, X, z)$. Then the procedure wit is applied to $\mathtt{trans}$ and the output is returned.

Definition 3.7 (tag-based SBcNMWI) *A family of commit-and-prove argument system* $\Pi = \{\langle P_{\mathtt{tag}}, V_{\mathtt{tag}}\rangle\}_{\mathtt{tag}}$ *is* tag-based simulation-based concurrent non-malleable witness indistinguishable *(in short,* tag-based SBcNMWI*) for the language* L*, if for all polynomials* $m = \mathsf{poly}(k)$ *and* $n = \mathsf{poly}(k)$*, for all probabilistic polynomial-time concurrent man-in-the-middle adversaries*

$\mathcal{A}$, there exists a simulator S running in expected polynomial time, such that

$\{\mathsf{wmim}^{\mathcal{A}}(T, X, W, z)\}_{T\in\{0,1\}^{ml}, X\in L_n^m, W\in W(X), z\in\{0,1\}^\star}$ and $\{\mathsf{wsta}^{S}(T, X, z)\}_{T\in\{0,1\}^{ml}, X\in L_n^m, z\in\{0,1\}^\star}$

are computationally indistinguishable.

The notion of a simulation-based non-malleable witness indistinguishable commit-and-prove *argument of knowledge* can be obtained by further requiring that S is able to extract witnesses from the right proofs whenever they use tags different from the left proofs.

As done with cNMZK arguments, the notion of one-left many-right SBcNMWI argument can be obtained by restricting the adversary to be involved only in one left proof. The next theorem proves that any one-left many-right SBcNMWI argument of knowledge is also a one-left many right cNMWI argument of knowledge.

Theorem 3.8 *Any one-left many-right tag-based SBcNMWI commit-and-prove argument of knowledge for an $\mathbb{NP}$-language L is a one-left many-right tag-based cNMWI commit-and-prove argument of knowledge for L.*

Proof. Let $\{\langle P_{\mathtt{tag}}, V_{\mathtt{tag}}\rangle\}$ be a one-left many-right tag-based SBcNMWI commit-and-prove argument of knowledge for the $\mathbb{NP}$ language L. Assume by contradiction that the claim does not hold. Therefore, there exists a one-left many-right concurrent man-in-the-middle adversary $\mathcal{A}$ that violates Definition 3.6. Now, let $x \in L$ and let w and w' be two witnesses for x and consider distributions $\mathsf{wmim}^{\mathcal{A}}(t, x, w, z)$ and $\mathsf{wmim}^{\mathcal{A}}(t, x, w', z)$. Then each of them is computationally indistinguishable from $\mathsf{wsta}^{S}(t, x, z)$, where S is the simulator associated with $\mathcal{A}$ (which exists since $\langle P, V\rangle$ is a one-left many-right tag-based SBcNMWI commit-and-prove argument of knowledge for L). Therefore, using $\mathcal{A}$ it will be possible to distinguish either between $\mathsf{wmim}^{\mathcal{A}}(t, x, w, z)$ and $\mathsf{wsta}^{S}(t, x, z)$ or between $\mathsf{wmim}^{\mathcal{A}}(t, x, w', z)$ and $\mathsf{wsta}^{S}(t, x, z)$ either one of which contradicts the hypothesis. □

4 Constant-Round cNMWI Arguments of Knowledge

In this section we present a tag-based constant-round cNMWI commit-and-prove argument of knowledge for all languages in $\mathbb{NP}$. For our construction we need the following tools.

Statistically binding commitments. A commitment scheme is a pair of probabilistic polynomial-time algorithms: the commitment algorithm Com and the verification algorithm Open. The committer obtains a pair $(\mathsf{com}, \mathsf{dec})$ of commitment and decommitment keys by running Com on input message m and randomness r. The commitment com is published by the committer. It is useful to think of com as a sealed envelop containing the message m. To reveal the commitment, the committer publishes the triple $(\mathsf{com}, \mathsf{dec}, m)$. The verification algorithm Open is then run on input the triple to verify that com was properly opened as m. The hiding property requires that the commitment com does not reveal any information on the committed message m to an adversary that has access to com. The binding property requires that an adversary can not produce a commitment com for which there exists two messages m_0 and m_1 and two decommitment keys $(\mathsf{dec}_0, \mathsf{dec}_1)$ such that com can be opened as m_0 using dec_0 and as m_1 using dec_1 (that is, $\mathsf{Open}(\mathsf{com}, \mathsf{dec}_0, m_0) = 1$ and $\mathsf{Open}(\mathsf{com}, \mathsf{dec}_1, m_1) = 1$).

We will consider non-interactive *statistically binding* commitment schemes where the binding property holds regardless of the computational power of the adversarial senders, the hiding

property holds only with respect to polynomial-time adversarial receivers and the commitment stage is a non-interactive message played from the sender to the receiver. Such commitment schemes can be constructed using 1-1 one way functions (see [19]).

Secure signature schemes. A secure signature scheme $SS = (\mathtt{SG}, \mathtt{Sig}, \mathtt{SVer})$ is a triple of efficient algorithms. The key generation algorithm $\mathtt{SG}$ on input a security parameter 1^k returns a pair $(\mathsf{pk}, \mathsf{sk})$ that are respectively a public and a secret key. The secret key sk is used to sign a message m by running the signature algorithm $\mathtt{Sig}$ on input sk and the message m and obtains the signature s. The public key pk instead is used to verify signatures by means of the $\mathtt{SVer}$ algorithm that runs on input the public key pk, the message m and the signature s and outputs a bit. The security requirement guarantees that no polynomial-time adversary that is given access to a signature oracle is able to produce a signature of a message for which it has not queried the oracle, or to produce a new signature of a message for which it has queried the oracle (this last requirement defines a *strong* secure signature scheme). See [36] for a construction of a secure signature scheme based on one-way functions.

4.1 Constructing a One-Left Many-Right Tag-Based SBcNMWI

In this section we construct a one-left many-right tag-based SBcNMWI commit-and-prove argument of knowledge for any $\mathbb{NP}$-language.

Fix a language $L \in \mathbb{NP}$ and non-interactive statistically binding commitment $(\mathsf{Com}, \mathsf{Open})$ and define language L_1 as consisting of the pairs (x, com) such that there exist (dec, w) for which $\mathsf{Open}(\mathsf{com}, \mathsf{dec}, w) = 1$ and $w \in W(x)$. Let $\Pi = \{\langle \mathcal{P}_{\mathtt{tag}}, \mathcal{V}_{\mathtt{tag}} \rangle\}_{\mathtt{tag}}$ be a tag-based one-left many-right concurrent non-malleable perfect zero-knowledge argument of knowledge for L_1 and let S^{Π} be the associated simulator extractor. Theorem 2.13 gives sufficient conditions for the existence of Π.

In Figure 1, we present a family $\Gamma = \{\langle P_t, V_t \rangle\}_t$ of constant-round commit-and-prove arguments of knowledge for the language $L \in \mathbb{NP}$. We shall prove that Γ is a one-left many-right SBcNMWI argument. The argument $\langle P_t, V_t \rangle$ is similar to the protocol for non-malleable commitments of [31], the only difference being that in [31] the statement used in the underlying tag-based non-malleable perfect zero-knowledge argument of knowledge is about knowledge of the decommitment while we also require that the committed message satisfies relation R with respect to input x.

Lemma 4.1 *Let $L \in \mathbb{NP}$. If $\Pi = \{\langle \mathcal{P}_{\mathtt{tag}}, \mathcal{V}_{\mathtt{tag}} \rangle\}_{\mathtt{tag}}$ is a constant-round tag-based one-left many-right concurrent non-malleable perfect zero-knowledge argument of knowledge for L_1, SS is a secure signature scheme and $(\mathsf{Com}, \mathsf{Open})$ is a non-interactive statistically binding commitment scheme, then $\Gamma = \{\langle P_t, V_t \rangle\}_t$, depicted in Fig. 1, is a constant-round tag-based one-left many-right SBcNMWI commit-and-prove argument of knowledge for L.*

Proof. It is straightforward to see that, for all tags t, $\langle P_t, V_t \rangle$ is a constant-round commit-and-prove argument for L.

Let us now prove that $\Gamma = \{\langle P_t, V_t \rangle\}_t$ is a tag-based one-left many-right SBcNMWI commit-and-prove argument for L. Let $\mathcal{A}$ be a one-left many-right concurrent man-in-the-middle adversary.

In this proof (and in the proofs of Section 5) we use the fact that an adversary $\mathcal{A}$ for the protocol is actually an adversary for Π since the commitment stage consists of just one message that actually corresponds to the statement for Π (for more details, see [32]).

Tag: t.
Common input: x.
Private input to P_t: $w \in W(x)$.

1. P_t: Compute $(\mathsf{com}, \mathsf{dec}) \leftarrow \mathsf{Com}(w)$.

   Set $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathtt{SG}(1^k)$.

   Send the pair $(\mathsf{com}, \mathsf{pk})$ to V_t.

2. $P_t \rightarrow V_t$: P_t and V_t engage in an execution of Π with tag pk, where P_t runs $\mathcal{P}_{\mathsf{pk}}$ to prove to V_t (running $\mathcal{V}_{\mathsf{pk}}$) knowledge of witness (w, dec) that $(x, \mathsf{com}) \in L_1$. Let $\mathtt{trans}$ be transcript of the protocol so far.

3. P_t: Compute a signature σ of tag t concatenated with transcript $\mathtt{trans}$ by setting $\sigma \leftarrow \mathtt{Sig}(t \circ \mathtt{trans}, \mathsf{sk})$. Send σ to V_t.

4. V_t: Accept if and only if $\mathtt{SVer}(t \circ \mathtt{trans}, \sigma, \mathsf{pk}) = 1$ and the interaction with $\mathcal{P}_{\mathsf{pk}}$ was accepted by $\mathcal{V}_{\mathsf{pk}}$.

Figure 1: A constant-round tag-based one-left many-right SBcNMWI argument of knowledge $\Gamma = \{\langle P_t, V_t \rangle\}_t$ for $L \in \mathbb{NP}$.

Consider the simulator S defined as follows. S, on input tag t and common input x for the left proof and auxiliary information z, interacts with $\mathcal{A}$ and then outputs the transcript of the interaction. In the left proof S commits to string of 0s and uses the simulator S^{Π} for Π to obtain a transcript of the proof that the commitment is the commitment of a valid witness. In the right proofs, S uses S^{Π} to extract witnesses for the right proofs.

