Concurrent Non-Malleable Commitments

Rafael Pass *　　Alon Rosen †

Abstract

We present a non-malleable commitment scheme that retains its security properties even when concurrently executed a polynomial number of times. That is, a man-in-the-middle adversary who is simultaneously participating in multiple concurrent *commitment phases* of our scheme, both as a sender and as a receiver, cannot make the values he commits to depend on the values he receives commitments to. Our result is achieved without assuming an a-priori bound on the number of executions and without relying on any set-up assumptions.

Our construction relies on the existence of *standard* claw-free permutations and only requires a constant number of communication rounds.

1 Introduction

The notion of commitment is central in cryptographic protocol design. Often described as the "digital" analogue of sealed envelopes, commitment schemes 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 stage.

For some applications, the above security guarantees are not sufficient and additional properties are required. For instance, the definition of commitments does not rule out the possibility that an adversary, upon seeing a commitment to a specific value v, is able to commit to a related value (say, $v - 1$), even though it does not know the actual value of v. This kind of attack might have devastating consequences if the underlying application relies on the *independence* of committed values (e.g., consider a case in which the commitment scheme is used for securely implementing a contract bidding mechanism). The state of affairs is even worsened by the fact that many of the known commitment schemes are actually susceptible to this kind of attack.

1.1 Non-Malleable Commitments

In order to address the above concerns, Dolev, Dwork and Naor (DDN) introduced the concept of *non-malleable commitments* [15]. Loosely speaking, a commitment scheme is said to be non-malleable if no adversary can succeed in the attack described above. That is, it is infeasible for the adversary to maul a commitment to a value v into a commitment to a "related" value $\tilde{v}$.

The first non-malleable commitment protocol was constructed by Dolev, Dwork and Naor [15]. The security of their protocol relies on the existence of one-way functions, and requires $O(\log n)$ rounds of interaction, where $n \in N$ is a security parameter. A more recent result by Barak presents a constant-round protocol for non-malleable commitment, whose security relies on the existence of trapdoor permutations and hash functions that are collision-resistant against sub-exponential sized circuits [2]. Even more recently, Pass and Rosen present a constant-round protocol for the same task, assuming only collision resistant hash function secure against polynomial sized circuits [38].

*Cornell University. E-mail: `pass@csail.mit.edu`. Work done while at CSAIL, MIT.

†Harvard DEAS, Cambridge, MA. E-mail: `alon@eecs.harvard.edu`. Part of work done while at CSAIL, MIT.

1.2 Concurrent Non-Malleable Commitments

The basic definition of non-malleable commitments only considers a scenario in which two executions take place at the same time. A natural extension of this scenario (already suggested in [15]) is one in which more than two invocations of the commitment protocol take place concurrently. In the concurrent scenario, the adversary is receiving commitments to multiple values $v_1, \ldots, v_m$, while attempting to commit to related values $\tilde{v}_1, \ldots, \tilde{v}_m$. As argued in [15], non-malleability with respect to two executions can be shown to guarantee *individual* independence of any $\tilde{v}_i$ from any v_j. However, it does not rule out the possibility of an adversary to create *joint* dependencies between more than a single individual pair (see [15], Section 3.4.1 for an example in the context of non-malleable encryption). Resolving this issue has been stated as a major open problem in [15].

Partially addressing this issue, prior work has demonstrated the existence of commitment schemes that remain non-malleable under *bounded concurrent* composition [36]. That is, for any (predetermined) polynomial $p(\cdot)$, there exists a non-malleable commitment that remains secure as long as it is not executed more than $p(n)$ times, where $n \in N$ is a security parameter.

One evident disadvantage of the above solution is that it requires that the number of executions is fixed *before* the protocol is specified, or otherwise no security guarantee is provided. Less evidently, the length of the messages in the protocols has to grow linearly with the number of executions. Thus, from both a theoretical and a practical point of view, the solution is still not satisfactory. What we would like to have is a *single* protocol that preserves its non-malleability even when it is executed concurrently for *any* (not predetermined) polynomial number of times.

1.3 Our Results

We present a new protocol for *concurrent non-malleable* commitments. Our protocol remains non-malleable even when concurrently executed an (unbounded) polynomial number of times. We do not rely on any kind of set-up assumption (such as the existence of a common reference string).

The resulting commitment is *statistically binding*, and satisfies non-malleability *with respect to commitment*. The former condition implies that, except with negligible probability, a transcript of a commitment corresponds to a unique value, whereas the latter implies that, upon concurrently participating in polynomially many commitments, both as a receiver and as a sender, the adversary is not able to *commit* to a sequence of related values.[1] Here we assume that the adversary does not get to see the de-commitment to any of the values he is receiving a commitment to until he is done with committing to all of his values.

Theorem 1 (Concurrent non-malleable commitment) *Suppose that there exists a family of pairs of claw-free permutations.*[2] *Then, there exists a constant-round statistically-binding commitment scheme that is concurrently non malleable with respect to commitment.*

To the best of our knowledge, this result yields the first instance of a non-trivial protocol that simultaneously satisfies non-malleability and concurrency without relying on set-up assumptions.

[1] In a different variant, called non-malleable commitment *with respect to opening* [18], the adversary is considered to have succeeded only if it manages to *de-commit* to a related value. This paper only considers the notion of non-malleability with respect to commitments.

[2] The existence of claw-free permutations follows from the assumption that factoring Blum integers is hard (or from the hardness of finding discrete-logarithms modulo a prime). They are required for obtaining *perfectly* hiding-commitments, as well as collision resistant hashing.

Additional contributions. Our proof also yields the first commitment scheme that is *strictly* non-malleable with respect to commitment.[3] Strict non-malleability means that the simulation used to prove non-malleability runs in strict (as opposed to expected) polynomial time. This was the security notion originally defined (but not achieved in) the DDN paper [15].

Our definitions of non-malleable commitments are somewhat different (stronger) than the ones appearing in the DDN paper [15]. Specifically, we formalize the notion of two values being unrelated through the concept of computational indistinguishability (rather than using polynomial time computable relations). The main reason for strengthening the definition is that it yields a notion that is more intuitive and easier to work with (especially in the concurrent setting). We stress that any protocol satisfying our definition also satisfies the original one.

Techniques and ideas Our construction follows the paradigm introduced by Pass and Rosen (PR), of using a protocol for non-malleable zero-knowledge in order to obtain (single execution) non-malleable commitments [38]. While our construction relies on the same high-level structure, the analysis of the protocol is significantly different. The central observation that enables the analysis is that concurrent simulation of the underlying (non-malleable) zero-knowledge protocol is not actually necessary for proving concurrent non-malleability of our commitments. Indeed, for our analysis to go through, it will be sufficient to simulate only a *single* execution of the underlying zero-knowledge protocol. This will be performed while concurrently extracting *multiple* witnesses for the statements proved by the adversary. We call the above property *one-many simulation extractability.* We prove that this property is indeed satisfied by the non-malleable zero-knowledge protocols of [36, 38]. To show this, we rely on a *non-black box* simulation argument, which is delicately combined with a *black-box* extraction technique. (Here we use the fact that concurrent extraction is significantly easier than concurrent simulation (cf. [30]).)

1.4 Related Work

A large body of previous work deals with the construction of non-malleable protocols assuming various kinds of trusted set-up. Known constructions include non-malleable commitment schemes assuming the existence of a common reference string [18, 10], as well as non-malleable commitment schemes and non-interactive non-malleable $\mathcal{ZK}$ protocols assuming the existence of a common random string [14, 13, 12].

Several of the above works explicitly address the issue of multiple executions of non-malleable schemes [12, 10, 8] (also called *reusability* in the terminology of [10]). Perhaps most notable amongst the works addressing concurrency, is the one on Universally composable commitments [8]. Universal composability implies concurrent non-malleability. However, it is impossible to construct universally composable commitments without making set-up assumptions [8].

Other related works involve the task of session-key generation in a setting where the honest parties share a password that is taken from a relatively small dictionary [21, 35, 3]. These protocols are designed having a man-in-the-middle adversary in mind, and only require the usage of a "mild" set-up assumption (namely the existence of a "short" password). Some of these works explicitly address the issue of multiple protocol execution (cf. [21]), but their treatment is limited to the case of sequential composition. A treatment of the full concurrent case appears in [28] (see also [9, 3]), but it relies on the existence of a common reference string.

[3]This should not be confused with a previous result showing the existence of commitment schemes that are strictly non-malleability with respect to *opening* [38].

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 [19].

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 $(x,y) \in R_L$. 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 for each language $L \in \mathcal{NP}$.

