New and Improved Constructions of Non-Malleable Cryptographic Protocols [*]

Rafael Pass [†]   Alon Rosen [‡]

Abstract

We present a new constant round protocol for non-malleable zero-knowledge. Using this protocol as a subroutine, we obtain a new constant-round protocol for non-malleable commitments. Our constructions rely on the existence of (standard) collision resistant hash functions. Previous constructions either relied on the existence of trapdoor permutations and hash functions that are collision resistant against sub-exponential sized circuits, or required a super-constant number of rounds. Additional results are the first construction of a non-malleable commitment scheme that is statistically hiding (with respect to opening), and the first non-malleable commitments that satisfy a strict polynomial-time simulation requirement.

Our approach differs from the approaches taken in previous works in that we view non-malleable zero-knowledge as a building-block rather than an end goal. This gives rise to a modular construction of non-malleable commitments and results in a somewhat simpler analysis.

Keywords: Cryptography, zero-knowledge, non-malleability, man-in-the-middle, round-complexity, non black-box simulation

[*] Preliminary version appeared in STOC 2005, pages 533–542.

[†] Department of Computer Science. Cornell University, Ithaca, NY. E-mail: `rafael@cs.cornell.edu`. Part of this work done while at CSAIL, MIT, Cambridge, MA.

[‡] Center for Research on Computation and Society (CRCS). DEAS, Harvard University, Cambridge, MA. E-mail: `alon@eecs.harvard.edu` . Part of this work done while at CSAIL, MIT, Cambridge, MA.

Contents

1 Introduction

Consider the execution of two-party protocols in the presence of an adversary that has full control of the communication channel between the parties. The adversary has the power to omit, insert or modify messages at its choice. It has also full control over the scheduling of the messages. The honest parties are not necessarily aware to the existence of the adversary, and are not allowed to use any kind of trusted set-up (such as a common reference string).

The above kind of attack is often referred to as a *man-in-the-middle* attack. It models a natural scenario whose investigation is well motivated. Protocols that retain their security properties in face of a man-in-the-middle are said to be *non-malleable* [16]. Due to the hostile environment in which they operate, the design and analysis of non-malleable protocols is a notoriously difficult task. The task becomes even more challenging if the honest parties are not allowed to use any kind of trusted set-up. Indeed, only a handful of such protocols have been constructed so far.

The rigorous treatment of two-party protocols in the man-in-the-middle setting has been initiated in the seminal paper by Dolev, Dwork and Naor [16]. The paper contains definitions of security for the tasks of non-malleable commitment and non-malleable zero-knowledge. It also presents protocols that meet these definitions. The protocols rely on the existence of one-way functions, and require $O(\log n)$ rounds of interaction, where $n \in N$ is a security parameter.

A more recent result by Barak presents constant-round protocols for non-malleable commitment and non-malleable zero-knowledge [2]. This is achieved by constructing a coin-tossing protocol that is secure against a man in the middle, and then using the outcome of this protocol to instantiate known constructions for non-malleable commitment and zero-knowledge in the common reference string model. The proof of security makes use of non black-box techniques and is highly complex. It relies on the existence of trapdoor permutations and hash functions that are collision-resistant against sub-exponential sized circuits.

In this paper we continue the line of research initiated by the above papers. We will be interested constructions of new constant-round protocols for non-malleable commitment and non-malleable zero-knowledge, and will not rely on any kind of set-up assumption.

1.1 Non-Malleable Protocols

In accordance with the above discussion, consider a man-in-the-middle adversary A that is simultaneously participating in two executions of a two-party protocol. These executions are called the left and the right interaction. Besides controlling the messages that it sends in the left and right interactions, A has control over the scheduling of the messages. In particular, it may delay the transmission of a message in one interaction until it receives a message (or even multiple messages) in the other interaction.

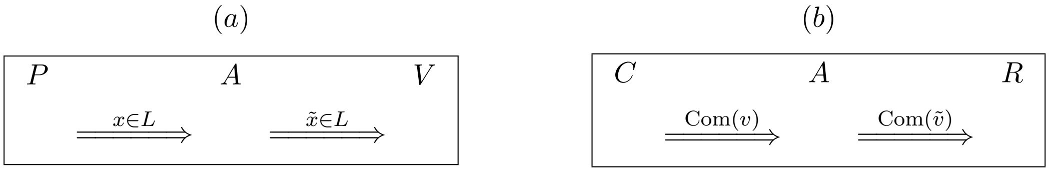

Figure 1: The man-in-the-middle adversary. (a) Interactive proofs. (b) Commitments.

The adversary is trying to take advantage of its participation in the left interaction in order to violate the security of the protocol executed in the right interaction, where the exact interpretation of the term ”violate the security” depends on the specific task at hand.

A two-party protocol is said to be non-malleable if the left interaction does not "help" the adversary in violating the security of the right interaction. Following the simulation paradigm [26, 27, 23, 24], this is formalized by defining appropriate "real" and "idealized" executions.

In the real execution, called the man-in-the-middle execution, the adversary participates in both the left and the right interactions. In the idealized execution, called the stand-alone execution, the adversary is only participating in a single interaction. Security is defined by requiring that the adversary cannot succeed better in the man-in-the middle execution than he could have in the stand-alone execution. In the specific instances of zero-knowledge and string commitment, the definition of security takes the following forms.

Non-malleable zero-knowledge [16]. Let $\langle P, V\rangle$ be an interactive proof system. In the left interaction the adversary A is verifying the validity of a statement x by interacting with an honest prover P. In the right interaction A proves the validity of a statement $\tilde{x} \neq x$ to the honest verifier V (see Figure 1.a). The objective of the adversary is to convince the verifier in the right interaction that $\tilde{x} \in L$. Non-malleability of $\langle P, V\rangle$ is defined by requiring that for any man-in-the-middle adversary A, there exists a stand-alone prover S that manages to convince the verifier with essentially the same probability as A. The interactive proof $\langle P, V\rangle$ is said to be non-malleable zero-knowledge if it is non-malleable and (stand-alone) zero-knowledge.

Non-malleable commitments [16]. Let $\langle C, R\rangle$ be a commitment scheme. In the left interaction the adversary A is receiving a commitment to a value v from the committer C. In the right interaction A is sending a commitment to a value $\tilde{v}$ to the receiver R (see Figure 1.b). The objective of the adversary is to succeed in committing in the right interaction to a value $\tilde{v} \neq v$ that satisfies $\mathcal{R}(v, \tilde{v}) = 1$ for some poly-time computable relation $\mathcal{R}$. Non-malleability of $\langle C, R\rangle$ is defined by requiring that for any man-in-the-middle adversary A, there exists a stand-alone committer S that manages to commit to the related $\tilde{v}$ with essentially the same probability as A.

Schemes that satisfy the above definition are said to be non-malleable with respect to commitment. In a different variant, called non-malleable commitment with respect to opening [19], the adversary is considered to have succeeded only if it manages to *decommit* to a related value $\tilde{v}$.

1.2 Our Contributions

Our main result is the construction of a new constant-round protocol for non-malleable $\mathcal{ZK}$. The proof of security relies on the existence of (ordinary) collision resistant hash functions and does not rely on any set-up assumption.

Theorem 1 (Non-malleable $\mathcal{ZK}$) *Suppose that there exists a family of collision resistant hash functions. Then, there exists a constant-round non-malleable $\mathcal{ZK}$ argument for every $L \in \mathcal{NP}$.*

Theorem 1 is established using the notion of *simulation extractability.* A protocol is said to be simulation extractable if for any man-in-the-middle adversary A, there exists a simulator-extractor that can simulate the views of both the left and the right interactions for A, while outputting a witness for the statement proved by A in the right interaction. Any protocol that is simulation-extractable is also non-malleable $\mathcal{ZK}$. The main reason for using simulation extractability (which is more technical in flavor than non-malleability) is that it is easier to work with.

Using our new simulation extractable protocols as a subroutine, we construct constant round protocols for non-malleable string commitment. One of our constructions achieves statistically binding commitments that are non-malleable w.r.t. commitment, and the other achieves statistically hiding commitments that are non-malleable w.r.t. opening.

Theorem 2 (Statistically binding non-malleable commitment) *Suppose that there exists a family of collision-resistant hash functions. Then, there exists a constant-round statistically binding commitment scheme that is non malleable with respect to commitment.*

Theorem 3 (Statistically hiding non-malleable commitment) *Suppose that there exists a family of collision-resistant hash functions. Then, there exists a constant-round statistically hiding commitment scheme that is non malleable with respect to opening.*

Underlying cryptographic assumptions. The main quantitative improvement of our construction over the constant round protocols in [2] is in the underlying cryptographic assumption. Our constructions rely on the existence of ordinary collision resistant hash functions. The protocols in [2] relied on the existence of both trapdoor permutations and hash functions that are collision resistant against sub exponential sized circuits. The constructions in [16] assumed only the existence of one-way functions, but had a super-constant number of rounds.

Statistically hiding non-malleable commitments. Theorem 3 gives the first construction of a non-malleable commitment scheme that is statistically hiding and that does not rely on set-up assumptions. We mention that the existence of collision resistant hash functions is the weakest assumption currently known to imply constant round statistically hiding commitment schemes (even those that are not of the non-malleable kind) [36, 12].

Strict vs. liberal non-malleability. The notion of non malleability that has been considered so far in all works, allows the stand alone adversary S to run in expected polynomial time. A stronger ("tighter") notion of security, named strict non-malleability [16], requires S to run in strict polynomial time. In the context of strict non-malleability, we have the following result.

Theorem 4 (Strict non-malleability) *Suppose that there exists a family of collision resistant hash functions. Then, There exists a constant-round statistically binding commitment scheme that is strictly non-malleable with respect to commitment.*

1.3 Techniques and New Ideas

Our protocols rely on non black-box techniques used by Barak to obtain constant-round public-coin $\mathcal{ZK}$ argument for $\mathcal{NP}$ [1] (in a setting where no man in the middle is considered). They are closely related to previous works by Pass [38], and Pass and Rosen [39] that appeared in the context of bounded-concurrent two-party and multi-party computation; in particular our protocols rely and further explore the technique from [38] of using *message-lengths* to obtain non-malleability. Our techniques are different than the ones used by Barak in the context of non-malleable coin-tossing [2].

The approach we follow in this work is fundamentally different than the approach used in [16]. Instead of viewing non-malleable commitments as a tool for constructing non-malleable $\mathcal{ZK}$ protocols, we reverse the roles and use non-malleable $\mathcal{ZK}$ protocols in order to construct non-malleable commitments. Our approach is also different from the one taken by [2], who uses a coin-tossing protocol to instantiate constructions that rely on the existence of a common reference string.

Our approach gives rise to a modular and natural construction of non-malleable commitments. This construction emphasizes the role of non-malleable $\mathcal{ZK}$ as a building block for other non-malleable cryptographic primitives. In proving the security of our protocols, we introduce the notion of *simulation extractability*, which is a convenient form of non-malleability (in particular, it enables a more modular construction of proofs). A generalization of simulation-extractability, called one-many simulation extractability, has already been found to be useful in constructing commitment schemes that retain their non-malleability properties even if executed concurrently an unbounded (polynomial) number of times [40].

In principle, our definitions of non-malleability are compatible with the ones appearing in [16]. However, the presentation is more detailed and somewhat different (see Section 3). Our definitional approach, as well as our construction of non-malleable $\mathcal{ZK}$ highlights a distinction between the notions of non-malleable interactive proofs and non-malleable $\mathcal{ZK}$. This distinction was not present in the definitions given in [16].

1.4 Related Work

Assuming the existence of a common random string, Di Crescenzo, Ishai and Ostrovsky [15], and Di Crescenzo, Katz, Ostrovsky, and Smith [14] construct non-malleable commitment schemes. Sahai [41], and De Santis, Di Crescenzo, Ostrovsky, Persiano and Sahai [13] construct a non-interactive non-malleable $\mathcal{ZK}$ protocol under the same assumption. Fischlin and Fischlin [19], and Damgård and Groth [11] construct non-malleable commitments assuming the existence of a common reference string. We note that the non-malleable commitments constructed in [15] and [19] only satisfy non-malleability with respect to opening [19]. Canetti and Fischlin [9] construct a universally composable commitment assuming a common random string. Universal composability implies non malleability. However, it is impossible to construct universally composable commitments without making set-up assumptions [9].

Goldreich and Lindell [22], and Nguyen and Vadhan [37] consider the task of session-key generation in a setting where the honest parties share a password that is taken from a relatively small dictionary. Their protocols are designed having a man-in-the-middle adversary in mind, and only requires the usage of a "mild" set-up assumption (namely the existence of a "short" password).

1.5 Future and Subsequent Work

Our constructions (and even more so the previous ones) are quite complex. A natural question is whether they can be simplified. A somewhat related question is whether non-black box techniques are necessary for achieving constant-round non-malleable $\mathcal{ZK}$ or commitments. Our constructions rely on the existence of collision resistant hash functions, whereas the non constant-round construction in [16] relies on the existence of one-way functions. We wonder whether the collision resistance assumption can be relaxed.

Another interesting question (which has been already addressed in subsequent work – see below) is whether it is possible to achieve non-malleability under concurrent executions. The techniques used in this paper do not seem to extend to the (unbounded) concurrent case and new ideas seem to be required. Advances in that direction might shed light on the issue of concurrent composition of general secure protocols.

In subsequent work [40], we show that (a close variant of) the commitments presented here will retain their non-malleability even if executed concurrently an unbounded (polynomial) number of times. We note that besides using an almost identical protocol, the proof of this new result heavily relies on a generalization of simulation extractability (called "one-many" simulation extractability). This notion has proved itself very useful in the context of non-malleability, and we believe that it will find more applications in scenarios where a man-in-the-middle adversary is involved. We additionally mention that the presentation of some of the results in this version of the paper incorporate simplification developed by us in [40].

2 Preliminaries

2.1 Basic notation

We let N denote the set of all integers. For any integer $m \in N$, denote by $[m]$ the set $\{1, 2, \ldots, m\}$. For any $x \in \{0,1\}^*$, we let $|x|$ denote the size of x (i.e., the number of bits used in order to write it). For two machines M, A, we let $M^A(x)$ denote the output of machine M on input x and given oracle access to A. The term negligible is used for denoting functions that are (asymptotically) smaller than one over any polynomial. More precisely, a function $\nu(\cdot)$ from non-negative integers to reals is called negligible if for every constant $c > 0$ and all sufficiently large n, it holds that $\nu(n) < n^{-c}$.

2.2 Witness Relations

We recall the definition of a witness relation for an $\mathcal{NP}$ language [20].

Definition 2.1 (Witness relation) *A witness relation for a language $L \in \mathcal{NP}$ is a binary relation R_L that is polynomially bounded, polynomial time recognizable and characterizes L by*

$$L = \{x : \exists y\, s.t.\, (x, y) \in R_L\}$$

We say that y is a witness for the membership $x \in L$ if $(x, y) \in R_L$ (also denoted $R_L(x, y) = 1$). We will also let $R_L(x)$ denote the set of witnesses for the membership $x \in L$, i.e.,

$$R_L(x) = \{y : (x, y) \in L\}$$

In the following, we assume a fixed witness relation $R_L(x, y)$ for each language $L \in \mathcal{NP}$.

2.3 Probabilistic notation

Denote by $x \overset{\text{R}}{\leftarrow} X$ the process of uniformly choosing an element x in a set X. If $B(\cdot)$ is an event depending on the choice of $x \overset{\text{R}}{\leftarrow} X$, then $\Pr_{x \leftarrow X}[B(x)]$ (alternatively, $\Pr_x[B(x)]$) denotes the probability that $B(x)$ holds when x is chosen with probability $1/|X|$. Namely,

$$\Pr_{x \leftarrow X}[B(x)] = \sum_x \frac{1}{|X|} \cdot \chi(B(x))$$

where χ is an indicator function so that $\chi(B) = 1$ if event B holds, and equals zero otherwise. We denote by U_n the uniform distribution over the set $\{0,1\}^n$.

2.4 Computational indistinguishability and statistical closeness

The following definition of (computational) indistinguishability originates in the seminal paper of Goldwasser and Micali [26].

Let X be a countable set of strings. A probability ensemble indexed by X is a sequence of random variables indexed by X. Namely, any $A = \{A_x\}_{x \in X}$ is a random variable indexed by X.

Definition 2.2 ((Computational) Indistinguishability) *Let X and Y be countable sets. Two ensembles $\{A_{x,y}\}_{x \in X, y \in Y}$ and $\{B_{x,y}\}_{x \in X, y \in Y}$ are said to be* computationally indistinguishable over X*, if for every probabilistic "distinguishing" machine D whose running time is polynomial in its first input, there exists a negligible function $\nu(\cdot)$ so that for every $x \in X, y \in Y$:*

$$|\Pr[D(x, y, A_{x,y}) = 1] - \Pr[D(x, y, B_{x,y}) = 1]| < \nu(|x|)$$

$\{A_{x,y}\}_{x \in X, y \in Y}$ and $\{B_{x,y}\}_{x \in X, y \in Y}$ are said to be statistically close *over X if the above condition holds for all (possibly unbounded) machines D.*

2.5 Interactive Proofs, Zero-Knowledge and Witness-Indistinguishability

We use the standard definitions of interactive proofs (and interactive Turing machines) [27, 20] and arguments [8]. Given a pair of interactive Turing machines, P and V, we denote by $\langle P, V \rangle(x)$ the random variable representing the (local) output of V when interacting with machine P on common input x, when the random input to each machine is uniformly and independently chosen.

Definition 2.3 (Interactive Proof System) *A pair of interactive machines $\langle P, V \rangle$ is called an* interactive proof system *for a language L if machine V is polynomial-time and the following two conditions hold with respect to some negligible function $\nu(\cdot)$:*

- Completeness: *For every $x \in L$,*

$$\Pr[\langle P, V \rangle(x) = 1] \geq 1 - \nu(|x|)$$

- Soundness: *For every $x \notin L$, and every interactive machine B,*

$$\Pr[\langle B, V \rangle(x) = 1] \leq \nu(|x|)$$

In case that the soundness condition is required to hold only with respect to a computationally bounded prover, the pair $\langle P, V \rangle$ is called an interactive argument *system.*

Zero-knowledge. An interactive proof is said to be *zero-knowledge* ($\mathcal{ZK}$) if it yields nothing beyond the validity of the assertion being proved. This is formalized by requiring that the view of every probabilistic polynomial-time adversary V^* interacting with the honest prover P can be simulated by a probabilistic polynomial-time machine S (a.k.a. the *simulator*). The idea behind this definition is that whatever V^* might have learned from interacting with P, he could have actually learned by himself (by running the simulator S).

The notion of $\mathcal{ZK}$ was introduced by Goldwasser, Micali and Rackoff [27]. To make $\mathcal{ZK}$ robust in the context of protocol composition, Goldreich and Oren [25] suggested to augment the definition so that the above requirement holds also with respect to all $z \in \{0,1\}^*$, where both V^* and S are allowed to obtain z as auxiliary input. The verifier's view of an interaction consists of the common input x, followed by its random tape and the sequence of prover messages the verifier receives during the interaction. We denote by $\text{view}_{V^*}^{P}(x, z)$ a random variable describing $V^*(z)$'s view of the interaction with P on common input x.

Definition 2.4 (Zero-knowledge) *Let $\langle P, V \rangle$ be an interactive proof system. We say that $\langle P, V \rangle$ is* zero-knowledge*, if for every probabilistic polynomial-time interactive machine V^* there exists a probabilistic polynomial-time algorithm S such that the ensembles $\{\text{view}_{V^*}^{P}(x, z)\}_{x \in L, z \in \{0,1\}^*}$ and $\{S(x, z)\}_{x \in L, z \in \{0,1\}^*}$ are computationally indistinguishable over L.*

A stronger variant of zero-knowledge is one in which the output of the simulator is statistically close to the verifier's view of real interactions. We focus on *argument* systems, in which the soundness property is only guaranteed to hold with respect to polynomial time provers.

Definition 2.5 (Statistical zero-knowledge) *Let $\langle P, V\rangle$ be an interactive argument system. We say that $\langle P, V\rangle$ is* statistical zero-knowledge*, if for every probabilistic polynomial-time V^* there exists a probabilistic polynomial-time S such that the ensembles $\{\text{view}_{V^*}^P(x,z)\}_{x\in L, z\in\{0,1\}^*}$ and $\{S(x,z)\}_{x\in L, z\in\{0,1\}^*}$ are statistically close over L.*

In case that the ensembles $\{\text{view}_{V^*}^P(x,z)\}_{x\in L, z\in\{0,1\}^*}$ and $\{S(x,z)\}_{x\in L, z\in\{0,1\}^*}$ are identically distributed, the protocol $\langle P, V\rangle$ is said to be *perfect* zero-knowledge.

Witness Indistinguishability. An interactive proof is said to be *witness indistinguishable* ($\mathcal{WI}$) if the verifier's view is "computationally independent" (resp. "statistically independent") of the witness used by the prover for proving the statement (the notion of statistical $\mathcal{WI}$ will be used in Section 4.2). In this context, we focus our attention to languages $L \in \mathcal{NP}$ with a corresponding witness relation R_L. Namely, we consider interactions in which on common input x the prover is given a witness in $R_L(x)$. By saying that the view is computationally (resp. statistically) independent of the witness, we mean that for any two possible $\mathcal{NP}$-witnesses that could be used by the prover to prove the statement $x \in L$, the corresponding views are computationally (resp. statistically) indistinguishable.

Let V^* be a probabilistic polynomial time adversary interacting with the prover, and let $\text{view}_{V^*}^P(x,w,z)$ denote V^*'s view of an interaction in which the witness used by the prover is w (where the common input is x and V^*'s auxiliary input is z).

Definition 2.6 (Witness-indistinguishability) *Let $\langle P, V\rangle$ be an interactive proof system for a language $L \in \mathcal{NP}$. We say that $\langle P, V\rangle$ is* witness-indistinguishable *for R_L, if for every probabilistic polynomial-time interactive machine V^* and for every two sequences $\{w_x^1\}_{x\in L}$ and $\{w_x^2\}_{x\in L}$, such that $w_x^1, w_x^2 \in R_L(x)$ for every $x \in L$, the probability ensembles $\{\text{view}_{V^*}^P(x,w_x^1)\}_{x\in L, z\in\{0,1\}^*}$ and $\{\text{view}_{V^*}^P(x,w_x^2)\}_{x\in L, z\in\{0,1\}^*}$ are computationally indistinguishable over L.*

In case that the ensembles $\{\text{view}_{V^*}^P(x,w_x^1)\}_{x\in L, z\in\{0,1\}^*}$ and $\{\text{view}_{V^*}^P(x,w_x^2)\}_{x\in L, z\in\{0,1\}^*}$ are statistically close over L, the proof system $\langle P, V\rangle$ is said to be *statistically witness indistinguishable.* If the ensembles are identically distributed, the proof system is said to be *witness independent.*

2.6 Proofs of Knowledge

Informally an interactive proof is a proof of knowledge if the prover convinces the verifier not only of the validity of a statement, but also that it possesses a witness for the statement. This notion is formalized by the introduction of an machine E, called a knowledge extractor. As the name suggests, the extractor E is supposed to extract a witness from any malicious prover P^* that succeeds in convincing an honest verifier. More formally,

Definition 2.7 *Let (P, V) be an interactive proof system for the language L with witness relation R_L. We say that (P, V) is a proof of knowledge if there exists a polynomial q and a probabilistic oracle machine E, such that for every probabilistic polynomial-time interactive machine P^*, there exists some negligible function $\mu(\cdot)$ such that for every $x \in L$ and every $y, r \in \{0,1\}^*$ such that $\Pr[\langle P_{x,y,r}^*, V(x)\rangle = 1] > 0$, where $P_{x,y,r}^*$ denotes the machine P^* with common input fixed to x, auxiliary input fixed to y and random tape fixed to r, the following holds*

1. *The expected number of steps taken by $E^{P^*_{x,y,r}}$ is bounded by*

$$\frac{q(|x|)}{\Pr[\langle P^*_{x,y,r}, V(x)\rangle = 1]}$$

where $E^{P^*_{x,y,r}}$ denotes the machine E with oracle access to $P^*_{x,y,r}$.

2. *Furthermore,*
$$\Pr[\langle P^*_{x,y,r}, V(x)\rangle = 1 \wedge E^{P^*_{x,y,r}} \notin R_L(x)] \leq \mu(|x|)$$

The machine E is called a (knowledge) extractor*.*

We remark that as our definition only considers computationally bounded provers, we only get a "computationally convincing" notion of a proof of knowledge (a.k.a *arguments of knowledge*) [8]. In addition, our definition is slightly different from the definition of [4] in that we require that the expected running-time of the extractor is always bounded by poly$(|x|)/p$, where p denotes the success probability of P^*, whereas [4] allows for some additional slackness in the running-time. On the other hand, whereas [4] requires the extractor to always output a valid witness, we instead allow the extractor to fail with some negligible probability. We will rely on the following theorem:

Theorem 2.8 ([6, 8]) *Assume the existence of claw-free permutations. Then there exists a constant-round public-coin witness independent argument of knowledge for $\mathcal{NP}$.*

Indeed, standard techniques can be used to show that the parallelized version of the protocol of [6], using perfectly-hiding commitments, is an argument of knowledge (as defined above). As usual, the knowledge extractor E proceeds by feeding new "challenges" to the prover P^* until it gets two accepting transcripts. If the two accepting challenges contain the same challenge, or if the prover manages to open up a commitment in two different ways, the extractor outputs `fail`; otherwise it can extract a witness.