Suppose, for sake of contradiction, that there exists a probabilistic polynomial-time algorithm D such that for a positive constant c, $z \in \{0,1\}^\star$ and for infinitely many tags t, $x \in L$ and $w \in W(x)$ it holds that

$$|\text{Prob}[\ \mathtt{trans} \leftarrow \mathsf{View}^{\mathcal{A}}(t, x, w, z); W \leftarrow \mathsf{wit}(\mathtt{trans}) : D(t, x, W, z) = 1\ ] - \text{Prob}[\ \mathtt{trans} \leftarrow S(t, x, z); W \leftarrow \mathsf{wit}(\mathtt{trans}) : D(t, x, W, z) = 1\ ]| > \tfrac{1}{|x|^c}.$$

We denote by $E(\mathtt{trans})$ the predicate that is false if and only if the one-left many-right transcript $\mathtt{trans}$ contains an accepting right proof that uses the same signature public key of the left proof but a different tag. We can then write

$$\begin{aligned}
&\text{Prob}[\ \mathtt{trans} \leftarrow \mathsf{View}^{\mathcal{A}}(t, x, w, z); W \leftarrow \mathsf{wit}(\mathtt{trans}) : D(t, x, W, z) = 1\ ] = \\
&\quad \text{Prob}[\ \mathtt{trans} \leftarrow \mathsf{View}^{\mathcal{A}}(t, x, w, z) : E(\mathtt{trans})\ ] \cdot \\
&\qquad \text{Prob}[\ \mathtt{trans} \leftarrow \mathsf{View}^{\mathcal{A}}(t, x, w, z); W \leftarrow \mathsf{wit}(\mathtt{trans}) : D(t, x, W, z) = 1 | E(\mathtt{trans})\ ] + \\
&\quad \text{Prob}[\ \mathtt{trans} \leftarrow \mathsf{View}^{\mathcal{A}}(t, x, w, z) : \neg E(\mathtt{trans})\ ] \cdot \\
&\qquad \text{Prob}[\ \mathtt{trans} \leftarrow \mathsf{View}^{\mathcal{A}}(t, x, w, z); W \leftarrow \mathsf{wit}(\mathtt{trans}) : D(t, x, W, z) = 1 | \neg E(\mathtt{trans})\ ].
\end{aligned}$$

Now observe that $\text{Prob}[\ \mathtt{trans} \leftarrow \mathsf{View}^{\mathcal{A}}(t, x, w, z) : \neg E(\mathtt{trans})\ ]$ is negligible. Suppose it is not. Then $\mathcal{A}$ has succeeded in producing a signature for a new message (since tags are different) which is valid with respect to the public key chosen by the honest prover. Therefore $\mathcal{A}$ can easily be used

to forge signatures. A similar argument shows that Prob[$\texttt{trans} \leftarrow S(t,x,w,z) : \neg E(\texttt{trans})$] is negligible.

By the above reasoning, we can assume that D distinguishes on transcripts $\texttt{trans}$ in which $E(\texttt{trans})$ is true. That is, we can assume that for some positive constant c',

$$|\text{Prob}[\ \texttt{trans} \leftarrow \mathsf{View}^{\mathcal{A}}(t,x,w,z); W \leftarrow \mathsf{wit}(\texttt{trans}) : D(t,x,W,z) = 1 | E(\texttt{trans})\ ] - \text{Prob}[\ \texttt{trans} \leftarrow S(t,x,z); W \leftarrow \mathsf{wit}(\texttt{trans}) : D(t,x,W,z) = 1 | E(\texttt{trans})\ ]| > \frac{1}{|x|^{c'}}.$$

We denote by $\mathsf{Expt}_0^{\mathcal{A},D}(t,x,w,z)$ the following experiment and by $p_0^{\mathcal{A},D}(t,x,w,z)$ the probability that it outputs 1.

$\mathsf{Expt}_0^{\mathcal{A},D}(t,x,w,z)$

Interaction. Adversary $\mathcal{A}$ interacts on the left with the honest prover $P_t(x,w)$ and on the right with honest verifiers.

Output. If in one of the right proofs $\mathcal{A}$ uses the same signature public key used by P_t in the left proof but a different tag then **return** 0.

Else, let W be the sequence of witnesses encoded by $\mathcal{A}$ in the right proofs. **return** $D(t,x,W,z)$.

Clearly, we have

$$p_0^{\mathcal{A},D}(t,x,w,z) = \text{Prob}[\ \texttt{trans} \leftarrow \mathsf{View}^{\mathcal{A}}(t,x,w,z); W \leftarrow \mathsf{wit}(\texttt{trans}) : D(t,x,W,z) = 1 | E(\texttt{trans})\ ].$$

We next consider the following experiment $\mathsf{Expt}_1^{\mathcal{A},D}(t,x,w,z)$ and denote by $p_1^{\mathcal{A},D}(t,x,w,z)$ the probability that it outputs 1.

$\mathsf{Expt}_1^{\mathcal{A},D}(t,x,w,z)$

Interaction. Adversary $\mathcal{A}$ interacts on the left with the a prover that follows algorithm P_t with the only exception that the simulator S^{Π} is used in the proof at Step 2. Adversary $\mathcal{A}$ interacts on the right with honest verifiers.

Output. If in one of the right proofs $\mathcal{A}$ uses the same signature public key used by P_t in the left proof but a different tag then **return** 0.

Else, let W be the sequence of witnesses encoded by $\mathcal{A}$ in the right proofs. **return** $D(t,x,W,z)$.

Obviously, $|p_0^{D,\mathcal{A}}(t,x,w,z) - p_1^{D,\mathcal{A}}(t,x,w,z)| = 0$ as the left proof produced by the simulator has the same distribution as the view of $\mathcal{A}$.

We next consider the following experiment $\mathsf{Expt}_2^{\mathcal{A},D}(t,x,w,z)$ and denote by $p_2^{\mathcal{A},D}(t,x,w,z)$ the probability that it outputs 1.

$\mathsf{Expt}_2^{\mathcal{A},D}(t,x,w,z)$

Interaction. Adversary $\mathcal{A}$ interacts on the left with the prover that follows algorithm P_t with the following two exceptions: at Step 1, com is computed as a commitment of $0^{|w|}$ and the simulator S^{Π} is used instead of P_t in the interaction at Step 2. Adversary $\mathcal{A}$ interacts on the right with simulator S^{Π}.

Output. If in one of the right proofs $\mathcal{A}$ uses the same signature public key used by P_t in the left proof but a different tag then **return** 0.

Else, let W be the sequence of witnesses as extracted by the simulator S^{Π} from the right proofs. **return** $D(t,x,W,z)$.

Suppose now that the difference $|p_2^{D,\mathcal{A}}(t,x,w,z) - p_1^{D,\mathcal{A}}(t,x,w,z)|$ is non-negligible. Then we can break the security of the commitment scheme in the following way. Consider an algorithm B that receives a commitment com' and has to decide whether com' is the commitment of w or of $0^{|w|}$. Algorithm B will perform $\mathsf{Expt}_2^{\mathcal{A},D}$ (notice that this experiment can be executed in polynomial time) with the only exception that com is set equal to com'. Then, depending on whether com is a commitment of w or a commitment of $0^{|w|}$, D will be fed the witnesses used by $\mathcal{A}$ in the right proofs of $\mathsf{Expt}_1^{\mathcal{A},D}$ and $\mathsf{Expt}_2^{\mathcal{A},D}$, respectively. We can therefore conclude that $|p_2^{D,\mathcal{A}}(t,x,w,z) - p_1^{D,\mathcal{A}}(t,x,w,z)|$ is negligible. This implies that $|p_2^{D,\mathcal{A}}(t,x,w,z) - p_0^{D,\mathcal{A}}(t,x,w,z)|$ is negligible and the observation that

$$p_2^{D,\mathcal{A}}(t,x,w,z) = \mathrm{Prob}[\ \mathtt{trans} \leftarrow S(t,x,z); W \leftarrow \mathsf{wit}(\mathtt{trans}) : D(t,x,W,z) = 1 | E(\mathtt{trans})\ ]$$

concludes the proof that $\{\langle P_t, V_t\rangle\}_t$ is a tag-based one-left many-right SBcNMWI commit-and-prove argument for L.

All that it is left to prove is that for each accepting proof given by the adversary, the simulator outputs the corresponding witnesses. This, as in [31, 30, 32] follows from the simulation-extraction property of Π. □

There is a subtle point in the proof on which we would like to draw the reader's attention. The security properties of Π are guaranteed to hold only if Π is concurrently composed with itself. Specifically, simulator S^{Π} is only guaranteed to extract and simulate when Π is concurrently composed with itself. Instead in our proof of security of Γ, we consider Π composed with commitments and signatures. However, this is easily seen not to constitute a problem since the commitment consists of just one message that actually corresponds to the statement for Π (see [32] where the same issue is discussed).

By combining Lemma 4.1 and Theorem 2.13 we obtain the following result.

Theorem 4.2 *Assume that there exists a family of claw-free permutations. Then there exists a constant-round tag-based one-left many-right SBcNMWI commit-and-prove argument of knowledge for all* $\mathbb{NP}$.

Proof. If claw-free permutations exist then we can construct secure signature schemes (see [36]), constant-round tag-based one-left many-right concurrent non-malleable perfect zero-knowledge arguments of knowledge for $\mathbb{NP}$ (see Theorem 2.13) and statistically binding commitment schemes. The result then follows by Lemma 4.1. □

4.2 Constant-Round cNMWI Arguments for all $\mathbb{NP}$

In this section we give a tag-based constant-round cNMWI argument of knowledge for all $\mathbb{NP}$. We start by proving that any one-left many-right tag-based cNMWI commit-and-prove argument of knowledge for an $\mathbb{NP}$-language L is actually a (many-left many-right) cNMWI commit-and-prove argument of knowledge for L.

Lemma 4.3 *Let $\Pi = \{\langle P_t, V_t\rangle\}_t$ be a one-left many-right tag-based cNMWI commit-and-prove argument of knowledge for a language L. Then Π is a cNMWI commit-and-prove argument of knowledge for L.*

Proof. Assume by contradiction that there exists a successful legal tag-based concurrent man-in-the-middle adversary $\mathcal{A}$ for Π and consider any input sequence X for the left proofs and any two witness sequences for the left proofs W and W' for which $\mathcal{A}$ is successful. That is, the distribution of the witnesses encoded in the proofs given by $\mathcal{A}$ in the right proofs when witnesses W are used in the left proofs is distinguishable from the distribution of the witnesses encoded in the proofs given by $\mathcal{A}$ in the right proofs when witnesses W' are used in the left proofs. Using standard hybrid arguments we can reduce to the case in which W and W' only differ in one component which we call the *special* component and we let x and w and w' be the input and the witnesses of the special components of X, W and W', respectively.