2.3 Probabilistic notation

Denote by $x \overset{\mathrm{R}}{\leftarrow} X$ the process of uniformly choosing an element x in a set X. If $B(\cdot)$ is an event depending on the choice of $x \overset{\mathrm{R}}{\leftarrow} X$, 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

Let $S \subseteq \{0,1\}^*$ be a set of strings. A probability ensemble indexed by S is a sequence of random variables indexed by S. Namely, any $X = \{X_w\}_{w \in S}$ is a random variable indexed by S.

Definition 2.2 (Computational indistinguishability) *Two ensembles $X = \{X_w\}_{w \in S}$ and $Y = \{Y_w\}_{w \in S}$ are said to be* computationally indistinguishable *if for every probabilistic polynomial-time algorithm D, there exists a negligible function $\nu(\cdot)$ so that for every $w \in S$:*

$$|\Pr[D(X_w, w) = 1] - \Pr[D(Y_w, w) = 1]| < \nu(|w|)$$

The algorithm D is often referred to as the distinguisher. For more details on computational indistinguishability see Section 3.2 of [19].

Definition 2.3 (Statistical Closeness) *Two ensembles $X = \{X_w\}_{w \in S}$ and $Y = \{Y_w\}_{w \in S}$ are said to be* statistically close *if there exists a negligible function $\nu(\cdot)$ so that for every $w \in S$:*

$$\max_D\{\Pr[D(X_w, w) = 1] - \Pr[D(Y_w, w) = 1]\} < \nu(|w|)$$

Note that the definition does not require that the functions D are computable in polynomial time.

2.5 Interactive Proofs, Zero-Knowledge and Witness-Indistinguishability

We use the standard definitions of interactive proofs (and interactive Turing machines) [26, 19] and arguments [4]. 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 2.4 (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 $x \in L$,*

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

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

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

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.*

Definition 2.4 can be relaxed to require only soundness error that is bounded away from $1-\nu(|x|)$. This is so, since the soundness error can always be made negligible by sufficiently many parallel repetitions of the protocol. However, in the case of interactive arguments, we do not know whether this condition can be relaxed. In particular, in this case parallel repetitions do not necessarily reduce the soundness error (cf. [7]).

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 [26]. To make $\mathcal{ZK}$ robust in the context of protocol composition, Goldreich and Oren [24] 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 $\mathrm{view}_{V^*}^P(x, z)$ a random variable describing $V^*(z)$'s view of the interaction with P on common input x.

Definition 2.5 (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 $\{\mathrm{view}_{V^*}^P(x,z)\}_{z\in\{0,1\}^*,x\in L}$ and $\{S(x,z)\}_{z\in\{0,1\}^*,x\in L}$ are computationally indistinguishable.*

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 2.6 (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 $\{\mathrm{view}_{V^*}^P(x,z)\}_{z\in\{0,1\}^*,x\in L}$ and $\{S(x,z)\}_{z\in\{0,1\}^*,x\in L}$ are statistically close.*

In case that the ensembles $\{\mathrm{view}_{V^*}^P(x,z)\}_{z\in\{0,1\}^*,x\in L}$ and $\{S(x,z)\}_{z\in\{0,1\}^*,x\in L}$ 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" of the witness used by the prover for proving the statement. 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 independent of the witness, we mean that for any two possible $\mathcal{NP}$-witnesses that could ne used by the prover to prove the statement $x \in L$, the corresponding views are computationally indistinguishable.

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

Definition 2.7 (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 $w_x^1, w_x^2 \in R_L(x)$, the ensembles $\{\mathrm{view}_{V^*}^P(x,w_x^1)\}_{x\in L}$ and $\{\mathrm{view}_{V^*}^P(x,w_x^2)\}_{x\in L}$ are computationally indistinguishable.*

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

2.6 Universal Arguments

Universal arguments (introduced in [5] and closely related to the notion of CS-proofs [31]) 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).[4]

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

Definition 2.8 (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* $p(|y|)$*.*
- Completeness by a relatively efficient prover: *For every* $((M, x, t); w)$ *in* R_U*,*

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

  Furthermore, there exists a polynomial p *such that the total time spent by* $P(w)$*, on common input* (M, x, t)*, is at most* $p(T_M(x, w)) \leq p(t)$*.*
- Computational Soundness: *For every polynomialsize circuit family* $\{P_n^*\}_{n \in N}$*, and every triplet* $(M, x, t) \in \{0, 1\}^n \setminus L_U$*,*

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

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

2.7 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 [19].

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. Loosely speaking, 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 [19]). 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 [32, 27]. 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 [33]. However, *constant-round* schemes are only known to exist under stronger assumptions; specifically, assuming the existence of a collection of certified clawfree functions [20].

2.8 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 2.9 *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, a so called* knowledge extractor*, such that for every interactive machine P', every $x \in L$ and every $y, r \in \{0,1\}^*$ the following two properties hold:*

1. *Except with negligible probability, the machine E with oracle access to $P'_{x,y,r}$ outputs a solution $s \in R_L(x)$.*

2. *Furthermore, the expected number of steps taken by E is bounded by*

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

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.

3 Concurrent Non-Malleable Commitments

Non-malleable commitments were introduced by Dolev, Dwork and Naor (DDN) [15]. Our definitions of non-malleability are somewhat stronger that the ones proposed by DDN [15]. Specifically, we formalize the notion of two values being unrelated through the concept of computational indistinguishability (rather than using polynomial time computable relations).

3.1 The General Setting

Let $\langle C, R\rangle$ be a commitment scheme and consider a man-in-the-middle adversary A that is simultaneously participating in multiple concurrent executions of $\langle C, R\rangle$. Executions in which A is playing the role of the receiver are said to belong to the left interaction, whereas executions in which A is playing the role of the sender are said to belong to the right interaction. We assume for simplicity, and without loss of generality, that the number of commitment schemes taking place in the left and right interactions is identical. The total number of the interactions in which the adversary is involved (either as a sender or as a receiver) is not a-priori bounded by any polynomial (though it is assumed to be polynomial in the security parameter). We assume that the adversary does not get to see the de-commitment to any of the values he is receiving a commitment to until he is done with committing to all of his values.

Besides controlling the messages that it sends in the left and right interactions, A has control over their scheduling. 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. It can also arbitrarily interleave messages that belong to different executions within an interaction.

The adversary A is trying to take advantage of his participation in the commitments taking place in the left interaction in order to violate the security of the commitments executed in the right interaction. The honest sender and receiver are not necessarily aware to the existence of the adversary, and might be under the impression that they are interacting one with the other. We let $v_1, \ldots, v_m$ denote the values committed to in the left interaction and $\tilde{v}_1, \ldots, \tilde{v}_m$ denote the values committed to in the right interaction. The above scenario is depicted in Figure 1 (with no explicit demonstration of possible interleavings of messages between different executions).

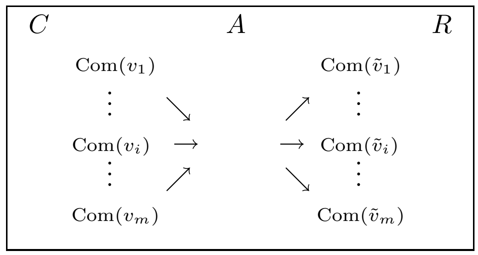

Figure 1: A concurrent man-in-the-middle adversary.

The traditional definition of non-malleable commitments [15] considers the case when $m = 1$. Loosely speaking, it requires that the left interaction does not "help" the adversary A in committing to a value $\tilde{v}_1$ that is somehow correlated with the value v_1. In this work we focus on non-malleability with respect to commitment [15], where the adversary is said to succeed if it manages to *commit* to a related value (even without being able to later de-commit to this value). Note that this notion makes sense only in the case of statistically-binding commitments.

3.2 Non-Malleability via Indistinguishability

Following the simulation paradigm [25, 26, 22, 23], the notion of non-malleability is formalized by comparing between a *man-in-the-middle* and a *simulated* execution. In the man-in-the-middle execution the adversary is simultaneously acting as a receiver in one interaction and as a committer in another interaction. In the simulated execution the adversary is engaged in a single interaction where it is acting as a committer.

The original definition of non malleability required that for any polynomial time computable (non-reflexive) relation $\mathcal{R}$, the value $\tilde{v}$ committed to by the adversary in the simulated execution is no (significantly) less likely to satisfy $\mathcal{R}(v, \tilde{v}) = 1$ than the value committed to by the adversary in the man-in-the-middle execution [15].