2.7 Universal Arguments

Universal arguments (introduced in [3] and closely related to the notion of CS-proofs [33]) are used in order to provide "efficient" proofs to statements of the form $y = (M, x, t)$, where y is considered to be a true statement if M is a non-deterministic machine that accepts x within t steps. The corresponding language and witness relation are denoted $L_\mathcal{U}$ and $R_\mathcal{U}$ respectively, where the pair $((M, x, t), w)$ is in $R_\mathcal{U}$ if M (viewed here as a two-input deterministic machine) accepts the pair (x, w) within t steps. Notice that every language in $\mathcal{NP}$ is linear time reducible to $L_\mathcal{U}$. Thus, a proof system for $L_\mathcal{U}$ allows us to handle all $\mathcal{NP}$-statements. In fact, a proof system for $L_\mathcal{U}$ enables us to handle languages that are presumably "beyond" $\mathcal{NP}$, as the language $L_\mathcal{U}$ is $\mathcal{NE}$-complete (hence the name universal arguments).[1]

Definition 2.9 (Universal argument) *A pair of interactive Turing machines (P, V) is called a* universal argument *system if it satisfies the following properties:*

- Efficient verification: *There exists a polynomial p such that for any $y = (M, x, t)$, the total time spent by the (probabilistic) verifier strategy V, on common input y, is at most $p(|y|)$. In particular, all messages exchanged in the protocol have length smaller than $p(|y|)$.*

[1] Furthermore, every language in $\mathcal{NEXP}$ is polynomial-time (but not linear-time) reducible to $L_\mathcal{U}$

- Completeness by a relatively efficient prover: *For every* $((M,x,t),w)$ *in* $R_{\mathcal{U}}$*,*

$$\Pr[(P(w),V)(M,x,t)=1]=1$$

Furthermore, there exists a polynomial q *such that the total time spent by* $P(w)$*, on common input* (M,x,t)*, is at most* $q(T_M(x,w)) \leq q(t)$*, where* $T_M(x,w)$ *denotes the running time of* M *on input* (x,w)*.*

- Computational Soundness: *For every polynomial size circuit family* $\{P^*_n\}_{n\in N}$*, and every triplet* $(M,x,t) \in \{0,1\}^n \setminus L_{\mathcal{U}}$*,*

$$\Pr[(P^*_n,V)(M,x,t)=1] < \nu(n)$$

where $\nu(\cdot)$ *is a negligible function.*

- Weak proof of knowledge: *For every positive polynomial* p *there exists a positive polynomial* p' *and a probabilistic polynomial-time oracle machine* E *such that the following holds: for every polynomial-size circuit family* $\{P^*_n\}_{n\in N}$*, and every sufficiently long* $y=(M,x,t)\in\{0,1\}^*$ *if* $\Pr[(P^*_n,V)(y)=1] > 1/p(|y|)$ *then*

$$\Pr[\exists w = w_1, \ldots w_t \in R_{\mathcal{U}}(y) \;\; s.t. \;\; \forall i \in [t],\; E_r^{P^*_n}(y,i)=w_i] > \frac{1}{p'(|y|)}$$

where $R_{\mathcal{U}}(y) \stackrel{\text{def}}{=} \{w : (y,w) \in R_{\mathcal{U}}\}$ *and* $E_r^{P^*_n}(\cdot,\cdot)$ *denotes the function defined by fixing the random-tape of* E *to equal* r*, and providing the resulting* E_r *with oracle access to* P^*_n*.*

2.8 Commitment Schemes

Commitment schemes are used to enable a party, known as the *sender*, to commit itself to a value while keeping it secret from the *receiver* (this property is called hiding). Furthermore, the commitment is binding, and thus in a later stage when the commitment is opened, it is guaranteed that the "opening" can yield only a single value determined in the committing phase. Commitment schemes come in two different flavors, statistically-binding and statistically-hiding. We sketch the properties of each one of these flavors. Full definitions can be found in [20].

Statistically-binding: In statistically binding commitments, the binding property holds against unbounded adversaries, while the hiding property only holds against computationally bounded (non-uniform) adversaries. The statistical-binding property asserts that, with overwhelming probability over the coin-tosses of the receiver, the transcript of the interaction fully determines the value committed to by the sender. The computational-hiding property guarantees that the commitments to any two different values are computationally indistinguishable.

Statistically-hiding: In statistically-hiding commitments, the hiding property holds against unbounded adversaries, while the binding property only holds against computationally bounded (non-uniform) adversaries. Loosely speaking, the statistical-hiding property asserts that commitments to any two different values are statistically close (i.e., have negligible statistical distance). In case the statistical distance is 0, the commitments are said to be *perfectly-hiding.* The computational-binding property guarantees that no polynomial time machine is able to open a given commitment in two different ways.

Non-interactive statistically-binding commitment schemes can be constructed using any 1–1 one-way function (see Section 4.4.1 of [20]). Allowing some minimal interaction (in which the receiver first sends a single random initialization message), statistically-binding commitment schemes can be obtained from any one-way function [34, 28]. We will think of such commitments as a *family* of non-interactive commitments, where the description of members in the family will be the initialization message. Perfectly-hiding commitment schemes can be constructed from any one-way permutation [35]. However, *constant-round* schemes are only known to exist under stronger assumptions; specifically, assuming the existence of a collection of certified clawfree functions [21].

3 Non-Malleable Protocols

The notion of non-malleability was introduced by Dolev, Dwork and Naor [16]. In this paper we focus on non malleability of zero-knowledge proofs and of string commitment. The definitions are stated in terms of interactive proofs, though what we actually construct are non-malleable argument systems. The adaptation of the definitions to the case of arguments can be obtained by simply replacing the word "proof" with "argument," whenever it appears.

In principle, our definitions are compatible with the ones appearing in [16]. However, the presentation is more detailed and somewhat different (see Section 3.4 for a discussion on the differences between our definition and previous ones).

3.1 Non-Malleable Interactive Proofs

Let $\langle P, V\rangle$ be an interactive proof. Consider a scenario where a man-in-the-middle adversary A is simultaneously participating in two interactions. These interactions are called the left and the right interaction. In the left interaction the adversary A is verifying the validity of a statement x by interacting with an honest prover P. In the right interaction A proves the validity of a statement $\tilde{x}$ to the honest verifier V. The statement $\tilde{x}$ is chosen by A, possibly depending on the messages it receives in the left interaction.

Besides controlling the messages sent by the verifier in the left interaction and by the prover in the right interaction, A has control over the scheduling of the messages. In particular, it may delay the transmission of a message in one interaction until it receives a message (or even multiple messages) in the other interaction. Figure 2 describes two representative scheduling strategies.

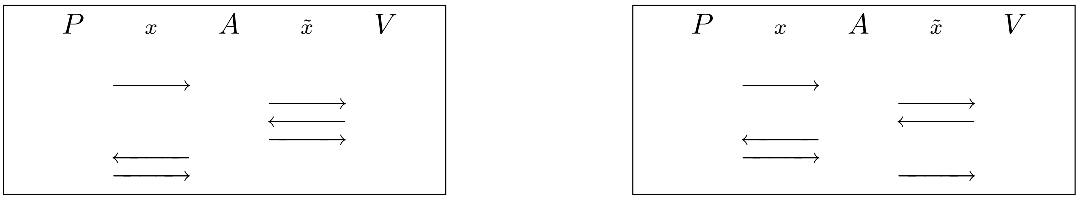

Figure 2: Two scheduling strategies.

The interactive proof $\langle P, V\rangle$ is said to be non-malleable if, whenever $x \neq \tilde{x}$, the left interaction does not "help" the adversary in convincing the verifier in the right interaction.[2] Following the simulation paradigm [26, 27, 23], this is formalized by defining appropriate "real" and "idealized"

[2] Notice that requiring that $x \neq \tilde{x}$ is necessary, since otherwise the adversary can succeed to convince the verifier in the right interaction by simply forwarding messages back and forth between the interactions.

executions. In the real execution, called the man-in-the-middle execution, the adversary participates in both the left and the right interactions with common inputs x and $\tilde{x}$ respectively. In the idealized execution, called the stand-alone execution, the adversary is playing the role of the prover in a single interaction with common input $\tilde{x}$. Security is defined by requiring that the adversary cannot succeed better in the man-in-the middle execution than he could have done in the stand-alone execution. More formally, we consider the following two executions.

Man-in-the-middle execution. The man-in-the-middle execution consists of the scenario described above. The input of P is an instance-witness pair (x, w), and the input of V is an instance $\tilde{x}$. A receives x and an auxiliary input z. Let $\mathsf{mim}_V^A(x, w, z)$ be a random variable describing the output of V in the above experiment when the random tapes of P, A and V are uniformly and independently chosen. In case that $x = \tilde{x}$, the view $\mathsf{mim}_V^A(x, w, z)$ is defined to be $\perp$.

Stand-alone execution. In the stand-alone execution only one interaction takes place. The stand-alone adversary S directly interacts with the honest verifier V. As in the man-in-the-middle execution, V receives as input an instance $\tilde{x}$. S receives instances $x, \tilde{x}$ and auxiliary input z. Let $\mathsf{sta}_V^S(x, \tilde{x}, z)$ be a random variable describing the output of V in the above experiment when the random tapes of S and V are uniformly and independently chosen.

Definition 3.1 (Non-malleable interactive proof) *An interactive proof $\langle P, V\rangle$ for a language L is said to be* non-malleable *if for every probabilistic polynomial time man-in-the-middle adversary A, there exists a probabilistic expected polynomial time stand-alone prover S and a negligible function $\nu : N \to N$, such that for every $(x, w) \in L \times R_L(x)$, every $\tilde{x} \in \{0,1\}^{|x|}$ so that $\tilde{x} \neq x$, and every $z \in \{0,1\}^*$:*

$$\Pr\Big[\mathsf{mim}_V^A(x, \tilde{x}, w, z) = 1\Big] \ < \ \Pr\Big[\mathsf{sta}_V^S(x, \tilde{x}, z) = 1\Big] \ + \ \nu(|x|)$$

Non-malleability with respect to tags. Definition 3.1 rules out the possibility that the statement proved on the right interaction is identical to the one on the left. Indeed, if the same protocol is executed on the left and on the right this kind of attack cannot be prevented, as the man-in-the-middle adversary can always copy messages between the two executions (cf., the chess-master problem [16]). Still, in many situations it might be important to be protected against an attacker that attempts to prove even the same statement. In order to deal with this problem, one could instead consider a "tag-based" variant of non-malleability (see [31] for a definition of tag-base non-malleability in the context of encryption).

We consider a family of interactive proofs, where each member of the family is labeled with a tag string $\textsc{tag} \in \{0,1\}^m$, and $t = t(n)$ is a parameter that potentially depends on the length of the common input (security parameter) $n \in N$. As before, we consider a MIM adversary A that is simultaneously participating in a left and a right interaction. In the left interaction, A is verifying the validity of a statement x by interacting with a prover $P_{\textsc{tag}}$ while using a protocol that is labeled with a string $\textsc{tag}$. In the right interaction A proves the validity of a statement $\tilde{x}$ to the honest verifier $V_{\tilde{\textsc{tag}}}$ while using a protocol that is labeled with a string $\tilde{\textsc{tag}}$. Let $\mathsf{mim}_V^A(\textsc{tag}, \tilde{\textsc{tag}}, x, \tilde{x}, w, z)$ be a random variable describing the output of V in the man-in-the-middle experiment. The stand-alone execution is defined as before with the only difference being that in addition to their original inputs, the parties also obtain the corresponding tags. Let $\mathsf{sta}_V^S(\textsc{tag}, \tilde{\textsc{tag}}, x, \tilde{x}, z)$ be a random variable describing the output of V in the stand-alone experiment.

The definition of non-malleability with respect to tags is essentially identical to Definition 3.1. The only differences in the definition is that instead of requiring non-malleability (which compares the success probability of $\mathsf{mim}_V^A(\text{TAG}, \tilde{\text{TAG}}, x, \tilde{x}, w, z)$ and $\mathsf{sta}_V^S(\text{TAG}, \tilde{\text{TAG}}, x, \tilde{x}, z)$) whenever $x \neq \tilde{x}$, we will require non-malleability whenever $\text{TAG} \neq \tilde{\text{TAG}}$. For convenience, we repeat the definition:

Definition 3.2 (Tag-based non-malleable interactive proofs) *A family of interactive proofs $\langle P_{\text{TAG}}, V_{\text{TAG}} \rangle$ for a language L is said to be* non-malleable with respect to tags of length m *if for every probabilistic polynomial time man-in-the-middle adversary A, there exists a probabilistic expected polynomial time stand-alone prover S and a negligible function $\nu : N \to N$, such that for every $(x, w) \in L \times R_L(x)$, every $\tilde{x} \in \{0,1\}^{|x|}$, every $\text{TAG}, \tilde{\text{TAG}} \in \{0,1\}^m$ so that $\text{TAG} \neq \tilde{\text{TAG}}$, and every $z \in \{0,1\}^*$:*

$$\Pr\Big[\mathsf{mim}_V^A(\text{TAG}, \tilde{\text{TAG}}, x, \tilde{x}, w, z) = 1\Big] \; < \; \Pr\Big[\mathsf{sta}_V^S(\text{TAG}, \tilde{\text{TAG}}, x, \tilde{x}, z) = 1\Big] \; + \; \nu(|x|)$$

Tags vs. statements. A non-malleable interactive proof can be turned into a tag-based one by simply concatenating the tag to the statement being proved. On the other hand, an interactive proof that is non-malleable with respect to tags of length $t(n) = n$ can be turned into a non-malleable interactive proof by using the statement $x \in \{0,1\}^n$ as tag.

The problem of constructing a tag-based non-malleable interactive proof is already non-trivial for tags of length, say $t(n) = O(\log n)$ (and even for $t(n) = O(1)$), but is still potentially easier than for tags of length n. This opens up the possibility of reducing the construction of interactive proofs that are non-malleable w.r.t. long tags into interactive proofs that are non-malleable w.r.t. shorter tags. Even though we do not know whether such a reduction is possible in general, our work follows this path and demonstrates that in specific cases such a reduction is indeed possible.

Non-malleability with respect to other protocols. Our definitions of non-malleability refer to protocols that are non-malleable *with respect to themselves*, since the definitions consider a setting where the same protocol is executed in the left and the right interaction. In principle, one could consider two different protocols that are executed on the left and on the right which are be non-malleable *with respect to each other*. Such definitions are not considered in this work.

3.2 Non-malleable Zero-Knowledge

Non-malleable $\mathcal{ZK}$ proofs are non-malleable interactive proofs that additionally satisfy the $\mathcal{ZK}$ property (as stated in Definitions 2.4)

Definition 3.3 (Non-malleable zero-knowledge) *A family $\{\langle P_{\text{TAG}}, V_{\text{TAG}} \rangle\}_{\text{TAG} \in \{0,1\}^*}$ of interactive proofs is said to be* non malleable zero-knowledge *if it is both non malleable and zero knowledge.*

3.3 Non-malleable Commitments

Informally, a commitment scheme is non-malleable if a man-in-the-middle adversary that receives a commitment to a value v will not be able to "successfully" commit to a related value $\tilde{v}$. The literature discusses two different interpretations of the term"success":

Non-malleability with respect to commitment [16]. The adversary is said to succeed if it manages to commit to a related value, even without being able to later decommit to this value. This notion makes sense only in the case of statistically-binding commitments.

Non-malleability with respect to opening [15]. The adversary is said to succeed only if it is able to both commit and decommit to a related value. This notion makes sense both in the case of statistically-binding and statistically-hiding commitments.

As in the case of non-malleable zero-knowledge, we formalize the definition by comparing a man-in-the-middle and a stand-alone execution. Let $n \in N$ be a security parameter. Let $\langle C, R \rangle$ be a commitment scheme, and let $\mathcal{R} \subseteq \{0,1\}^n \times \{0,1\}^n$ be a polynomial-time computable irreflexive relation (i.e., $\mathcal{R}(v, v) = 0$). As before, we consider man-in-the-middle adversaries that are simultaneously participating in a left and a right interaction in which a commitment scheme is taking place. The adversary is said to succeed in mauling a left commitment to a value v, if he is able to come up with a right commitment to a value $\tilde{v}$ such that $\mathcal{R}(v, \tilde{v}) = 1$. Since we cannot rule out copying, we will only be interested in relations where copying is not considered success, and we therefore require that the relation $\mathcal{R}$ is irreflexive. The man-in-the-middle and the stand-alone executions are defined as follows.

The man-in-the-middle execution. In the man-in-the-middle execution, the man-in-the-middle adversary A is simultaneously participating in a left and a right interaction. In the left interaction the man-in-the-middle adversary A interacts with C receiving a commitment to a value v. In the right interaction A interacts with R attempting to commit to a related value $\tilde{v}$. Prior to the interaction, the value v is given to C as local input. A receives an auxiliary input z, which in particular might contain a-priori information about v.[3] The success of A is defined using the following two Boolean random variables:

- $\mathsf{mim}^A_{\mathsf{com}}(\mathcal{R}, v, z) = 1$ if and only if A produces a valid commitment to $\tilde{v}$ such that $\mathcal{R}(v, \tilde{v}) = 1$.
- $\mathsf{mim}^A_{\mathsf{open}}(\mathcal{R}, v, z) = 1$ if and only if A decommits to a value $\tilde{v}$ such that $\mathcal{R}(v, \tilde{v}) = 1$.

The stand-alone execution. In the stand-alone execution only one interaction takes place. The stand-alone adversary S directly interacts with R. As in the man-in-the-middle execution, the value v is chosen prior to the interaction and S receives some a-priori information about v as part of its an auxiliary input z. S first executes the commitment phase with R. Once the commitment phase has been completed, S receives the value v and attempts to decommit to a value $\tilde{v}$. The success of S is defined using the following two Boolean random variables:

- $\mathsf{sta}^S_{\mathsf{com}}(\mathcal{R}, v, z) = 1$ if and only if S produces a valid commitment to $\tilde{v}$ such that $\mathcal{R}(v, \tilde{v}) = 1$.
- $\mathsf{sta}^S_{\mathsf{open}}(\mathcal{R}, v, z) = 1$ if and only if A decommits to a value $\tilde{v}$ such that $\mathcal{R}(v, \tilde{v}) = 1$.

Definition 3.4 (Non-malleable commitment) *A commitment scheme $\langle C, R \rangle$ is said to be* non-malleable with respect to commitment *if for every probabilistic polynomial-time man-in-the-middle adversary A, there exists a probabilistic expected polynomial time stand-alone adversary S and a negligible function $\nu : N \to N$, such that for every irreflexive polynomial-time computable relation $\mathcal{R} \subseteq \{0,1\}^n \times \{0,1\}^n$, every $v \in \{0,1\}^n$, and every $z \in \{0,1\}^*$, it holds that:*

$$\Pr\Big[\mathsf{mim}^A_{\mathsf{com}}(\mathcal{R}, v, z) = 1\Big] \;<\; \Pr\Big[\mathsf{sta}^S_{\mathsf{com}}(\mathcal{R}, v, z) = 1\Big] \;+\; \nu(n)$$

Non-malleability with respect to opening *is defined in the same way, while replacing the random variables $\mathsf{mim}^A_{\mathsf{com}}(\mathcal{R}, v, z)$ and $\mathsf{sta}^S_{\mathsf{com}}(\mathcal{R}, v, z)$ with $\mathsf{mim}^A_{\mathsf{open}}(\mathcal{R}, v, z)$ and $\mathsf{sta}^S_{\mathsf{open}}(\mathcal{R}, v, z)$.*

[3] The original definition by Dwork et al. [16] accounted for such a-priori information by providing the adversary with the value $\mathsf{hist}(v)$, where the function $\mathsf{hist}(\cdot)$ be a polynomial-time computable function.

Content-based v.s. tag-based commitments. Similar to the definition of interactive proofs non-malleable with respect to statements, the above definitions only require that the adversary should not be able to commit to a value that is related, *but different*, from the value it receives a commitment of. Technically, the above fact can be seen from the definitions by noting that the relation $\mathcal{R}$, which defines the success of the adversary, is required to be irreflexive. This means that the adversary is said to fail if it only is able to produce a commitment to the same value.[4] Indeed, if the same protocol is executed in the left and the right interaction, the adversary can always copy messages and succeed in committing to the same value on the right as it receives a commitment of, on the left. To cope with this problem, the definition can be extended to incorporate tags, in analogy with the definition of interactive proofs non-malleable with respect to tags. The extension is straightforward and therefore omitted.

We note that any commitment scheme that satisfies Definition 3.4 can easily be transformed into a scheme which is tag-based non-malleable, by prepending the tag to the value before committing. Conversely, in analogy with non-malleable interactive proof, commitment schemes that are non-malleable with respect to tags of length $t(n) = \text{poly}(n)$ can be transformed into commitment schemes non-malleable with respect to content in a standard way (see e.g., [16, 31]).

3.4 Comparison with Previous Definitions

Our definitions of non-malleability essentially follow the original definitions by Dwork et al.[16]. However, whereas the definitions by Dwork et al. quantifies the experiments over all distributions D of inputs for the left and the right interaction (or just left interaction in the case of commitments), we instead quantify over all possible input values $x, \tilde{x}$ (or, in the case of commitments over all possible input values v for the left interaction). Our definitions can thus be seen as non-uniform versions of the definitions of [16].

Our definition of non-malleability with respect to opening is, however, different from the definition of [15] in the following ways: (1) The definition of [15] does not take into account possible a-priori information that the adversary might have about the commitment, while ours (following [16]) does. (2) In our definition of the stand-alone execution the stand-alone adversary receives the value v after having completed the commitment phase and is thereafter supposed to decommit to a value related to v. The definition of [15] does not provide the simulator with this information.

In our view, the "a-priori information" requirement is essential in many situations and we therefore present a definition that satisfies it. (Consider, for example, a setting where the value v committed to is determined by a different protocol, which "leaks" some information about v.) In order to be able to satisfy this stronger requirement we relax the definition of [15] by allowing the stand-alone adversary to receive the value v before de-committing.

3.5 Simulation-Extractability

A central tool in our constructions of non-malleable interactive-proofs and commitments is the notion of *simulation-extractability.* Loosely speaking, an interactive protocol is said to be simulation extractable if for any man-in-the-middle adversary A, there exists a probabilistic polynomial time machine (called the simulator-extractor) that can simulate both the left and the right interaction for A, while outputting a witness for the statement proved by the adversary in the right interaction.

[4] Potentially, one could consider a slightly stronger definition, which also rules out the case when the adversary is able to construct a *different* commitment to the *same* value. Nevertheless, we here adhere to the standard definition of non-malleable commitments which allows the adversary to produce a different commitment to the same value.

Simulation-extractability can be thought of a technical (and stronger) variant of non-malleability. The main reason for introducing this notion is that it enables a more modular analysis (and in particular is easier to work with). At the end of this section, we argue that any protocol that is simulation-extractable is also a non-malleable zero-knowledge proof of knowledge. In Section 6 we show how to use simulation-extractable protocols in order to obtain non-malleable commitments.

Let A be a man-in-the middle adversary that is simultaneously participating in a left interaction of $\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle$ while acting as verifier, and a right interaction of $\langle P_{\tilde{\text{TAG}}}, V_{\tilde{\text{TAG}}}\rangle$ while acting as prover.

Let $\mathsf{view}_A(x, z, \text{TAG})$ denote the *joint* view of $A(x, z)$ and the honest verifier $V_{\tilde{\text{TAG}}}$ when A is verifying a left-proof of the statement x, using identity TAG, and proving on the right a statement $\tilde{x}$ using identity $\tilde{\text{TAG}}$. (The view consists of the messages sent and received by A in both left and right interactions, and the random coins of A, and $V_{\tilde{\text{TAG}}}$).[5] Both $\tilde{x}$ and $\tilde{\text{TAG}}$ are chosen by A. Given a function $t = t(n)$ we use the notation $\{\cdot\}_{z,x,\text{TAG}}$ as shorthand for $\{\cdot\}_{z\in\{0,1\}^*,x\in L,\text{TAG}\in\{0,1\}^{t(|x|)}}$.

Definition 3.5 (Simulation-extractable protocol) *A family $\{\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle\}_{\text{TAG}\in\{0,1\}^*}$ of interactive proofs is said to be* simulation extractable *with tags of length $t = t(n)$ if for any man-in-the-middle adversary A, there exists a probabilistic expected poly-time machine $\mathcal{S}$ such that:*

1. *The probability ensembles $\{\mathcal{S}_1(x, z, \text{TAG})\}_{x,z,\text{TAG}}$ and $\{\mathsf{view}_A(x, z, \text{TAG})\}_{x,z,\text{TAG}}$ are statistically close over L, where $\mathcal{S}_1(x, z, \text{TAG})$ denotes the first output of $\mathcal{S}(x, z, \text{TAG})$.*
2. *Let $x \in L, z \in \{0,1\}^*, \text{TAG} \in \{0,1\}^{t(|x|)}$ and let (view, w) denote the output of $\mathcal{S}(x, z, \text{TAG})$ (on input some random tape). Let $\tilde{x}$ be the right-execution statement appearing in* view *and let $\tilde{\text{TAG}}$ denote the right-execution tag. Then, if the right-execution in* view *is accepting AND $\text{TAG} \neq \tilde{\text{TAG}}$, then $R_L(\tilde{x}, w) = 1$.*

We note that the above definition refers to protocols that are simulation extractable *with respect to themselves.* A stronger variant (which is not considered in the current work) would have required simulation extractability even in the presence of protocols that do not belong to the family.

We next argue that in order to construct non-malleable zero-knowledge protocols, it will be sufficient to come up with a protocol that is simulation-extractable. To do so, we prove that any protocol that is simulation-extractable (and has an efficient prover strategy) is also non-malleable zero-knowledge (i.e., it satisfies Definitions 3.1 and 3.3).

Proposition 3.6 *Let $\{\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle\}_{\text{TAG}\in\{0,1\}^*}$ be a family of simulation-extractable protocols with tags of length $t = t(n)$ (with respect to the language L and the witness relation R_L) with an efficient prover strategy. Then, $\{\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle\}_{\text{TAG}\in\{0,1\}^*}$ is also a non-malleable zero-knowledge (with tags of length t).*

Proof: Let $\{\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle\}_{\text{TAG}\in\{0,1\}^*}$ be a family of simulation-extractable protocols with tags of length t, with respect to the language L and the witness relation R_L. We argue that $\{\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle\}_{\text{TAG}\in\{0,1\}^*}$ is both a non-malleable interactive proof and stand alone zero-knowledge.