We use $\mathcal{A}$ to construct a legal tag-based one-left many-right concurrent man-in-the-middle adversary M for Π thus contradicting the hypothesis. M has as auxiliary information the sequence W of witnesses and interacts with $\mathcal{A}$. Specifically, for all left proofs except the special one, M runs the prover's algorithm using the witnesses of W. For all the right proofs M interacts with $\mathcal{A}$ by executing the code of the honest verifier. For the special left proof, M performs a relay of messages with an external honest prover P' which is given (x, w, w') and can use either w or w' as witness. We then consider the distribution of the witnesses encoded in the right proofs.

Two cases are possible. If P' uses w then the view of $\mathcal{A}$ is exactly the same as in the game in which $\mathcal{A}$ interacts in the left proofs with a prover P that uses witnesses W. Therefore, the distribution of the witnesses encoded in the right proofs output by M is exactly the same as the distribution of the witnesses encoded in the right proofs output by $\mathcal{A}$. Similarly, if P' uses w'. Therefore, the distributions in the witnesses encoded in the right proofs by M when w and w' are used in the left proof can be distinguished, thus contradicting the hypothesis. □

We are now ready for the main result of this section.

Theorem 4.4 *Assume that there exists a family of claw-free permutations. Then there exists a constant-round tag-based cNMWI commit-and-prove argument of knowledge for all $\mathbb{NP}$.*

Proof. If claw-free permutations exist then by Lemma 4.1 there exists a one-left many-right SBcNMWI commit-and-prove argument of knowledge Π for all $\mathbb{NP}$. By Theorem 3.8, Π is a one-left many-right cNMWI commit-and-prove argument of knowledge for all $\mathbb{NP}$. Finally, by Lemma 4.3, Π is a (many-left many-right) cNMWI commit-and-prove argument of knowledge for all $\mathbb{NP}$. □

5 Separations

In this section we show the surprising separation between witness indistinguishability and zero knowledge with respect to man-in-the-middle attacks. In all previously known notions, zero

knowledge implies witness indistinguishability.

Since the protocol of Section 3 for non-malleable witness indistinguishability and the one by Pass and Rosen [31] for non-malleable zero knowledge have a similar structure, we show the separation between the two notions by constructing ad-hoc protocols that are variations of the above two protocols.

5.1 NMZK Argument Systems $\not\Rightarrow$ NMWI Argument Systems

We now show that when non-malleability is considered, zero knowledge does not imply witness indistinguishability. Specifically, we exhibit an argument system that is NMZK and for which there exists a man-in-the-middle adversary that breaks the non-malleable witness indistinguishability property.

Fix a non-interactive statistically binding commitment scheme $(\mathsf{Com}, \mathsf{Open})$ and an $\mathbb{NP}$ language L. Let L_1 be the language consisting of pairs (x, com) for which there exist (w, dec) such that $\mathsf{Open}(\mathsf{com}, \mathsf{dec}, w) = 1$ and $w \in W(x)$. Let $\Pi = \{\langle \mathcal{P}_{\mathtt{tag}}, \mathcal{V}_{\mathtt{tag}} \rangle\}_{\mathtt{tag}}$ be a tag-based perfect NMZK argument of knowledge for L_1. By Theorem 2.13, Π exists under the assumption of existence of claw-free permutations. In our construction we also need a secure signature scheme $SS = (\mathtt{SG}, \mathtt{Sig}, \mathtt{SVer})$.

Consider the following family of commit-and-prove argument systems $\Gamma = \{\langle P_{t_0,t_1}, V_{t_0,t_1} \rangle\}_{t_0,t_1}$.

Tag: (t_0, t_1);
Common input: x;
Private input to prover: witness w for $x \in L$.

1. P_{t_0,t_1} computes $(\mathsf{com}_0, \mathsf{dec}_0) \leftarrow \mathsf{Com}(w)$ and sends com_0 to V;
2. P_{t_0,t_1} computes $(\mathsf{sk}_0, \mathsf{pk}_0) \leftarrow \mathtt{SG}(1^k)$ and sends pk_0 to V;
3. P_{t_0,t_1} computes $(\mathsf{com}_1, \mathsf{dec}_1) \leftarrow \mathsf{Com}(w)$ and sends com_1 to V;
4. P_{t_0,t_1} computes $(\mathsf{sk}_1, \mathsf{pk}_1) \leftarrow \mathtt{SG}(1^k)$ and sends pk_1 to V;
5. $P_{t_0,t_1} \Leftrightarrow V_{t_0,t_1}$ run NMZK argument of knowledge $\Pi_{\mathsf{pk}_0 \circ 0}$ on common input (x, com_0) in which P_{t_0,t_1} runs algorithm $\mathcal{P}_{\mathsf{pk}_0 \circ 0}$ with private input (w, dec_0) and V_{t_0,t_1} runs algorithm $\mathcal{V}_{\mathsf{pk}_0 \circ 0}$. Let $\mathtt{trans}_0$ be the transcript of this interaction.
6. $P_{t_0,t_1} \Leftrightarrow V_{t_0,t_1}$ run NMZK argument of knowledge $\Pi_{\mathsf{pk}_1 \circ 1}$ on common input (x, com_1) in which P_{t_0,t_1} runs algorithm $\mathcal{P}_{\mathsf{pk}_1 \circ 1}$ with private input (w, dec_1) and V_{t_0,t_1} runs algorithm $\mathcal{V}_{\mathsf{pk}_1 \circ 1}$. Let $\mathtt{trans}_1$ be the transcript of this interaction.
7. P sets $\sigma_0 \leftarrow \mathtt{Sig}(t_0 \circ \mathsf{com}_0 \circ \mathtt{trans}_0 \circ 0, \mathsf{sk}_0)$, $\sigma_1 \leftarrow \mathtt{Sig}(t_1 \circ \mathsf{com}_1 \circ \mathtt{trans}_1 \circ 1, \mathsf{sk}_1)$ and send (σ_0, σ_1) to V;
8. V accepts if and only if $\mathtt{SVer}(t_0 \circ \mathsf{com}_0 \circ \mathtt{trans}_0 \circ 0, \sigma_0, \mathsf{pk}_0) = 1$, $\mathtt{SVer}(t_1 \circ \mathsf{com}_1 \circ \mathtt{trans}_1 \circ 1, \sigma_1, \mathsf{pk}_1) = 1$ and executions of $\Gamma_{\mathsf{pk}_0 \circ 0}$ and $\Gamma_{\mathsf{pk}_1 \circ 1}$ were accepted by $\mathcal{V}_{\mathsf{pk}_0 \circ 0}$ and $\mathcal{V}_{\mathsf{pk}_1 \circ 1}$.

Γ is a commit-and-prove argument system. The transcript of Γ on input x consists of a commitment com_0 of a witness w for $x \in L$ (this is the first line in the description of Γ)

and of a subprotocol Γ' that corresponds to the remaining lines of the description of Γ. Γ' is an argument system for proving that com_0 is indeed the commitment of a valid witness. Completeness of Γ' follows by inspection. Soundness of Γ' follows by observing that $\Pi_{\mathsf{pk}_0 \circ 0}$ is an argument of knowledge for L_1 and for each accepting statement, a corresponding witness is extracted. Therefore, Γ is a commit-and-prove argument system.

Γ is NMZK. Now we prove that Γ is a tag-based NMZK argument of knowledge (see Definition 2.12) by exhibiting, for each man-in-the-middle adversary $\mathcal{A}$, a simulator-extractor S. In the description of S we denote by S^{Π} the perfect simulator-extractor for Π. We assume wlog that witnesses have the same length of the instance.

Simulator S interacts with $\mathcal{A}$ both in the left proof and in the right proof. On input tag (t_0, t_1) and $x \in L$ for the left proof and auxiliary information z, S computes commitments com_0 and com_1 of $0^{|x|}$, picks pairs of public and secret keys $(\mathsf{sk}_0, \mathsf{pk}_0)$ and $(\mathsf{sk}_1, \mathsf{pk}_1)$ for the signature scheme and sends $\mathsf{com}_0, \mathsf{com}_1, \mathsf{pk}_0, \mathsf{pk}_1$ to $\mathcal{A}$. Then it uses twice simulator S^{Π} of Π on input auxiliary information z and the first time on input (x, com_0) and tag $\mathsf{pk}_0 \circ 0$ and the second time on input (x, com_1) with tag $\mathsf{pk}_1 \circ 1$ to complete the two subprotocols of the left proof and to extract a witness $\tilde{w}$ for the the two executions of Π run by $\mathcal{A}$ in the right proof. Finally, S signs the transcripts of the two subprotocols of the left proofs using sk_0 and sk_1 and sends the signatures to $\mathcal{A}$. S outputs the transcripts of the left and the right proof and the witness extracted by S^{Π} in the right proof.

Suppose for sake of contradiction that there exists a distinguisher D that for infinitely many $x \in L$, $w \in W(x)$ and z distinguishes the output of S from the view of $\mathcal{A}$. Let us consider the following three experiments. $\mathsf{Expt}_0(x, w, z)$ is exactly the experiment of S interacting with $\mathcal{A}$. In $\mathsf{Expt}_1(x, w, z)$ we execute the code of S with the following exception: com_0 is computed as a commitment of w. Finally, in $\mathsf{Expt}_2(x, w, z)$ both com_0 and com_1 are commitments of w. We observe that, since S^{Π} is a perfect simulator, the output of $\mathsf{Expt}_2(x, w, z)$ is perfectly identical to the view of $\mathcal{A}$ when interacting with the real prover. Therefore, D distinguishes Expt_0 from Expt_1 or Expt_1 from Expt_2. In both case we can use D to break the hiding property of the commitment scheme.

We now show that S actually outputs a witness for an accepting right proof whenever $\mathcal{A}$ uses for the right proof tag $(\tilde{t}_0, \tilde{t}_1)$ different from the tag (t_0, t_1) used by S in the left proof. Indeed, simulator S^{Π} fails only in case the tag used by $\mathcal{A}$ for the subprotocol of the right proof is equal to one of the tags used by S in the left proof. Observe that tags used for the first subprotocol end in 0 and tags used for the second subprotocol end in 1 and therefore S^{Π} fails in extracting the witness only if $\mathcal{A}$ picks the same signature public key used by S in the left proof. In this case though, since we are assuming that $(\tilde{t}_0, \tilde{t}_1) \neq (t_0, t_1)$, and since the right proof is accepting $\mathcal{A}$ has succeeded in producing a signature of a new message for a public key chosen by S. This contradicts the security of the underlying signature scheme.

Γ is not non-malleable witness indistinguishable. Now we prove that Γ is not tag-based NMWI. Consider the following man-in-the-middle adversary $\mathcal{A}$ that manages to produce an accepting right proof that encodes the same witness encoded in the left proof and uses a tag different from the one used in the left proof.