To facilitate the formalization for $m > 1$, we choose to adopt a slightly different definitional approach and will actually require an even stronger condition (which we are still able to satisfy with our protocol). Specifically, we require that for any adversary in a man-in-the-middle execution, there exists an adversary that commits to essentially the same value in the simulated execution. By essentially the same value, we mean that the value committed to by the simulator is computationally indistinguishable from the value committed to by the adversary in the man-in-the middle execution.

Since copying cannot be ruled out, we will only be interested in the case where copying is not considered success. We therefore impose the condition that whenever the adversary has *fully* copied a transcript of an interaction in which it acts as a receiver, the value $\tilde{v}$ that he has committed to in the corresponding execution is set to be a special "failure" symbol, denoted $\perp$.

3.3 The Actual Definition

Let $\langle C, R \rangle$ be a commitment scheme, and let $n \in N$ be a security parameter. Consider man-in-the-middle adversaries that are participating in left and right interactions in which $m = \text{poly}(n)$ commitments take place. We compare between a *man-in-the-middle* and a *simulated* execution.

The man-in-the-middle execution. In the man-in-the-middle execution, the adversary A is simultaneously participating in m left and right interactions. In the left interactions the man-in-the-middle adversary A interacts with C receiving commitments to values $v_1, \ldots, v_m$. In the right interaction A interacts with R attempting to commit to a sequence of related values $\tilde{v}_1, \ldots, \tilde{v}_m$. Prior to the interaction, the values $v_1, \ldots, v_m$ are given to C as local input. A receives an auxiliary input z, which may contain a-priori information about $v_1, \ldots, v_m$. Let $\mathsf{mim}^A_{\mathsf{com}}(v_1, \ldots, v_m, z)$ denote a random variable that describes the values $\tilde{v}_1, \ldots, \tilde{v}_m$ to which the adversary has committed in the right interaction. (Since we are dealing with statistically binding commitments, $\tilde{v}_1, \ldots, \tilde{v}_m$ are (almost always) well defined. Whenever the value of the commitment is not uniquely defined (which can happen with some negligible probability in case of statistically binding commitments), the value of the commitment is defined to be $\perp$.) If the transcript of the i^{th} right commitment is identical to the transcript of *any* of the left interactions (which means that adversary has fully copied a specific commitment that has taken place on the left), the value $\tilde{v}_i$ is set to be $\perp$.[5]

The simulated execution. In the simulated execution a simulator S directly interacts with R. As in the man-in-the-middle execution, the values $v_1, \ldots, v_m$ are chosen prior to the interaction and S receives some a-priori information about $v_1, \ldots, v_m$ as part of its an auxiliary input z. We let $\mathsf{sim}^S_{\mathsf{com}}(v_1, \ldots, v_m, z)$ denote a random variable that describes the values committeed to in the output of S (which consists of a sequence of values $\tilde{v}_1, \ldots, \tilde{v}_m$).

Definition 3.1 *A commitment scheme $\langle C, R \rangle$ is said to be* concurrent non-malleable with respect to commitment *if for every polynomial $p(\cdot)$, and every probabilistic polynomial-time man-in-the-middle adversary A that participates in at most $m = p(n)$ concurrent executions, there exists a probabilistic polynomial time simulator S such that the following ensembles are computationally indistinguishable:*

- $\left\{\mathsf{mim}^A_{\mathsf{com}}(v_1, \ldots, v_m, z)\right\}_{v_1,\ldots,v_m \in \{0,1\}^n, z \in \{0,1\}^*}$
- $\left\{\mathsf{sim}^S_{\mathsf{com}}(z)\right\}_{v_1,\ldots,v_m \in \{0,1\}^n, z \in \{0,1\}^*}$

It can be seen that for $m = 1$ any protocol that satisfies Definition 3.1 also satisfies the original (relation based) definition of non-malleability. Loosely speaking, this is because the existence of a polynomial time computable relation $\mathcal{R}$ that violates the original definition of non-malleability could be used to distinguish between the values of $\mathsf{mim}^A_{\mathsf{com}}(v, z)$ and $\mathsf{sim}^S_{\mathsf{com}}(v, z)$.

3.4 One-Many Concurrent Non-Malleable Commitments

A seemingly more relaxed (and thus potentially easier to satisfy) notion of concurrent non-malleable commitments is one in which the man-in-the-middle adversary A engages in only a *single* commitment protocol in the left interaction (but still polynomially many in the right interaction). Such a

[5] This approach allows $\tilde{v}_i = v$, as long as the man-in-the-middle does not fully copy the messages from one of the left executions. This is in contrast to the original definition which does not handle the case of $\tilde{v} = v$ (as $\mathcal{R}$ is non-reflexive). This means that the new approach takes into consideration a potentially larger class of attacks.

notion is a special case of Definition 3.1 in which the adversary A participates in only one commitment sessions on the left hand side (instead of m sessions).

A commitment protocol that satisfies the relaxed definition is said to be *one-many concurrent non-malleable.* As we argue below, the relaxed notion turns out to imply full-fledged non-malleable commitments. In particular, in order to construct concurrent non-malleable commitments, it will be sufficient to come up with a protocol that is one-many concurrent non-malleable.

Proposition 3.2 *Let $\langle C, R\rangle$ be a one-many concurrent non-malleable commitment. Then, $\langle C, R\rangle$ is also a (full fledged) concurrent non-malleable commitment.*

Proof: Let A be a man-in-the-middle adversary that participates in at most $m = p(n)$ concurrent executions. We show the existence of a simulator S such that the following ensembles are computationally indistinguishable:

- $\left\{\mathsf{mim}^A_{\mathsf{com}}(v_1, \ldots, v_m, z)\right\}_{v_1,\ldots,v_m\in\{0,1\}^n, z\in\{0,1\}^*}$
- $\left\{\mathsf{sim}^S_{\mathsf{com}}(z)\right\}_{v_1,\ldots,v_m\in\{0,1\}^n, z\in\{0,1\}^*}$

The simulator S proceeds as follows on input z. S incorporates $A(z)$ and internally emulates all the left interactions for A by simply *honestly* committing to the string 0^n (i.e., in order to emulate the i^{th} left interaction, S executes the algorithm C on input 0^n). Messages from the right interactions are instead forwarded externally, except for the following difference: If A wishes to send the *last* message m in one right interaction, and if the complete transcript of this interaction (including the messages m) would be identical to the transcript of one of the emulated left interactions (i.e., if A has fully copied one of the left interactions), S instead externally sends the message $\perp$ (to invalidate the right commitment).

We show that the values that S commits to are indistinguishable from the values that A commits to. Suppose, for contradiction, that this is not the case. That is, there exists a polynomial-time distinguisher D and a polynomial $p(n)$ such that for infinitely many n, there exist strings $v_1, \ldots, v_m \in \{0,1\}^n, z \in \{0,1\}^*$ such that

$$\Pr\left[D(\mathsf{mim}^A_{\mathsf{com}}(v_1, \ldots, v_m, z) = 1\right] - \Pr\left[D(\mathsf{sim}^S_{\mathsf{com}}(z)) = 1\right] \geq \frac{1}{p(n)}$$

Fix a generic n for which this happens. We provide a hybrid argument that will contradict the one-many non-malleability of $\langle C, R\rangle$. The "hybrid" random variable $H_k(v_1, \ldots, v_m, z)$ involves an execution where $A(z)$ is participating in m left and m right interactions, and is defined in the following way:

- For $j \leq k$, the j^{th} session in the left interaction consists of a commitment to 0^n.
- For $j > k$, the j^{th} session in the left interaction consists of a commitment to v_j.
- Output the values $\tilde{v}_1, \ldots, \tilde{v}_m$ committed to by A in the right interactions with an honest R.

Note that the values $\tilde{v}_1, \ldots, \tilde{v}_m$ are not efficiently computable, but are well defined nevertheless. Just as in Definition 3.1, if a commitment can be opened to two (or more) different values, we set its value to $\perp$. It can be seen that:

$$H_0(v_1, \ldots, v_m, z) = \mathsf{mim}^A_{\mathsf{com}}(v_1, \ldots, v_m, z)$$

$$H_m(v_1, \dots, v_m, z) = \mathsf{sim}^S_{\mathsf{com}}(z)$$

If follows by a standard hybrid argument that there exists an i such that

$$\Pr\Big[D(H_{i-1}(v_1, \dots, v_m, z) = 1\Big] - \Pr\Big[D(H_i(v_1, \dots, v_m, z)) = 1\Big] \geq \frac{1}{p(n)m}$$

Note that the only difference between the experiments $H_{i-1}(v_1, \dots, v_m, z)$ and $H_i(v_1, \dots, v_m, z)$, is that in the former A receives a commitment to v_i in session i, whereas in the latter it receives a commitment to 0^n. Now, consider the one-many adversary $\tilde{A}$ that when receiving $\tilde{z} = (i, v_1, \dots, v_m, z)$ as auxiliary input proceeds as follows. $\tilde{A}$ internally incorporates $A(z)$ and emulates the left and right interactions for A.