Non-malleability. Assume for contradiction that there exist a prob. poly–time man-in-the-middle adversary A, and a polynomial $p(n)$ such for infinitely many n, there exists $x, \tilde{x} \in \{0,1\}^n$, $w, z \in \{0,1\}^*$, and $\text{TAG}, \tilde{\text{TAG}} \in \{0,1\}^{t(n)}$ such that $(x, w) \in L \times R_L(x)$, $\text{TAG} \neq \tilde{\text{TAG}}$ and

$$\Pr\Big[\mathsf{mim}_V^A(\text{TAG}, \tilde{\text{TAG}}, x, \tilde{x}, w, z) = 1\Big] \;\geq\; \Pr\Big[\mathsf{sta}_V^S(\text{TAG}, \tilde{\text{TAG}}, x, \tilde{x}, z) = 1\Big] \;+\; \frac{1}{p(n)} \qquad (1)$$

[5] Since the messages sent by A are fully determined given the code of A and the messages it receives, including them as part of the view is somewhat redundant. The reason we have chosen to do so is for convenience of presentation.

By Definition 3.5, there exist a probabilistic polynomial time machine $\mathcal{S}$ for A that satisfies the definition's conditions. We show how to use $\mathcal{S}$ in order to construct a stand alone prover S for $\langle P_{\text{TAG}}, V_{\text{TAG}} \rangle$. On input $\text{TAG}, \tilde{\text{TAG}}, x, \tilde{x}, z$, the machine S runs the simulator extractor $\mathcal{S}$ on input x, z, TAG and obtains the view *view* and witness $\tilde{w}$. In the event that the *view* contains an accepting right-execution of the statement $\tilde{x}$ using tag TAG, S executes the honest prover strategy P_{TAG} on input x and the witness w.

It follows directly from the simulation property of $\mathcal{S}$ that the probability that *view* contains an accepting right-execution proof of $\tilde{x}$ using tag $\tilde{\text{TAG}}$ is negligibly close to

$$p_A = \Pr\left[\mathsf{mim}_V^A(\text{TAG}, \tilde{\text{TAG}}, x, \tilde{x}, w, z) = 1\right]$$

Since $\mathcal{S}$ always outputs a witness when the right-execution is accepting and the tag of the right-execution is different from the tag of the left execution, we conclude that success probability of S also is negligbly close to p_A (since $\text{TAG} \neq \tilde{\text{TAG}}$). This contradicts equation 1.

Zero Knowledge. Consider any probabilistic poly-time verifier V^*. Construct the man-in-the-middle adversary A that internally incorporates V and relays its left execution unmodified to V^*. In the right execution, A simply outputs $\perp$. By the simulation-extractability property of $\langle P_{\text{TAG}}, V_{\text{TAG}} \rangle$, there exists a simulator-extractor $\mathcal{S}$ for A. We describe a simulator S for V^*.

On input x, z, TAG, S runs $\mathcal{S}$ on input x, z, TAG to obtain (view, w). Given the view *view*, S outputs the view of V^* in *view* (which is a subset of *view*). It follows directly from the simulation property of $\mathcal{S}$, and from the fact that S outputs an (efficiently computable) subset of *view* that the output of S is indistinguishable from the view of V^* in an honest interaction with a prover. ■

4 A Simulation-Extractable Protocol

We now turn to describe our construction of simulation extractable protocols. At a high level, the construction proceeds in two steps:

1. For any $n \in N$, construct a family $\{\langle P_{\mathsf{tag}}, V_{\mathsf{tag}} \rangle\}_{\mathsf{tag} \in [2n]}$ of simulation-extractable arguments with tags of length $t(n) = \log n + 1$.

2. For any $n \in N$, use the family $\{\langle P_{\mathsf{tag}}, V_{\mathsf{tag}} \rangle\}_{\mathsf{tag} \in [2n]}$ to construct a family $\{\langle P_{\text{TAG}}, V_{\text{TAG}} \rangle\}_{\text{TAG} \in \{0,1\}^n}$ of simulation extractable arguments with tags of length $t(n) = 2^n$.

The construction of the family $\{\langle P_{\mathsf{tag}}, V_{\mathsf{tag}} \rangle\}_{\mathsf{tag} \in [2n]}$ relies on Barak's non black-box techniques for obtaining constant-round public-coin $\mathcal{ZK}$ for $\mathcal{NP}$ [1], and are very similar in structure to the $\mathcal{ZK}$ protocols used by Pass in [38]. We start by reviewing the ideas underlying Barak's protocol. We then proceed to present our protocols.

4.1 Barak's non-black-box protocol

Barak's protocol is designed to allow the simulator access to "trapdoor" information that is not available to the prover in actual interactions. Given this "trapdoor" information, the simulator will be able to produce convincing interactions even without possessing a witness for the statement being proved. The high-level idea is to enable the usage of the verifier's code as a "fake" witness in the proof. In the case of the honest verifier V (which merely sends over random bits), the code consists of the verifier's random tape. In the case of a malicious verifier V^*, the code may also consist of a program that generates the verifier's messages (based on previously received messages).

Since the actual prover does not have a-priori access to V's code in real interactions, this will not harm the soundness of the protocol. The simulator, on the other hand, will be always able to generate transcripts in which the verifier accepts since, by definition, it obtains V^*'s code as input.

Let $n \in N$, and let $T : N \rightarrow N$ be a "nice" function that satisfies $T(n) = n^{\omega(1)}$. To make the above ideas work, Barak's protocol relies on a "special" $\mathbf{NTIME}(T(n))$ relation. It also makes use of a witness-indistinguishable universal argument ($WIUARG$) [18, 17, 29, 33, 3]. We start by describing a variant of Barak's relation, which we denote by R_{sim}. Usage of this variant will facilitate the presentation of our ideas in later stages.

Let $\{\mathcal{H}_n\}_n$ be a family of hash functions where a function $h \in \mathcal{H}_n$ maps $\{0,1\}^*$ to $\{0,1\}^n$ (cf. [32, 10]), and let Com be a statistically binding commitment scheme for strings of length n, where for any $\alpha \in \{0,1\}^n$, the length of $\mathrm{Com}(\alpha)$ is upper bounded by $2n$. The relation R_{sim} is described in Figure 3.

Instance: A triplet $\langle h, c, r\rangle \in \mathcal{H}_n \times \{0,1\}^n \times \{0,1\}^{\mathrm{poly}(n)}$.

Witness: A program $\Pi \in \{0,1\}^*$, a string $y \in \{0,1\}^*$ and a string $s \in \{0,1\}^{\mathrm{poly}(n)}$.

Relation: $\mathbf{R}_{\mathsf{sim}}(\langle h, c, r\rangle, \langle \Pi, y, s\rangle) = 1$ if and only if:

1. $|y| \leq |r| - n$.
2. $c = \mathrm{Com}(h(\Pi); s)$.
3. $\Pi(y) = r$ within $T(n)$ steps.

Figure 3: R_{sim} - A variant of Barak's relation.

Remark 4.1 (Simplifying assumptions) *The relation presented in Figure 3 is slightly oversimplified and will make Barak's protocol work only when $\{\mathcal{H}_n\}_n$ is collision resistant against "slightly" super-polynomial sized circuits* [1]. *To make it work assuming collision resistance against polynomial sized circuits, one should use a "good" error-correcting code* $\mathtt{ECC}$ (*i.e., with constant distance and with polynomial-time encoding and decoding*), *and replace the condition $c = \mathrm{Com}(h(\Pi); s)$ with $c = \mathrm{Com}(h(\mathtt{ECC}(\Pi)); s)$* [3]. *We also assume that* Com *is a one-message commitment scheme. Such schemes can be constructed based on any 1-1 one-way function. At the cost of a small complication, the one-message scheme could have been replaced by the 2-message commitment scheme of* [34]*, which can be based on "ordinary" one-way functions* [28].

Let L be any language in $\mathcal{NP}$, let $n \in N$, and let $x \in \{0,1\}^n$ be the common input for the protocol. The idea is to have the prover claim (in a witness indistinguishable fashion) that either $x \in L$, or that $\langle h, c, r\rangle$ belongs to the language L_{sim} that corresponds to R_{sim}, where $\langle h, c, r\rangle$ is a triplet that is jointly generated by the prover and the verifier. As will turn out from the analysis, no polynomial-time prover will be able to make $\langle h, c, r\rangle$ belong to L_{sim}. The simulator, on the other hand, will use the verifier's program in order to make sure that $\langle h, c, r\rangle$ is indeed in L_{sim} (while also possessing a witness for this fact).

A subtle point to be taken into consideration is that the verifier's running-time (program size) is not a-priori bounded by any specific polynomial (this is because the adversary verifier might run in arbitrary polynomial time). This imposes a choice of $T(n) = n^{\omega(1)}$ in R_{sim}, and implies that the corresponding language does not lie in $\mathcal{NP}$ (but rather in $\mathbf{NTIME}(n^{\omega(1)})$). Such languages are beyond the scope of the "traditional" witness indistinguishable proof systems (which were originally designed to handle "only" $\mathcal{NP}$-languages), and will thus require the usage of a Witness Indistinguishable Universal Argument. Barak's protocol is described in Figure 4.

Common Input: An instance $x \in \{0,1\}^n$

Security parameter: 1^n.

Stage 1:

$V \to P$: Send $h \overset{\text{R}}{\leftarrow} \mathcal{H}_n$.

$P \to V$: Send $c = \text{Com}(0^n)$.

$V \to P$: Send $r \overset{\text{R}}{\leftarrow} \{0,1\}^{3n}$.

Stage 2 (Body of the proof):

$P \Leftrightarrow V$: A *WI UARG* $\langle P_{\mathsf{UA}}, V_{\mathsf{UA}} \rangle$ proving the OR of the following statements:

1. $\exists\ w \in \{0,1\}^{\text{poly}(|x|)}$ s.t. $R_L(x,w) = 1$.
2. $\exists\ \langle \Pi, y, s \rangle$ s.t. $R_{\mathsf{sim}}(\langle h, c, r\rangle, \langle \Pi, y, s\rangle) = 1$.

Figure 4: Barak's $\mathcal{ZK}$ argument for $\mathcal{NP}$ - $\langle P_B, V_B \rangle$.

Soundness. The idea behind the soundness of $\langle P_B, V_B \rangle$ is that any program Π (be it efficient or not) has only one output for any given input. This means that Π, when fed with an input y, has probability 2^{-3n} to "hit" a string $r \overset{\text{R}}{\leftarrow} \{0,1\}^{3n}$. Since the prover sends c before actually receiving r, and since R_{sim} imposes $|y| \le |r| - n = n$, then it is not able to "arrange" that both $c = \text{Com}(h(\Pi))$ and $\Pi(y) = r$ with probability significantly greater than $2^{2n} \cdot 2^{-3n} = 2^{-n}$ (as $|c| \le 2n$). The only way for a prover to make the honest verifier accept in the $WIUARG$ is thus to use a witness w for R_L. This guarantees that whenever the verifier is convinced, it is indeed the case that $x \in L$.

Zero-knowledge. Let V^* be the program of a potentially malicious verifier. The $\mathcal{ZK}$ property of $\langle P_B, V_B \rangle$ follows by letting the simulator set $\Pi = V^*$ and $y = c$. Since $|c| \le 2n \le |r| - n$ and since, by definition $V^*(c)$ always equals r, the simulator can set $c = \text{Com}(h(V^*); s)$ in Stage 1, and use the triplet $\langle V^*, c, s \rangle$ as a witness for R_{sim} in the $WIUARG$. This enables the simulator to produce convincing interactions, even without knowing a valid witness for $x \in L$. The $\mathcal{ZK}$ property then follows (with some work) from the hiding property of Com and the $\mathcal{WI}$ property of $\langle P_{\mathsf{UA}}, V_{\mathsf{UA}} \rangle$.

4.2 A "Special-Purpose" Universal Argument

Before we proceed with the construction of our new protocol, we will need to present a universal argument that is specially tailored for our purposes. The main distinguishing features of this universal argument, which we call the *special purpose* argument, are: (1) it is *statistically* witness indistinguishable; and (2) it will enable us to prove that our protocols satisfy the proof of knowledge property of Definition 2.7.[6]

[6] The "weak" proof of knowledge property of a universal argument (as defined in [3]) is not sufficient for our purposes. Specifically, while in a weak proof of knowledge it is required that the extractor succeeds with probability

Let **Com** be a statistically-hiding commitment scheme for strings of length n. Let $\mathbf{R}_{\mathsf{sim}}$ be a variant of the relation R_{sim} (from Figure 3) in which the statistically-binding commitment Com is replaced with the commitment **Com**, let $\langle P_{\mathsf{sWI}}, V_{\mathsf{sWI}} \rangle$ be a statistical witness indistinguishable argument of knowledge, and let $\langle P_{\mathsf{UA}}, V_{\mathsf{UA}} \rangle$ be a 4-message, public-coin universal argument where the length of the messages is upper bounded by n.[7] The special purpose $UARG$, which we denote by $\langle P_{\mathsf{sUA}}, V_{\mathsf{sUA}} \rangle$, handles statements of the form $(x, \langle h, c_1, c_2, r_1, r_2 \rangle)$, where the triplets $\langle h, c_1, r_1 \rangle$ and $\langle h, c_2, r_2 \rangle$ correspond to instances for $\mathbf{R}_{\mathsf{sim}}$. The protocol $\langle P_{\mathsf{sUA}}, V_{\mathsf{sUA}} \rangle$ is described in Figure 5.

Parameters: Security parameter 1^n.

Common Input: $x \in \{0,1\}^n$, $\langle h, c_1, c_2, r_1, r_2 \rangle$ where for $i \in \{1,2\}$, $\langle h, c_i, r_i \rangle$ is an instance for $\mathbf{R}_{\mathsf{sim}}$.

Stage 1 (Encrypted UARG):

$V \to P$: Send $\alpha \overset{\text{R}}{\leftarrow} \{0,1\}^n$.

$P \to V$: Send $\widehat{\beta} = \mathbf{Com}(0^n)$.

$V \to P$: Send $\gamma \overset{\text{R}}{\leftarrow} \{0,1\}^n$.

$P \to V$: Send $\widehat{\delta} = \mathbf{Com}(0^n)$.

Stage 2 (Body of the proof):

$P \leftrightarrow V$: A statistical $WIAOK$ $\langle P_{\mathsf{sWI}}, V_{\mathsf{sWI}} \rangle$ proving the OR of the following statements:

1. $\exists\, w \in \{0,1\}^{\mathrm{poly}(|x|)}$ so that $R_L(x, w) = 1$.
2. $\exists\, \langle \beta, \delta, s_1, s_2 \rangle$ so that:
   - $\widehat{\beta} = \mathbf{Com}(\beta; s_1)$.
   - $\widehat{\delta} = \mathbf{Com}(\delta; s_2)$.
   - $(\alpha, \beta, \gamma, \delta)$ is an accepting transcript for $\langle P_{\mathsf{UA}}, V_{\mathsf{UA}} \rangle$ proving the statement:
     - $\exists\, \langle i, \Pi, y, s \rangle$ so that $\mathbf{R}_{\mathsf{sim}}(\langle h, c_i, r_i \rangle, \langle \Pi, y, s \rangle) = 1$

Figure 5: A special-purpose universal argument $\langle P_{\mathsf{sUA}}, V_{\mathsf{sUA}} \rangle$.

4.3 A family of $2n$ protocols

We next present a family of protocols $\{\langle P_{\mathsf{tag}}, V_{\mathsf{tag}} \rangle\}_{\mathsf{tag} \in [2n]}$ (with tags of length $t(n) = \log n + 1$).[8] The protocols are based on $\langle P_B, V_B \rangle$, and are a variant of the $\mathcal{ZK}$ protocols introduced by Pass [38]. There are two key differences between $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}} \rangle$ and $\langle P_B, V_B \rangle$: in the $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}} \rangle$ protocol (1) the prover (simulator) is given two opportunities to guess the verifier's "next message", and (2) the *lengths* of the verifier's "next messages" depend on the tag of the protocol.We mention that the idea of using a multiple slot version of $\langle P_B, V_B \rangle$ already appeared in [39, 38], and the message-length technique appeared in [38]. However, our protocols $\{\langle P_{\mathsf{tag}}, V_{\mathsf{tag}} \rangle\}_{\mathsf{tag} \in [2n]}$ differ from the protocol of [38] in two aspects: our protocols are required to satisfy (1) a *statistical* secrecy property, and (2) a *proof of knowledge* property. Towards this end, we replace the statistically-binding commitments, Com, used in the presentation of $\langle P_B, V_B \rangle$ with statistically-hiding commitments, and replace the

that is polynomially related to the success probability of the prover, in our proof of security we will make use of an extractor that succeeds with probability negligibly close to the success probability of the prover.

[7] Both statistical witness indistinguishable arguments of knowledge, and 4-message, public-coin, universal arguments can be constructed assuming a family $\mathcal{H}_n$ of standard collision resistant hash functions (cf. [20] and [29, 33, 3]).

[8] A closer look at the construction will reveal that it will in fact work for any $t(n) = O(\log n)$. The choice of $t(n) = \log n + 1$ is simply made for the sake of concreteness (as in our constructions it is the case that $\mathsf{tag} \in [2n]$).

use of a $WIUARG$ with the use of a "special-purpose" $UARG$. Let $\mathbf{Com}$ be a statistically-hiding commitment scheme for strings of length n, where for any $\alpha \in \{0,1\}^n$, the length of $\mathbf{Com}(\alpha)$ is upper bounded by $2n$. Let $\mathbf{R}_{\mathsf{sim}}$ be the statistical variant of the relation R_{sim}, and let $\langle P_{\mathsf{sUA}}, V_{\mathsf{sUA}}\rangle$ be the special purpose universal argument (both $\mathbf{R}_{\mathsf{sim}}$ and $\langle P_{\mathsf{sUA}}, V_{\mathsf{sUA}}\rangle$ are described in Section 4.2). Protocol $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$ is described in Figure 6.

Common Input: An instance $x \in \{0,1\}^n$

Parameters: Security parameter 1^n, length parameter $\ell(n)$.

Tag String: $\mathsf{tag} \in [2n]$.

Stage 0 (Set-up):

$V \to P$: Send $h \overset{\text{R}}{\leftarrow} \mathcal{H}_n$.

Stage 1 (Slot 1):

$P \to V$: Send $c_1 = \mathbf{Com}(0^n)$.

$V \to P$: Send $r_1 \overset{\text{R}}{\leftarrow} \{0,1\}^{\mathsf{tag}\cdot\ell(n)}$.

Stage 1 (Slot 2):

$P \to V$: Send $c_2 = \mathbf{Com}(0^n)$.

$V \to P$: Send $r_2 \overset{\text{R}}{\leftarrow} \{0,1\}^{(2n+1-\mathsf{tag})\cdot\ell(n)}$.

Stage 2 (Body of the proof):

$P \Leftrightarrow V$: A special-purpose $UARG$ $\langle P_{\mathsf{sUA}}, V_{\mathsf{sUA}}\rangle$ proving the OR of the following statements:

1. $\exists\ w \in \{0,1\}^{\text{poly}(|x|)}$ s.t. $R_L(x,w) = 1$.
2. $\exists\ \langle \Pi, y, s\rangle$ s.t. $\mathbf{R}_{\mathsf{Sim}}(\langle h, c_1, r_1\rangle, \langle \Pi, y, s\rangle) = 1$.
3. $\exists\ \langle \Pi, y, s\rangle$ s.t. $\mathbf{R}_{\mathsf{Sim}}(\langle h, c_2, r_2\rangle, \langle \Pi, y, s\rangle) = 1$.

Figure 6: Protocol $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$.

Note that the only difference between two protocols $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$ and $\langle P_{\tilde{\mathsf{tag}}}, V_{\tilde{\mathsf{tag}}}\rangle$ is the length of the verifier's "next messages": in fact, the length of those messages in $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$ is a parameter that depends on tag (as well as on the length parameter $\ell(n)$). This property will be crucial for the analysis of these protocols in the man in the middle setting.

Using similar arguments to the ones used for $\langle P_B, V_B\rangle$, it can be shown that $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$ is computationally sound. The main difference to be taken into consideration is the existence of multiple slots in Stage 1 (see Lemma A.1 for a proof of an even stronger statement).

The $\mathcal{ZK}$ property of $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$ is proved exactly as in the case of $\langle P_B, V_B\rangle$, by letting the simulator pick either $i = 1$ or $i = 2$, and use $\langle V^*, c_i, s_i\rangle$ as the witness for $\langle h, c_i, r_i\rangle \in L_{\mathsf{sim}}$ (where L_{sim} is the language that corresponds to $\mathbf{R}_{\mathsf{sim}}$). Since for every $\mathsf{tag} \in [m]$, $|r_i| - |c_i| \geq \ell(n) - 2n$, we have that as long as $\ell(n) \geq 3n$, the protocol $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$ is indeed $\mathcal{ZK}$.

We wish to highlight some useful properties of $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$. These properties will turn out to be relevant when dealing with a man in the middle.

Freedom in the choice of the slot: The simulator described above has the freedom to choose which $i \in \{1,2\}$ it will use in order to satisfy the relation $\mathbf{R}_{\mathsf{sim}}$. In particular, for the simulation to succeed, it is sufficient that $\langle h, c_i, r_i\rangle \in L_{\mathsf{sim}}$ for *some* $i \in \{1,2\}$.

Using a longer y in the simulation: The stand-alone analysis of $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$ only requires

$\ell(n) \geq 3n$. Allowing larger values of $\ell(n)$ opens the possibility of using a longer y in the simulation. This will turn out to be useful if the verifier is allowed to receive "outside" messages that do not belong to the protocol (as occurs in the man-in-the-middle setting).

Statistical secrecy: The output of the simulator described above is *statistically* close to real interactions (whereas the security guaranteed in $\langle P_B, P_B \rangle$ is only computational). A related property will turn out to be crucial for the use of $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}} \rangle$ as a subroutine in higher level applications (such as non-malleable commitments).

Proof of knowledge: $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}} \rangle$ is a proof of knowledge. That is, for any prover P^* and for any $x \in \{0,1\}^n$, if P^* convinces the honest verifier V that $x \in L$ with non-negligible probability then one can extract a witness w that satisfies $R_L(x, w) = 1$ in (expected) polynomial time.

4.4 A family of 2^n protocols

Relying on the protocol family $\{\langle P_{\mathsf{tag}}, V_{\mathsf{tag}} \rangle\}_{\mathsf{tag} \in [2n]}$, we now show how to construct a family $\{\langle P_{\textsc{tag}}, V_{\textsc{tag}} \rangle\}_{\textsc{tag} \in \{0,1\}^n}$ with tags of length $t(n) = n$. The protocols are *constant-round* and involve n parallel executions of $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}} \rangle$, with appropriately chosen tags. This new family of protocols is denoted $\{\langle P_{\textsc{tag}}, V_{\textsc{tag}} \rangle\}_{\textsc{tag} \in \{0,1\}^n}$ and is described in Figure 7.

Common Input: An instance $x \in \{0,1\}^n$

Parameters: Security parameter 1^n, length parameter $\ell(n)$

Tag String: $\textsc{tag} \in \{0,1\}^n$. Let $\textsc{tag} = \textsc{tag}_1, \ldots, \textsc{tag}_n$.

The protocol:

$P \leftrightarrow V$: For all $i \in \{1, \ldots, n\}$ (in parallel):

1. Set $\mathsf{tag}_i = (i, \textsc{tag}_i)$.
2. Run $\langle P_{\mathsf{tag}_i}, V_{\mathsf{tag}_i} \rangle$ with common input x and length parameter $\ell(n)$.

V: Accept if and only if all runs are accepting.

Figure 7: Protocol $\langle P_{\textsc{tag}}, V_{\textsc{tag}} \rangle$.

Notice that $\langle P_{\textsc{tag}}, V_{\textsc{tag}} \rangle$ has a constant number of rounds (since each $\langle P_{\mathsf{tag}_i}, V_{\mathsf{tag}_i} \rangle$ is constant-round). Also notice that for $i \in [n]$, the length of $\mathsf{tag}_i = (i, \textsc{tag}_i)$ is

$$|i| + |\textsc{tag}_i| = \log n + 1 = \log(2n).$$

Viewing $(i, \textsc{tag}_i)$ as elements in $[2n]$ we infer that the length of verifier messages in $\langle P_{\mathsf{tag}_i}, V_{\mathsf{tag}_i} \rangle$ is upper bounded by $2n\ell(n)$. Hence, as long as $\ell(n) = \mathrm{poly}(n)$ the length of verifier messages in $\langle P_{\textsc{tag}}, V_{\textsc{tag}} \rangle$ is $2n^2\ell(n) = \mathrm{poly}(n)$.

We now turn to show that for any $\textsc{tag} \in 2^n$, the protocol $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}} \rangle$ is an interactive argument. In fact, what we show is a stronger statement. Namely, that the protocols $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}} \rangle$ are proofs (actually arguments) of knowledge (as in Definition 2.7). For simplicity of exposition, we will show how to prove the above assuming a family of hash functions that is collision resistant against $T(n) = n^{\omega(1)}$-sized circuits. As mentioned in Remark 4.1, by slightly modifying $\mathbf{R}_{\mathsf{sim}}$, one can prove the same statement under the more standard assumption of collision resistance against polynomial-sized circuits.