$\mathcal{A}$ acts as a verifier in a left proof on input x with tag (t_0, t_1) and starts a right proof (in which $\mathcal{A}$ acts as a prover) on input x and tags $(t_0, \tilde{t}_1)$, for $\tilde{t}_1 \neq t_1$. We also assume that $\mathcal{A}$ has a witness w for $x \in L$. When the left proof starts $\mathcal{A}$ receives as a verifier of the left proof messages $\mathsf{com}_0, \mathsf{pk}_0, \mathsf{com}_1$ and pk_1. $\mathcal{A}$ then sets $\widetilde{\mathsf{com}}_0 = \mathsf{com}_0$, $\widetilde{\mathsf{pk}}_0 = \mathsf{pk}_0$, $(\widetilde{\mathsf{com}}_1, \widetilde{\mathsf{dec}}_1) \leftarrow \mathsf{Com}(w)$ and $(\widetilde{\mathsf{pk}}_1, \widetilde{\mathsf{sk}}_1) \leftarrow \mathtt{SG}(1^k)$. $\mathcal{A}$ then sends to the verifier of the right proof messages

$\widetilde{\mathsf{com}}_0, \widetilde{\mathsf{pk}}_0, \widetilde{\mathsf{com}}_1, \widetilde{\mathsf{pk}}_1$. Then for the first subprotocol of the right proof (this is the argument that proves that $(x, \widetilde{\mathsf{com}}_0) = (x, \mathsf{com}_0) \in L$), $\mathcal{A}$ acts as a message relayer between the verifier of the right proof and the prover of the left proof. Instead for the second NMZK argument (this is the argument that proves that $(x, \widetilde{\mathsf{com}}_1) \in L$), $\mathcal{A}$ executes the algorithm of the honest prover (since $\mathcal{A}$ has witness $(w, \widetilde{\mathsf{dec}}_1)$ for $(x, \widetilde{\mathsf{com}}_1) \in L$). Finally, $\mathcal{A}$ receives signatures σ_0 and σ_1 in the left proof. Then $\mathcal{A}$ sets $\widetilde{\sigma}_0 = \sigma_0$ and computes $\widetilde{\sigma}_1$ by running algorithm $\mathtt{Sig}$ with the secret key sk_1 on the transcript of the second subprotocol. It is clear that $\mathcal{A}$ has managed to produce a right proof with respect to tag $(\widetilde{t}_0, \widetilde{t}_1)$ different from the tag (t_0, t_1) of the left proof and the witness encoded in the right proof is the same as the witness encoded in the left proof. We stress that, even though $\mathcal{A}$ knew a witness w and used it for computing the right proof, the witness encoded in the right proof is not necessarily w but it is precisely same witness encoded by the prover in the left proof. This implies that Γ is not NMWI.

We have thus proved the following theorem.

Theorem 5.1 *Assume that there exists a family of claw-free permutations. Then for any non-trivial $\mathbb{NP}$ language there exists a NMZK argument system that is not non-malleable witness indistinguishable.*

5.2 NMWI Argument Systems $\not\Rightarrow$ NMZK Argument Systems

In this section we show that, even when non-malleability is considered, witness indistinguishability does not imply zero knowledge. Specifically, we exhibit a commit-and-prove argument that is NMWI but not NMZK unless $\mathbb{NP} \subseteq \mathbb{BPP}$. This second separation is expected as witness indistinguishability is in general weaker than zero knowledge. Indeed, we exploit this by plugging a zap, i.e., a witness indistinguishable proof system that is not zero knowledge, in a NMWI argument system.

Again, as done for the proof of Lemma 4.1 in the proofs of the previous separations we use the simulator of the one-left many-right perfect non-malleable zero-knowledge argument of knowledge of [31, 30, 32] even though the protocol is composed with other primitives. Still, as in [32], the proof of security works even though it requires additional analysis.

We will use the following lower bound on the round-complexity of black-box zero-knowledge.

Theorem 5.2 ([20]) *If a language L has a three-round black-box zero knowledge argument system then $L \in \mathbb{BPP}$.*

Fix language $L \in \mathbb{NP} \setminus \mathbb{BPP}$ and consider the auxiliary language L_1 defined in the previous section. Also denote by $\Gamma = \{\langle \mathcal{P}_t, \mathcal{V}_t \rangle\}_t$ a NMZK argument of knowledge for L_1. Moreover we use a zap (that is, a 2-round witness indistinguishable argument) for L (see [24, 14]). Consider the following tag-based commit-and-prove argument system $\Pi = \{\langle P_t, V_t \rangle\}_t$.

Common input: x;
Tag: t;
Private input to prover: witness w for $x \in L$.

1. P_t sets $(\mathsf{com}, \mathsf{dec}) \leftarrow \mathsf{Com}(w)$ and sends com to V;
2. V_t sends the first message Σ for the zap to P;
3. P_t computes and send to V_t message π obtained by running the prover's algorithm for the zap on input x, witness w and message Σ received from V_t;
4. $P_t \Leftrightarrow V_t$ run NMZK argument of knowledge Γ for L_1 on common input (x, com) in which P_t runs algorithm $\mathcal{P}_t$ with private input (w, dec) and V_t runs algorithm $\mathcal{V}_t$;

Π is a commit-and-prove argument system. Π has the correct form since the first round is a commitment to a message and the remaining rounds form an argument system Π' for proving that the committed message is a witness for the given statement. Indeed, completeness follows by inspection while soundness follows by the argument of knowledge property of the last subprotocol that allows one to obtain a witness for any accepting proof given by the adversary.

Π is non-malleable witness indistinguishable. Let $x \in L$ and consider $w_1, w_2 \in W(x)$. Fix a man-in-the-middle adversary $\mathcal{A}$ that violates the non-malleable witness indistinguishability of Π. We consider a sequence of experiments $\mathsf{Expt}_1^{\mathcal{A}}(x, w_1, w_2), \cdots, \mathsf{Expt}_6^{\mathcal{A}}(x, w_1, w_2)$ in which $\mathsf{Expt}_1^{\mathcal{A}}(x, w_1, w_2)$ is the experiment of $\mathcal{A}$ interacting in the left proof with a honest prover that uses w_1 as a witness and $\mathsf{Expt}_6(x, w_1, w_2)$ is the experiment of $\mathcal{A}$ interacting in the left proof with a honest prover that uses w_2 as witness. We will show that for $i = 2, \cdots, 6$, the distribution of the witness encoded in the right proof of $\mathsf{Expt}_i^{\mathcal{A}}(x, w_1.w_2)$ (which we denote by $W_i^{\mathcal{A}}(x, w_1, w_2)$) is indistinguishable from $W_{i-1}^{\mathcal{A}}(x, w_1, w_2)$.

$\mathsf{Expt}_2^{\mathcal{A}}(x, w_1, w_2)$ is the experiment in which $\mathcal{A}$ interact on the left with the simulator S^{Γ} of Γ. Being the simulator perfect, $W_1^{\mathcal{A}}(x, w_1, w_2)$ and $W_2^{\mathcal{A}}(x, w_1, w_2)$ are indistinguishable.

$\mathsf{Expt}_3^{\mathcal{A}}(x, w_1, w_2)$ is the experiment in which $\mathcal{A}$ interacts on the left and on the right with the simulator S^{Γ} and the commitment com is computed as $\mathsf{com} \leftarrow \mathsf{Com}(0^{|w_1|})$. Standard arguments prove that, if Com is secure, then $W_2^{\mathcal{A}}(x, w_1, w_2)$ and $W_3^{\mathcal{A}}(x, w_1, w_2)$ are indistinguishable. Notice that in this experiment S^{Γ} manages to extract the witness of the right proof (unless $\mathcal{A}$ uses the same tag in the right proof in which case NMWI is not violated). This powerful technique is referred to as *simulation-extraction* and has been shown in [31, 32] for proving the non-malleability of their commitment scheme. The obtained message can therefore be given as input to D and its output can therefore be used to guess the committed message.

$\mathsf{Expt}_4^{\mathcal{A}}(x, w_1, w_2)$ is similar to the previous experiment with the only exception that the zap uses w_2 as witness. Again, the witness indistinguishability of the zap proves that $W_3^{\mathcal{A}}(x, w_1, w_2)$ and $W_4^{\mathcal{A}}(x, w_1, w_2)$ are indistinguishable. As before, notice that in this experiment S^{Γ} manages to extract the witness of the right proof (unless $\mathcal{A}$ uses the same tag in the right proof in which case NMWI is not violated). The extraction involves rewinding and thus the prover of the zap will be subject to rewind. This is however admissible as zap can be made resettably-sound resettable witness indistinguishable [14, 4][4].

[4] Another possible approach to overcome the problems introduced by these rewinds is to use two executions

$\mathsf{Expt}_5^{\mathcal{A}}(x, w_1, w_2)$ is similar to the previous experiment with the only exception that com is computed as a commitment of w_2 (and not of $0^{|w_2|}$). Again we use security of the commitment scheme as we did for $\mathsf{Expt}_3^{\mathcal{A}}(x, w_1, w_2)$

Finally, $W_5^{\mathcal{A}}(x, w_1, w_2)$ and $W_6^{\mathcal{A}}(x, w_1, w_2)$ are indistinguishable as S^{Γ} is perfect zero-knowledge.

Π is not non-malleable zero knowledge. Assume by contradiction that Π is NMZK, we show how to compute a simulator M for the zap. We would thus have a black-box 2-round zero-knowledge argument which does not exist for $L \in \mathbb{NP} \setminus \mathbb{BPP}$. M receives the first message Σ of a zap and constructs an adversary $\mathcal{A}$ for NMZK that runs against the honest prover P, the honest verifier algorithm with the only exception that Σ is played as first round of the zap, independently of the random tape. For this adversary there must be a simulator S of NMZK that therefore will compute a transcript that includes an accepting zap for $x \in L$ with first round Σ, otherwise a distinguisher that will detect the difference between the transcript of the real game (that contains Σ as first round of the zap in all proofs given by P to $\mathcal{A}$) and the transcript of S (that instead never contains left proofs where Σ is in the first round of the zap). From the transcript of the simulation computed by S, M can pick the computed zap π that can be used for the original purpose of completing the simulation of the zap without using any witness. Notice that M only accesses to the verifier of the zap as a black-box for obtaining message Σ. Therefore, if Π is non-malleable zero knowledge then the we have a 2-round black-box zero-knowledge proof system for a language L not in $\mathbb{BPP}$ which contradicts Theorem 5.2. We thus have the following theorem.

Theorem 5.3 *Assume that there exists a family of claw-free permutations and zaps for all $\mathbb{NP}$. Then there exists a NMWI argument system for L that is not NMZK.*

Proof. We observe that claw-free permutations are sufficient for the existence of zap for all $\mathbb{NP}$. The theorem then follows from the above discussion. □

6 cNMZK in the BPK Model

We start by reviewing the BPK model [7] and then we define the notion of cNMZK in the BPK model.