1. $\tilde{A}$ forwards messages in its j^{th} right session directly to A (as part of its j^{th} right session).
2. $\tilde{A}$ forwards messages from its left session directly to A (as part of its i^{th} session).
3. $\tilde{A}$ emulates all left sessions $j \neq i$, by committing to v_j if $j > i$, and committing to 0^n otherwise.

Note that

$$\mathsf{mim}^{\tilde{A}}_{\mathsf{com}}(v_i, \tilde{z}) = H_{i-1}(v_1, \dots, v_m, z)$$
$$\mathsf{mim}^{\tilde{A}}_{\mathsf{com}}(0, \tilde{z}) = H_i(v_1, \dots, v_m, z)$$

This contradicts the fact that there exists a simulator $\tilde{S}$ for $\tilde{A}$ such that both:

1. $\mathsf{mim}^{\tilde{A}}_{\mathsf{com}}(v_i, \tilde{z})$ and $\mathsf{sim}^{\tilde{S}}_{\mathsf{com}}(\tilde{z})$ are indistinguishable, and
2. $\mathsf{mim}^{\tilde{A}}_{\mathsf{com}}(0, \tilde{z})$ and $\mathsf{sim}^{\tilde{S}}_{\mathsf{com}}(\tilde{z})$ are indistinguishable.

We conclude that $\langle C, R\rangle$ is not one-many concurrent non-malleable. ■

4 The Protocol

Our construction of concurrent non-malleable commitments follows the paradigm introduced by Pass and Rosen for obtaining (single execution) non-malleable commitments [38]. The commit phase of the Pass–Rosen protocol consists of having the sender engage in a (standard) statistically binding commitment with the receiver and thereafter also provide a *non-malleable* $\mathcal{ZK}$ proof of knowledge of the value committed to (a step which we subsequently refer to as *compilation*). The reveal phase consists of sending the de-commitment information of the statistically binding commitment used in the commit phase.

The non-malleable $\mathcal{ZK}$ protocols that are used in our construction are identical to the ones presented in [38]. The basic scenario in which the $\mathcal{ZK}$ protocols take place involves a man-in-the-middle adversary A is simultaneously participating in two executions of the protocol. These executions are called the left and the right interaction.

The left interaction is tagged by an identity string $\text{TAG} \in \{0,1\}^n$, and the right interaction is tagged by an identity $\widetilde{\text{TAG}} \in \{0,1\}^n$. The instructions of the protocol executed in each of the interactions depend on the corresponding identities. (The way in which the identity strings are determined and used will become clear at a later stage.)

In the left interaction, the adversary A is verifying the validity of a statement x by interacting with an honest prover P_{TAG} using a protocol $\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle$. In the right interaction A proves the validity of a statement $\tilde{x}$ to the honest verifier $V_{\widetilde{\text{TAG}}}$. The statement $\tilde{x}$ is chosen by A, possibly depending on the messages it receives in the left interaction. As in the case of concurrent non-malleable commitments, A has control over the scheduling of the messages.

4.1 Non-Malleability and Simulation-Extractability

In [38] it is shown that a commitment is non-malleable as long as it is compiled using $\mathcal{ZK}$ protocols that satisfy a *simulation exractability* property (a strengthening of non-malleability). Loosely speaking, simulation extractability requires that for any man-in-the-middle adversary A, there exists a 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.

For the purpose of the current work we will need to show that the $\mathcal{ZK}$ protocols used in the compilation satisfy an even stronger property, which we call *one-many simulation-extractability*. This is a strengthening of the simulation extractability property in that it guarantees simulation and extraction (of all witnesses on the right) even if there is an *unbounded* number of concurrent *right* interactions (but still with only one left interaction).

As we will show later, a (non-interactive) commitment scheme that is compiled with one-many simulation-extractable $\mathcal{ZK}$ will result in a one-many concurrent non-malleable commitment protocol $\langle C, R\rangle$. By Proposition 3.2, this implies that $\langle C, R\rangle$ is also concurrent non-malleable.

Let A be a man-in-the middle adversary that is simultaneously participating in one left interaction of $\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle$ while acting as verifier, and an (unbounded) polynomial number of right-interactions of $\langle P_{\tilde{\text{TAG}}_i}, V_{\tilde{\text{TAG}}_i}\rangle_{i=1}^m$ while acting as prover. Let $\mathsf{view}_A(x, z, \text{TAG})$ denote the view of $A(x, z)$ when verifying a left-proof of the statement x, using identity TAG, and proving on the right interaction statements of its choice and using identities $\tilde{\text{TAG}}_1, \ldots, \tilde{\text{TAG}}_m$ of its choice.

Definition 4.1 (One-many Simulation-extractability) *A family* $\{\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle\}_{\text{TAG}\in\{0,1\}^n}$*, is said to be* one-many simulation extractable *if for any polynomial* $p(\cdot)$ *and any man-in-the-middle adversary* A *that participates in one left interaction and at most* $m = p(n)$ *right interactions, there exists a probabilistic polynomial time,* $S = (\mathsf{SIM}, \mathsf{EXT})$ *such that for every* $\text{TAG} \in \{0,1\}^n$*, every* $x \in L$ *and every auxiliary input* $z \in \{0,1\}^*$*, the following holds:*

1. $\mathsf{SIM}(x, z, \text{TAG})$ *is statistically indistinguishable from* $\mathsf{view}_A(x, z, \text{TAG})$.
2. $\mathsf{EXT}(x, z, \text{TAG})$ *contains witnesses* w_i *for all statements* x_i *proved in accepting right sessions in the view* $\mathsf{SIM}(x, z, \text{TAG})$*, for which* $\tilde{\text{TAG}}_i \neq \text{TAG}$ *(where* $\tilde{\text{TAG}}_i$ *is the identity string used in the* i^{th} *session in the right hand side view of* $\mathsf{SIM}(x, z, \text{TAG})$*).*

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.

4.2 A Simulation Extractable Protocol

We now turn to describe our construction of simulation extractable $\mathcal{ZK}$ protocols. At a high level, the construction proceeds in two steps:

1. Construct a family of n zero-knowledge protocols that are simulation-extractable with respect to each other (i.e., with identity strings $\mathsf{tag} \in [n]$). These protocols are identical to the protocols introduced by Pass in [36].
2. Transform the above protocol into a family of 2^n protocols with the same property (i.e., with identity strings $\text{TAG} \in \{0,1\}^n$). The technique used in the transformation is from [38].

For simplicity of exposition we will start by describing a "weak" version of our $\mathcal{ZK}$ protocols (which is quite complicated by itself). We then turn to describe a stronger version, which enjoys statistical (actually perfect) security, and is the one which is required for our proof to go through.

4.2.1 A family of n protocols

The family of n protocols is denoted $\{\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle\}_{\mathsf{tag}\in[n]}$, and relies on the $\mathcal{ZK}$ protocol of Barak [1] (see also [1, 36, 38]). Let $n \in N$, and let $T : N \to N$ be a "nice" function that satisfies $T(n) = n^{\omega(1)}$. $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$ relies on a "special" $\mathbf{NTIME}(T(n))$ relation, which we denote by R_{sim}. Let $\{\mathcal{H}_n\}_n$ be a family of hash functions where a function $h \in \mathcal{H}_n$ maps $\{0,1\}^*$ to $\{0,1\}^n$, 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 2.

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: $R_{\mathsf{Sim}}(\langle h, c, r\rangle, \langle \Pi, y, s\rangle) = 1$ 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 2: The relation R_{sim}.

Similarly to [1], the protocol $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$ makes use of a witness-indistinguishable universal argument ($WIUARG$) [17, 16, 29, 31, 5]. Let L be any language in $\mathcal{NP}$, let $n \in N$, let $x \in \{0,1\}^n$ be the common input for the protocol, and let $\mathsf{tag} \in [n]$. $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$ is described in Figure 3.

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

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

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

Stage 0 (Set-up):

$V \to P$: Send $h \xleftarrow{\mathrm{R}} \mathcal{H}_n$.

Stage 1 (Slot 1):

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

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

Stage 1 (Slot 2):

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

$V \to P$: Send $r_2 \xleftarrow{\mathrm{R}} \{0,1\}^{(m+1-\mathsf{tag})\cdot\ell(n)}$.