Proposition 4.2 (Argument of knowledge) *Let $\langle P_{\mathsf{sWI}}, V_{\mathsf{sWI}}\rangle$ and $\langle P_{\mathsf{UA}}, V_{\mathsf{UA}}\rangle$ be the protocols used in the construction of $\langle P_{\mathsf{sUA}}, V_{\mathsf{sUA}}\rangle$. Suppose that $\{\mathcal{H}_n\}_n$ is collision resistant for $T(n)$-sized circuits, that* **Com** *is statistically hiding, that $\langle P_{\mathsf{sWI}}, V_{\mathsf{sWI}}\rangle$ is a statistical witness indistinguishable argument of knowledge, and that $\langle P_{\mathsf{UA}}, V_{\mathsf{UA}}\rangle$ is a universal argument. Then, for any $\text{TAG} \in \{0,1\}^n$, $\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle$ is an interactive argument of knowledge.*

Similar arguments to the ones used to prove Proposition 4.2 have already appeared in the works of Barak [1], and Barak and Goldreich [3]. While our proof builds on these arguments, it is somewhat more involved. For the sake of completeness, the full proof appears in Appendix A.

5 Proving Simulation-Extractability

Our central technical Lemma states that the family of protocols $\{\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle\}_{\text{TAG}\in\{0,1\}^n}$ is simulation extractable. As shown in Proposition 3.6 this implies that these protocols are also non-malleable zero-knowledge.

Lemma 5.1 (Simulation extractability) *Suppose that* **Com** *are statistically hiding, that $\{\mathcal{H}_n\}_n$ is a family of collision-resistant hash functions, that $\langle P_{UA}, V_{UA}\rangle$ is a special-purpose WIUARG, and that $\ell(n) \geq 2n^2 + 2n$. Then, $\{\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle\}_{\text{TAG}\in\{0,1\}^n}$ is simulation extractable.*

The proof of Lemma 5.1 is fairly complex. To keep things manageable, we first give an overview of the proof, describing the key ideas used for establishing the simulation extractability of the family $\{\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle\}_{\mathsf{tag}\in[2n]}$. This is followed by a full proof for the case of $\{\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle\}_{\text{TAG}\in\{0,1\}^n}$.

5.1 Proof Overview

Consider a man-in-the-middle adversary A that is playing the role of the verifier of $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$ in the left interaction while simultaneously playing the role of the prover of $\langle P_{\tilde{\mathsf{tag}}}, V_{\tilde{\mathsf{tag}}}\rangle$ in the right interaction. Recall that in order to prove simulation-extractability we have to show that for any such A, there exists a combined simulator-extractor $\mathcal{S} = (\mathsf{SIM}, \mathsf{EXT})$ that is able to simulate both the left and the right interactions for A, while simultaneously extracting a witness to the statement $\tilde{x}$ proved in the right interaction.

Towards this goal, we will construct a simulator S that is able to "internally" generate P_{tag} messages for the left interaction of A, even if the messages in the right interaction are forwarded to A from an "external" verifier $V_{\tilde{\mathsf{tag}}}$. The simulator S is almost identical to the simulator of [38] and exploits the difference in message lengths between the protocols $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$ and $\langle P_{\tilde{\mathsf{tag}}}, V_{\tilde{\mathsf{tag}}}\rangle$. As the analysis will demonstrate, the left view produced by the simulator S is statistically indistinguishable from A's actual interactions with an honest left prover P_{tag}. Furthermore, we show that:

1. It will be possible to construct a procedure SIM that faithfully simulates A's view in a man-in-the-middle execution. To do so, we will honestly play the role of $V_{\tilde{\mathsf{tag}}}$ in the right interaction and use S to simulate A's left interaction with P_{tag} (pretending that the messages from the right interaction came from an external $V_{\tilde{\mathsf{tag}}}$).

2. It will be possible to construct a procedure EXT that extracts witnesses for the statements $\tilde{x}$ proved in the right interactions of the views generated by the above SIM. To do so, we will use S to transform A into a stand alone prover $P^*_{\tilde{\mathsf{tag}}}$ for the statement $\tilde{x}$. This will be done by having $P^*_{\tilde{\mathsf{tag}}}$ internally emulate A's execution, while forwarding A's messages to an external

honest verifier $V_{\tilde{\mathsf{tag}}}$, and using S to simulate A's left interaction with P_{tag}. We can then invoke the knowledge extractor that is guaranteed by the (stand-alone) proof of knowledge property of $\langle P_{\tilde{\mathsf{tag}}}, V_{\tilde{\mathsf{tag}}}\rangle$ and obtain a witness for $\tilde{x} \in L$.

It is important to have both SIM and EXT use the same simulator program S (with same random coins) in their respective executions. Otherwise, we are not guaranteed that the statement $\tilde{x}$ appearing in the output of SIM is the same one EXT extracts a witness from.[9]

The execution of S (with one specific scheduling of messages) is depicted in Figure 8 below. In order to differentiate between the left and right interactions, messages m in the right interaction are labeled as $\tilde{m}$. Stage 2 messages in the left and right interactions are denoted u and $\tilde{u}$ respectively.

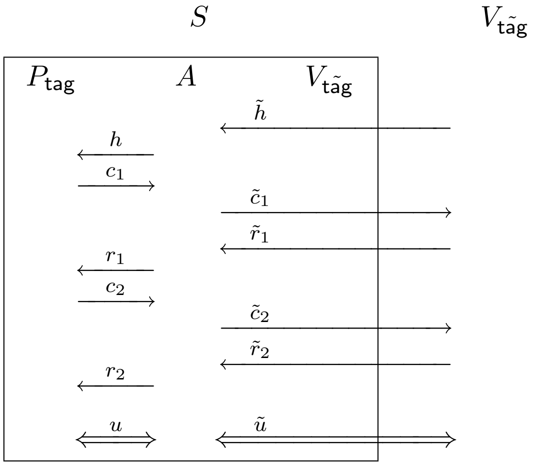

Figure 8: The simulator S.

The main hurdle in implementing S is in making the simulation of the left interaction work. The problem is that the actual code of the verifier whose view we are simulating is only partially available to S. This is because the messages sent by A in the left interaction also depend on the messages A receives in the right interaction. These messages are sent by an "external" $V_{\tilde{\mathsf{tag}}}$, and $V_{\tilde{\mathsf{tag}}}$'s code (randomness) is not available to S.

Technically speaking, the problem is implied by the fact that the values of the r_i's do not necessarily depend only on the corresponding c_i, but rather may also depend on the "external" right messages $\tilde{r_i}$. Thus, setting $\Pi = A$ and $y = c_i$ in the simulation (as done in Section 4.3) will not be sufficient, since in some cases it is simply not true that $r_i = A(c_i)$.

Intuitively, the most difficult case to handle is the one in which $\tilde{r_1}$ is contained in Slot 1 of $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$ and $\tilde{r_2}$ is contained in Slot 2 of $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$ (as in Figure 8 above). In this case $r_i = A(c_i, \tilde{r_i})$ and so $r_i = A(c_i)$ does not hold for either $i \in \{1,2\}$. As a consequence, the simulator will not be able to produce views of convincing Stage 2 interactions with A. In order to overcome the difficulty, we will use the fact that for a given instance $\langle h, c_i, r_i\rangle$, the string c_i is short enough to be "augmented" by $\tilde{r_i}$ while still satisfying the relation $\mathbf{R}_{\mathsf{sim}}$.

Specifically, as long as $|c_i| + |\tilde{r_i}| \leq |r_i| - n$ the relation $\mathbf{R}_{\mathsf{sim}}$ can be satisfied by setting $y = (c_i, \tilde{r_i})$. This guarantees that indeed $\Pi(y) = r_i$. The crux of the argument lies in the following fact.

Fact 5.2 *If* $\mathsf{tag} \neq \tilde{\mathsf{tag}}$ *then there exists* $i \in \{1,2\}$ *so that* $|\tilde{r_i}| \leq |r_i| - \ell(n)$.

[9] The statement $\tilde{x}$ will remain unchanged because $\tilde{x}$ occurs prior to any message in the right interaction (and hence does not depend on the external messages received by $P^*_{\tilde{\mathsf{tag}}}$).

By setting $y = (c_i, \tilde{r_i})$ for the appropriate i, the simulator is thus always able to satisfy $\mathbf{R}_{\mathsf{sim}}$ for some $i \in \{1,2\}$. This is because the "auxiliary" string y used in order to enable the prediction of r_i is short enough to pass the inspection at Condition 1 of $\mathbf{R}_{\mathsf{sim}}$ (i.e., $|y| = |c_i| + |\tilde{r_i}| \leq |r_i| - n$).[10] Once $\mathbf{R}_{\mathsf{sim}}$ can be satisfied, the simulator is able to produce views of convincing interactions that are computationally indistinguishable from real left interactions.[11]

The extension of the above analysis to the case of $\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle$ has to take several new factors into consideration. First, each execution of $\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle$ consists of n parallel executions of $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$ (and not only one). This imposes the constraint $\ell(n) \geq 2n^2+2n$, and requires a careful specification of the way in which the left simulation procedure handles the verifier challenges in $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$. Secondly, and more importantly, the simulation procedure will not be able to handle a case in which $\langle P_{\tilde{\text{TAG}}}, V_{\text{TAG}}\rangle$ messages of the right interaction are forwarded from an external verifier $V_{\tilde{\text{TAG}}}$ (because these messages are too long for the simulation to work).

While this does not seem to pose a problem for the SIM procedure, it suddenly becomes unclear how to construct a stand alone prover $P^*_{\tilde{\text{TAG}}}$ for the EXT procedure (since this involves forwarding messages from $V_{\tilde{\text{TAG}}}$). The way around this difficulty will be to construct a stand-alone prover $P^*_{\tilde{\mathsf{tag}}}$ for a *single* sub-protocol $\langle P_{\tilde{\mathsf{tag}}}, V_{\tilde{\mathsf{tag}}}\rangle$ instead. This will guarantee that the only messages that end up being forwarded are sent by an external verifier $V_{\tilde{\mathsf{tag}}}$, whose messages are short enough to make the simulation work. Once such a $P^*_{\tilde{\mathsf{tag}}}$ is constructed it is possible to use the knowledge extractor for $\langle P_{\tilde{\mathsf{tag}}}, V_{\tilde{\mathsf{tag}}}\rangle$ in order to obtain a witness for $\tilde{x}$.

5.2 Many-to-One Simulation-Extractabiity

We now proceed with the proof of Lemma 5.1. To establish simulation-extractability of $\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle$, we first consider what happens when a man-in-the-middle adversary is simultaneously involved in the verification of *many* different (parallel) executions of $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$ on the left while proving a *single* interaction $\langle P_{\tilde{\mathsf{tag}}}, V_{\tilde{\mathsf{tag}}}\rangle$ on the right. As it turns out, as long as the number of left executions is bounded in advance, we can actually guarantee simulation-extractability even in this scenario.

For any $\text{TAG} = (\mathsf{tag}_1, \ldots, \mathsf{tag}_n) \in [2n]^n$ we consider a left interaction in which the protocols $\langle P_{\mathsf{tag}_1}, V_{\mathsf{tag}_1}\rangle, \ldots, \langle P_{\mathsf{tag}_n}, V_{\mathsf{tag}_n}\rangle$ are executed in parallel with common input $x \in \{0,1\}^n$, and a right interaction in which $\langle P_{\tilde{\mathsf{tag}}}, V_{\tilde{\mathsf{tag}}}\rangle$ is executed with common input $\tilde{x} \in \{0,1\}^n$. The strings $\tilde{\mathsf{tag}}$ and $\tilde{x}$ are chosen adaptively by the man-in-the-middle A. The witness used by the prover in the left interaction is denoted by w, and the auxiliary input used by A is denoted by z.

Proposition 5.3 *Let A be a MIM adversary as above, and suppose that $\ell(n) \geq 2n^2+2n$. Then, there exists a probabilistic expected polynomial time, $\mathcal{S}$ such that the following conditions hold:*

1. *The probability ensembles $\{\mathcal{S}_1(x,z,\text{TAG})\}_{x,z,\text{TAG}}$ and $\{\mathsf{view}_A(x,z,\text{TAG})\}_{x,z,\text{TAG}}$ are statistically close over L, where $\mathcal{S}_1(x,z,\text{TAG})$ denotes the first output of $\mathcal{S}(x,z,\text{TAG})$.*

2. *Let $x \in L, z \in \{0,1\}^*, \text{TAG} \in \{0,1\}^{t(|x|)}$ and let (view, w) denote the output of $\mathcal{S}(x,z,\text{TAG})$ (on input some random tape). Let $\tilde{x}$ be the right-execution statement appearing in* view *and let* $\tilde{\mathsf{tag}}$ *denote the right-execution tag. Then, if the right-execution in* view *is accepting AND* $\mathsf{tag}_j \neq \tilde{\mathsf{tag}}$ *for all $j \in [n]$, then $R_L(\tilde{x}, w) = 1$.*

[10] This follows from the fact that $\ell(n) \geq 3n$ and $|c_i| = 2n$.

[11] In the above discussion we have been implicitly assuming that $\tilde{h}, \tilde{u}$ are not contained in the two slots of $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$ (where $\tilde{h}$ denotes the hash function in the right interaction and $\tilde{u}$ denotes the sequence of messages sent in the right *WIUARG*). The case in which $\tilde{h}, \tilde{u}$ are contained in the slots can be handled by setting $\ell(n) \geq 4n$, and by assuming that both $|\tilde{h}|$ and the total length of the messages sent by the verifier in the *WIUARG* is at most n. We mention that the latter assumption is reasonable, and is indeed satisfied by known protocols (e.g. the *WIUARG* of [3]).

Proof: As discussed in Section 5.1, we construct a "many-to-one" simulator S that internally generates a left view of $\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle = (\langle P_{\mathsf{tag}_1}, V_{\mathsf{tag}_1}\rangle, \ldots, \langle P_{\mathsf{tag}_n}, V_{\mathsf{tag}_n}\rangle)$ for A while forwarding messages from the right interaction to an external honest verifier $V_{\tilde{\mathsf{tag}}}$. This simulator is essentially identical to the simulator of [38].[12] We then show how to use S to construct the procedures $(\mathsf{SIM}, \mathsf{EXT})$.

5.2.1 The Many-to-One Simulator

The many-to-one simulator S invokes A as a subroutine. It attempts to generate views of the left and right interactions that are indistinguishable from A's view in real interactions. Messages in the right interaction are forwarded by S to an "external" honest verifier $V_{\tilde{\mathsf{tag}}}$ for $\langle P_{\tilde{\mathsf{tag}}}, V_{\tilde{\mathsf{tag}}}\rangle$, whose replies are then fed back to A. Messages in the left interaction are handled by n "sub-simulators" $S_1, \ldots, S_n$, where each S_j is responsible for generating the messages of the sub-protocol $\langle P_{\mathsf{tag}_j}, V_{\mathsf{tag}_j}\rangle$. The execution of the simulator is depicted in Figure 9 (for simplicity, we ignore the messages $h^1, \ldots, h^n, u^1, \ldots, u^n$ and $\tilde{h}, \tilde{u}$).

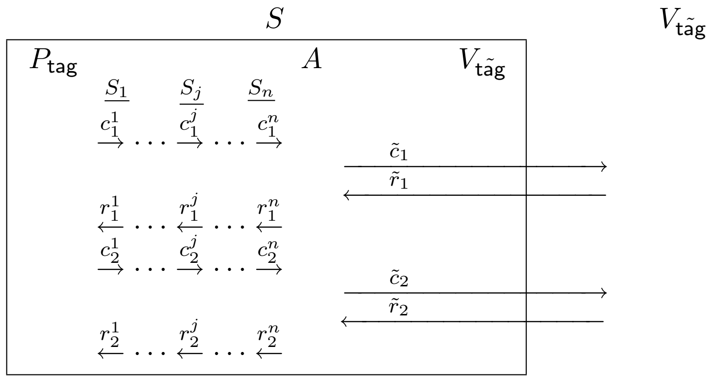
S
$V_{\tilde{\mathsf{tag}}}$
P_{tag}
A
$V_{\tilde{\mathsf{tag}}}$
S_1 S_j S_n
$c_1^1 \ldots c_1^j \ldots c_1^n$
$\tilde{c}_1$
$\tilde{r}_1$
$r_1^1 \ldots r_1^j \ldots r_1^n$
$c_2^1 \ldots c_2^j \ldots c_2^n$
$\tilde{c}_2$
$\tilde{r}_2$
$r_2^1 \ldots r_2^j \ldots r_2^n$

Figure 9: The "many-to-one" simulator S.

The specific actions of a sub-simulator S_j depend on the scheduling of Stage 1 messages as decided by A. The scheduling of left and right messages are divided into three separate cases which are depicted in Figure 10 below. In all three cases we make the simplifying assumption that $\tilde{h}$ is scheduled in the right interaction before $h^1 \ldots, h^n$ are scheduled in the left interaction. We also assume that the $\langle P_{\mathsf{sUA}}, V_{\mathsf{sUA}}\rangle$ messages $\tilde{u}$ in the right interaction are scheduled after the $\langle P_{\mathsf{sUA}}, V_{\mathsf{sUA}}\rangle$ messages $u^1, \ldots, u^n$ in the left interaction. We later argue how these assumptions can be removed.

Let A_j be a program that acts exactly like A, but for any $i \in \{1, 2\}$ instead of outputting $r_i^1, \ldots, r_i^n$ it outputs only r_i^j. Given a string $\alpha \in \{0,1\}^*$, let $A(\alpha, \cdot)$ denote the program obtained by "hardwiring" α into it (i.e., $A(\alpha, \cdot)$ evaluated on β equals $A(\alpha, \beta)$). We now describe S_j's actions in each of the three cases:

None of $\tilde{r}_1, \tilde{r}_2$ is contained in Slot 1 of $\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle$: Assume for simplicity that $\tilde{c_1}, \tilde{r_1}, \tilde{c_2}, \tilde{r_2}$ are all contained in Slot 2 of $\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle$ (Fig. 10a). The simulator S_j sets $c_1 = \text{Com}(h(\Pi_1); s_1)$ and $c_2 = \text{Com}(0^n; s_2)$ where $\Pi_1 = A_j(x, \cdot)$. It then sets the triplet $\langle \Pi_1, (c_1^1, \ldots, c_1^n), s_1\rangle$ as witness for $\langle h^j, c_1^j, r_1^j\rangle \in L_{\mathsf{sim}}$.

[12] In fact, the simulator presented here is somewhat simplified in that we only consider n *parallel* executions of $\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle$, whereas [38] shows a simulator also for n *concurrent* executions of the protocols.

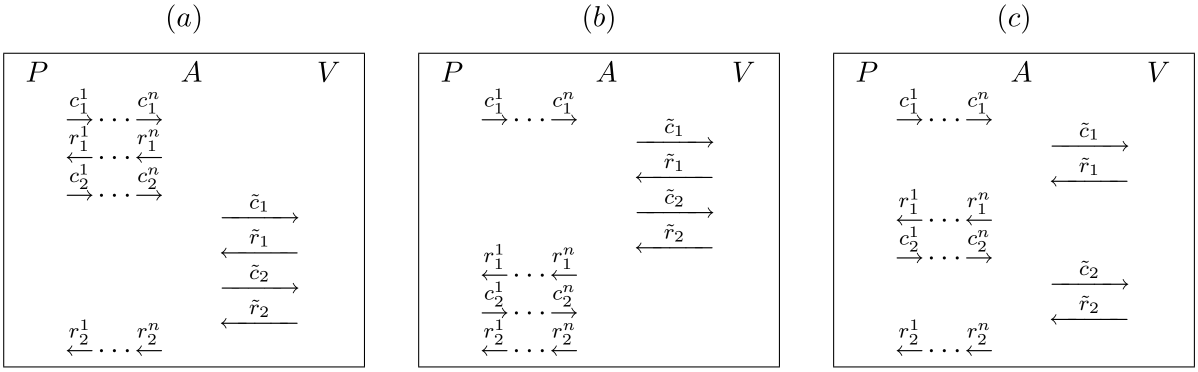

Figure 10: Three "representative" schedulings.

None of $\tilde{r_1}, \tilde{r_2}$ is contained in Slot 2 of $\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle$: Assume for simplicity that $\tilde{c_1}, \tilde{r_1}, \tilde{c_2}, \tilde{r_2}$ are all contained in Slot 1 of $\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle$ (Fig. 10b). The simulator S_j sets $c_1 = \text{Com}(0^n; s_1)$ and $c_2 = \text{Com}(h(\Pi_2); s_2)$ where $\Pi_2 = A_j(x, c_1^1, \ldots, c_1^n, \tilde{r_1}, \tilde{r_2}, \cdot)$. It then sets the triplet $\langle \Pi_2, (c_2^1, \ldots, c_2^n), s_2\rangle$ as witness for $\langle h^j, c_2^j, r_2^j\rangle \in L_{\mathsf{sim}}$.

$\tilde{r_1}$ is contained in Slot 1 and $\tilde{r_2}$ is contained in Slot 2 of $\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle$: In this case $\tilde{c_1}, \tilde{r_1}$ are both contained in Slot 1 of $\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle$, and $\tilde{c_2}, \tilde{r_2}$ are both contained in Slot 2 of $\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle$ (Fig. 10c). The simulator S_j sets $c_1 = \text{Com}(h(\Pi_1); s_1)$ and $c_2 = \text{Com}(h(\Pi_2); s_2)$ where $\Pi_1 = A_j(x, \cdot)$ and $\Pi_2 = A_j(x, c_1^1, \ldots, c_1^n, \tilde{r_1}, \cdot)$. Then:

- If $\mathsf{tag}_j > \tilde{\mathsf{tag}}$, the simulator sets $\langle \Pi_1, (c_1^1, \ldots, c_1^n, \tilde{r_1}), s_1\rangle$ as witness for $\langle h^j, c_1^j, r_1^j\rangle \in L_{\mathsf{sim}}$.
- If $\mathsf{tag}_j < \tilde{\mathsf{tag}}$, the simulator sets $\langle \Pi_2, (c_2^1, \ldots, c_2^n, \tilde{r_2}), s_2\rangle$ as witness for $\langle h^j, c_2^j, r_2^j\rangle \in L_{\mathsf{sim}}$.

In all cases, combining the messages together results in a Stage 1 transcript $\tau_1^j = \langle h^j, c_1^j, r_1^j, c_2^j, r_2^j\rangle$. By definition of $\langle P_{\mathsf{tag}_j}, V_{\mathsf{tag}_j}\rangle$, the transcript τ_1^j induces a Stage 2 special-purpose $WIUARG$ with common input $(x, \langle h^j, c_1^j, r_1^j\rangle, \langle h^j, c_2^j, r_2^j\rangle)$. The sub-simulator S_j now follows the prescribed prover strategy P_{sUA} and produces a Stage 2 transcript τ_2^j for $\langle P_{\mathsf{tag}_j}, V_{\mathsf{tag}_j}\rangle$.

Remark 5.4 (Handling $\tilde{h}$ and $\tilde{u}$) *To handle the case in which either $\tilde{h}$ or $\tilde{u}$ are contained in one of the slots, we set $\ell(n) \geq 2n^2 + 2n$ and let the simulator append either $\tilde{h}$ or $\tilde{u}$ to the auxiliary string y (whenever necessary). This will guarantee that the program committed to by the simulator indeed outputs the corresponding "challenge" r_i^j, when fed with y as input. The crucial point is that even after appending $\tilde{h}$ or $\tilde{u}$ to y, it will still be the case that $|y| \leq |r_i^j| - n$. This just follows from the fact that the total length of $\tilde{h}$ and the messages $\tilde{u}$ sent in $\langle P_{\mathsf{sUA}}, V_{\mathsf{sUA}}\rangle$ is upper bounded by, say n, and that the gap between $|r_i^j|$ and the "original" $|y|$ (i.e. before appending $\tilde{h}$ or $\tilde{u}$ to it) is guaranteed to be at least n (this follows from the requirement $\ell(n) \geq 2n^2 + 2n$).*

Output of S. To generate its output, which consists of a verifier view of a $\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle$ interaction, S combines all the views generated by the S_j's. Specifically, letting $\sigma_1^j = (c_1^j, c_2^j)$ be the verifier's view of τ_1^j, and σ_2^j be the verifier's view of τ_2^j, the output of S consists of $(\sigma_1, \sigma_2) = ((\sigma_1^1, \ldots, \sigma_1^n), (\sigma_2^1, \ldots, \sigma_2^n))$.

5.2.2 The Simulator-Extractor

Using S, we construct the simulator-extractor $\mathcal{S} = (\mathsf{SIM}, \mathsf{EXT})$. We start with the machine SIM. In the right interaction SIM's goal is to generate messages by a verifier $V_{\tilde{\mathsf{tag}}}$. This part of the simulation is quite straightforward, and is performed by simply playing the role of an honest verifier in the execution of the protocol (with the exception of cases in which $\mathsf{tag}_j = \tilde{\mathsf{tag}}$ for some $j \in [n]$ – see below for details). In the left interaction, on the other hand, SIM is supposed to act as a prover P_{TAG}, and this is where S is invoked.

The machine SIM. On input (x, z, TAG), and given a man-in-the-middle adversary A, SIM starts by constructing a man-in-the-middle adversary A' that acts as follows:

Internal messages: Pick random $M' = (\tilde{h}, \tilde{r}_1, \tilde{r}_2, \tilde{u})$ verifier messages for the right interaction.

Right interaction: The statement $\tilde{x}$ proved is the same as the one chosen by A. If there exists $j \in [n]$ so that $\mathsf{tag}_j = \tilde{\mathsf{tag}}$, use the messages in M' in order to internally emulate a right interaction for A (while ignoring external $V_{\tilde{\mathsf{tag}}}$ messages M). Otherwise, forward A's messages in the right interaction to an external $V_{\tilde{\mathsf{tag}}}$ and send back his answers M to A.

Left interaction: As induced by the scheduling of messages by A, forward the messages sent by A in the left interaction to an external prover P_{TAG}, and send back his answers to A.