6.1 The BPK Model

In the BPK model, each verifier registers some public information (called the *public key*) in a public file during a preprocessing stage. Each public key is associated with some secret information (called the *secret key*) that is known only to the owner of the public key. After the preprocessing is completed, parties engage in the proof stage in which the actual argument will be executed.

We will define and construct in the BPK model constant-round arguments for any $\mathbb{NP}$-language that are secure with respect to a BPK concurrent man-in-the-middle adversary $\mathcal{A}$ which during the preprocessing stage has complete control over the public file where keys are registered (that is, $\mathcal{A}$ can modify, omit and, add new adaptively chosen keys to the public

of Γ_t in Π. In this case, the zap computed in the left proof can be affected by at most one of the two possible extraction procedures of the right proofs. The extraction therefore will be run on the right execution of Γ_t that is "safe" for the zap. Finally, we stress that at the cost of using number-theoretic assumptions, it is possible to use a one-round zap [24].

file) and, once the preprocessing stage is completed, $\mathcal{A}$ acts as a concurrent man-in-the-middle adversary. We stress that no form of key-authentication is required thus making the BPK model a very weak model.

The BPK model for interactive argument systems. We now review the definition of an interactive argument system in the BPK model that were previously given in [29] and the extension to the concurrent man-in-the-middle case.

Formally, a BPK *pair* is a pair $\langle P, V \rangle$ where P is a probabilistic polynomial-time algorithm and V is a pair $V = (V_0, V_1)$ of probabilistic polynomial-time algorithms. The interaction between provers and verifiers takes place in two stages. In the first stage, called the *set-up* stage, verifiers run algorithm V_0, on input a security parameter 1^k, to obtain a pair $(\mathsf{pk}, \mathsf{sk})$ consisting of a public and a secret key. Each verifier publishes his public key pk in a public file F. The second stage, called the *proof* stage, consists of polynomially (in the security parameter) many proofs. In each of them a prover interacts with a verifier; specifically, the prover runs algorithm P on input x (of length polynomial in the security parameter), some auxiliary information w (typically w is a witness for x to be member of some fixed language L) and the public key pk chosen by the verifier. The verifier instead runs algorithm V_1 on input x and sk.

Definition 6.1 *A* BPK *pair* $\langle P, V \rangle$ *is* complete *for the language* L *if in any interaction on common input* $x \in L$ *and* pk *constructed by* V_0*, where* P *receives as additional input* $w \in W(x)$*, and* V_1 *secret key* sk *associated with* pk*,* V_1 *accepts with probability negligible close to* 1*.*

The definitions of argument systems in the BPK model can be found in [7], in particular in [29, 34] the notions of concurrent zero-knowledge and concurrent soundness have been defined. We will focus on concurrent non-malleable zero-knowledge argument of knowledge in the BPK model that implies both concurrent zero knowledge and concurrent soundness. Indeed, concurrent zero-knowledge corresponds to a special case where the man-in-the-middle does not run any right proof. Instead, concurrent soundness corresponds to the special case where the man-in-the-middle does not run any left proof and is implied by the fact that we require that a legal $\mathbb{NP}$ witness is obtained for any accepting proof given by the adversary.

6.2 Concurrent Non-Malleable Zero Knowledge in the BPK Model

We next define the concept of *concurrent non-malleable zero-knowledge argument of knowledge* in the BPK model.

A BPK concurrent man-in-the-middle adversary $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$ is a pair of probabilistic algorithms. $\mathcal{A}_0$ on input an auxiliary information z receives the public file F containing the public keys as computed by the honest verifiers and outputs a modified public file F'. In computing F', $\mathcal{A}_0$ is allowed to add new adaptively chosen keys and to remove some of the keys of the honest verifiers. $\mathcal{A}_0$ also outputs some secret auxiliary information Z relative to F'. Once F' is made public by $\mathcal{A}_0$, it cannot be changed and the control passes to $\mathcal{A}_1$ that runs on input F' and Z. In the proof stage, $\mathcal{A}_1$ behaves like a concurrent man-in-the-middle adversary with the only restriction that he can start right proofs in which he plays as a prover with honest verifiers only with respect to entries of F' that were chosen by the honest verifiers and not modified by $\mathcal{A}_0$.

We define the view $\mathsf{BView}_{\mathcal{A}}(X, W, z)$ of a BPK concurrent man-in-the-middle adversary $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$ with respect to the vector X of left inputs with witnesses W as consisting of the initial public file received by $\mathcal{A}_0$, of all messages received by $\mathcal{A}_1$ in the proof stage both in the left

proofs run on input X and right proofs run on inputs adaptively chosen by $\mathcal{A}_1$, along with the sequence of internal states of $\mathcal{A}_0$ and $\mathcal{A}_1$ and coin tosses.

Definition 6.2 (cNMZK arguments of knowledge in the BPK) *A BPK pair $\Pi = \langle P, V \rangle$ complete for the language L is a* BPK concurrent non-malleable zero-knowledge argument of knowledge *(a* BPK cNMZK argument of knowledge*) if for every probabilistic polynomial-time* BPK *concurrent man-in-the-middle adversary $\mathcal{A}$, there exists a probabilistic algorithm S running in expected polynomial time such that, for all $m = \mathsf{poly}(k)$ and $n = \mathsf{poly}(k)$, by denoting with $S(X, z) = (S_0(X, z), S_1(X, z))$ the output of S on input (X, z), we have*

1. *$\{S_0(X, z)\}_{X \in L_n^m, z \in \{0,1\}^\star}$ and $\{\mathsf{BView}_{\mathcal{A}}(X, W, z)\}_{X \in L_n^m, W \in W(X), z \in \{0,1\}^\star}$ are computationally indistinguishable.*

2. *Writing the second component of S's output as $S_1(X, z) = (\tilde{w}_1, \ldots, \tilde{w}_m)$, we have that, for all accepting right proofs j of $S_0(X, z)$ with common input $\tilde{x}_j \notin X$, $\tilde{w}_j \in W(\tilde{x}_j)$ except with negligible probability.*

As a concurrent verifier and a concurrent prover are both special cases of a concurrent man-in-the-middle adversary, then it is obvious that a cNMZK argument of knowledge in the BPK model is both concurrent zero-knowledge and concurrently sound.

High-level idea. In next section we present our construction for a constant-round cNMZK argument of knowledge in the BPK model. We start with an informal discussion.

The main idea is that we want to use the FLS paradigm so that a prover actually proves knowledge of either a legal witness or of the secret key of the adversary. In order to give to the simulator such a secret key, we need a proof of knowledge of the secret key given by the verifier. However malleability attacks here are dangerous even when concurrent soundness only is considered [11]. When man-in-the-middle attacks are considered the implementation of the FLS paradigm is even more complex and requires new ideas. The concurrent NMWI argument of knowledge however can play a central role to solve this problem, and we use it along with a known technique by [18, 16] that suggests to use pairs of public keys.

More in details, in the preprocessing stage, each verifier computes a pair of public keys along with the corresponding secret keys. He randomly chooses one of the two secret keys and discards the other one. This step can be implement by using a one-way function f in the following way: randomly pick two messages $\mathsf{sk}_0, \mathsf{sk}_1$ in the domain of f; compute public keys $\mathsf{pk}_0 = f(\mathsf{sk}_0), \mathsf{pk}_1 = f(\mathsf{sk}_1)$; randomly select $b \leftarrow \{0, 1\}$; set $\mathsf{sk} = (b, \mathsf{sk}_b)$.

The protocol for the relation R and common input x is a sequential composition of two instances of the tag-based constant-round cNMWI commit-and-prove argument of knowledge presented in Appendix 4. In the first execution the verifier proves knowledge of one of the two secret keys associated to his entry in the public file (this is obviously done by $\mathbb{NP}$-reducing this instance to the $\mathbb{NP}$-complete language used by the subprotocol). This subprotocol is run using $x \circ 0$ as tag. Obviously the honest verifier uses his knowledge of one of two secret keys to successfully complete this subprotocol. In the second execution the prover proves knowledge of either w such that $R(x, w) = 1$ or of one of the two secret keys associated with the two public keys of the verifier. The tag used in this subprotocol is $x \circ 1$. Obviously the honest prover uses knowledge of a witness w for $R(x, \cdot)$ to complete the protocol.

Let us explain how we plan to perform simulation of the protocol. Simulation is easy for right proofs where the simulator plays the role of the honest verifier. Indeed right proofs are executed

relatively to entry of the public file that have been constructed by the simulator itself and thus it knows one of the secret keys to perform the first subprotocol of a right proof. Simulating the second subprotocol of right proofs and the first subprotocol of the left proofs is trivial as the simulator can simply play the honest verifier algorithm of the subprotocol. In order to simulate the second subprotocol of left proofs instead the simulator needs to know either a witness for "$x \in L$" or one of the secret keys associated with the corresponding entries of the public file that are used by the adversary. However, the adversary has just proved knowledge of at least one of the two keys in the first subprotocol of the same proof. Therefore we plan on extracting one of these keys from the adversary and then use it to perform the second subprotocol. The use of rewinds is dangerous in concurrent setting but not in the BPK model as shown in [7]. Indeed the number of extraction procedures that have to be successfully run is independent of the number of concurrent proofs, since it is bounded by the size of the public file. Once the simulator knows at least one secret key for each of the entries of the public file used by the adversary, the simulation is straight-line.

6.3 The protocol in details

Let L be an $\mathbb{NP}$-language with polynomial-time relation R and let f be a one-way function. Associated with L and f, we consider two auxiliary $\mathbb{NP}$-languages L_1 and L_2 with polynomial-time relations R_1 and R_2 defined as follows.

- $(\mathsf{pk}_0, \mathsf{pk}_1) \in L_1$ iff there exist b and sk such that $\mathsf{pk}_b = f(\mathsf{sk})$.
- $(x, \mathsf{pk}_0, \mathsf{pk}_1) \in L_2$ iff $x \in L$ or $(\mathsf{pk}_0, \mathsf{pk}_1) \in L_1$.

In the description of our BPK cNMZK argument of knowledge (P, V) for any $\mathbb{NP}$-language L we will use a tag-based cNMWI argument of knowledge $\Pi = \{\langle \mathcal{P}_{\mathtt{tag}}, \mathcal{V}_{\mathtt{tag}} \rangle\}_{\mathtt{tag}}$ for an $\mathbb{NP}$-complete language Λ. When we say that we execute Π for proving that $\tau \in L_1$ (or $\sigma \in L_2$) we actually mean that τ (or σ) is reduced to an instance of Λ and $\mathcal{P}_{\mathtt{tag}}$ and $\mathcal{V}_{\mathtt{tag}}$ are executed on input this instance. We also remark that known reductions have the property that, if a witness for $\tau \in L_1$ (or for $\sigma \in L_2$) is known then a witness for the new instance can be constructed in polynomial time.