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 three statements:

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

Figure 3: The Pass protocol – $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$.

What differentiates between two protocols $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$ and $\langle P_{\tilde{\mathsf{tag}}}, V_{\tilde{\mathsf{tag}}}\rangle$ is the fact that the length of the verifier's messages in $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$ is a parameter that depends on tag. The length of verifier messages is also dictated by the parameter $\ell(n)$.

4.2.2 A family of 2^n protocols

For the zero-knowledge protocols to be useful, it will be required to handle identity strings of length n, and in particular construct a family of 2^n protocols, whereas the family of protocols described above was of size n (or at most $O(n)$).

One approach for obtaining a family of 2^n simulation-extractable protocols is to rely on an "n-slot" version of the protocols in $\{\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle\}_{n\in[n]}$ (this was suggested in [36]). An alternative approach (from [38]), shows how to obtain a family of 2^n *constant-round* protocols by running n parallel executions of $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$, using appropriately chosen tags. This new family of protocols is denoted $\{\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle\}_{\text{TAG}\in\{0,1\}^n}$ and is described in Figure 4.

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

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

Tag String: $\text{TAG} \in \{0,1\}^n$. Let $\text{TAG} = \text{TAG}_1, \dots, \text{TAG}_n$.

The protocol:

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

1. Set $\mathsf{tag}_i = (i, \text{TAG}_i)$.
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 4: The Pass-Rosen protocol – $\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle$.

4.2.3 Strengthening the protocols

The above protocols are not strong enough for our analysis go through. To strengthen the protocols we make the following two modifications (cf. [38]):

Proof of knowledge. We require that the $WIUARG$ $\langle P_{UA}, V_{UA}\rangle$ that is used in Stage 2 of $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$ is of a special form (which we call a *special purpose* $WIUARG$). The main distinguishing feature of the special purpose $WIUARG$ (which is not necessarily satisfied by an "ordinary" $WIUARG$) is that it satisfies the proof of knowledge property of Definition 2.9. This was proved in [38] for a specific construction, which is described in Figure 11.

Perfect Zero-Knowledge. The current version of the protocols is computational zero-knowledge. For the proof to go through, however, we need to required that the protocols are *perfect* zero-knowledge. This it is obtained by:

1. Using a perfectly hiding commitment scheme **Com** in Stage 1 of the protocol, and
2. using a $UARG$ that is perfectly witness indistinguishable (a.k.a. *witness independent*) in Stage 2 of the protocol (this is achieved by using perfectly hiding commitments).

In the sequel we denote by $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$ (resp. $\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle$) the protocols obtained by performing the above modifications. As shown in [38], the second modification results in a perfect zero-knowledge version of the protocols $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$ (as well as $\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle$).[6]

[6]Actually $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$ is known to be sound only assuming that the family $\{\mathcal{H}_k\}_n$ is collision resistant against $T(n)$-sized circuits. Nevertheless, using ideas from [5], it is possible to show how by slightly modifying the relation R_{sim}, one can guarantee soundness under "standard" collision resistance. See [5, 38] for more details.

Lemma 4.2 ([38]) *Suppose that* **Com** *are perfectly hiding, that* $\{\mathcal{H}_n\}_n$ *is a family of collision-resistant hash functions, that* $\langle P_{UA}, V_{UA}\rangle$ *is a witness independent UARG, and that* $\ell(n) \geq 2n^2+n$. *Then, for any* $\text{TAG} \in \{0,1\}^n$, *the protocol* $\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle$ *is a perfect zero-knowledge argument.*

The main technical contribution of the current paper consists of proving that, as a result of the above two modifications, the protocol $\{\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle\}_{\text{TAG}\in\{0,1\}^n}$ is one-many simulation extractable.

Lemma 4.3 (Main technical lemma) *Suppose that* **Com** *are perfectly 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 witness independent UARG, and that* $\ell(n) \geq 2n^3+n$. *Then,* $\{\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle\}_{\text{TAG}\in\{0,1\}^n}$ *is one-many simulation extractable.*

Before we go on and prove Lemma 4.3, we turn to describe our commitment protocol and to show how one many simulation extractability is used in order to establish its one-many concurrent non-malleability. The full proof of Lemma 4.3 can be found in Section 5.

4.3 The Commitment Protocol

Using protocols from $\{\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle\}_{\text{TAG}\in\{0,1\}^n}$ as a subroutine, we present the construction of concurrent non-malleable commitments. Let $\{\text{Com}_r\}_{r\in\{0,1\}^*}$ be a family of *non-interactive* statistically binding commitment schemes (e.g., Naor's commitment [32]). Let $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ be a one-time signature scheme secure against a chosen-message attack. Consider the following protocol (which is a variant of the non-malleable commitment of Pass and Rosen [38]).[7]

Security Parameter: 1^k.

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

Commit Phase:

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

$C \to R$: Let $vk, sk \leftarrow \mathsf{Gen}(1^k)$. Pick uniformly $s \in \{0,1\}^k$.
Set $\text{TAG} = vk$ and send $c = \text{Com}_r(v;s), \text{TAG}$.

$C \leftrightarrow R$: Prove using $\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle$ that there exist $v, s \in \{0,1\}^k$ so that $c = \text{Com}_r(v;s)$.

$C \to R$: Let T denote the transcript of the above interaction.
Compute $\sigma = \mathsf{Sign}(sk, T)$ and send σ.

R: Verify that $\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle$ is accepting and that $\mathsf{Verify}(vk, T, \sigma) = 1$.

Reveal Phase:

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

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

Figure 5: Concurrent non-malleable commitment - $\langle C, R\rangle$.

As argued in [38], the statistical binding property of $\langle C, R\rangle$ follows directly from the statistical binding of Com. The computational hiding property follows from the computational hiding of Com, as well as from the (stand alone) $\mathcal{ZK}$ property of $\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle$ (see Lemma 4.2). Hence, we have.

[7]The difference between the this protocol and the protocol of [38] is that here we also employ a signature scheme. We note that the important difference, nevertheless, lies in the analysis of the protocol.

Proposition 4.4 ([38]) *Suppose that* $\{\mathrm{Com}_r\}_{r\in\{0,1\}^*}$ *is a family of* non-interactive *statistically binding commitment schemes, and that all members in the family* $\{\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle\}_{\text{TAG}\in\{0,1\}^k}$ *are (stand-alone) zero-knowledge. Then,* $\langle C, R\rangle$ *is a statistically-binding commitment protocol.*

4.4 Concurrent Non-Malleablity

Relying on the one-many simulation extractability of $\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle$, we next argue that $\langle C, R\rangle$ is one-many non-malleable.

Theorem 4.5 *Suppose that* $\{\mathrm{Com}_r\}_{r\in\{0,1\}^*}$ *is a family of* non-interactive *statistically binding commitment schemes, and that* $\{\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle\}_{\text{TAG}\in\{0,1\}^n}$ *is one-many simulation extractable. Then,* $\langle C, R\rangle$ *is a one-many concurrent non-malleable commitment.*

Proof: Consider a man-in-the-middle adversary A that participates in one left execution and $m = m(n)$ right executions. 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).

Consider the simulator S that 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., S executes the algorithm C on input 0^n). Messages from the right interactions are instead forwarded externally, with the following exception: If A wishes to send the *last* message m in one right interactions, and the if the complete transcript of this interaction (including the messages m) would be identical to the transcript of the emulated left interaction (i.e., if A has fully copied the left interactions), S instead externally sends the message $\perp$ (to invalidate the right commitment).

We show that the following distributions are indistinguishable.

- $\left\{\mathsf{mim}^A_{\mathsf{com}}(v, z)\right\}_{v\in\{0,1\}^n, z\in\{0,1\}^*}$
- $\left\{\mathsf{sim}^S_{\mathsf{com}}(z)\right\}_{v\in\{0,1\}^n, z\in\{0,1\}^*}$

Suppose, for contradiction, that this is not the case. That is, there exists a polynomial-time distinguisher D 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

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

Fix a generic n for which this happens. We show how this contradicts the simulation-extractability property of $\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle$. We start by providing an (oversimplified) sketch. On a high-level the proof consists of the following steps:

1. We first note that since the commit phase of (C, R) "essentially" only consists of a statement c (i.e., the commitment) and a proof of the validity of c, A can be interpreted as a one-many simulation-extractability 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 S' for A' that outputs a view that is statistically close to that of A', while at the same time outputting witnesses to all accepting right proofs.

3. Since the view output by the simulator-extractor S' is *statistically* close to the view of A' in the real interaction, it follows that also the *values* committed to in that view are are statistically close to the values 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 except with negligible probability, the simulator-extractor S' will output also the witnesses to all accepting right executions.[8] We conclude that S' additionally outputs the values *committed to* in the right executions.