Fig. 11.a describes the behavior of A' in case $\mathsf{tag}_j \neq \tilde{\mathsf{tag}}$ for all $j \in [n]$, whereas Fig. 11.b describes its behavior otherwise. The purpose of constructing such an A' is to enable us to argue that for all "practical purposes" the man-in-the-middle adversary never uses a $\tilde{\mathsf{tag}}$ that satisfies $\mathsf{tag}_j = \tilde{\mathsf{tag}}$ for some $j \in [n]$ (as in such cases A' ignores all messages M in the right interaction anyway).

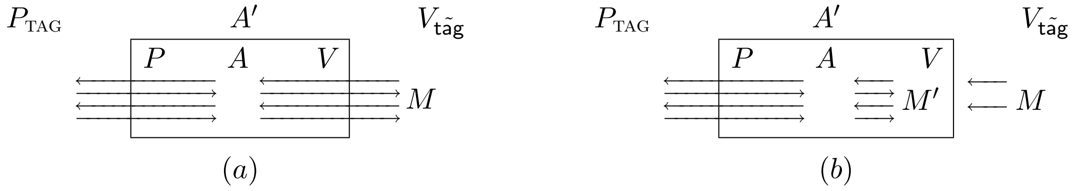

Figure 11: The adversary A'.

Given the new adversary A', the machine SIM picks random $V_{\tilde{\mathsf{tag}}}$ messages M, and invokes the simulator S with random coins $\bar{s}$. The simulator's goal is to generate a view of a left interaction for an A' that receives messages M in the right interaction.

Let (σ_1, σ_2) be the view generated by S for the left $\langle P_{\text{TAG}}, V_{\text{TAG}} \rangle$ interaction, and let $\tilde{\mathsf{tag}}$ be the tag sent by A in the right interaction when $(\sigma_1 \sigma_2)$ is its view of the left interaction. If there exists $j \in [n]$ so that $\mathsf{tag}_j = \tilde{\mathsf{tag}}$ then SIM outputs M' as the right view of A. Otherwise, it outputs M. SIM always outputs (σ_1, σ_2) as a left view for A.

The machine EXT. The machine EXT starts by sampling a random execution of SIM, using random coins $\bar{s}, M', M$. Let $\tilde{x}$ be the right hand side common input that results from feeding the output of SIM to A. Our goal is to extract a witness to the statement $\tilde{x}$. At a high level, EXT acts the following way:

1. If the right session was *not* accepting or $\mathsf{tag}_j = \tilde{\mathsf{tag}}$ for some $j \in [n]$, EXT will assume that no witness exists for the statement $\tilde{x}$, and will refrain from extraction.

2. Otherwise, EXT constructs a stand-alone prover $P^*_{\tilde{\mathsf{tag}}}$ for the right interaction $\langle P_{\tilde{\mathsf{tag}}_i}, V_{\tilde{\mathsf{tag}}_i}\rangle$, and from which it will later attempt to extract the witness (see Figure 12).

   In principle, the prover $P^*_{\tilde{\mathsf{tag}}}$ will follow SIM's actions using the same random coins $\bar{s}$ used for initially sampling the execution of SIM. However, $P^*_{\tilde{\mathsf{tag}}}$'s execution will differ from SIM's execution in that it will not use the messages M in the right interaction of A, but will rather forward messages receives from an external verifier $V_{\tilde{\mathsf{tag}}}$

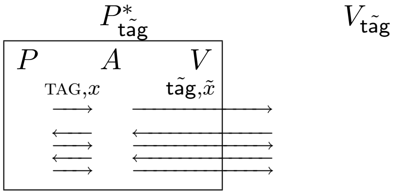
$P^*_{\tilde{\mathsf{tag}}}$
$V_{\tilde{\mathsf{tag}}}$
P
A
V
TAG,x
$\tilde{\mathsf{tag}}$,$\tilde{x}$

Figure 12: The prover $P^*_{\tilde{\mathsf{tag}}}$.

3. Once $P^*_{\tilde{\mathsf{tag}}}$ is constructed, EXT can apply the knowledge extractor, guaranteed by the proof of knowledge property of $\langle P_{\tilde{\mathsf{tag}}}, V_{\tilde{\mathsf{tag}}}\rangle$, and extract a witness w to the statement $\tilde{x}$. In the unlikely event that the extraction failed, EXT outputs `fail`. Otherwise, it outputs w.

Remark 5.5 *It is important to have a prover $P^*_{\tilde{\mathsf{tag}}}$ for the* entire *protocol $\langle P_{\tilde{\mathsf{tag}}_i}, V_{\tilde{\mathsf{tag}}_i}\rangle$ (and not just for $\langle P_{\mathsf{sUA}}, V_{\mathsf{sUA}}\rangle$). This is required in order to argue that the witness extracted is a witness for $\tilde{x}$ and not a witness to $\langle \tilde{h}, \tilde{c}_1, \tilde{r}_1\rangle \in L_{\mathsf{sim}}$ or to $\langle \tilde{h}, \tilde{c}_2, \tilde{r}_2\rangle \in L_{\mathsf{sim}}$ (which could indeed be the case if we fixed the messages $\langle \tilde{h}, \tilde{c}_1, \tilde{r}_1, \tilde{c}_2, \tilde{r}_2\rangle$ in advance).*

Output of simulator-extractor $\mathcal{S}$. The combined simulator-extractor $\mathcal{S}$ runs EXT and outputs `fail` whenever EXT does so. Otherwise, it output the view output by SIM (in the execution by EXT) followed by the output of EXT.

5.2.3 Correctness of Simulation-Extraction

We start by showing that the view of A in the simulation by SIM is statistically close to its view in an actual interaction with P_{TAG} and $V_{\tilde{\mathsf{tag}}}$.

Lemma 5.6 *$\{\mathcal{S}_1(x, z, \text{TAG})\}_{x\in L, z\in\{0,1\}^*, \text{TAG}\in\{0,1\}^m}$ and $\{\mathsf{view}_A(x, z, \text{TAG})\}_{x\in L, z\in\{0,1\}^*, \text{TAG}\in\{0,1\}^m}$ are statistically close over L, where $\mathcal{S}_1(x, z, \text{TAG})$ denotes the first output of $\mathcal{S}(x, z, \text{TAG})$.*

Proof: Recall $\mathcal{S}$ proceeds by first computing a joint view $\langle(\sigma_1, \sigma_2), M\rangle$ (by running SIM) and then outputs this view only if EXT does not output `fail`. Below we show that the output of SIM is statistically close to the view of A in real interactions. This concludes that, since EXT outputs `fail` only in the event that the extraction fails, and since by the proof of knowledge property of $\langle P_{\tilde{\mathsf{tag}}}, V_{\tilde{\mathsf{tag}}}\rangle$ the extraction fails only with negligible probability, the first output of $\mathcal{S}$ is also statistically close to the view of A.

Let the random variable $\{\mathsf{SIM}(x, z, \text{TAG})\}$ denote the view $\langle(\sigma_1, \sigma_2), M\rangle$ output by $\{\mathsf{SIM}(x, z, \text{TAG})\}$ in the execution by $\mathcal{S}$.

Claim 5.7 *$\{\mathsf{SIM}(x,z,\textsc{tag})\}_{x\in L,z\in\{0,1\}^*,\textsc{tag}\in\{0,1\}^m}$ and $\{\mathsf{view}_A(x,z,\textsc{tag})\}_{x\in L,z\in\{0,1\}^*,\textsc{tag}\in\{0,1\}^m}$ are statistically close over L.*

Proof: Recall that the output of SIM consists of the tuple $\langle(\sigma_1,\sigma_2),M\rangle$, where (σ_1,σ_2) is the left view generated by the simulator S and $M=(\tilde{h},\tilde{r}_1,\tilde{r}_2,\tilde{u})$ are uniformly chosen messages that are fed to A during the simulation. In other words:

$$\{\mathsf{SIM}(x,z,\textsc{tag})\}_{x,z,\textsc{tag}} = \{(S(x,z,\textsc{tag}),U_{|M|})\}_{x,z,\textsc{tag}}.$$

Let $x\in L$, $\textsc{tag}\in[2n]^n$ and $z\in\{0,1\}^*$. To prove the claim, we will thus compare the distribution $(S(x,z,\textsc{tag}),U_{|M|})$ with real executions of the man-in-the-middle $A(x,z,\textsc{tag})$.

We start by observing that, whenever M is chosen randomly, a distinguisher between real and simulated views of the left interaction of A yields a distinguisher between $S(x,z,\textsc{tag})$ and $\mathsf{view}_A(x,z,\textsc{tag})$. This follows from the following two facts (both facts are true regardless of whether $\mathsf{tag}_j=\tilde{\mathsf{tag}}$ for some $j\in[n]$):

1. The messages M (resp, M') appearing in its output are identically distributed to messages in a real right interaction of A with $V_{\tilde{\mathsf{tag}}}$ (by construction of SIM).

2. The simulation of the left interaction in SIM is done with respect to an A whose right hand side view consists of the messages M (resp. M').

In particular, to distinguish whether a tuple $(\sigma_1,\sigma_2),M$ was drawn according to $S(x,z,\textsc{tag})$ or according to $\mathsf{view}_A(x,z,\textsc{tag})$ one could simply take M, hardwire it into A and invoke the distinguisher for the resulting stand alone verifier for $\langle P_{\textsc{tag}},V_{\textsc{tag}}\rangle$ (which we denote by V^*_{tag}). Thus, all we need to prove is the indistinguishability of real and simulated views of an arbitrary stand alone verifier $V^*_{\textsc{tag}}$ (while ignoring the messages M). We now proceed with the proof of the claim.

Consider a random variable (σ_1,σ_2) that is distributed according to the output of $S(x,z,\textsc{tag})$, and a random variable (π_1,π_2) that is distributed according to the verifier view of $\langle P_{\textsc{tag}},V_{\textsc{tag}}\rangle$ in $\{\mathsf{view}_A(x,z,\textsc{tag})\}_{z,x,\textsc{tag}}$ (where π_1,π_2 are A's respective views of Stages 1 and 2). We will show that both (σ_1,σ_2) and (π_1,π_2) are statistically close to the hybrid distribution (σ_1,π_2). This distribution is obtained by considering a hybrid simulator that generates σ_1 exactly as S does, but uses the witness w for $x\in L$ in order to produce π_2.

Sub Claim 5.8 *The distribution (π_1,π_2) is statistically close to (σ_1,π_2).*

Proof: The claim follows from the (parallel) statistical hiding property of **Com**. Specifically, suppose that there exists a (possibly unbounded) D that distinguishes between the two distributions with probability ϵ. Consider a distinguisher D' that has the witness w for $x\in L$ and $h=V^*_{\textsc{tag}}(x)$ hardwired, and acts as follows. Whenever D' gets an input $(\bar{c}_1,\bar{c}_2)$, it starts by generating $\bar{r}_1=V^*_{\textsc{tag}}(x,\bar{c}_1)$ and $\bar{r}_2=V^*_{\textsc{tag}}(x,\bar{c}_1,\bar{c}_2)$. It then emulates a Stage 2 interaction between $V^*_{\textsc{tag}}(x,\bar{c}_1,\bar{c}_2)$ and the honest provers P_{sUA}, where $(x,\langle h,c_1^j,r_1^j\rangle,\langle h,c_2^j,r_2^j\rangle)$ is the common input for the j^{th} interaction, and P_{sUA} is using w as a witness for $x\in L$. Let π_2 denote the resulting verifier view. D' outputs whatever D outputs on input $(\bar{c}_1,\bar{c}_2,\pi_2)$.

Notice that if $(c_1^j,c_2^j)=(\mathbf{Com}(0^n),\mathbf{Com}(0^n))$ for all $j\in[n]$, then the input fed to D is distributed according to (π_1,π_2), whereas if $(c_1^j,c_2^j)=(\mathbf{Com}(h(\Pi_1^j)),\mathbf{Com}(h(\Pi_2^j)))$, then the input fed to D is distributed according to (σ_1,π_2). Thus, D' has advantage ϵ in distinguishing between two tuples of n committed values. Hence, if ϵ is non-negligible we reach contradiction to the statistical hiding property of **Com**. ■

Sub Claim 5.9 *The distribution (σ_1, σ_2) is statistically close to (σ_1, π_2).*

Proof: The claim follows from the (parallel) statistical witness indistinguishability property of $\langle P_{\mathsf{sWI}}, V_{\mathsf{sWI}}\rangle$ and the (parallel) statistical hiding property of **Com** (both used in $\langle P_{\mathsf{sUA}}, V_{\mathsf{sUA}}\rangle$). Let $\sigma_2 = (\sigma_{2,1}, \sigma_{2,2})$, where $\sigma_{2,1}$ corresponds to a simulated view of Stage 1 of $\langle P_{\mathsf{sUA}}, V_{\mathsf{sUA}}\rangle$ and $\sigma_{2,2}$ corresponds to a Stage 2 of $\langle P_{\mathsf{sUA}}, V_{\mathsf{sUA}}\rangle$. Similarly, let $\pi_2 = (\pi_{2,1}, \pi_{2,2})$ correspond to the real views of Stages 1 and 2 of $\langle P_{\mathsf{sUA}}, V_{\mathsf{sUA}}\rangle$. We will show that both (σ_1, σ_2) and (σ_1, π_2) are statistically close to the hybrid distribution $(\sigma_1, (\sigma_{2,1}, \pi_{2,2}))$.

Suppose that there exists a (possibly unbounded) algorithm D that distinguishes between (σ_1, σ_2) and $(\sigma_1, (\sigma_{2,1}, \pi_{2,2}))$ with probability ϵ. Then there must exist a Stage 1 view $(\bar{c}_1, \bar{c}_2) = (c_1^1, \ldots, c_1^n, c_2^1, \ldots, c_2^n)$ for $\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle$, and a Stage 1 view $(\bar{\beta}, \bar{\delta}) = (\widehat{\beta}^1, \ldots, \beta^n, \widehat{\delta}^1, \ldots, \widehat{\delta}^n)$ for the sub-protocol $\langle P_{\mathsf{sUA}}, V_{\mathsf{sUA}}\rangle$ so that D has advantage ϵ in distinguishing between $\langle(\sigma_1, \sigma_2)\rangle$ and $\langle(\sigma_1, \pi_2)\rangle$ conditioned on $(\sigma_1, \sigma_{2,1}) = ((\bar{c}_1, \bar{c}_2), (\bar{\beta}, \bar{\delta}))$.

Consider a Stage 2 execution of $\langle P_{\mathsf{sUA}}, V_{\mathsf{sUA}}\rangle$ with $V^*_{\mathsf{sWI}} = V^*_{\text{TAG}}(x, \bar{c}_1, \bar{c}_2, \bar{\beta}, \bar{\delta}, \cdot)$ as verifier. Then, a distinguisher $D(\bar{c}_1, \bar{c}_2, \bar{\beta}, \bar{\delta}, \cdot)$ (i.e., D with $(\bar{c}_1, \bar{c}_2, \bar{\beta}, \bar{\delta})$ hardwired as part of its input) has advantage ϵ in distinguishing between an interaction of V^*_{sWI} with n honest P_{sWI} provers that use accepting $\langle P_{\mathsf{UA}}, V_{\mathsf{UA}}\rangle$ transcripts $(\alpha, \beta, \gamma, \delta)$, and an interaction of V^*_{sUA} with honest P_{sUA} provers that uses w as witness.[13] Thus, if ϵ is non-negligible we reach contradiction to the (parallel) statistical witness indistinguishability of $\langle P_{\mathsf{sWI}}, V_{\mathsf{sWI}}\rangle$.

Suppose now that there exists a (possibly unbounded) algorithm D that distinguishes between (σ_1, π_2) and $(\sigma_1, (\sigma_{2,1}, \pi_{2,2}))$ with probability ϵ. Consider a distinguisher D' that has the witness w for $x \in L$ and $(h, \bar{c}_1, \bar{c}_2)$ hardwired, and acts as follows. Whenever D' gets an input $(\bar{\beta}, \bar{\delta})$, it starts by generating $\bar{\alpha} = V^*_{\text{TAG}}(x, \bar{c}_1, \bar{c}_2)$ and $\bar{\gamma} = V^*_{\text{TAG}}(x, \bar{c}_1, \bar{c}_2, \bar{\beta})$. It then emulates a Stage 2 interaction between the V_{sWI} verifiers $V^*_{\text{TAG}}(x, \bar{c}_1, \bar{c}_2, \bar{\beta}, \bar{\delta})$ and the honest provers P_{sWI}, where $(x, \langle h, c_1^j, r_1^j\rangle, \langle h, c_2^j, r_2^j\rangle, \langle \alpha, \bar{\beta}, \gamma, \bar{\delta}\rangle)$ is the common input for the j^{th} interaction, and P_{sWI} is using w as a witness for $x \in L$. Let $\pi_{2,2}$ denote the resulting verifier view. D' outputs whatever D outputs on input $(\bar{c}_1, \bar{c}_2, \bar{\beta}, \bar{\delta}, \pi_{2,2})$.

Notice that if $(\widehat{\beta}^j, \widehat{\delta}^j) = (\mathbf{Com}(0^n), \mathbf{Com}(0^n))$ for all $j \in [n]$, then the input fed to D is distributed according to $(\sigma_1, (\pi_{2,1}, \pi_{2,2})) = (\sigma_1, \pi_2)$, whereas if $(\widehat{\beta}^j, \widehat{\delta}^j) = \mathbf{Com}(\beta^j), \mathbf{Com}(\delta^j)\rangle$, for some β^j, δ^j for which $(\alpha^j, \beta^j, \gamma^j, \delta^j)$ is an accepting $\langle P_{\mathsf{UA}}, V_{\mathsf{UA}}\rangle$ then the input fed to D is distributed according to $(\sigma_1, (\sigma_{2,1}, \pi_{2,2}))$. Thus, D' has advantage ϵ in distinguishing between two tuples of n committed values. Hence, if ϵ is non-negligible we reach contradiction to the statistical hiding property of **Com**. ■

Combining Sub-claims 5.9 and 5.8 we conclude that the ensembles $\{\mathsf{SIM}(x, z, \text{TAG})\}_{x,z,\text{TAG}}$ and $\{\mathsf{view}_A(x, z, \text{TAG})\}_{x,z,\text{TAG}}$ are statistically close. ■

This completes the proof of Lemma 5.6 ■

Lemma 5.10 *Let $x \in L$, $z \in \{0,1\}^*$, $\text{TAG} \in \{0,1\}^m$ and let (view, w) denote the output of $\mathcal{S}(x, z, \text{TAG})$ (on input some random tape). Let $\tilde{x}$ be the right-execution statement appearing in* view *and let* $\tilde{\mathsf{tag}}$ *denote the right-execution tag. Then, if the right-execution in* view *is accepting AND* $\mathsf{tag}_j \neq \tilde{\mathsf{tag}}$ *for all $j \in [n]$, then $R_L(\tilde{x}, w) = 1$.*

Proof: We start by noting that since $\mathcal{S}$ outputs `fail` whenever the extraction by EXT fails, the claim trivially holds in the event that the extraction by EXT fails.

[13] In accordance with the specification of $\langle P_{\mathsf{sUA}}, V_{\mathsf{sUA}}\rangle$, the transcripts $(\alpha, \beta, \gamma, \delta)$ are generated using programs Π_i^j as witnesses, where Π_i^j is the program chosen by the simulator S_j.

Observe that the right hand side input-tag pair $(\tilde{x}, \tilde{\mathsf{tag}})$ used in EXT is exactly the same as the one generated by SIM. This follows from the following two reasons: (1) Both EXT and SIM use the same random coins $\bar{s}$ in the simulation. (2) The input-tag pair $\tilde{x}, \tilde{\mathsf{tag}}$ is determined before any external message is received in the right interaction. In particular, the pair $(\tilde{x}, \tilde{\mathsf{tag}})$ is *independent* of the messages M (which is the only potential difference between the executions of SIM and EXT).

Since whenever the properties described in the hypothesis hold EXT performs extraction, and since the extraction by EXT proceeds until a witness is extracted (or until the extraction fails in which case, we are already done), we infer that $\mathcal{S}$ always outputs a witness to the statement $\tilde{x}$ proved in the right-interaction in the view output. ■

We conclude the proof by bounding the running time of the combined simulator-extractor $\mathcal{S}$.

Lemma 5.11 *$\mathcal{S}$ runs in expected polynomial time.*

Proof: We start by proving that the running time of SIM is polynomial. Recall that the SIM procedure invokes the simulator S with the adversary A'. Thus, we need to show that S runs in polynomial time. To this end, it will be sufficient to show that every individual sub-simulator S_j runs in polynomial time. We first do so assuming that $\mathsf{tag}_j \neq \tilde{\mathsf{tag}}$. We then argue that, by construction of the adversary A' this will be sufficient to guarantee polynomial running time even in cases where $\mathsf{tag}_j = \tilde{\mathsf{tag}}$.

Claim 5.12 *Suppose that $\mathsf{tag}_j \neq \tilde{\mathsf{tag}}$. Then, S_j completes the simulation in polynomial time.*

Proof: We start by arguing that, in each of the three cases specified in the simulation, the witness used by the simulator indeed satisfies the relation $\mathbf{R}_{\mathsf{sim}}$. A close inspection of the simulator's actions in the first two cases reveals that the simulator indeed commits to a program Π_i that on input $y = (c_i^1, \ldots, c_i^n)$ outputs the corresponding r_i. Namely,

- $\Pi_1(y) = A_j(x, c_1^1, \ldots, c_1^n) = r_1^j$.
- $\Pi_2(y) = A_j(x, c_1^1, \ldots, c_1^n, \tilde{r_1}, \tilde{r_2}, c_2^1, \ldots, c_2^n) = r_2^j$.

Since in both cases $|y| = n|c_j^i| = 2n^2 \leq \ell(n) - n \leq |r_i^j| - n$ it follows that $\mathbf{R}_{\mathsf{sim}}$ is satisfied. As for the third case, observe that for both $i \in \{1, 2\}$, if S_j sets $y = (c_i^1, \ldots, c_i^n, \tilde{r}_i)$ then

- $\Pi_i(y) = A_j(c_i^1, \ldots, c_i^n, \tilde{r}_i) = r_i^j$.

Since $\mathsf{tag}_j \neq \tilde{\mathsf{tag}}$ we can use Fact 5.2 and infer that there exists $i \in \{1, 2\}$ so that $|\tilde{r_i}| \leq |r_i^j| - \ell(n)$. This means that for every $j \in [n]$ the simulator S_j will choose the $i \in \{1, 2\}$ for which:

$$\begin{aligned} |y| &= |c_i^1| + \ldots + |c_i^n| + |\tilde{r}_i^i| \\ &= 2n^2 + |\tilde{r}_i^i| \\ &\leq 2n^2 + |r_i^j| - \ell(n) \qquad (2) \\ &\leq |r_i^j| - n \qquad (3) \end{aligned}$$

where Eq. (2) follows from $|\tilde{r_i}| \leq |r_i^j| - \ell(n)$ and Eq. (3) follows from the fact that $\ell(n) \geq 2n^2 + n$. Thus, $\mathbf{R}_{\mathsf{sim}}$ can always be satisfied by S_j.

Since the programs Π_i are of size poly(n) and satisfy $\Pi_i(y) = r_i^j$ in poly(n) time (because A_j does), then the verification time of $\mathbf{R}_{\mathsf{sim}}$ on the instance $\langle h, c_i^j, r_i^j \rangle$ is polynomial in n. By the perfect completeness and relative prover efficiency of $\langle P_{\mathsf{sUA}}, V_{\mathsf{sUA}} \rangle$, it then follows that the simulator is always able to make a verifier accept in polynomial time. ■

Remark 5.13 (Handling the case $\mathsf{tag}_j = \tilde{\mathsf{tag}}$) *When invoked by* SIM*, the simulator S will output an accepting left view even if A chooses $\tilde{\mathsf{tag}}$ so that $\mathsf{tag}_j = \tilde{\mathsf{tag}}$ for some $j \in [n]$.*[14] *This is because in such a case the A' whose view S needs to simulate ignores all right hand side messages, and feeds the messages M' to A internally. In particular, no external messages will be contained in neither Slot 1 or Slot 2 of $\langle P_{\text{TAG}}, V_{\text{TAG}} \rangle$. A look at the proof of Claim 5.12, reveals that in such cases the simulator can indeed always produce an accepting conversation (regardless of whether $\mathsf{tag}_j = \tilde{\mathsf{tag}}$ or not).*

It now remains to bound the *expected* running time of EXT. Recall that EXT uses the view sampled by SIM and proceeds to extract a witness only if the right interaction is accepting and $\mathsf{tag}_j \neq \tilde{\mathsf{tag}}$ for all $j \in [n]$. Using Claim 5.12, we know that the simulation internally invoked by the stand alone prover $P^*_{\tilde{\mathsf{tag}}}$ will always terminate in polynomial time (since $\mathsf{tag}_j \neq \tilde{\mathsf{tag}}$ for all $j \in [n]$ and A is a poly-time machine). We now argue that the extraction of the witness from $P^*_{\tilde{\mathsf{tag}}}$ conducted by EXT will terminate in expected polynomial time.

Let p denote the probability that A produces an accepting proof in the right execution in the simulation by SIM. Let p' denote the probability that A produces an accepting proof in the right execution in the internal execution of $P^*_{\tilde{\mathsf{tag}}}$ (constructed in EXT). By the POK property of $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}} \rangle$ it holds that the expected running-time of the knowledge extractor is bounded by

$$\frac{\text{poly}(n)}{p'}.$$

Since the probability of invoking the extraction procedure is p, the expected number of steps used to extract a witness is

$$p \frac{\text{poly}(n)}{p'}.$$

Now, in both SIM and EXT the left view is generated by S, and the right view is uniformly chosen. This in particular means that $p = p'$. It follows that the expected number of steps used to extract a witness is

$$p \frac{\text{poly}(n)}{p} = \text{poly}(n)$$

This completes the proof of Lemma 5.11 . ■

This completes the proof of "many-to-one" simulation extractability (Proposition 5.3). ■

5.3 "Full-Fledged" Simulation-Extractability

Let $\text{TAG} \in \{0,1\}^m$, let $x \in \{0,1\}^n$, and let A be the corresponding MIM adversary. We consider a left interaction in which $\langle P_{\text{TAG}}, V_{\text{TAG}} \rangle$ is executed with common input $x \in \{0,1\}^n$, and a right interaction in which $\langle P_{\tilde{\text{TAG}}}, V_{\tilde{\text{TAG}}} \rangle$ is executed with common input $\tilde{x} \in \{0,1\}^n$ The strings $\tilde{\mathsf{tag}}$ and $\tilde{x}$ are chosen adaptively by the man-in-the-middle A. The witness used by the prover in the left interaction is denoted by w, and the auxiliary input used by the adversary is denoted by z.