The protocol is formally described in Figure 2. We have the following lemma.

Lemma 6.3 *If f is a one-way function and Π is a cNMWI argument of knowledge then the protocol $(\mathcal{P}, \mathcal{V})$ of Figure 2 is a cNMZK argument of knowledge in the BPK model for any $\mathbb{NP}$ language.*

Proof. Completeness is straightforward. We describe the simulator S as required by Definition 6.2 of cNMZK argument of knowledge.

S receives in input the vector X of left inputs and auxiliary information z and interacts with the BPK concurrent man-in-the-middle adversary $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$ running on input z. We let k denote the security parameter and denote by $n = \mathsf{poly}(k)$ the length of the inputs of X, by $m = \mathsf{poly}(k)$ the number of left sessions (that is the length of vector X), and by $\tilde{m} = \mathsf{poly}(k)$ the number of right sessions. The simulator S performs the preprocessing stage by running the preprocessing stage of the honest verifier on input 1^k. S then receives the modified public file from $\mathcal{A}_0$ and it knows all secret keys associated to the public key that have not been removed by $\mathcal{A}_0$. Some of the entries of the public file output by $\mathcal{A}_0$ are actually entries computed by S (we call these entries *S-controlled*) and some have been added by $\mathcal{A}_0$ (we call these entries

PREPROCESSING STAGE:
Entry l of the public file is constructed by V_0 as follows:

Input: security parameter 1^k.

pick $\mathsf{sk}_0^l, \mathsf{sk}_1^l \leftarrow \{0,1\}^k$, compute $\mathsf{pk}_0^l = f(\mathsf{sk}_0^l)$ and $\mathsf{pk}_1^l = f(\mathsf{sk}_1^l)$, randomly pick $b^l \leftarrow \{0,1\}$, set $\mathsf{pk}^l = (\mathsf{pk}_0^l, \mathsf{pk}_1^l)$ and $\mathsf{sk}^l = (b_l, \mathsf{sk}_{b_l}^l)$.

output: $(\mathsf{pk}, \mathsf{sk})$.

PROOF STAGE:
Sub-protocol: a tag-based cNMWI argument of knowledge $\Pi = \{\langle \mathcal{P}_{\mathtt{tag}}, \mathcal{V}_{\mathtt{tag}} \rangle\}_{\mathtt{tag}}$ for a $\mathbb{NP}$-complete language Λ.

Common input: the public file F, entry $\mathsf{pk}^l = (\mathsf{pk}_0^l, \mathsf{pk}_1^l)$ of F, security parameter 1^k, $n = \mathsf{poly}(k)$-bit string $x \in L$.

P's private input: a witness w for $x \in L$.

V_1's private input: secret key $\mathsf{sk}^l = (b_l, \mathsf{sk}_{b_l}^l)$;

$V_1 \longrightarrow P$: V_1 and P engage in an execution of Π with tag $x \circ 0$ where V_1 runs $\mathcal{P}_{x\circ 0}$ to prove to P (running $\mathcal{V}_{x\circ 0}$) knowledge of a witness (b_l, sk^l) for $\sigma = (\mathsf{pk}_0^l, \mathsf{pk}_1^l) \in L_1$.

$P \longrightarrow V_1$: P and V_1 engage in an execution of Π with tag $x \circ 1$ where P runs $\mathcal{P}_{x\circ 1}$ to prove to V_1 (running $\mathcal{V}_{x\circ 1}$) knowledge of a witness for $\tau = (x, \mathsf{pk}_0^l, \mathsf{pk}_1^l) \in L_2$.

Figure 2: The constant-round BPK cNMZK argument of knowledge $\langle P, V \rangle$ for any $\mathbb{NP}$-language L.

$\mathcal{A}$-controlled). We denote by $\ell = \mathsf{poly}(k)$ the total number of entries in the public file output by $\mathcal{A}$. S then interacts with $\mathcal{A}_1$ in the proof stage on input the public file output by $\mathcal{A}_o$, the security parameter 1^k and the vector X of the left inputs.

We adopt the following notation. We denote by x_i the input of the i-th left proof and by σ_i and τ_i the inputs to the two subprotocols of the i-th left proof. We denote by l_i the entry $(\mathsf{pk}_0^{l_i}, \mathsf{pk}_1^{l_i})$ of the public file of the verifier that is active in the i-th left proof. We use $\tilde{x}_j, \tilde{\sigma}_j, \tilde{\tau}_j$ to denote the corresponding entities for j-th right proofs and r_j to denote the entry of the public file of the verifier that is active in the j-th right proof. Notice that $\tilde{x}_j$ is adaptively chosen by $\mathcal{A}$ whereas x_i is given as input to S. Also, if l-th entry of the public file is S-controlled, we denote by $\mathsf{sk}_{b_l}^l$ the secret key that is known to S. Actually, S could know both secret keys but, as prescribed by the honest verifier algorithm, S only retains a randomly chosen one.

Simulating the view of $\mathcal{A}$. The right proofs involving $\mathcal{A}$-controlled entries of the public file (and thus associated with corrupted verifiers) need not to be simulated; i.e., they are *internal* to $\mathcal{A}$. Similarly, we concentrate on left proofs where the verifier is corrupted and do not show simulation of left proofs relative to honest verifiers as they are internal to S. The interaction in the proof stage between S and $\mathcal{A}$ involves four different types of protocols.

1. $\mathsf{prot}_1^l(i)$, first subprotocol of i-th left proof where $\mathcal{A}$ acts as a prover and S as a verifier. The common input to this protocol is $\sigma_i = (\mathsf{pk}_0^{l_i}, \mathsf{pk}_1^{l_i})$ and entry l_i of public file is $\mathcal{A}$-controlled.

2. $\mathsf{prot}_2^l(i)$, second subprotocol of i-th left proof where $\mathcal{A}$ acts as a verifier and S as a prover.

The common input to this protocol is $\tau_i = (x_i, \mathsf{pk}_0^{l_i}, \mathsf{pk}_1^{l_1})$ and entry l_i of public file is $\mathcal{A}$-controlled.

3. $\mathsf{prot}_1^r(j)$, first subprotocol of j-th right proof where $\mathcal{A}$ acts as a verifier and S as a prover. The common input to this protocol is $\tilde{\sigma}_j = (\mathsf{pk}_0^{r_j}, \mathsf{pk}_1^{r_j})$ and entry r_j of public file is S-controlled.

4. $\mathsf{prot}_2^r(j)$, second subprotocol of j-th right proof where $\mathcal{A}$ acts as a prover and S as a verifier. The common input to this protocol is $\tilde{\tau}_j = (\tilde{x}_j, \mathsf{pk}_0^{r_j}, \mathsf{pk}_1^{r_j})$ and entry r_j of public file is S-controlled.

Simulating right proofs does not pose a problem to S. Indeed in $\mathsf{prot}_1^r(j)$, S executes the code of the honest prover $\mathcal{P}$ of Π for proving that $\tilde{\sigma}_j \in L_1$ using $\mathsf{sk}_{b_{r_j}}^{r_j}$ as a witness. In $\mathsf{prot}_2^r(j)$, S executes the code of the honest verifier $\mathcal{V}$ of Π to verify the proof given by $\mathcal{A}$ that $\tilde{\tau}_j \in L_2$.

Simulating left proofs instead is more problematic. S uses $\mathsf{prot}_1^l(i)$, where $\mathcal{A}$ proves knowledge of one of the two secret keys associates with the l_i-th entry of the public file, to extract one of the two secret keys to be used by S to run the code of the honest prover in $\mathsf{prot}_2^l(i)$. More formally, for the i-th left proof, two cases are possible.

1. S knows $\mathsf{sk}_{\mathcal{A}}^{l_i}$, one of the secret keys corresponding to the l_i-th entry of the public file. In this case, S uses $\mathsf{sk}_{\mathcal{A}}^{l_i}$ as a witness and carries out the first subprotocol as a honest verifier and the second subprotocol as a prover.

2. S does not know any secret key corresponding to the l_i-th entry of public file. In this case S, after the first subprotocol of the i-th proof is completed, stores the current transcript `trans` and starts the extraction procedure of the first subprotocol to obtain one of the two secret keys of that entry. Then S goes back to transcript `trans` and continues as described in item 1 above.

At the end of the simulation S then outputs the transcript `trans`.

The extraction procedure involves rewinding $\mathcal{A}$. We stress though that, since entries of the public file can not be changed once the proof stage has started, each time S obtains a secret key, it can be used for all proofs relative to the same entry of the public file and that the number of entries is polynomially bounded.

Before describing how S extracts the witnesses for the right proofs, we show that the transcript output by S is indistinguishable from the view of $\mathcal{A}$.

S is a good simulator. We observe that the difference between the view of $\mathcal{A}$ during a concurrent man-in-the-middle attack and the transcript output by S is in the witnesses encoded in $\mathsf{prot}_2^l(i)$, $i = 1, \ldots, m$. Indeed there the simulator uses as witness a secret key of $\mathcal{A}$ that has been previously extracted. However, if the output of S and the view of $\mathcal{A}$ can be distinguished then it is possible to break the adaptive concurrent witness indistinguishability of the subprotocol (here we do not need to break the non-malleable witness indistinguishability of Π).