5. We finally note that if D can distinguish between the values committed to by A and by S, then D can also distinguish the second output (which consists of the committed values) of S' when run on input a commitment (using Com) to v, and the second output of S' when run on input a commitment to 0. 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 the use a family of commitment schemes $\{\mathrm{Com}_r\}_{r\in\{0,1\}^*}$ and not a single non-interactive commitment scheme. To address this issue we make use of non-uniformity to show the existence of particular "prefix" of right-interactions such that A always chooses the instance Com_r in its left interaction, yet A commits to different values when receiving a commitments to v (as in mim) and 0 (as in sim).

More precisely, since in both experiments mim and sim the right executions are generated identically, there must exists some *fixed* prefix transcript τ of A's right interactions such that

1. A sends its first message r_τ in its *left* interaction directly after receiving the messages in τ (as part of its right executions).

2. D distinguishes between $\mathsf{mim}^A_{\mathsf{com}}(v, z)$ and $\mathsf{sim}^S_{\mathsf{com}}(z)$ with probability $p(n)$, conditioned on the event that the right executions are consistent with τ.

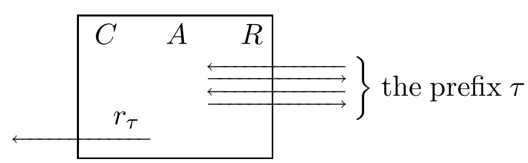

Figure 6: The prefix τ.

Note that in particular τ contains all first messages r sent by R in all right executions. Let $\mathtt{committed}_\tau$ (which is a subset of $[m]$) denote the set of executions i such that $A(z)$ has sent its first message in the i'th execution in τ. (Recall that the first message sent by A, playing the "role" of the C, consists of a commitment using Com.) For each $i \in \mathtt{committed}_\tau$, let $\mathtt{value}_\tau(i)$ denote the value committed to in the first message of execution i in τ. (If this value is not uniquely defined, set $\mathtt{value}_\tau(i) = \perp$).

We next define a one-many simulation-extractability adversary A' for $\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle$. On input $x', \text{TAG}', z' = (z, \tau, \mathtt{committed}_\tau, \mathtt{value}_\tau, sk)$, A' proceeds as follows. A' internally incorporates $A(z)$ and emulates the left and right interactions for A as follows.

[8]More precisely, the simulator-extractor only outputs witnesses to all right-executions that use a different tag than the left interaction. We rely on the use of the digital signature to handle the case when A copies the tag of the left interaction.

1. It starts by feeding A all messages in τ as part of its right executions.

2. For each $i \in \texttt{committed}_\tau$, A' internally emulates the (rest of the) i'th right execution for A by honestly following the strategy of R. (Note that this is possible, given any partial transcript τ, since R only uses public coins.)

3. For each $i \in [m], i \notin \texttt{committed}_\tau$, A' externally forwards messages in the i'th right interaction. (Note that since $i \notin \texttt{committed}_\tau$, A has not yet sent any messages in execution i. Thus A is expected to produce a statement c, a tag vk and a proof of the statement c, as part of its i'th execution.)

4. Messages in A's left interaction are forwarded externally as part of A''s left interaction. Once A' has feed A the last external message, A' signs the transcript of the left interaction using sk (received as auxiliary input) and feeds the signature to A. (Recall that whereas A' receives only a proof as part of its left interaction, A expects to see a commitment using $\langle C, R\rangle$. It is therefore essential that A' adds a signature to the proof.)

Now, define the hybrid experiment $H(v')$:

1. Pick $vk, sk \leftarrow \mathsf{Gen}(1^\mathsf{k})$, and pick uniformly $s \in \{0,1\}^n$.

2. Let r_τ denote the first message sent by $A(z)$ in its left execution, when feed the messages in τ (as part of its right interaction) and let $c = \mathrm{Com}_{r_\tau}(v', s)$.

3. Let $x' = c, \textsc{tag}' = vk, z' = (z, \tau, \texttt{committed}_\tau, \texttt{value}_\tau, sk)$. Emulate an execution for $A'(x', \textsc{tag}', z')$ by honestly providing a proof of x' (using tag $\textsc{tag}'$ and the witness (v', s)) as part of its left interaction, and honestly verifying all right interactions.

4. Finally, given the view of A' in the above emulation, reconstruct the view of A in the emulation by A'. Output the pair $(\mathit{view}, \bar{v})$ where view denotes the reconstructed view of A and $\bar{v}$ denotes the values committed to in the view of A. (As in definition 3.1, if a commitment is undefined, invalid, or if the transcript of the commitment is identical to the transcript of the commitment received by A' on the left, its value is set to $\perp$). Note that although the values committed to are not necessarily efficiently computable from the view of A, they are determined.

Note that H is not efficiently samplable, since the last step in the description of H is not efficient. However, except for that last step, every other operation in H is indeed efficient. (This will be useful to us at a later stage). We start by showing the following claim.

Claim 4.6 *D distinguishes the second output of $H(0)$ from the second output of $H(v)$ with probability $p(n)$.*

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

1. The first output of $H(v)$ is identically distributed to the view of A in $\mathsf{mim}^A_{\mathsf{com}}(v, z)$, conditioned on the event that the right interactions are consistent with τ.

2. The first output of $H(0^n)$ is identically distributed to the view of S in $\mathsf{sim}^S_{\mathsf{com}}(z)$, conditioned on the event that the right interactions are consistent with τ.

Since the second output of H is determined by the first output, it thus holds that:

1. The second output of $H(v)$ is identically distributed to the output of $\mathsf{mim}^A_{\mathsf{com}}(v,z)$, conditioned on the event that the right interactions in the view of A are consistent with τ.

2. The second output of $H(0^n)$ is identically distributed to the output of $\mathsf{sim}^S_{\mathsf{com}}(v,z)$, conditioned on the event that the right interactions in the view of S are consistent with τ.

The claim now follows from the fact that D distinguishes between $\mathsf{mim}^A_{\mathsf{com}}(v,z)$ and $\mathsf{sim}^S_{\mathsf{com}}(z)$ with probability $p(n)$, conditioned on the event that the right interactions are consistent with τ. ■

We next define an additional hybrid experiment H', that proceeds just as H, except that instead of emulating the left and right interactions for A', H runs the combined simulator extractor S' for A' to generate the view of A'.

Claim 4.7 *For any string v', the output of $H(v)$ is statistically close to the output of $H'(v')$.*

Proof: It following directly from the statistical indistinguishability property of S' that the first output of H is statistically close to the first output of H'. The claim is concluded by observing that the second output is determined by the first output. ■

Remark: *Note that the proof of Claim 4.7 inherently relies on the* statistical *indistinguishability property of S'. Indeed, if the simulation had only been computationally indistinguishable, we would not have been able to argue indistinguishability of the second output of H and H''. This follows from the fact that the second output (which consists of the* actual *committed values) is not efficiently computable from the view alone.*

We define a final hybrid experiment H'' that proceeds just as H' with the exception that instead of setting its second output $\bar{v}$ to the actual values committed to in the view of A, H'' efficiently computes values $\bar{v} = v_1, .., v_m$ as follows. Recall that the combined-simulator extractor S' outputs both a view and witnesses to all accepting right interactions. For each accepting right interaction i in the reconstructed view of A, H'' lets $v_i = \mathtt{value}_\tau(\mathtt{i})$ if $i \in \mathtt{committed}_\tau$ and otherwise set v_i to be consistent with the witness output for execution i by the combined simulator-extractor S'. For all right execution j for which the reconstructed view of A is rejecting, set $v_j = \perp$.[9]

Note that in contrast to H', H'' is efficiently computable. Furthermore it holds that:

Claim 4.8 *For any string v', the output of $H'(v)$ is statistically close to the output of $H''(v')$.*

Proof: Recall that the first output of H' and H'' are identical. We show that except with negligible probability, the second output of H'' consists of the values committed to in the first output of H'.

We start by noting that it directly follows that the procedure by H'' finds the correct values for all rejecting right-executions and all accepting right-executions i such that $i \in \mathtt{committed}_\tau$. We proceed to consider accepting right-executions i such that $i \notin \mathtt{committed}_\tau$. Recall that for these executions v_i is obtained from the witnesses output by the combined simulator-extractor S'. Also, recall that the combined simulator-extractor outputs witnesses for all accepting right interactions that use a different tag than the one used in the left interaction. Namely, values for all (non-rejected) right-commitments that use a verification key vk for the signature scheme that is different from the one used in the left-commitment, are extracted. Also, note that values for right-commitments that have *exactly* the same transcript as the left-commitment are "trivially" extracted (as they are just $\perp$).