Proposition 5.14 *Let A be a MIM adversary as above, and suppose that $\ell(n) \geq 2n^2 + 2n$. Then, there exists a probabilistic expected polynomial time machine $\mathcal{S}$ such that the following conditions hold:*

[14] Note that this is not necessarily true in general. For example, when $\mathsf{tag}_j = \tilde{\mathsf{tag}}$ for some $j \in [n]$, and the messages that A sees are forwarded from an external source (e.g., when S is used by EXT in order to construct the stand alone prover $P^*_{\tilde{\mathsf{tag}}}$), we cannot guarantee anything about the running time of S. Indeed, the definition of simulation-extractability does not require EXT to output a witness when $\mathsf{tag}_j = \tilde{\mathsf{tag}}$ for some $j \in [n]$.

1. *The probability ensembles* $\{\mathcal{S}_1(x, z, \text{TAG})\}_{x,z,\text{TAG}}$ *and* $\{\mathsf{view}_A(x, z, \text{TAG})\}_{x,z,\text{TAG}}$ *are statistically close over L, where* $\mathcal{S}_1(x, z, \text{TAG})$ *denotes the first output of* $\mathcal{S}(x, z, \text{TAG})$*.*

2. *Let* $x \in L, z \in \{0,1\}^*, \text{TAG} \in \{0,1\}^{t(|x|)}$ *and let* (view, w) *denote the output of* $\mathcal{S}(x, z, \text{TAG})$ *(on input some random tape). Let* $\tilde{x}$ *be the right-execution statement appearing in* view *and let* $\tilde{\text{TAG}}$ *denote the right-execution tag. Then, if the right-execution in* view *is accepting AND* $\text{TAG} \neq \tilde{\text{TAG}}$*, then* $R_L(\tilde{x}, w) = 1$*.*

Proof: The construction of the simulator-extractor $\mathcal{S}$ proceeds in two phases and makes use of the many-to-one simulator extractor guaranteed by Lemma 5.3. In the first phase, the adversary A is used in order to construct a many-to-one adversary A' with the protocol $\langle P_{\text{TAG}}, V_{\text{TAG}} \rangle$ on its left and with one of the sub-protocols $\langle P_{\tilde{\mathsf{tag}}_i}, V_{\tilde{\mathsf{tag}}} \rangle$ on its right. In the second phase, the many-to-one simulation-extractability property of $\langle P_{\text{TAG}}, V_{\text{TAG}} \rangle$ is used in order to generate a view for A along with a witness for the statement $\tilde{x}$ appearing in the simulation.

The many-to-one adversary. On input (x, z, TAG), and given a man-in-the-middle adversary A, the many-to-one adversary A' acts as follows:

Internal messages: For all $j \in [n]$, pick random messages $(\tilde{h}^j, \tilde{r}_1^j, \tilde{r}_2^j, \tilde{u}^j)$ for the right interaction.

Right interaction The statement $\tilde{x}$ proved is the same as the one chosen by A. If there exists $i \in [n]$ so that $\mathsf{tag}_j \neq \tilde{\mathsf{tag}}_i$ for all $j \in [n]$, forward A's messages in the i^{th} right interaction to an external $V_{\tilde{\mathsf{tag}}_i}$ and send back his answers to A. Use the messages $\{(\tilde{h}^j, \tilde{r}_1^j, \tilde{r}_2^j, \tilde{u}^j)\}_{j \neq i}$ to internally emulate all other right interactions $\{\langle P_{\tilde{\mathsf{tag}}_j}, V_{\tilde{\mathsf{tag}}_j} \rangle\}_{j \neq i}$.

Otherwise, (i.e. if for all $i \in [n]$ there exists $j \in [n]$ such that $\mathsf{tag}_j = \tilde{\mathsf{tag}}_i$), pick an arbitrary $i \in [n]$, forward A's messages in the i^{th} right interaction to an external $V_{\tilde{\mathsf{tag}}_i}$ and send back his answers to A. Use the messages $\{(\tilde{h}^j, \tilde{r}_1^j, \tilde{r}_2^j, \tilde{u}^j)\}_{j \neq i}$ to internally emulate all other right interactions $\{\langle P_{\tilde{\mathsf{tag}}_j}, V_{\tilde{\mathsf{tag}}_j} \rangle\}_{j \neq i}$.

Left interaction: As induced by the scheduling of messages by A, forward the messages sent by A in the left interaction to an external prover P_{TAG}, and send back his answers to A.

The many-to-one adversary A' is depicted in Fig. 13. Messages from circled sub-protocols are the ones who get forwarded externally.

The simulator-extractor. By Lemma 5.3 there exists a simulator $\mathcal{S}'$ that produces a view that is statistically close to the real view of A', and outputs a witness provided that the right interaction is accepting and $\tilde{\mathsf{tag}}$ is different from all the left side tags $\mathsf{tag}_1, \ldots, \mathsf{tag}_n$.

The simulator-extractor $\mathcal{S}(x, z, \text{TAG})$ for A invokes $\mathcal{S}'(x, z, \text{TAG})$. If $\mathcal{S}'$ outputs `fail` so does $\mathcal{S}$. Otherwise, let view', w' denote the output of $\mathcal{S}'$. $\mathcal{S}$ now outputs view, w', where view is the view view' (output by $\mathcal{S}'$) augmented with the right hand side messages $\{(\tilde{h}^j, \tilde{r}_1^j, \tilde{r}_2^j, \tilde{u}^j)\}_{j \neq i}$ that were used in the internal emulation of A'.

Claim 5.15 *$\mathcal{S}$ runs in expected polynomial time.*

Proof: Notice that whenever A is polynomial time then so is A'. By Lemma 5.11, this implies that $\mathcal{S}'$ runs in expected polynomial time, and hence so does $\mathcal{S}$. ■

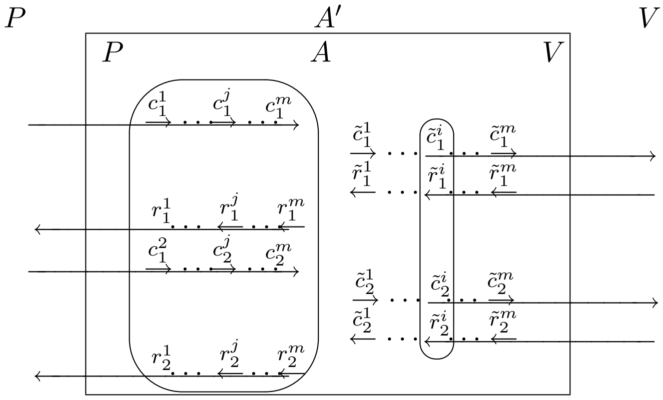

Figure 13: The "many-to-one" MIM adversary A'.

Claim 5.16 *$\{\mathcal{S}_1(x,z,\text{TAG})\}_{x\in L, z\in\{0,1\}^*, \text{TAG}\in\{0,1\}^m}$ and $\{\mathsf{view}_A(x,z,\text{TAG})\}_{x\in L, z\in\{0,1\}^*, \text{TAG}\in\{0,1\}^m}$ are statistically close over L, where $\mathcal{S}_1(x,z,\text{TAG})$ denotes the first output of $\mathcal{S}(x,z,\text{TAG})$.*

Proof: Given a distinguisher D between $\{\mathcal{S}_1(x,z,\text{TAG})\}_{x,z,\text{TAG}}$ and $\{\mathsf{view}_A(x,z,\text{TAG})\}_{z,x,\text{TAG}}$, we construct a distinguisher D' between $\{\mathcal{S}'_1(x,z,\text{TAG})\}_{x,z,\text{TAG}}$ and $\{\mathsf{view}_{A'}(x,z,\text{TAG})\}_{x,z,\text{TAG}}$. This will be in contradiction to Lemma 5.6. The distinguisher D' has the messages $\{(\tilde{h}^j, \tilde{r}_1^j, \tilde{r}_2^j, \tilde{u}^j)\}_{j\neq i}$ hardwired.[15] Given a joint view $\langle(\sigma'_1,\sigma'_2), M'\rangle$ of a left $\langle P_{\widetilde{\text{TAG}}}, V_{\widetilde{\text{TAG}}}\rangle$ interaction and a $\langle P_{\tilde{\mathsf{tag}}_i}, V_{\tilde{\mathsf{tag}}_i}\rangle$ right interaction, D' augments the view with the right interaction messages $\{(\tilde{h}^j, \tilde{r}_1^j, \tilde{r}_2^j, \tilde{u}^j)\}_{j\neq i}$. The distinguisher D' feeds the augmented view to D and outputs whatever D outputs.

Notice that if $\langle(\sigma'_1,\sigma'_2), M'\rangle$ is drawn according to $\{\mathcal{S}'_1(x,z,\text{TAG})\}_{x,z,\text{TAG}}$ then the augmented view is distributed according to $\{\mathsf{SIM}(x,z,\text{TAG})\}_{x,z,\text{TAG}}$. On the other hand, if $\langle(\sigma'_1,\sigma'_2), M'\rangle$ is drawn according to $\{\mathsf{view}_{A'}(x,z,\text{TAG})\}_{z,x,\text{TAG}}$ then the augmented view is distributed according to $\{\mathsf{view}_A(x,z,\text{TAG})\}_{z,x,\text{TAG}}$. Thus D' has exactly the same advantage as D. ■

Claim 5.17 *Let $x\in L$, $z\in\{0,1\}^*$, $\text{TAG}\in\{0,1\}^m$ and let (view, w) denote the output of $\mathcal{S}(x,z,\text{TAG})$ (on input some random tape). Let $\tilde{x}$ be the right-execution statement appearing in view and let $\widetilde{\text{TAG}}$ denote the right-execution tag. Then, if the right-execution in view is accepting AND $\text{TAG}\neq\widetilde{\text{TAG}}$, then $R_L(\tilde{x}, w) = 1$.*

Proof: Recall that the right interaction in the view output of $\mathcal{S}$ is accepting if and only if the right interaction in the view output of $\mathcal{S}$ is accepting. In addition, the statement proved in the right interaction output by $\mathcal{S}$ is identical to the one proved in the right interaction of $\mathcal{S}'$.

Observe that if $\text{TAG}\neq\widetilde{\text{TAG}}$, there must exist $i\in[n]$ for which $(i,\widetilde{\text{TAG}}_i)\neq(j,\text{TAG}_j)$ for all $j\in[n]$ (just take the i for which $\widetilde{\text{TAG}}_i\neq\text{TAG}_i$). Recall that by construction of the protocol $\langle P_{\widetilde{\text{TAG}}}, V_{\widetilde{\text{TAG}}}\rangle$, for every $i\in[n]$, the value tag_i is defined as (i,TAG_i) Thus, whenever $\text{TAG}\neq\widetilde{\text{TAG}}$ there exists a $\tilde{\mathsf{tag}}_i = (i,\widetilde{\text{TAG}}_i)$ that is different than $\mathsf{tag}_j = (j,\text{TAG}_j)$ for all $j\in[n]$. In particular, the tag used by A' in the right interaction will satisfy $\mathsf{tag}_j\neq\tilde{\mathsf{tag}}_i$ for all $j\in[n]$. By Lemma 5.10, we then have that if the right interaction in view output by $\mathcal{S}$ is accepting (and hence also in $\mathcal{S}'$), $\mathcal{S}'$ will output a witness for $\tilde{x}$. The proof is complete, since $\mathcal{S}$ outputs whatever witness $\mathcal{S}'$ outputs. ■

This completes the proof of Proposition 5.14. ■

[15] One could think of D as a family of distinguishers that is indexed by $\{(\tilde{h}^j, \tilde{r}_1^j, \tilde{r}_2^j, \tilde{u}^j)\}_{j\neq i}$, and from which a member is drawn at random.

6 Non-malleable Commitments

In this section we present two simple constructions of non-malleable commitments. The approach we follow is different than the approach used in [16]. Instead of viewing non-malleable commitments as a tool for constructing non-malleable zero-knowledge protocols, we reverse the roles and use a non-malleable zero-knowledge protocol (in particular any simulation-extractable protocol will do) in order to construct a non-malleable commitment scheme. Our approach is also different from the approach taken by [2].

6.1 A statistically-binding scheme (NM with respect to commitment)

We start by presenting a construction of a statistically binding scheme which is non-malleable with respect to commitment. Our construction relies on the following two building blocks:

- a family of (possibly malleable) non-interactive statistically binding commitment schemes,
- a simulation-extractable zero-knowledge argument.

The construction is conceptually very simple: The committer commits to a string using the statistically binding commitment scheme, and then proves knowledge of the string committed to using a simulation-extractable argument.

We remark that the general idea behind this protocol is not new. The idea of enhancing a commitment scheme with a proof of knowledge protocols was explored already in [16]. However, as pointed out in [16], this approach cannot work with *any* proof of knowledge protocol, as this proof of knowledge protocol itself might be malleable. (As mentioned above, Dolev et al therefore rely on a quite different approach to construct non-malleable commitments [16]). What we show here, is that this approach in fact works if the proof of knowledge protocol is simulation-extractable.

Let $\{\mathrm{Com}_r\}_{r\in\{0,1\}^*}$ be a family of *non-interactive* statistically binding commitment schemes (e.g., Naor's commitment [34]) and let $\langle P_{\mathrm{TAG}}, V_{\mathrm{TAG}}\rangle$ be a simulation extractable protocol. Consider the protocol in Figure 14.[16]

We start by sketching why the scheme is non-malleable. Note that the commit phase of the scheme only consists of a message specifying an NP-statement (i.e., the "statement" $c = \mathrm{Com}_r(v; s)$), and an accompanying "proof" of this statement. Thus, intuitively, an adversary that is able to successfully maul a commitment, must be able to maul both the commitment Com and also the accompanying proof $\langle P_{\mathrm{TAG}}, V_{\mathrm{TAG}}\rangle$. The former might be easy as Com might be malleable. However, as $\langle P_{\mathrm{TAG}}, V_{\mathrm{TAG}}\rangle$ is simulation-extractable (and thus non-malleable) the latter will be impossible.

At first sight, it seems like it would be sufficient to simply assume that $\langle P_{\mathrm{TAG}}, V_{\mathrm{TAG}}\rangle$ is non-malleable to conclude non-malleability of $\langle C, R\rangle$. Note, however, that the definition of a non-malleable proofs only considers a setting where the statement proven by the man-in-the-middle adversary is fixed ahead of the interaction. In our scenario, we instead require security also w.r.t an *adaptively* chosen statement (as the statement proven is related to the commitment chosen by the adversary). This gap is addressed in the definition of simulation-extractability; here the man-in-the-middle adversary is also allowed to adaptively chose the statements it will attempt to prove.

[16] It is interesting to note that the protocol $\langle C, R\rangle$ is statistically-binding even though the protocol $\langle P_{\mathrm{TAG}}, V_{\mathrm{TAG}}\rangle$ is "only" computationally sound. At first sight, this is somewhat counter intuitive since the statistical binding property is typically associated with all-powerful committers.

Protocol $\langle C, R\rangle$

Security Parameter: 1^n.

String to be committed to: $v \in \{0,1\}^n$.

Commit Phase:

$R \to C$: Pick uniformly $r \in \{0,1\}^n$.

$C \to R$: Pick $s \in \{0,1\}^n$ and send $c = \mathrm{Com}_r(v; s)$.

$C \leftrightarrow R$: Let TAG $= (r, c)$. Prove using $\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle$ that there exist $v, s \in \{0,1\}^n$ so that $c = \mathrm{Com}_r(v; s)$. Formally, prove the statement (r, c) with respect to the witness relation

$$R_L = \{(r, c), (v, s) | c = \mathrm{Com}_r(v; s)\}$$

R: Verify that $\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle$ is accepting.

Reveal Phase:

$C \to R$: Send v, s.

R: Verify that $c = \mathrm{Com}_r(v; s)$.

Figure 14: A statistically binding non-malleable string commitment protocol - $\langle C, R\rangle$

Interestingly, to formalize the above argument, we will be required to rely on the *statistical indistinguishability* property of the definition of simulation-extractability. Intuitively, the reason for this is the following. For any given man-in-the-middle adversary we are required to construct a stand-alone adversary that succeeds in committing to "indistinguishable" values. In proving that this stand-alone adversary succeeds in this task, we will rely on the simulator-extractor for the man-in-the-middle adversary; however, to do this, we need to make sure that the simulator-extractor indeed will commit to indistinguishable values. Note that it is not sufficient that the simulator-extractor simply proves a statement that is indistinguishable from the statement proved by the man-in-the-middle adversary, as the value committed to is not efficiently computable from the statement. However, the value committed to is computable (although not efficiently) from the statement. Thus, by relying on the statistical indistinguishability property of the simulator-extractor, we can make sure that also the value committed by the simulator-extractor is indistinguishable from that committed to by the man-in-the-middle adversary.[17]

Theorem 6.1 (*nm𝒞* with respect to commitment) *Suppose that* $\{\mathrm{Com}_r\}_{r\in\{0,1\}^*}$ *is a family of* non-interactive *statistically binding commitment schemes, and that* $\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle$ *is simulation extractable. Then,* $\langle C, R\rangle$ *is a statistically binding non-malleable commitment scheme with respect to commitment.*

Proof: We need to prove that the scheme satisfies the following three properties: statistical binding, computational hiding and non-malleability with respect to commitment.

Statistical Binding: The binding property of the scheme directly follows from the binding property of the underlying commitment scheme Com: to break the binding property of $nm\mathcal{C}_{\text{TAG}}$

[17] Note that in the case of non-malleability with respect to opening, this complication does not arise.

requires first breaking the binding of Com. More formally, given any adversary A that breaks the binding property of $nm\mathcal{C}_{\text{TAG}}$, we construct an adversary A', which on input a message r, feeds r to A, and then internally honestly emulates all the verification messages in the proof part of $nm\mathcal{C}_{\text{TAG}}$. It directly follows that the success probability of A' is the same as that of A.

Computational Hiding: The hiding property follows from the hiding property of Com combined with the $\mathcal{ZK}$ property of $\langle P_{\text{TAG}}, V_{\text{TAG}} \rangle$. More formally, recall that the notion of simulation-extractability implies $\mathcal{ZK}$ (see Proposition 3.6) and that $\mathcal{ZK}$ implies *strong*-witness indistinguishability[18] [20]. Since the scheme Com produces indistinguishable commitments, it thus directly follows by the definition of strong witness indistinguishability that the protocol $\langle C, R \rangle$ also produces indistinguishable commitments.

Non-malleability: Consider a man-in-the middle adversary A. We assume without loss of generality that A is deterministic (this is w.l.o.g since A can obtain its "best" random tape as auxiliary input). We show the existence of a probabilistic polynomial-time stand-alone adversary S and a negligible function $\nu : N \to N$, such that for every irreflexive polynomial-time computable relation $\mathcal{R} \subseteq \{0,1\}^n \times \{0,1\}^n$, every $v \in \{0,1\}^n$, and every $z \in \{0,1\}^*$, it holds that:

$$\Pr\Big[\mathsf{mim}^A_{\mathsf{com}}(\mathcal{R}, v, z) = 1\Big] \;<\; \Pr\Big[\mathsf{sta}^S_{\mathsf{com}}(\mathcal{R}, v, z) = 1\Big] \;+\; \nu(n) \tag{4}$$

Description of the stand-alone adversary. The stand-alone adversary S uses A as a black-box and emulates the left and right interactions for A as follows: the left interaction is emulated internally, while the right interaction is forwarded externally. More precisely, S proceeds as follows on input z. S incorporates $A(z)$ and internally emulates the left interactions for A by simply *honestly* committing to the string 0^n; i.e., to emulate the left interaction, S executes the algorithm C on input 0^n. Messages from the right interactions are instead forwarded externally. Note that S is thus a stand-alone adversary that expects to act as a committer for the scheme $\langle C, R \rangle$.

Analysis of the stand-alone adversary. We proceed to show that equation 4 holds. Suppose, for contradiction, that this is not the case. That is, there exists an irreflexive polynomial-time relation $\mathcal{R}$ and a polynomial $p(n)$ such that for infinitely many n, there exists strings $v \in \{0,1\}^n$, $z \in \{0,1\}^*$ such that

$$\Pr\left[\mathsf{mim}^A_{\mathsf{com}}(\mathcal{R}, v, z) = 1\right] - \Pr\left[\mathsf{sta}^S_{\mathsf{com}}(\mathcal{R}, v, z) = 1\right] \geq \frac{1}{p(n)}$$

Fix generic n, v, z for which the above holds. We show how this contradicts the simulation-extractability property of $\langle P_{\text{TAG}}, V_{\text{TAG}} \rangle$. On a high-level, our proof consists of the following steps:

1. We first note that since the commit phase of $\langle C, R \rangle$ "essentially" only consists of a statement (r, c) (i.e., the commitment) and a proof of the "validity" of (r, c), A can be interpreted as a simulation-extractability man-in-the-middle adversary A' for $\langle P_{\text{TAG}}, V_{\text{TAG}} \rangle$.

2. It follows from the simulation-extractability property of $\langle P_{\text{TAG}}, V_{\text{TAG}} \rangle$ that there exist a combined simulator-extractor $\mathcal{S}$ for A' that outputs a view that is statistically close to that of A',

[18] Intuitively, the notion of strong-witness indistinguishability requires that proofs of indistinguishable statements are indistinguishable. This is, in fact, exactly the property we need in order to prove that the commitment scheme is computationally hiding.

while at the same time outputting a witness to all accepting right proofs which use a different right tag $\tilde{\text{TAG}}$ than the tag TAG of the left interaction, i.e., if $\tilde{\text{TAG}} \neq \text{TAG}$.

3. Since the view output by the simulator-extractor $\mathcal{S}$ is *statistically* close to the view of A' in the real interaction, it follows that also the *value* committed to in that view is statistically close to value committed to by A'. (Note that computational indistinguishability would not have been enough to argue the indistinguishability of these values, since they are not efficiently computable from the view.)

4. It also follows that the simulator-extractor $\mathcal{S}$ will output also the witness to each *accepting* right executions such that $\tilde{\text{TAG}} \neq \text{TAG}$. We conclude that $\mathcal{S}$ additionally outputs the value *committed to* in the right execution (except possibly when the value committed to in the right interaction is the same as that committed to in the left).

5. We finally note that if $\mathcal{R}$ (which is irreflexive) "distinguishes" between the value committed to by A and by S, then $\mathcal{R}$ also "distinguishes" the second output of $\mathcal{S}$ (which consists of the committed values) when run on input a commitment (using Com) to v, and the second output of $\mathcal{S}$ when run on input a commitment to 0. But, this contradicts the hiding property of Com.

We proceed to a formal proof. One particular complication that arises with the above proof sketch is that in the construction of $\langle C, R\rangle$ we are relying on a family of commitment schemes $\{\text{Com}_r\}_{r\in\{0,1\}^*}$ and not a single non-interactive commitment scheme. Thus, strictly speaking, A is not a man-in-the-middle adversary for the interactive proof $\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle$. However, the only difference is that A additionally expects to receive a message $\tilde{r}$ on the right, and also to send a message r on the left. To get around this problem we rely on the *non-uniform* computational hiding property of $\langle C, R\rangle$.

Note that since in both experiments mim and sta the right execution is identically generated, there must exists some *fixed* message $\tilde{r}$ such that conditioned on the event that the first message sent in the right execution is $\tilde{r}$, it holds that the success probability in $\mathsf{mim}^A_{\mathsf{com}}(\mathcal{R}, v, z)$ is $\frac{1}{p(n)}$ higher than in $\mathsf{sta}^S_{\mathsf{com}}(\mathcal{R}, v, z)$. In fact, by the statistical binding property of Com, it follows that there must exists some message $\tilde{r}$ such $\text{Com}_{\tilde{r}}$ is *perfectly binding* and additionally, conditioned on the event that the first message sent in the right execution is $\tilde{r}$, it holds that the success probability in $\mathsf{mim}^A_{\mathsf{com}}(\mathcal{R}, v, z)$ is $\frac{1}{2p(n)}$ higher than in $\mathsf{sta}^S_{\mathsf{com}}(\mathcal{R}, v, z)$

Given this message $\tilde{r}$, we must now consider two cases:

1. Either A (which by assumption is deterministic) sends its first message r in left interaction directly after receiving $\tilde{r}$ (see Figure 15),

2. or A first send its first message $\tilde{c}$ in the right interaction.

We first show that in the second case, A can be used to break the non-uniform hiding property of $\langle C, R\rangle$. Intuitively, this follows from the fact that the value committed to by A on the right is "essentially" determined by the message $\tilde{c}$ sent by A before it has received a single message on the left; it is not "fully" determined by $\tilde{c}$ as A can decide whether it fails in the interactive proof, and thus potentially change the value determined by $\tilde{c}$ to $\perp$. However, in this case, A can be used to violate the hiding property of $\langle C, R\rangle$ (as the probability of failing the proof must depend on the value it receives a commitment to).