More precisely, suppose that for a concurrent man-in-the-middle adversary $\mathcal{A}$ there exists a distinguisher D that distinguishes the transcript output by $S(X, z)$ from $\mathsf{BView}_{\mathcal{A}}(X, W, z)$. That is, there exists a constant $c > 0$ and $m = \mathsf{poly}(k)$ and $n = \mathsf{poly}(k)$ such that for infinitely many k there exists a vector $X = (x_1, \cdots, x_m)$ of m elements of L of length n and a vector $W = (w_1, \cdots, w_m)$ of witnesses for X such that

$$|\text{Prob}[\ \alpha \leftarrow \mathsf{BView}_{\mathcal{A}}(X, W, z) : D(\alpha) = 1\ ] - \text{Prob}[\ \alpha \leftarrow S(X, z) : D(\alpha) = 1\ ]| > k^{-c}. \quad (1)$$

Then we can construct the following adaptive concurrent witness indistinguishability adversary M for Π. M uses D and $\mathcal{A}$ as subroutine and, on input 1^k, M has as auxiliary information a triplet (X, W, z) for which Equation 1 holds. In addition M has access to a prover $\mathcal{P}^\star$ that can be invoked on adaptively selected input τ along with two witnesses (ω_0, ω_1) for $\tau \in L_2$. Prover $\mathcal{P}^\star$ has random bit $b \in \{0,1\}$ wired-in and, for each invocation on input τ and witnesses (ω_0, ω_1), it executes the algorithm of the honest prover $\mathcal{P}$ on input τ and ω_b. M's goal is to guess b. M works in three phases. In the first phase M executes S's algorithm so to construct a public file that is then given as input to $\mathcal{A}_0$. $\mathcal{A}_0$ outputs a modified public file which will be used for the two following phases. In the second phase, M interacts with $\mathcal{A}_1$ (still following S's algorithm) and obtains, by using rewinding, a secret key for each entry of the public file used by $\mathcal{A}$ in a left interaction. In the third phase, M starts a new interaction with $\mathcal{A}_1$ (using the same public file) by executing S's algorithm with the following exception: for left proof i, instead of executing $\mathsf{prot}_2^l(i)$, M invokes $\mathcal{P}^\star$ on input $\tau_i = (x_i, \mathsf{pk}_0^{l_i}, \mathsf{pk}_1^{l_i})$ and $(w_i, \mathsf{sk}_{\mathcal{A}}^{l_i})$ and has $\mathcal{A}$ interact with $\mathcal{P}^\star$. Here w_i is a witness for $x_i \in L$ and $\mathsf{sk}_{\mathcal{A}}^{l_i}$ is a secret key associated with the public keys $(\mathsf{pk}_0^{l_i}, \mathsf{pk}_1^{l_i})$ of the l_i-th entry of the public file. Also, notice that $\mathcal{A}$ picks the pair $(\mathsf{pk}_0^{l_i}, \mathsf{pk}_1^{l_i})$ and thus τ_i is adaptively chosen by $\mathcal{A}$. For the right proofs instead M just relays messages between $\mathcal{A}$ and honest external verifiers. Notice that M does not need any rewind in the third phase. At the end of the third phase, M returns the output of D on input the transcript of the interaction (including left and right proofs and the public file produced by $\mathcal{A}_0$).

Observe that if $b = 0$ then $\mathcal{P}^\star$ uses the first witness provided (that is w_i) in each invocation and thus the obtained transcript is perfectly distributed according to $\mathsf{BView}_{\mathcal{A}}(X, W, z)$. If instead $b = 1$ then $\mathcal{P}^\star$ always uses the second witness provided (that is $\mathsf{sk}_{\mathcal{A}}^{l_i}$) and thus the obtained transcript is perfectly distributed according to the output of $S(X, z)$. Therefore M breaks the concurrent witness indistinguishability of $(\mathcal{P}, \mathcal{V})$.

Extracting the witnesses. We now show how S extracts witnesses from the right proofs, thus concluding the description of simulator S. S obtains the witnesses for the right proofs of `trans` by executing the algorithm of the honest prover in the left proofs (using as witnesses the extracted secret keys of the adversary) and by running the extractor for the adaptive cNMWI argument of knowledge one-by-one for each right proof, sequentially. Note that a rewind procedure could fail since new left proofs could be opened by the adversary with respect to an entry of the public file that has not been used previously (and thus the simulator does not know any secret key for it). However, the extraction procedure can be simply repeated a polynomial number of times and there will be an execution that will not suffer this problem. This follows from the fact that in the first execution the adversary chose only some entries of the public file, and thus we can assume that a subset of those entries will be chosen again with non-negligible probability. Let now w_j be the the witness extracted by S for the j-th right proof. We now show that if $\mathcal{A}$ convinces $\mathcal{V}$ in the j-th right proof on input $\tilde{x}_j$ that does not appear as an input in any left proof then, except with negligible probability, w_j is a witness for $\tilde{x}_j \in L$. Suppose that there are infinitely many (X, z, k) such that, with some non-negligible probability, there exists some j for which w_j is not a witness for $\tilde{x}_j \in L$ and $\tilde{x}_j$ is not an input in any left proof. We call such a triplet (X, z, k) a *problematic* triplets. We distinguish the following two cases.

Case 1. There exist infinitely many problematic triplets (X, z, k) for which S outputs $(w_1, \ldots, w_m)$ and, with non-negligible probability, for at least one j we have $w_j = \mathsf{sk}_{1-b_{r_j}}^{r_j}$. We call such a triplet (X, z, k) an *inverting* triplet. Then, it is easy to see that $\mathcal{A}$ can be used to invert the one-way function f used to compute the public keys. More precisely, consider the following in-

verting algorithm I for a one-way function f. I, for inputs of length k, has wired-in (X, z) such that (X, z, k) is an inverting triplet (if no such inverting triplet exists then I fails for length k). I receives as a challenge y and has to compute $x \in \{0,1\}^k$ such that $y = f(x)$. I constructs the public file in the following way: I picks r at random and computes the r-th entry of the public file by picking $b_r \leftarrow \{0,1\}$ at random and $\mathsf{sk}^r_{b_r}$ at random and setting $\mathsf{pk}^r_{b_r} = f(\mathsf{sk}^r_{b_r})$ and $\mathsf{pk}^r_{1-b_r} = y$. All remaining entries of the public file are computed following the code of honest verifier. Then I has S interact with $\mathcal{A}$ on input (X, z). By assumption, with some non-negligible probability, entry r is used in a right proof from which S extracts sk^{1-b_r} such that $f(\mathsf{sk}^{1-b_r}) = \mathsf{pk}^r_{1-b_r} = y$. This violates the one-wayness of f as I succeeds with non-negligible probability for infinitely many k.

Case 2. Suppose now that there exist only finitely many inverting triplets and thus, for infinitely many problematic triplets (X, z, k), S outputs $(w_1, \ldots, w_m)$ and, with some non-negligible probability, for all j for which w_j is not a witness for $\tilde{x}_j \in L$ we have $w_j = \mathsf{sk}^{r_j}_{b_{r_j}}$. We call such a triplet (X, z, k) a *copying* triplet. Clearly a problematic triplet that is not inverting must be copying.

For a vector $B = (b_1, \cdots, b_\ell)$ we define by S^B the algorithm S where we fix the secret key $\mathsf{sk}^i_{b_i}$ associated with the i-th S-controlled entry known to S. Then, for a random vector B and a copying triplet (X, z, k), S^B, on input (X, z, k), extracts $(w_1, \ldots, w_m)$ and the following event occurs with some non-negligible probability: there exists at least a j such that w_j is not a witness for $\tilde{x}_j$ and for all such j's we have $w_j = \mathsf{sk}^{r_j}_{b_{r_j}}$. We next show how to turn $\mathcal{A}$ into an adversary M that breaks the concurrent non-malleable witness indistinguishability of the subprotocol Π.

The interaction between S^B and $\mathcal{A}$ involves four types of subprotocols:

1. $\mathsf{prot}^l_1(i)$, first subprotocol of i-th left proof where $\mathcal{A}$ acts as a prover and S^B behaves like a honest verifier;

2. $\mathsf{prot}^l_2(i)$, second subprotocol of i-th left proof where $\mathcal{A}$ acts as a verifier and S^B uses knowledge of a secret key $\mathsf{sk}^{l_i}_A(B)$ extracted from $\mathcal{A}$ in order to execute the code of a honest prover;

3. $\mathsf{prot}^r_1(j)$, first subprotocol of j-th right proof where $\mathcal{A}$ acts as a verifier and S uses knowledge of $\mathsf{sk}^{r_j}_{b_{r_j}}$ to execute the code of the honest verifier;

4. $\mathsf{prot}^r_2(j)$, second subprotocol of j-th right proof where $\mathcal{A}$ acts as a prover and S^B as a honest verifier.

By assumption we know that, for a copying triplet, the witness encoded in $\mathsf{prot}^r_2(j)$ by $\mathcal{A}$ is related to $\mathsf{sk}^{r_i}_{b_{r_i}}$ and we will exploit this dependency to break the concurrent non-malleable witness indistinguishability of the subprotocol Π. To do so, we show an adaptive concurrent man-in-the-middle adversary M that receives in input security parameter 1^k, has wired in a copying triplet (X, z, k), and has oracle access to prover $\mathcal{P}'$ that has a randomly chosen $b \in \{0,1\}$ wired-in. $\mathcal{P}'$, when invoked on input x and witnesses (ω_0, ω_1) for x, executes the code of the honest prover $\mathcal{P}$ on input x and ω_b [5]. We pick two vectors B_0 and B_1 and show that when M interacts with $\mathcal{P}^\star$ the witness encoded in the right proofs produced by M are distributed like the witnesses

[5] We stress that Π is an argument system for an $\mathbb{NP}$-complete language Λ and in the description of M we will invoke $\mathcal{P}'$ to prove membership in an $\mathbb{NP}$-language L. What we actually mean is that the input (and the witnesses for membership to L) are first reduced to an instance of Λ (and to witnesses for membership to Λ) and $\mathcal{P}'$ is invoked for using witness ω_b.

encoded in the proofs produced by $\mathcal{A}$ when interacting with S^{B_b} (b is the hidden bit of $\mathcal{P}^\star$). This contradicts the concurrent non-malleable witness indistinguishability of Π. Let us now describe M.

M starts interacting with $\mathcal{A}$ on input 1^k by constructing a public file and submitting it to $\mathcal{A}_0$. For each entry constructed by M, M retains both associated secret keys. Once M receives the modified public file from $\mathcal{A}_0$, M starts interacting with $\mathcal{A}_1$ on input (X, z). Specifically, M picks two vectors $B_0 = (b_1^0, \cdots, b_\ell^0)$ and $B_1 = (b_1^1, \cdots, b_\ell^1)$ and has $\mathcal{A}_1$ interact with S^{B_0}. At the end of the interaction M obtains a secret key for each $\mathcal{A}$-controlled entry of the public file used by $\mathcal{A}_1$ in the interaction; we denote by $\mathsf{sk}_{\mathcal{A}}^l(B_0)$ the obtained secret key relative to the l-th $\mathcal{A}$-controlled entry of the public file. Then M starts again (with the same public file) and has $\mathcal{A}_1$ interact with S^{B_1}. As before we obtain secret keys associated to $\mathcal{A}$-controlled entries of the public file that have been used in the interaction and we denote by $\mathsf{sk}_{\mathcal{A}}^r(B_1)$ the obtained secret key relative to the r-th entry of the public file.

Finally M starts the actual attack. More precisely, M interacts with $\mathcal{P}^\star$ (on the left), with honest verifiers (on the right) and internally with $\mathcal{A}_1$ in the following way.