[9]Note that an interaction that is accepting in the view of A' can still be rejecting in the view of A, since in the latter we additionally require a valid signature.

It only remains to analyze what happens to right-commitments that use the same verification key as the left-commitment, but different transcripts. In this case, the values committed to are not extracted. However, based on the unforgeability of the signature scheme, this event only happens with negligible probability.[10]

We conclude that except with negligible probability H'' computes the actual values of the commitments in the view of A, and thus the output of H' and H'' are statistically close. ■

By combining the above claims we obtain that D distinguishes the second output of $H''(v)$ and $H''(0^n)$. However, since H'' is efficiently samplable, we conclude that this contradicts the (non-uniform) hiding property of $\mathrm{Com}_{\mathrm{r}_\tau}$. More formally, define a third hybrid experiment H''' that proceeds as follows on input a commitment c' using Com_{r_τ}. H''' performs the same operations as H'', except that instead of generating the commitment c, it simply sets $c = c'$. It directly follows from the constructions that $H'''(c')$ is identical to $H''(0^n)$ when c' is a commitment to 0^n, and identical to $H''(v)$ when c' is a commitment to v. We conclude that D together with H''' (both of which are efficient) can be used to distinguish commitments (using Com_{r_τ}) to 0^n and v. ■

5 Simulation-Extractability

We now turn to prove Lemma 4.3. We start by proving an analogous lemma for the "small" family $\{\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle\}_{\mathsf{tag}\in[n]}$ of n protocols. Then we show how to extend the analysis to the family $\{\langle P_{\mathrm{TAG}}, V_{\mathrm{TAG}}\rangle\}_{\mathrm{TAG}\in\{0,1\}^n}$.

Recall that one-many simulation-extractability (Definition 4.1) means that there exists a combined simulator-extractor $S = (\mathsf{SIM}, \mathsf{EXT})$ that is able to simulate both the left and the right interactions for a man-in-the-middle adversary A, while simultaneously extracting witnesses to the m statements proved in the right interaction. The construction of S is fairly complex. To keep things simple, we decompose the description of the simulator into three simulation procedures, where each procedure relies on the previous (simpler) ones:

Basic simulator. This consists of the simulator that is used for establish the traditional (stand-alone) zero-knowledge property of $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$. The simulator is similar to the one used in Barak's original protocol [1].

Alternative simulator. This consists of the simulator that is used for establishing the "simulation soundness" (cf. [39]) of $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$. The simulator is designed to work in the presence of a man-in-the-middle adversary that is conducting a single left interaction of $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$ concurrently with a single right interaction of $\langle P_{\tilde{\mathsf{tag}}}, V_{\tilde{\mathsf{tag}}}\rangle$. It guarantees that an adversary whose left view consists of a simulated execution of $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$ cannot break the soundness of $\langle P_{\tilde{\mathsf{tag}}}, V_{\tilde{\mathsf{tag}}}\rangle$. The simulator is essentially identical to the one used by Pass [36].

Simulator-extractor. The description of the simulation extraction procedure $S = (\mathsf{SIM}, \mathsf{EXT})$ relies on the previous two simulators. The simulator SIM relies on the basic simulator to generate, whereas the extractor EXT (which also employs a simulation of the left interaction) makes use of the alternative simulator.

We turn to provide a description of the above simulation procedures. (We only provide a brief sketch of the basic and alternative simulators, and assume familiarity with the protocols of [1] and [36]. For completeness, the description of the simulator-extractor is nevertheless self-contained.)

[10]Note that even though the adversary has only seen *one* signed message using vk, we still need to rely on signature scheme that is secure against a *chosen message* attack. This follows from the fact that the adversary can influence the choice of this message (since the message is the transcript of the interaction).

5.1 Basic Simulator

Given the program, V^*_{tag}, of an adversary verifier, the basic simulator acts as follows. In Stage 1 of the protocol (i.e., in Slots 1 and 2), the simulator proceeds by committing to the program $\Pi \stackrel{\text{def}}{=} V^*_{\mathsf{tag}}$. Let s_1, s_2 the randomness used for the commitments.

In Stage 2 of the protocol, the simulator proves that it committed to the program of the verifier in Slot 1. More concretely, the simulator uses the tuple $\langle \Pi, c_1, s_1 \rangle$ as a witness for $\langle h, c_1, r_1 \rangle \in L_{\mathsf{sim}}$ (where L_{sim} is the language that corresponds to R_{sim}). This is a valid witness, since: (1) by the definition of Π it holds that $\Pi(c_1) = r_1$, and (2) as long as $\ell(n) \geq 3n$, for every $\mathsf{tag} \in [n]$, $|r_i| - |c_i| = \ell(n) - |c_i| \geq n$.

5.2 Alternative Simulator

The alternative simulator is constructed having a man-in-the middle adversary A in mind. Consider an A that manages to violate the soundness of protocol $\langle P_{\tilde{\mathsf{tag}}}, V_{\tilde{\mathsf{tag}}} \rangle$, while verifying a simulated proof of $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}} \rangle$. We show how to construct a cheating prover P^* for a single instance of $\langle P_{\tilde{\mathsf{tag}}}, V_{\tilde{\mathsf{tag}}} \rangle$ by forwarding A's messages in $\langle P_{\tilde{\mathsf{tag}}}, V_{\tilde{\mathsf{tag}}} \rangle$ to an external honest verifier V and internally simulating the messages of $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}} \rangle$ for A. The problem that arises in the attempt to simulate is that the code of the external verifier V is not available to the simulator. This means that a stand-alone simulation of the protocol $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}} \rangle$ cannot be completed as it is, since it explicitly requires possession of a "short" program Π that would have generated the corresponding verifier messages.

On a high-level, the prover P^* simulates the left interaction in the following way (see Section 5.3.2 for a more detailed description). In Slot 1 of the protocol, the simulator proceeds by committing to the program $\Pi_1 = A$. So far its instructions are just like the basic simulator. In Slot 2, however, the simulator commits to a program Π_2 which consists of both the code of A and all messages A has received from $V_{\tilde{\mathsf{tag}}}$ in the right interaction. In Stage 2 of the protocol, the simulator attempts to prove that it committed to the program of the verifier in either Slot 1 or Slot 2. Note that the simulator will succeed in this task if there exists a "short" message y (the actual required length of y is determined by the tag tag and the slot number) such that $\Pi_1(y) = r_1$ or $\Pi_2(y) = r_2$, where r_1, r_2 denotes the challenges receives in Slot 1 and 2 respectively (of $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}} \rangle$).

Now, except for the "long" challenges $\tilde{r_1}, \tilde{r_2}$ sent by the verifier of $\langle P_{\tilde{\mathsf{tag}}}, V_{\tilde{\mathsf{tag}}} \rangle$ we do have a description of all messages sent to the adversary A that is shorter than $\ell(n) - n$ (since $\ell(n) = \ell'(n) + n$, where $\ell'(n)$ upper bounds the total length of both prover and verifier messages, except for the challenges r_1, r_2). In order to show that we can still perform a simulation, even in the presence of these messages (for which we do not have a short description), we use the fact that it is sufficient to have a short description of the messages sent in *one* of the slots of $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}} \rangle$. As in [36], we separate between two different schedulings:

There exist one "free" slot j in $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}} \rangle$ in which neither of $\tilde{r_1}$, $\tilde{r_2}$ are contained. In this case the "free" slot j in $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}} \rangle$ can be used to perform a basic simulation (since in this case the simulator did indeed produce a commitment c_j to the code of a machine that on input c_j outputs the challenge r_j in slot j).

The messages r_1, r_2 in $\langle P_{\tilde{\mathsf{tag}}}, V_{\tilde{\mathsf{tag}}} \rangle$ occur in slot 1, 2 respectively in $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}} \rangle$. By construction of the protocols, the length of either the first or the second challenge in $\langle P_{\tilde{\mathsf{tag}}}, V_{\tilde{\mathsf{tag}}} \rangle$ is at least $\ell(n)$ bits longer than the corresponding challenge in $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}} \rangle$. Thus there exist a slot j in $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}} \rangle$ such that even if we include the verifier's challenge $\tilde{r}_j$ from the protocol $\langle P_{\tilde{\mathsf{tag}}}, V_{\tilde{\mathsf{tag}}} \rangle$ in the description y, we still have $\ell(n) - n$ bits to describe all other messages.