More formally, let $v_{\tilde{c}}$ denote the value committed to in $\tilde{c}$ (using Com). It then holds that the value committed to by A in its right interaction (of $\langle C, R\rangle$) will be $v_{\tilde{c}}$ if A succeeds in the

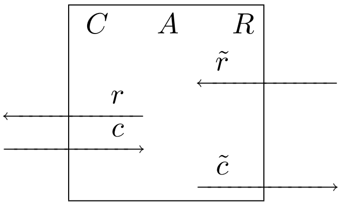

Figure 15: An interleaved scheduling of commitments.

proof following the message $\tilde{c}$ and $\perp$ otherwise. By our assumption that the success probability in $\mathsf{mim}^{A}_{\mathsf{com}}(\mathcal{R}, v, z)$ is $\frac{1}{2p(n)}$ higher than in $\mathsf{sta}^{S}_{\mathsf{com}}(\mathcal{R}, v, z)$, conditioned on the event that the first message sent in the right execution is $\tilde{r}$, it thus holds that A "aborts" the proof in the left interaction with different probability in experiments $\mathsf{mim}^{A}_{\mathsf{com}}(\mathcal{R}, v, z)$ and $\mathsf{sta}^{S}_{\mathsf{com}}(\mathcal{R}, v, z)$, conditioned on the first message in the right interaction being $\tilde{r}$. However, since the only difference in those experiments is that A receives a commitment to v in mim and a commitment to 0^n in sta, we conclude that A contradicts the (non-uniform) computational hiding property of Com. Formally, we construct a *non-uniform* distinguisher D for the commitment scheme $\langle C, R\rangle$: D incorporates A, z, r, v and $v_{\tilde{c}}$ and emulates the right execution for A by honestly running the verification procedure of $\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle$, and forwards messages in the left execution externally. D finally outputs $\mathcal{R}(v, v_{\tilde{c}})$ if the proof was accepting and 0 otherwise. The claim follows from the fact that D perfectly emulates $\mathsf{mim}^{A}_{\mathsf{com}}(\mathcal{R}, v, z)$ when receiving a commitment to v, and perfectly emulates $\mathsf{sta}^{S}_{\mathsf{com}}(\mathcal{R}, v, z)$ when receiving a commitment to 0^n.

We proceed to consider the first (and harder) case depicted in Figure 15—i.e., when A sends its first left message r directly after receiving the message $\tilde{r}$. In this case, we instead directly use A to contradict the hiding property of Com. Towards this goal, we proceed in three steps:

1. We first define a simulation-extractability adversary A'.

2. We next show that A' can be used to violate the non-malleability property of $\langle C, R\rangle$.

3. In the final step, we show how to use the simulator-extractor $\mathcal{S}$ for A' to violate the hiding property of Com.

Step 1: Defining a simulation-extractability adversary A'. We define a simulation-extractability adversary A' for $\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle$. On input $x, \text{TAG}, z' = (z, \tilde{r})$, A' internally incorporates $A(z)$ and emulates the left and right interactions for A as follows.

1. A' starts by feeding A the message $\tilde{r}$ as part of its right execution. All remaining messages in the right execution are forwarded externally.

2. All messages in A's left interaction are forwarded externally as part of A''s left interaction, except for the *first* message r.

Step 2: Show that A' violates non-malleability of $\langle C, R\rangle$. Towards the goal of showing that A violates non-malleability of $\langle C, R\rangle$, we define the hybrid experiment $\mathsf{hyb}_1(v')$: (relying on the definition of $v, z, r, \tilde{r}$)

1. Let s be a uniform random string and let $c = \text{Com}_r(v'; s)$.

2. Let $x = (r, c), \text{TAG} = (r, c), z' = (z, \tilde{r})$. Emulate an execution for $A'(x, \text{TAG}, z')$ by honestly providing a proof of x (using tag TAG and the witness (v', s)) as part of its left interaction, and honestly verifying the right interaction.

3. Given the view of A' in the above emulation, reconstruct the view *view* of A in the emulation by A'. If the commitment produced by A in the right-execution in *view* is valid, let $\tilde{v}$ denote the value committed to; recall that by our assumption on $\tilde{r}$, this value is uniquely defined (although it is not efficiently computable). If, on the other hand, the commitment produced by A is invalid, let $\tilde{v} = \perp$.

4. Finally, if $\tilde{v} = \perp$, output 0. Otherwise output $\mathcal{R}(v, \tilde{v})$.

Note that $\mathsf{hyb}_1(v)$ is not efficiently samplable, since the third step is not efficient. However, except for that step, every other operation in hyb_1 is efficient. (This will be useful to us at a later stage).

We have the following claim.

Claim 6.2

$$\Pr\left[\mathsf{hyb}_1(v) = 1\right] - \Pr\left[\mathsf{hyb}_1(0^n) = 1\right] \geq \frac{1}{2p(n)}$$

Proof: Note that by the construction of A' and hyb_1 it directly follows that:

1. The view of A in $\mathsf{hyb}_1(v)$ is identically distributed to the view of A in $\mathsf{mim}^A_{\mathsf{com}}(\mathcal{R}, v, z)$, conditioned on the event that the first message in the right execution is $\tilde{r}$.

2. The view of A in $\mathsf{hyb}_1(0^n)$ is identically distributed to the view of A in $\mathsf{sta}^A_{\mathsf{com}}(\mathcal{R}, v, z)$, conditioned on the event that the first message in the right execution is $\tilde{r}$.

Since the output of the experiments $\mathsf{hyb}, \mathsf{mim}, \mathsf{sta}$ is determined by applying the same fixed function (involving $\mathcal{R}$ and v) to the view of A in those experiments, the claim follows. ■

Step 3: Show that the simulator for A' violates the hiding property of Com. We next use the simulator-extractor S' for A' to construct an *efficiently computable* experiment that is statistically close to hyb_1.

Towards this goal, we first consider the following experiment hyb_2, which still is not efficient. $\mathsf{hyb}_2(v')$ proceeds just as $\mathsf{hyb}_1(v')$ except that instead of emulating the left and right interactions for A', hyb_2 runs the combined simulator extractor $\mathcal{S}$ for A' to generate the view of A'.

Claim 6.3 *There exists a negligible function $\nu'(n)$ such that for any string $v' \in \{0,1\}^n$,*

$$|\Pr\left[\mathsf{hyb}_1(v') = 1\right] - \Pr\left[\mathsf{hyb}_2(v') = 1\right]| \leq \nu'(n)$$

Proof: It follows directly from the statistical indistinguishability property of $\mathcal{S}$ that the view of A generated in hyb_1 is statistically close to the view of A generated in hyb_2. The claim is concluded by (again) observing that the success of both hyb_1 and hyb_2 is defined by applying the same (deterministic) function to the view of A. ■

Remark 6.4 Note that the proof of Claim 6.3 inherently relies on the *statistical* indistinguishability property of $\mathcal{S}$. Indeed, if the simulation had only been computationally indistinguishable, we would not have been able to argue indistinguishability of the outputs of hyb_1 and hyb_2. This follows from the fact that success in experiments hyb_1 and hyb_2 (which depends on the *actual* committed values in the view of A) is not efficiently computable from the view alone.

We next define the final experiment $\mathsf{hyb}_3(v')$ that proceeds just as $\mathsf{hyb}_2(v')$ with the following modification:

- Instead of letting $\tilde{v}$ be set to the actual value committed to in the view *view* of A, $\tilde{v}$ is computed as follows. Recall that the combined-simulator extractor $\mathcal{S}$ outputs both a view and a witness to each accepting right interaction. If the right execution in *view* is accepting[19] and if $\tilde{\text{TAG}} \neq \text{TAG}$ where $\tilde{\text{TAG}}$ is the tag used in the right interaction in *view*, simply set $\tilde{v}$ to be consistent with the witness output by $\mathcal{S}$ (i.e., if $\mathcal{S}$ outputs the witness (v', s'), let $\tilde{v} = v'$). Otherwise, (i.e., if $\tilde{\text{TAG}} = \text{TAG}$, or if the right execution was rejecting), let $\tilde{v} = \perp$.

Note that in contrast to hyb_2, hyb_3 is efficiently computable. Furthermore it holds that:

Claim 6.5 *For any string $v' \in \{0,1\}^n$,*

$$\Pr\left[\mathsf{hyb}_2(v') = 1\right] = \Pr\left[\mathsf{hyb}_3(v') = 1\right]$$

Proof: Recall that the view of A in hyb_2 and hyb_3 is identical; the only difference in the experiments is how the final output is computed. It holds by the definition of the simulator-extractor $\mathcal{S}$ that $\mathcal{S}$ *always* outputs the witness to the statement proved by A' if the right interaction is accepting and if $\tilde{\text{TAG}} \neq \text{TAG}$. Thus, whenever *view* contains an accepting right-execution proof such that $\tilde{\text{TAG}} \neq \text{TAG}$, it follows by our assumption that $\text{Com}_{\tilde{r}}$ is perfectly binding, that the output of hyb_2 and hyb_3 is identical. Furthermore, in case the right-execution proof is rejecting, it holds that $\tilde{v} = \perp$ in both hyb_2 and hyb_3, which again means the output in both experiments is identical. Finally, consider the case when the right-execution is accepting, but $\tilde{\text{TAG}} = \text{TAG}$. By definition, it holds that hyb_3 outputs 0. Now, recall that $\text{TAG} = (r, c)$ and $\tilde{\text{TAG}} = (\tilde{r}, \tilde{c})$; in other words, if $\tilde{\text{TAG}} = \text{TAG}$, it means that A fully copied the initial commitment using Com_r. Since $\mathcal{R}$ is irreflexive and $\text{Com}_{\tilde{r}}$ is perfectly binding, it follows that also hyb_2 outputs 0. We conclude that the outputs of hyb_2 and hyb_3 are identically distributed. ■

By combining the above claims we obtain that there exists some polynomial $p'(n)$ such that

$$\Pr\left[\mathsf{hyb}_3(v) = 1\right] - \Pr\left[\mathsf{hyb}_3(0^n) = 1\right] \geq \frac{1}{p'(n)}$$

However, since hyb_3 is efficiently samplable, we conclude that this contradicts the (non-uniform) hiding property of Com_r.

More formally, define an additional hybrid experiment $\mathsf{hyb}_4(c')$ that proceeds as follows on input a commitment c' using Com_r: hyb_4 performs the same operations as hyb_3, except that instead of generating the commitment c, it simply sets $c = c'$. It directly follows from the construction of hyb_4 that $\mathsf{hyb}_4(c')$ is identically distributed to $\mathsf{hyb}_3(0^n)$ when c' is a (random) commitment to 0^n (using Com_r), and is identically distributed to $\mathsf{hyb}_3(v)$ when c' is a commitment to v. We conclude that hyb_4 distinguishes commitments (using Com_r) to 0^n and v. ■

Remark 6.6 (Black-box v.s. Non Black-box Simulation) *Note that the stand-alone committer S constructed in the above proof only uses black-box access to the adversary A, even if the simulation-extractability property of $\langle P_{\text{TAG}}, V_{\text{TAG}} \rangle$ has been proven using a non black-box simulation. Thus, in essence, the simulation of our non-malleable commitment is* always *black-box. However, the analysis showing the correctness of the simulator relies on non black-box techniques, whenever the simulator-extractor for $\langle P_{\text{TAG}}, V_{\text{TAG}} \rangle$ is proven using non black-box techniques.*

[19] Note that since *view* is a *joint* view of A and the honest receiver R in the right-execution, one can efficiently determine whether R indeed accepted the commitment.

Since families of non-interactive statistically binding commitments schemes can be based on collision-resistant hash functions (in fact one-way functions are enough [34, 28]) we get the following corollary:

Corollary 6.7 (Statistical binding non-malleable commitment) *Suppose that there exists a family of collision resistant hash functions. Then, there exists a constant-round statistically-binding commitment scheme that is non malleable with respect to commitment.*

6.2 A statistically-hiding scheme (NM with respect to opening)

We proceed to the construction of a statistically-hiding commitment scheme $\langle \mathbf{C}, \mathbf{R} \rangle$ which is non-malleable with respect to opening. Our construction relies on a quite straight-forward combination of a (family) of non-interactive statistically-hiding commitments and a simulation-extractable argument.[20] Let $\{\mathbf{Com}_r\}_{r \in \{0,1\}^*}$ be a family of *non-interactive* statistically hiding commitment schemes (e.g., [12]) and let $\langle P_{\text{TAG}}, V_{\text{TAG}} \rangle$ be simulation extractable protocol. The protocol is depicted in Figure 16.

Protocol $\langle \mathbf{C}, \mathbf{R} \rangle$

Security Parameter: 1^n.

String to be committed to: $v \in \{0,1\}^n$.

Commit Phase:

$R \rightarrow C$: Pick uniformly $r \in \{0,1\}^n$.

$C \rightarrow R$: Pick $s \in \{0,1\}^n$ and send $c = \mathbf{Com}_r(v; s)$.

Reveal Phase:

$C \rightarrow R$: Send v.

$C \leftrightarrow R$: Let $\text{TAG} = (r, c, v)$. Prove using $\langle P_{\text{TAG}}, V_{\text{TAG}} \rangle$ that there exist $s \in \{0,1\}^n$ so that $c = \mathbf{Com}_r(v; s)$. Formally, prove the statement (r, c, v) with respect to the witness relation

$$R_L = \{(r, c, v), s | c = \mathbf{Com}_r(v; s)\})$$

R: Verify that $\langle P_{\text{TAG}}, V_{\text{TAG}} \rangle$ is accepting.

Figure 16: A statistically hiding non-malleable string commitment protocol - $\langle \mathbf{C}, \mathbf{R} \rangle$.

Theorem 6.8 (*nmC* with respect to opening) *Suppose that $\{\mathbf{Com}_r\}_{r \in \{0,1\}^*}$ is a family of* non-interactive *commitment schemes, and that $\langle P_{\text{TAG}}, V_{\text{TAG}} \rangle$ a simulation extractable argument with an efficient prover strategy. Then, $\langle \mathbf{C}, \mathbf{R} \rangle$ is a non-malleable commitment scheme with respect to opening. If furthermore $\{\mathbf{Com}_r\}_{r \in \{0,1\}^*}$ is statistically hiding, then $\langle \mathbf{C}, \mathbf{R} \rangle$ is so as well.*

Proof: We need to prove that the scheme satisfies the following three properties: computational binding, (statistical) hiding and non-malleability with respect to opening.

[20] Note that whereas our construction of statistically binding commitments required that the simulation-extractable argument provides a simulation that is statistically close, we here are content with a computationally indistinguishable simulation.

We start by proving the hiding and non-malleability properties, and then return to the proof of the binding property.

(Statistical) Hiding: The hiding property directly follows from the hiding property of **Com**. Note that if **Com** is statistically hiding then $\langle \mathbf{C}, \mathbf{R} \rangle$ is also statistically hiding.

Non-malleability: We show that for every probabilistic polynomial-time man-in-the-middle adversary A, there exists a probabilistic *expected* polynomial-time stand-alone adversary S and a negligible function $\nu : N \to N$, such that for every irreflexive polynomial-time computable relation $\mathcal{R} \subseteq \{0,1\}^n \times \{0,1\}^n$, every $v \in \{0,1\}^n$, and every $z \in \{0,1\}^*$, it holds that:

$$\Pr\Big[\mathsf{mim}^{A}_{\mathsf{open}}(\mathcal{R}, v, z) = 1\Big] \ < \ \Pr\Big[\mathsf{sta}^{S}_{\mathsf{open}}(\mathcal{R}, v, z) = 1\Big] \ + \ \nu(n)$$

We remark that the stand-alone adversary S constructed here will be conceptually quite different from the one constructed in the proof of Theorem 6.1.

Description of the stand-alone adversary. We proceed to describe the stand-alone adversary S. On a high-level, S internally incorporates A and emulates the commit phase of the left execution for adversary A by honestly committing to 0^n, while externally forwarding messages in the right execution. Once A has completed the commit phase, S interprets the residual adversary (after the completed commit phase) as a man-in-the middle adversary A' for $\langle P_{\text{TAG}}, V_{\text{TAG}} \rangle$. It then executes the simulator-extractor $\mathcal{S}$ for A' to obtain a witness to the statement proved in the right execution by A' (and thus A). Using this witness S can then complete the decommit phase of the external execution. (We here rely on the fact that $\langle P_{\text{TAG}}, V_{\text{TAG}} \rangle$ has an efficient prover strategy.).

More formally, the stand-alone adversary S proceeds as follows on input z.

1. S internally incorporates $A(z)$.
2. During the commit phase S proceeds as follows:
   (a) S internally emulates the left interaction for A by honestly committing to 0^n.
   (b) Messages from right execution are forwarded externally.
3. Once the commit phase has finished S receives the value v. Let $(r, c), (\tilde{r}, \tilde{c})$ denote the left and right-execution transcripts of A (recall that the left execution has been internally emulated, while the right execution has been externally forwarded).
4. Construct a man-in-the-middle adversary A' for $\langle P_{\text{TAG}}, V_{\text{TAG}} \rangle$. Informally, A' will simply consist of the residual machine resulting after the above-executed commit phase. More formally, $A'(x, \text{TAG}, z')$ proceeds as follows:
   (a) Parse z' as $(\tilde{r}, \tilde{c}, z)$.
   (b) Parse x as (r, c, v).
   (c) Internally emulate the commit phase $(r, c), (\tilde{r}, \tilde{c})$ for $A(z)$ (i.e., feed A the message $\tilde{r}$ as part of its right execution, and c as part of its left execution).
   (d) Once the commit phase has finished, feed v to A.
   (e) Externally forward all the remaining messages during the reveal phase.

5. Let $\mathcal{S}$ denote the simulator-extractor for A'.

6. Let $x = (r, c, v)$, $\text{TAG} = (r, c, v)$ and $z' = (\tilde{r}, \tilde{c}, z)$

7. Run $\mathcal{S}$ on input (x, TAG, z') to obtain the view *view* and the witness $\tilde{w}$.

8. Finally, if the statement proved in the right-execution of *view* is $\tilde{x} = (\tilde{r}, \tilde{c}, \tilde{v})$ (where $\tilde{v}$ is an arbitrary string), the right-execution proof is accepting and uses a tag $\tilde{\text{TAG}}$ such that $\tilde{\text{TAG}} \neq \text{TAG}$, and $\tilde{w}$ contains a valid witness for $\tilde{x}$, run the honest prover strategy P_{TAG} on input $\tilde{x}$ and the witness $\tilde{w}$. (Otherwise, simply abort.)

Analysis of the stand-alone adversary. Towards the goal of showing equation 5, we define a hybrid stand-alone adversary $\hat{S}$ that also receives v as auxiliary input. $\hat{S}$ proceeds exactly as S, but instead of feeding A a commitment to 0^n in the commit phase, $\hat{S}$ instead feeds A a commitment to v.

Since both the experiment $\mathsf{sta}_{\mathsf{open}}$ and $\hat{S}$ are efficiently computable, the following claim directly follows from the hiding property of **Com**.

Claim 6.9 *There exists some negligible function ν' such that*

$$\left| \Pr\left[\mathsf{sta}^{S}_{\mathsf{open}}(\mathcal{R}, v, z) = 1\right] - \Pr\left[\mathsf{sta}^{\hat{S}}_{\mathsf{open}}(\mathcal{R}, v, z) = 1\right]\right| \leq \nu'(k)$$

We proceed to show the following claim, which together with Claim 6.9 concludes Equation 5.

Claim 6.10 *There exist some negligible function ν'' such that*

$$\left| \Pr\left[\mathsf{mim}^{A}_{\mathsf{open}}(\mathcal{R}, v, z) = 1\right] - \left[\mathsf{sta}^{\hat{S}}_{\mathsf{open}}(\mathcal{R}, v, z) = 1\right]\right| \leq \nu''(k)$$

Proof: Towards the goal of showing this claim we introduce an additional hybrid experiment $\mathsf{hyb}(\mathcal{R}, v, z)$ which proceeds as follows: Emulate $\mathsf{sta}^{\hat{S}}_{\mathsf{open}}(\mathcal{R}, v, z)$ but instead of defining $\tilde{v}$ as the value (successfully) decommitted to by S, define $\tilde{v}$ as the value (successfully) decommitted to in the view *view* output by simulator-extractor $\mathcal{S}$ (in the execution by $\hat{S}$). We start by noting that it follows directly from the indistinguishability property of the simulator-extractor $\mathcal{S}$ that the following quantity is negligible.[21]

$$\left| \Pr\left[\mathsf{mim}^{A}_{\mathsf{open}}(\mathcal{R}, v, z) = 1\right] - \Pr\left[\mathsf{hyb}(\mathcal{R}, v, z) = 1\right]\right|$$

To conclude the claim, we show that

$$\Pr\left[\mathsf{hyb}(\mathcal{R}, v, z) = 1\right] = \left[\mathsf{sta}^{\hat{S}}_{\mathsf{open}}(\mathcal{R}, v, z) = 1\right]$$

Note that the only difference between experiments $\mathsf{hyb}(\mathcal{R}, v, z)$ and $\mathsf{sta}^{\hat{S}}_{\mathsf{open}}(\mathcal{R}, v, z)$ is that in hyb, the value $\tilde{v}$ is taken from the view output by $\mathcal{S}$, whereas in $\mathsf{sta}^{\hat{S}}_{\mathsf{open}}$ it is defined as the value successfully decommitted to by $\hat{S}$. Also recall that $\hat{S}$ "attempts" to decommit to the value $\tilde{v}$ successfully decommitted to in the output by $\mathcal{S}$; $\hat{S}$ is successful in this task whenever $\mathcal{S}$ also is able to extract a witness to the right-execution proof. Note that by the simulation-extractability property of $\langle P_{\text{TAG}}, V_{\text{TAG}} \rangle$ it follows that $\mathcal{S}$ *always* outputs a valid witness if the right-execution in

[21] We remark that it is here sufficient that the simulator-extractor outputs a view that is merely computationally indistinguishable from the view in a "real" execution.

view is accepting, as long as the tag $\tilde{\text{TAG}}$ of the right execution is different from TAG. Thus, in case $\tilde{\text{TAG}} \neq \text{TAG}$, we conclude by the perfect completeness of $\langle P_{\text{TAG}}, V_{\text{TAG}} \rangle$ that the output of experiments sta and hyb are defined in exactly the same way. In case $\tilde{\text{TAG}} = \text{TAG}$, sta will output 0 (as $\hat{S}$ will not even attempt to decommit). However, in this case, it holds that that $\tilde{v} = v$ (since $\text{TAG} = (r, c, v)$ and $\tilde{\text{TAG}} = (\tilde{r}, \tilde{c}, \tilde{v})$); this means that hyb will also output 0 (since $\mathcal{R}$ is irreflexive). The claim follows. ■

We now return to the binding property.

Computational Binding: The binding properties of the scheme intuitively follows from the binding property of the underlying commitment scheme **Com** and the "proof-of-knowledge" property implicitly guaranteed by the simulation-extractability property of $\langle P_{\text{TAG}}, V_{\text{TAG}} \rangle$. A formal proof proceeds along the lines of the proof of non-malleability (but is simpler.) More precisely, assume for contradiction that there exists some adversary A that is able to violate the binding property of $\langle \mathbf{C}, \mathbf{R} \rangle$. We show how to construct a machine $\hat{A}$ that violates the binding property of **Com**. $\hat{A}$ starts by running A, letting it complete the commit phase by externally forwarding its messages. Once A has completed the commit phase, $\hat{A}$ interprets the residual adversary (after the completed commit phase) as a man-in-the middle adversary A' (that ignores all left-execution messages) for $\langle P_{\text{TAG}}, V_{\text{TAG}} \rangle$; a formal description of this adversary is essentially identical to the one described in the proof of non-malleability. $\hat{A}$ then executes the simulator-extractor $\mathcal{S}$ for A' to obtain a witness to the statement proved in the right execution by A' (and thus A). Using this witness $\hat{A}$ can then complete the decommit phase of the external execution of **Com**. It directly follows by the indistinguishability property of the simulator-extractor $\mathcal{S}$ that $\hat{A}$ violates the binding property of **Com** with essentially the same probability as A violates the binding property of $\langle \mathbf{C}, \mathbf{R} \rangle$.

This completes the proof of Theorem 6.8. ■

Remark 6.11 (Black-box v.s. Non Black-box Simulation) *Note that the stand-alone adversary S constructed in the proof of Theorem 6.8 is very different from the stand-alone adversary constructed in the proof of Theorem 6.1. In particular S constructed above in fact runs the simulator-extractor $\mathcal{S}$ (whereas in the proof of Theorem 6.1 the simulator extractor is simply used in the analysis. As a consequence, (in contrast to the simulator constructed in 6.1) the stand-alone adversary S constructed above makes use of the man-in-the middle adversary in a non black-box way if relying on a simulation-extractable argument with a non-black box simulator.*

Since families of non-interactive statistically hiding commitments can be based on collision-resistant hash functions [36, 12] we obtain the following corollary:

Corollary 6.12 (Statistically hiding non-malleable commitment) *Suppose that there exists a family of collision resistant hash functions. Then there exists a constant-round statistically hiding commitment scheme which is non-malleable with respect to opening.*

7 Acknowledgments

We are grateful to Johan Håstad and Moni Naor for many helpful conversations and great advice. Thanks to Boaz Barak for useful clarifications of his works. The second author would also like to thank Marc Fischlin, Rosario Gennaro, Yehuda Lindell and Tal Rabin for insightful discussions regarding non-malleable commitments. Thanks to Oded Goldreich for useful feedback on an earlier version of this work, and to the anonymous referees for their thoughtful comments. Finally, thanks to Huijia Lin for pointing out a subtlety in the proof of Proposition 4.2.

References

[1] B. Barak. How to go Beyond the Black-Box Simulation Barrier. In *42nd FOCS*, pages 106–115, 2001.

[2] B. Barak. Constant-Round Coin-Tossing or Realizing the Shared Random String Model. In *43rd FOCS*, p. 345-355, 2002.

[3] B. Barak and O. Goldreich. Universal Arguments and their Applications. *17th CCC*, pages 194–203, 2002.

[4] M. Bellare and O. Goldreich. On Defining Proofs of Knowledge. In *CRYPTO'92*, Springer (LNCS 740), pages 390–420, 1993.