1. $\mathsf{prot}_1^l(i)$. Here $\mathcal{A}_1$ acts as a prover and M behaves like a honest verifier;
2. $\mathsf{prot}_2^l(i)$. M acts as intermediary between $\mathcal{P}^\star$ and $\mathcal{A}_1$. $\mathcal{P}^\star$ is run on input τ_i and is provided with witnesses $(\mathsf{sk}_{\mathcal{A}}^{l_i}(B_0), \mathsf{sk}_{\mathcal{A}}^{l_i}(B_1))$.
3. $\mathsf{prot}_1^r(j)$. M acts as intermediary between $\mathcal{P}^\star$ and $\mathcal{A}_1$. $\mathcal{P}^\star$ is run on input $\tilde{\sigma}_j$ and witnesses $(\mathsf{sk}_{B_{r_j}^0}^{r_j}, \mathsf{sk}_{B_{r_j}^1}^{r_j})$.
4. $\mathsf{prot}_2^r(j)$. Here M acts as intermediary between $\mathcal{A}_1$ and the honest verifier.

We notice that, while interacting with $\mathcal{P}'$, M does not perform any rewind. In the attack of M we consider executions of prot_1^r as left proofs (which are provided by $\mathcal{P}'$) and executions of $\mathsf{prot}_2^r(j)$ as right proofs. Therefore all right proofs are actually produced by $\mathcal{A}_1$. Moreover observe that all instances of prot_1^r are executions of Π with tag ending in 0 and all instances of prot_2^r are executions of Π with tag ending in 1. Therefore all right proofs will have tags different from those used in the left proofs and thus M is a legal tag-based concurrent man-in-the-middle adversary. Let us now look at the distributions of the witnesses encoded in the right proofs when $b = 0$ and when $b = 1$. If $b = 0$ then the view of $\mathcal{A}_1$ is exactly the same as the view of $\mathcal{A}_1$ when interacting with S^{B_0}. Therefore the witnesses encoded in the proofs produced by M are the same as the witnesses encoded in the proofs produced by $\mathcal{A}_1$ when interacting with S^{B_0}. By our hypothesis, with some non-negligible probability, these witnesses are either witnesses for membership in L or the keys used by S^{B_0}. Similarly if $b = 1$ then the view of $\mathcal{A}_1$ is exactly the same as the view of $\mathcal{A}_1$ when interacting with S^{B_1}. Therefore by our hypothesis, with some non-negligible probability, the witnesses encoded in the proofs given in output by $\mathcal{A}_1$ are either witnesses for membership in L or the keys used by S^{B_1}. The two distribution are clearly distinguishable thus breaking the tag-based concurrent non-malleable witness indistinguishability of Π. □

Theorem 6.4 *Assume that there exists a family of claw-free permutations. Then there exists a constant-round cNMZK* BPK *argument of knowledge for all* $\mathbb{NP}$.

Proof. The proof follows by Lemma 6.3, and by the observation that claw-free permutations imply the existence of one-way functions. □

Acknowledgments

We thank Rafael Pass and Alon Rosen for interesting and fruitful discussions.

References

[1] Boaz Barak. How to go beyond the black-box simulation barrier. In *42nd Annual Symposium on Foundations of Computer Science*, pages 106–115, Las Vegas, Nevada, USA, October 14–17, 2001. IEEE Computer Society Press.

[2] Boaz Barak. Constant-round coin-tossing with a man in the middle or realizing the shared random string model. In *43rd Annual Symposium on Foundations of Computer Science*, pages 345–355, Vancouver, British Columbia, Canada, November 16–19, 2002. IEEE Computer Society Press.

[3] Boaz Barak, Ran Canetti, Jesper Buus Nielsen, and Rafael Pass. Universally composable protocols with relaxed set-up assumptions. In *45th Annual Symposium on Foundations of Computer Science*, pages 186–195. IEEE Computer Society Press, 2004.

[4] Boaz Barak, Oded Goldreich, Shafi Goldwasser, and Yehuda Lindell. Resettably-sound zero-knowledge and its applications. In *42nd Annual Symposium on Foundations of Computer Science*, pages 116–125, Las Vegas, Nevada, USA, October 14–17, 2001. IEEE Computer Society Press.

[5] Boaz Barak, Manoj Prabhakaran, and Amit Sahai. Concurrent non-malleable zero knowledge. In *47th Annual Symposium on Foundations of Computer Science*. IEEE Computer Society Press, 2006.

[6] Boaz Barak and Amit Sahai. How to play almost any mental game over the net - concurrent composition via super-polynomial simulation. In *46th Annual Symposium on Foundations of Computer Science*, pages 543–552. IEEE Computer Society Press, 2005.

[7] Ran Canetti, Oded Goldreich, Shafi Goldwasser, and Silvio Micali. Resettable zero-knowledge. In *32nd Annual ACM Symposium on Theory of Computing*, pages 235–244, Portland, Oregon, USA, May 21–23, 2000. ACM Press.

[8] Ran Canetti, Yehuda Lindell, Rafail Ostrovsky, and Amit Sahai. Universally composable two-party and multi-party secure computation. In *34th Annual ACM Symposium on Theory of Computing*, pages 494–503, Montreal, Québec, Canada, May 19–21, 2002. ACM Press.

[9] 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*, volume 2139 of *Lecture Notes in Computer Science*, pages 566–598, Santa Barbara, CA, USA, August 19–23, 2001. Springer-Verlag, Berlin, Germany.

[10] Giovanni Di Crescenzo, Giuseppe Persiano, and Ivan Visconti. Constant-round resettable zero knowledge with concurrent soundness in the bare public-key model. In Matthew Franklin, editor, *Advances in Cryptology – CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 237–253, Santa Barbara, CA, USA, August 15–19, 2004. Springer-Verlag, Berlin, Germany.

[11] Giovanni Di Crescenzo and Ivan Visconti. Concurrent zero knowledge in the public-key model. In Lus Caires, Giuseppe F. Italiano, Lus Monteiro, Catuscia Palamidessi, and Moti Yung, editors, *Automata, Languages and Programming: 32nd International Colloquium*, volume 3580 of *Lecture Notes in Computer Science*, pages 816–827, Lisbon, Portugal, July 11–15, 2005. Springer-Verlag, Berlin, Germany.

[12] Danny Dolev, Cynthia Dwork, and Moni Naor. Non-malleable cryptography. In *23rd Annual ACM Symposium on Theory of Computing*, pages 542–552, New Orleans, Louisiana, USA, May 6–8, 1991. ACM Press.

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

[14] Cynthia Dwork and Moni Naor. ZAPs and their applications. In *41st Annual Symposium on Foundations of Computer Science*, pages 283–293, Redondo Beach, California, USA, November 12–14, 2000. IEEE Computer Society Press.

[15] Cynthia Dwork, Moni Naor, and Amit Sahai. Concurrent zero-knowledge. In *30th Annual ACM Symposium on Theory of Computing*, pages 409–418, Dallas, Texas, USA, May 23–26, 1998. ACM Press.

[16] Uriel Feige. *Alternative Models for Zero Knowledge Interactive Proofs.* Weizmann Institute of Science, 19990.

[17] Uriel Feige, Dror Lapidot, and Adi Shamir. Multiple NonInteractive Zero Knowledge Proofs under General Assumptions. *SIAM Journal on Computing*, 29:1–28, 1999.

[18] Uriel Feige and Adi Shamir. Witness indistinguishable and witness hiding protocols. In *22nd Annual ACM Symposium on Theory of Computing*, pages 416–426, Baltimore, Maryland, USA, May 14–16, 1990. ACM Press.

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

[20] Oded Goldreich and Hugo Krawczyk. On the composition of Zero-Knowledge Proof systems. *SIAM Journal on Computing*, 25(1):169–192, 1996.

[21] Oded Goldreich, Silvio Micali, and Avi Wigderson. Proofs that yield nothing but their validity or all languages in np have zero-knowledge proof systems. *Journal of the ACM*, 38(1):691–729, 1991.

[22] Oded Goldreich and Oren. Bit Commitment using Pseudorandomness. *Journal of Cryptology*, 4:151–158, 1994.

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

[24] Jens Groth, Rafail Ostrovsky, and Amit Sahai. Non-interactive ZAPs and New Techniques for NIZK. In Cynthia Dwork, editor, *Advances in Cryptology – CRYPTO 2006*, volume 4117 of *Lecture Notes in Computer Science*, pages 97–11, Santa Barbara, CA, USA, August 20–24, 2006. Springer-Verlag, Berlin, Germany.

[25] Yael Tauman Kalai, Yehuda Lindell, and Manoj Prabhakaran. Concurrent general composition of secure protocols in the timing model. In *37th Annual ACM Symposium on Theory of Computing*, pages 644–653. ACM Press, 2005.

[26] Joe Kilian. *Uses of randomness in Algorithms and Protocols.* MIT Press, Cambridge, MA, 1990.

[27] Joe Kilian and Erez Petrank. Concurrent and resettable zero-knowledge in poly-loalgorithm rounds. In *33rd Annual ACM Symposium on Theory of Computing*, pages 560–569, Crete, Greece, July 6–8, 2001. ACM Press.

[28] Silvio Micali, Rafael Pass, and Alon Rosen. Input-indistinguishable computation. In *47th Annual Symposium on Foundations of Computer Science*, pages 136–145. IEEE Computer Society Press, 2006.

[29] Silvio Micali and Leonid Reyzin. Soundness in the public-key model. In Joe Kilian, editor, *Advances in Cryptology – CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 542–565, Santa Barbara, CA, USA, August 19–23, 2001. Springer-Verlag, Berlin, Germany.

[30] Rafael Pass and Alon Rosen. Concurrent non-malleable commitments. In *46th Annual Symposium on Foundations of Computer Science*, pages 563–572. IEEE Computer Society Press, 2005.

[31] Rafael Pass and Alon Rosen. New and Improved Constructions of Non-Malleable Cryptographic Protocols. In *37th Annual ACM Symposium on Theory of Computing*, pages 533–542. ACM Press, 2005.

[32] Rafael Pass and Alon Rosen. Concurrent non-malleable commitments (full version). http://www.eecs.harvard.edu/∼alon/PAPERS/conc-nmc/conc-nmc.ps, 2006.

[33] Manoj Prabhakaran and Amit Sahai. New notions of security: achieving universal composability without trusted setup. In *36th Annual ACM Symposium on Theory of Computing*, pages 242–251. ACM Press, 2004.

[34] Leonid Reyzin. *Zero-Knowledge with Public Keys, Ph.D. Thesis.* MIT Press, Cambridge, MA, 2001.

[35] Ransom Richardson and Joe Kilian. On the concurrent composition of zero-knowledge proofs. In Jacques Stern, editor, *Advances in Cryptology – EUROCRYPT'99*, volume 1592 of *Lecture Notes in Computer Science*, pages 415–431, Prague, Czech Republic, May 2–6, 1999. Springer-Verlag, Berlin, Germany.

[36] John Rompel. One-way functions are necessary and sufficient for secure signatures. In *22nd Annual ACM Symposium on Theory of Computing*, pages 387–394, Baltimore, Maryland, USA, May 14–16, 1990. ACM Press.