[5] B. Barak and Y. Lindell. Strict Polynomial-Time in Simulation and Extraction. In *34th STOC*, p. 484–493, 2002.

[6] M. Blum. Coin Flipping by Telephone. In *CRYPTO 1981*, pages 11-15, 1981.

[7] M. Bellare, R. Impagliazzo and M. Naor. Does Parallel Repetition Lower the Error in Computationally Sound Protocols? In *38th FOCS*, pages 374–383, 1997.

[8] G. Brassard, D. Chaum and C. Crépeau. Minimum Disclosure Proofs of Knowledge. *JCSS*, Vol. 37, No. 2, pages 156–189, 1988. in *27th FOCS*, 1986.

[9] R. Canetti and M. Fischlin. Universally Composable Commitments. In *Crypto2001*, Springer LNCS 2139, pages 19–40, 2001.

[10] Ivan Damgård: A Design Principle for Hash Functions. *CRYPTO 1989*, pages 416-427, 1989.

[11] Ivan Damgård and Jens Groth. Non-interactive and Reusable Non-Malleable Commitment Schemes. In *35th STOC*, pages 426-437, 2003.

[12] I. Damgård, T. Pedersen and B. Pfitzmann. On the Existence of Statistically Hiding Bit Commitment Schemes and Fail-Stop Signatures. In *Crypto93*, pages 250–265, 1993.

[13] A. De Santis, G. Di Crescenzo, R. Ostrovsky, G. Persiano and A. Sahai. Robust Non-interactive Zero Knowledge. In *CRYPTO 2001*, pages 566-598, 2001.

[14] G. Di Crescenzo, J. Katz, R. Ostrovsky and A. Smith. Efficient and Non-interactive Non-malleable Commitment. In *EUROCRYPT 2001*, pages 40-59, 2001.

[15] G. Di Crescenzo, Y. Ishai and R. Ostrovsky. Non-Interactive and Non-Malleable Commitment. In *30th STOC*, pages 141-150, 1998

[16] D. Dolev, C. Dwork and M. Naor. Non-Malleable Cryptography. *SIAM Jour. on Computing*, Vol. 30(2), pages 391–437, 2000. Preliminary version in *23rd STOC*, pages 542-552, 1991

[17] U. Feige, D. Lapidot and A. Shamir. Multiple Noninteractive Zero Knowledge Proofs under General Assumptions. *Siam Jour. on Computing 1999*, Vol. 29(1), pages 1-28.

[18] U. Feige and A. Shamir. Witness Indistinguishability and Witness Hiding Protocols. In *22nd STOC*, p. 416–426, 1990.

[19] M. Fischlin and R. Fischlin. Efficient Non-malleable Commitment Schemes. *In CRYPTO 2000*, Springer LNCS Vol. 1880, pages 413-431, 2000.

[20] O. Goldreich. *Foundation of Cryptography – Basic Tools.* Cambridge University Press, 2001.

[21] O. Goldreich and A. Kahan. How to Construct Constant-Round Zero-Knowledge Proof Systems for NP. *Jour. of Cryptology*, Vol. 9, No. 2, pages 167–189, 1996.

[22] O. Goldreich and Y. Lindell. Session-Key Generation Using Human Passwords Only. In *CRYPTO 2001*, p. 408-432, 2001.

[23] O. Goldreich, S. Micali and A. Wigderson. Proofs that Yield Nothing But Their Validity or All Languages in NP Have Zero-Knowledge Proof Systems. *JACM*, Vol. 38(1), pages 691–729, 1991.

[24] O. Goldreich, S. Micali and A. Wigderson. How to Play any Mental Game – A Completeness Theorem for Protocols with Honest Majority. In *19th STOC*, pages 218–229, 1987.

[25] O. Goldreich and Y. Oren. Definitions and Properties of Zero-Knowledge Proof Systems. *Jour. of Cryptology*, Vol. 7, No. 1, pages 1–32, 1994.

[26] S. Goldwasser and S. Micali. Probabilistic Encryption. *JCSS*, Vol. 28(2), pages 270-299, 1984.

[27] S. Goldwasser, S. Micali and C. Rackoff. The Knowledge Complexity of Interactive Proof Systems. *SIAM Jour. on Computing*, Vol. 18(1), pages 186–208, 1989.

[28] J. Håstad, R. Impagliazzo, L.A. Levin and M. Luby. Construction of Pseudorandom Generator from any One-Way Function. *SIAM Jour. on Computing*, Vol. 28 (4), pages 1364–1396, 1999.

[29] J. Kilian. A Note on Efficient Zero-Knowledge Proofs and Arguments. In *24th STOC*, pages 723–732, 1992.

[30] Y. Lindell. Bounded-Concurrent Secure Two-Party Computation Without Setup Assumptions. In *34th STOC*, pages 683–692, 2003.

[31] P. D. MacKenzie, M. K. Reiter, K. Yang: Alternatives to Non-malleability: Definitions, Constructions, and Applications. TCC 2004, pages 171-190, 2004.

[32] R. C. Merkle. A Certified Digital Signature. In *CRYPTO 1989*, 218-238, 1989.

[33] S. Micali. CS Proofs. *SIAM Jour. on Computing*, Vol. 30 (4), pages 1253–1298, 2000.

[34] M. Naor. Bit Commitment using Pseudorandomness. *Jour. of Cryptology*, Vol. 4, pages 151–158, 1991.

[35] M. Naor, R. Ostrovsky, R. Venkatesan and M. Yung. Perfect Zero-Knowledge Arguments for NP Using any One-Way Permutation. *Jour. of Cryptology*, Vol. 11, pages 87–108, 1998.

[36] M. Naor and M. Yung. Universal One-Way Hash Functions and their Cryptographic Applications. In *21st STOC*, pages 33–43, 1989.

[37] M. Nguyen and S. Vadhan. Simpler Session-Key Generation from Short Random Passwords. In *1st TCC*, p. 428-445, 2004.

[38] R. Pass. Bounded-Concurrent Secure Multi-Party Computation with a Dishonest Majority. In *36th STOC*, 2004, pages 232-241, 2004.

[39] R. Pass and A. Rosen. Bounded-Concurrent Secure Two-Party Computation in a Constant Number of Rounds. In *34th FOCS*, pages 404-413, 2003.

[40] R. Pass and A. Rosen. Concurrent Non-Malleable Commitments. In *36th FOCS*, pages 563–572, 2005.

[41] A. Sahai. Non-Malleable Non-Interactive Zero Knowledge and Adaptive Chosen-Ciphertext Security. *In 40th FOCS*, pages 543-553, 1999.

Appendix

A Missing Proofs

Proposition 4.2 (Argument of knowledge) *Let $\langle P_{\mathsf{sWI}}, V_{\mathsf{sWI}}\rangle$ and $\langle P_{\mathsf{UA}}, V_{\mathsf{UA}}\rangle$ be the protocols used in the construction of $\langle P_{\mathsf{sUA}}, V_{\mathsf{sUA}}\rangle$. Suppose that $\{\mathcal{H}_n\}_n$ is collision resistant for $T(n)$-sized circuits, that* **Com** *is statistically hiding, that $\langle P_{\mathsf{sWI}}, V_{\mathsf{sWI}}\rangle$ is a statistical witness indistinguishable argument of knowledge, and that $\langle P_{\mathsf{UA}}, V_{\mathsf{UA}}\rangle$ is a universal argument. Then, for any* $\text{TAG} \in \{0,1\}^n$, *$\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle$ is an interactive argument of knowledge.*

Completeness of $\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle$ follows from the completeness property of $\langle P_{\mathsf{sUA}}, V_{\mathsf{sUA}}\rangle$. Specifically, an honest prover P, who possesses a witness w for $x \in L$ can always make the verifier accept by using w as the witness in the n parallel executions of $\langle P_{\mathsf{sUA}}, V_{\mathsf{sUA}}\rangle$. To demonstrate the argument of knowledge property of $\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle$, it will be sufficient to prove that $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$ is an argument of knowledge. This is because the prescribed verifier in $\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle$ will accept the proof only if *all* runs of $\langle P_{\mathsf{tag}_i}, V_{\mathsf{tag}_i}\rangle$ are accepting.[22]

Lemma A.1 *Suppose that $\{\mathcal{H}_n\}_n$ is collision resistant for $T(n)$-sized circuits, that* **Com** *is statistically hiding, that $\langle P_{\mathsf{sWI}}, V_{\mathsf{sWI}}\rangle$ is a statistical witness indistinguishable argument of knowledge, and that $\langle P_{\mathsf{UA}}, V_{\mathsf{UA}}\rangle$ is a universal argument. Then, for any* $\mathsf{tag} \in [2n]$, *$\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$ is an argument of knowledge.*

Proof: We show the existence of an extractor machine E for protocol $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$.

Description of the extractor machine. E proceeds as follows given oracle access to a malicious prover P^*. E, using black-box access to P^*, internally emulates the role of the honest verifier V_{tag} for P^* *until* P^* provides an accepting proof (i.e., if P^* fails, E restarts P^* and attempts a new emulation of V_{tag}). Let σ_s denote the messages received by P^*, in the successful emulation by E, up until protocol $\langle P_{\mathsf{WI}}, V_{\mathsf{WI}}\rangle$ is reached; let P^*_{WI} denote the residual prover $P^*(\sigma_s)$.

The extractor E next applies the witness extractor E_{WI} for $\langle P_{\mathsf{WI}}, V_{\mathsf{WI}}\rangle$ on P^*_{WI}. If $E_{\mathsf{WI}}^{P^*_{\mathsf{WI}}}$ outputs a witness w, such that $R_L(x,w) = 1$, E outputs the same witness w, otherwise it outputs `fail`. (The reason E constructs P^*_{WI} as above is to ensure that P^*_{WI} convinces V_{WI} with non-zero probability; otherwise E_{WI} is not guaranteed to extract a witness).

Analysis of the extractor. Let P^* be a non-uniform PPT that convinces the honest verifier V_{tag} of the validity of a statement $x \in \{0,1\}^n$ with probability $\epsilon(n)$. We assume without loss of generality that P^* is deterministic. We need to show the following two properties:

1. The probability that P^* succeeds in convincing V_{tag}, but E does not output a valid witness to x, is negligible.
2. The expected number of steps taken by E is bounded by $\frac{poly(n)}{\epsilon(n)}$.

We start by noting that since E perfectly emulates the role of the honest verifier V_{tag}, E requires in expectation $\frac{poly(n)}{\epsilon(n)}$ steps before invoking the extractor E_{WI}. Additionally, by the proof-of-knowledge property of $\langle P_{\mathsf{WI}}, V_{\mathsf{WI}}\rangle$ it follows that the expected running-time of E_{WI} is $\frac{poly(n)}{\epsilon(n)}$. To

[22] One could turn any cheating prover for $\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle$ into a cheating prover for $\langle P_{\mathsf{tag}_i}, V_{\mathsf{tag}_i}\rangle$ by internally emulating the role of the verifier V_{tag_j} for $j \neq i$ and forwarding the messages from P_{tag_i} to an external V_{tag_i}.

see this, let the random variable T denote the running-time of E_{WI} in the execution by E and let $T(\sigma)$ denote the running-time of E_{WI} given that E chooses the prefix σ. We abuse of notation and let σ denote the event that P^* receives the prefix σ in an interaction with V_{tag}; also let accept denote the event that P^* produces a convincing proof. We have,

$$E[T] = \sum_{\sigma} E[T(\sigma)] \Pr(\sigma|\mathsf{accept}) = \sum_{\sigma} \frac{poly(n)}{\Pr[\mathsf{accept}|\sigma]} \Pr(\sigma|\mathsf{accept}) =$$

$$poly(n) \sum_{\sigma} \frac{\Pr[\sigma]}{\Pr[\mathsf{accept}]} = \frac{poly(n)}{\Pr[\mathsf{accept}]} = \frac{poly(n)}{\epsilon(n)}$$

where the second equality follows from the definition of a proof of knowledge. We conclude by the linearity of expectations that the expected running-time of E is $\frac{poly(n)}{\epsilon(n)}$ and thus the second of the above properties holds.

We turn to show that also the first property holds. Assume that there exist some polynomial $p(n)$ such that for infinitely many n, $\epsilon(n) \geq \frac{1}{p(n)}$ but E^{P^*} fails to output a valid witness with probability $\frac{1}{p(n)}$. Note that E^{P^*} can fail for two reasons:

1. Either the extraction by E_{WI} fails, or
2. E_{WI} outputs $\langle \beta, \delta, s_1, s_2 \rangle$ so that:
   - $\hat{\beta} = \mathbf{Com}(\beta; s_1)$.
   - $\hat{\delta} = \mathbf{Com}(\delta; s_2)$.
   - $(\alpha, \beta, \gamma, \delta)$ is an accepting transcript for $\langle P_{\mathsf{UA}}, V_{\mathsf{UA}} \rangle$

If the second event occurs we say that E_{WI} outputs a *false witness.* Consider an alternative extractor E' which proceeds just as E except that if in the *first* emulation of V_{tag}, P^* fails in producing a convincing proof, E' directly aborts. Note that the only difference between E and E' is that E continues sampling executions until P^* produces a convincing proof, whereas E' only samples once. Since by our assumption, $\epsilon(n) \geq \frac{1}{p(n)}$, it follows that there exists some polynomial $p'(n)$ such that in the execution by E', E_{WI} either fails or outputs a false witness with probability $\frac{1}{p'(n)}$. It directly follows from the proof-of-knowledge property of $\langle P_{\mathsf{WI}}, V_{\mathsf{WI}} \rangle$ that E_{WI} fails only with negligible probability. We show below that E_{WI} outputs a false witness also with negligible probability; this is a contradiction.

Proposition A.2 *In the execution of E'^{P^*}, E_{WI} outputs a false witness with negligible probability.*

Proof: Towards proving the proposition we start by showing the following lemma.

Lemma A.3 *Let P^*_{sUA} be a non-uniform polynomial-time machine such that $E_{\mathsf{WI}}^{P^*_{\mathsf{sUA}}(\alpha,\gamma)}$ outputs a false witness to the statement $\bar{x} = (x, \langle h, c_1, c_2, r_1, r_2 \rangle)$ with probability $\epsilon(n) = \frac{1}{poly(n)}$ given uniformly chosen verifier messages α, γ. Then, there exists a* strict *polynomial-time machine* $\mathsf{extract}$ *such that with probability* $\mathrm{poly}(\epsilon(n))$, $\mathsf{extract}(P^*_{\mathsf{sUA}}, \bar{x})$ *outputs*

- *an index $i \in \{1, 2\}$*
- *strings y, s, z such that $z = h(\Pi)$, $r_i = \Pi(y, s)$, and $c_i = \mathbf{Com}(z; s)$*
- *a polynomial-time machine M such that $M(j)$ outputs the j'th bit of Π. (M is called the "implicit" representation of Π.)*

Proof: The proof of the lemma proceeds in the following two steps.

1. Using an argument by Barak and Goldreich [3], we use P^*_{sUA} and E_{WI} to construct a prover P^*_{UA} for the UARG $\langle P_{\mathsf{UA}}, V_{\mathsf{UA}}\rangle$ that succeeds with probability $poly(\epsilon(n))$.
2. Due to the weak proof of knowledge property of $UARG$ we then obtain an index $i \in \{1,2\}$, strings y, s, a hash $z = h(\Pi)$ s.t. $r_i = \Pi(y,s)$ and $c_i = \mathbf{C}(z;s)$. We furthermore obtain an "implicit" representation of the program Π.

Step 1. Constructing P^*_{UA}. P^*_{UA} proceeds as follows.

- P^*_{UA} starts by receiving a message α from the honest verifier V_{UA}.
- P^*_{UA} incorporates P^*_{sUA} and internally forwards the message α, to P^*_{sUA}, resulting in a residual prover $P^*_{\mathsf{sUA}}(\alpha)$.
- P^*_{UA} then internally emulates the role of the honest verifier for P^*_{sUA} until protocol $\langle P_{\mathsf{WI}}, V_{\mathsf{WI}}\rangle$ is reached (i.e., P^*_{UA} uniformly choses a random message $\bar{\gamma}$ that it forwards to P^*_{sUA}, resulting in a residual prover $P^*_{\mathsf{sUA}}(\alpha,\bar{\gamma})$). Thereafter, P^*_{UA} honestly emulates the verifier V_{WI} for $P^*_{\mathsf{sUA}}(\alpha,\bar{\gamma})$). If $P^*_{\mathsf{sUA}}(\alpha,\bar{\gamma})$ succeeds in providing an accepting proof, P^*_{UA} invokes the knowledge extractor E_{WI} on the prover $P^*_{\mathsf{sUA}}(\alpha,\bar{\gamma})$.
- In the event that $P^*_{\mathsf{sUA}}(\alpha,\bar{\gamma})$ does not produce an accepting proof, or if E_{WI} does not output an accepting tuple $\langle\beta,\delta,s_1,s_2\rangle$, P^*_{UA} halts. Otherwise, it externally forwards the message β to V_{UA} and receives as response γ.
- P^*_{UA} now rewinds P^*_{sUA} until the point where it awaits the message γ and internally forwards γ to P^*_{sUA}, resulting in a residual prover $P^*_{\mathsf{sUA}}(\alpha,\gamma)$. As before P^*_{UA} first honestly verifies the WI proof that $P^*_{\mathsf{UA}}(\alpha,\gamma)$ gives and in the case this proof is accepting applies the extractor E_{WI} to $P^*_{\mathsf{sUA}}(\alpha,\gamma)$.
- If E_{WI} outputs an accepting tuple $\langle\beta',\delta',s'_1,s'_2\rangle$, such that $\beta'=\beta$, P^*_{UA} forwards δ to V_{UA}, and otherwise it halts.

Since $\langle\alpha,\beta',\gamma,\delta'\rangle$ is an accepting transcript of $\langle P_{\mathsf{UA}}, V_{\mathsf{UA}}\rangle$, it follows that unless P^*_{UA} halts the execution, it succeeds in convincing the verifier V_{UA}.

We show that P^*_{UA} finishes the execution with probability $poly(\epsilon(n))$. Using the same argument as Barak and Goldreich [3] (of counting "good" verifier messages, i.e., messages that will let the prover succeed with "high" probability, see Claim 4.2.1 in [3]), it can shown that with probability $poly(\epsilon(n))$, P^*_{UA} reaches the case where the extractor outputs $\langle\beta',\delta',s'_1,s'_2\rangle$. Thus it only remains to show that conditioned on this event, $\beta'\neq\beta$ occurs with polynomial probability. In fact, the event that $\beta'=\beta$ can only occur with negligible probability or else we would contradict the binding property of **Com** (since $\hat{\beta} = \mathbf{Com}(\beta,s_1) = \mathbf{Com}(\beta',s'_1)$). We thus conclude that P^*_{UA} succeeds in convincing V_{UA} with probability $poly(\epsilon(n))$.

Furthermore, since the extractor E_{WI} is only applied when P^*_{UA} provides an accepting proof, it follows from the definition of a proof of knowledge that the *expected* running-time of P^*_{UA} is a polynomial, say $g(n)$. Finally, if we truncate the execution of P^*_{UA} after $2g(n)$ steps we get by the Markov inequality that (the truncated) P^*_{UA} still convinces produces convincing proofs with probability $poly(\epsilon)$.

Step 2: Extracting the "false" witness. By the weak proof of knowledge property of $\langle P_{\mathsf{UA}}, V_{\mathsf{UA}}\rangle$ there exists a strict PPT machine E_{UA} such that $E_{\mathsf{UA}}^{P^*_{\mathsf{UA}}}$ outputs an "implicit" representation of a "witness" to the statement $\bar{x} = (x, \langle h, c_1, c_2, r_1, r_2\rangle)$ proved by P^*_{UA}. Since the values i, y, s, z have fixed polynomial length, they can all be extracted in polynomial time. Note, however, that since there is not a (polynomial) bound on the length of the program Π, we can only extract an implicit representation of Π. The concludes the proof of the lemma. ■

Armed with Lemma A.3, we now turn to show that E_{WI} outputs a false witness with negligible probability in the execution of E'^{P^*}. Suppose for contradiction that there exist a polynomial $p(n)$ such that for infinitely many n's, E_{WI} outputs a false witness, with probability at least $\epsilon(n) = \frac{1}{p(n)}$. We construct a $T(n)^{O(1)}$-sized circuit family, $\{C_n\}_n$, that finds collisions for $\{\mathcal{H}_n\}_n$ with probability $\mathrm{poly}(\epsilon(n))$.

More specifically,

- On input $h \xleftarrow{\mathrm{R}} \mathcal{H}_n$, the circuit C_n incorporates P^* and internally emulates the honest verifier V for P^* until the protocol $\langle P_{\mathsf{sUA}}, V_{\mathsf{sUA}}\rangle$ is reached (i.e., C_n internally sends randomly chosen messages h, r_1, r_2 to P^*, resulting in a residual prover $P^*(h, r_1, r_2)$).

- C_n then invokes the knowledge extractor $\mathsf{extract}$, guaranteed by lemma A.3, on $P^*(h, r_1, r_2)$, extracting values i, y, s, z and an implicit representation of Π, given by a machine M.

- If $\mathsf{extract}$ fails, C_n outputs `fail`, otherwise it rewinds P^* until the point where it expects to receive the message r_i, and then continues the emulation of the honest verifier from this point (using new random coins).

- Once again, when P^* reaches $\langle P_{\mathsf{sUA}}, V_{\mathsf{sUA}}\rangle$, C_n invokes $\mathsf{extract}$ on the residual prover, extracting values i', y', s', z' and an implicit representation of Π', given by a machine M'.

- If the extraction fails or if $i' \neq i$, C_n outputs `fail`.

It remains to analyze the success probability of C_n. We start by noting that $y = y'$ occurs only with negligible probability. This follows from the computational binding property of **Com**. Since the probability that E_{WI} outputs a false witness is $\epsilon(n)$ it must hold that for a fraction $\epsilon(n)/2$ of the verifier messages before protocol $\langle P_{\mathsf{sUA}}, V_{\mathsf{sUA}}\rangle$, E_{WI} outputs a false witness with probability $\epsilon(n)/2$ when given oracle access to P^* having been feed messages in this set of "good" messages. Due to the correctness of $\mathsf{extract}$ it holds that when C_n selects verifier messages from this "good" set, the probability that extraction succeeds (in outputting a false witness) on P^* is

$$\epsilon' = poly(\epsilon)$$

Thus, given that C_n picks random verifier messages, it holds that the probability that $\mathsf{extract}$ succeeds is at least

$$\epsilon'' = \frac{\epsilon}{2}\epsilon' = poly(\epsilon)$$

There, thus, exists an index $\sigma \in \{1, 2\}$ such that the extraction outputs the index $i = \sigma$ with probability $\epsilon''' = \epsilon''/2$. Again, for a fraction $\epsilon'''/2$ of verifier messages before slot σ (when $\sigma = 1$, there is only one message, namely h, while when $\sigma = 2$, the messages are h, r_1), the residual prover ($P^*(h)$ when $\sigma = 1$, or $P^*(h, r_1)$ when $\sigma = 2$) succeeds in convincing the verifier with probability $\epsilon'''/2$. We conclude that with probability

$$\epsilon'''/2 \cdot (\epsilon'''/2)^2 = \mathrm{poly}(\epsilon)$$

C_n obtains an implicit representation of programs Π, Π' such that $\exists y, y' \in \{0,1\}^{(|r_i|-n)}$ for which $\Pi(y) = r_i$, $\Pi'(y') = r'_i$ and $h(\Pi) = h(\Pi')$. Using a simple counting argument it follows that with probability $(1 - 2^{-n})$ (over the choices of r_i, r'_i), $\Pi \neq \Pi'$.[23] Thus, by *fully* extracting the programs (from the implicit representation) C_n finds a collision with probability

$$\text{poly}(\epsilon) \cdot (1 - 2^{-n}) = \text{poly}(\epsilon)$$

Note that the time required for extracting these programs is upper bounded by $T(n)^{O(1)}$. Thus, any poly-time prover P^* that can make V accept $x \notin L$ with non-negligible probability can be used in order to obtain collisions for $h \overset{\text{R}}{\leftarrow} \mathcal{H}_n$ in time $T(n)^{O(1)}$ (note that we here additionally rely on the fact that $\mathsf{extract}$ is a strict polynomial-time machine), in contradiction to the collision resistance of $\{\mathcal{H}_n\}_n$ against $T(n)$-sized circuits.[24] This concludes the proof of the proposition. ■

This completes the proof of Lemma A.3. ■

Basing the construction on "standard" collision resistant hash functions Although the above analysis (for the proof of knowledge property of $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$) relies on the assumption that $\{\mathcal{H}_k\}_k$ is a family of hash functions that is collision resistant against $T(k)$-sized circuits, we note that by using the method of Barak and Goldreich [3], this assumption can be weakened to the (more) standard assumption of collision resistance against polynomial-sized circuits. The main idea in their approach is to replace the arbitrary hashing in Slot 1 and 2 with the following two step hashing procedure:

- Apply a "good"[25] error-correcting code ECC to the input string.
- Use tree-hashing [32, 10] to the encoded string.

This method has the advantage that in order to find a collision for the hash function in the "proof of knowledge" proof, the full description of programs Π, Π' is not needed. Instead it is sufficient to look at a randomly chosen position in the description (which can be computed in polynomial time from the implicit representation of Π, Π'). The analysis, here, relies on the fact that two different codewords differ in random position i with a (positive) constant probability.

[23] Fix Π, Π', y, r_i. Then, with probability 2^{-n} over the choice of r'_i, there exist a $y' \in \{0,1\}^{(|r'_i|-n)}$ s. t. $\Pi'(y') = r'_i$.

[24] We mention that by slightly modifying the protocol, following the approach by Barak and Goldreich [3], one can instead obtain a polynomial-sized circuit C_n finding a collisions for $\mathcal{H}_k$. More details follow after the proof.

[25] By "good" we here mean an error-correcting code correcting a constant fraction of error.