Remark 1 *As shown in [36], the protocols $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$ retain both their zero-knowledge and simulation soundness properties even if the adversary is allowed to participate in an a-priori* bounded *number of concurrent executions.*[11] *This additional property is useful to us in order to construct a family of 2^n protocols (see Section 5.5 and [38] for more details).*

5.3 Simulator-Extractor

Consider a man-in-the-middle adversary A. We assume without loss of generality that A is deterministic and has the auxiliary input z hardwired in. Let k denote the number of rounds in $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$, and let m be an upper-bound on the number of right interactions that A participates in. We describe a combined simulator-extractor $S = (\mathsf{SIM}, \mathsf{EXT})$, that proceeds as follows on input x (and auxiliary input z).

5.3.1 Simulation of view

We start by describing a machine SIM that simulates the view of A. This requires simulating all the left and the right interactions for A. In the right interactions SIM acts as a verifier. Thus, simulation is straightforward, and is performed by simply playing the role of an honest verifier in all the executions of the protocol. In the left interaction, on the other hand, SIM is supposed to act as a prover, and thus the simulation task is more involved. Towards its goals, SIM acts as follows.

1. For all $i \in [m]$, pick random $\bar{r}_i = (r_{i,1}, \ldots, r_{i,k})$ honest verifier messages for the right interactions. Messages in the right interactions are then emulated by playing the role of the honest verifiers with the fixed random messages $\bar{r}_1, \ldots, \bar{r}_m$. That is, in order to emulate the j^{th} message in the i^{th} right interaction, SIM forwards the message $r_{i,j}$ to A.

2. The left interaction is simulated as follows. SIM views the execution of A and the emulation of the right interactions (with the fixed messages $\bar{r}_1, \ldots, \bar{r}_m$) as a *stand-alone* verifier for the left interaction and applies a close variant of the basic simulator to this interaction. Let $\Pi(\cdot)$ denote the joint code of A and the emulation of the right interactions (including the coins $\bar{r}_1, \ldots, \bar{r}_m$). Whereas the basic simulator would have committed to $\Pi(\cdot)$, we instead let SIM commit to a program $\Pi'(b, \cdot)$ that is defined as follows:

   (a) if $b = 0$, execute $\Pi(\cdot)$;

   (b) if $b = 1$, execute $\Pi(\cdot)$ with the exception that messages $\bar{r}_i = (r_{i,1}, \ldots, r_{i,k})$ (i.e., messages of the i^{th} right interaction) are not emulated, but rather received externally as input.

   Thereafter, SIM proceeds exactly as the basic simulator, by additionally using both $b = 0$ and Π' as a witness in stage 2 of the protocol. More concretely, SIM starts by computing $h = \Pi(\cdot)$. It then generates prover commitments $c_1 = \mathrm{Com}(h(\Pi'); s_1)$ and $c_2 = \mathrm{Com}(h(\Pi'); s_2)$, where $s_1, s_2 \xleftarrow{\text{R}} \{0,1\}^{\mathrm{poly}(k)}$. Using c_1 and c_2, it computes $r_1 = \Pi(h, c_1)$, and $r_2 = \Pi(h, c_1, c_2)$. Combining the messages together, this results in a Stage 1 transcript $\tau_1 = \langle h, c_1, r_1, c_2, r_2\rangle$.

   By definition of $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$, the transcript τ_1 induces a Stage 2 $WIUARG$ with $\Pi(h, c_1, c_2)$ as verifier and $(x, \langle h, c_1, r_1\rangle, \langle h, c_2, r_2\rangle)$ as common input. Using $\langle \Pi', (0, h, c_1), s_1\rangle$ as witness for the statement $\langle h, c_1, r_1\rangle \in L_{\mathsf{sim}}$, the SIM follows the prescribed prover strategy of the $WIUARG$ and produces a convincing stage 2 transcript τ_2. Since $r_1 = \Pi(h, c_1) = \Pi'(0, h, c_1)$ and since $|0| + |h| + |c_1| \le \ell(k)$ it follows that SIM can always succeed in this task.

Figure 7 demonstrates the definition of SIM, as well as of the program $\Pi'(b, \cdot)$ (for simplicity the various sessions are depicted as if they were executed sequentially).

[11] We mention that this requires adjusting the length parameter $\ell(n)$ in a way that depends on the a-priori bound.

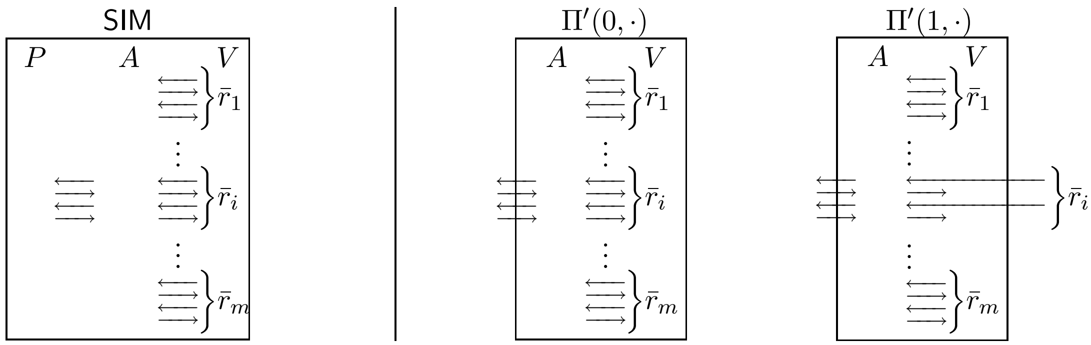

Figure 7: The simulator SIM and the program $\Pi'(b,\cdot)$.

5.3.2 Extraction of witnesses

Once the view of A has been simulated, we turn to the extraction of witnesses to the statements proved by A. Note that we need to extract witnesses to all concurrent right interactions. Towards this goal we rely on a variant of Lindell's concurrent extraction technique [30], combined with the alternative simulator technique described in Section 5.2. In a sense, this can be seen as a (non-trivial) extension of the method of Pass and Rosen [38] (which was used to show a similar property for the simpler case of only one right interaction).

The machine EXT fixes the random coins of the simulator SIM and iteratively extracts witnesses for each of the right interactions. More specifically, EXT starts by sampling a random execution of SIM, using random coins $\bar{s}, \bar{r}$. Let $x_1, \ldots, x_m$ be the inputs corresponding to the m sessions that have taken place in the right interaction.

For each $i \in [m]$ such that the i^{th} right session was *not* accepting, EXT will assume that no witness exists for the corresponding statement x_i, and will refrain from extraction. For all $i \in [m]$ so that the i^{th} right session *is* accepting in this execution of SIM, and for which the tag of the i^{th} session is different from the tag of the left session, EXT will attempt to extract a witness for the statement x_i being proved in the corresponding session.

To do so EXT constructs a stand-alone prover P_i for the i^{th} 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 8). In principle, the prover P_i will follow SIM's actions using the same random coins $\bar{s}, \bar{r}$ used for initially sampling the execution of SIM. However, P_i's execution will differ from SIM's execution in the following important ways:

1. Messages in the i^{th} right session are no longer emulated internally, but forwarded externally.

2. In the simulation of the left protocol, use the alternative simulator from Section 5.2 in order to complete stage 2 of the protocol.

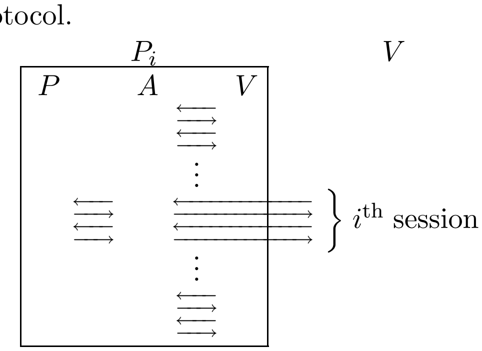

Figure 8: The prover P_i.

The reason for using the alternative simulation instead of the basic one is that the latter might not be able to commit to the external messages of the i^{th} right interaction (as it might not know these messages at the time it commits). Note that the way the simulation within P_i is defined, the program committed to in Stage 1 is Π'. To enable the alternative simulation with a commitment to Π' in Stage 1, we let the simulator additionally provide the input $b = 1$ to Π' as part of the witness in Stage 2 (this enables Π' to depend on the external messages in the i^{th} right session). The alternative simulation technique, combined with the fact that there is only *one* external interaction on the right hand side, are what eventually enables the simulation to go through.

The actual witness used by the simulator in Stage 2 depends on the scheduling of the messages. We distinguish between the following cases, depending on where the i^{th} session has started with relation to the messages c_1, c_2 in the left hand side protocol (see Figure 9).

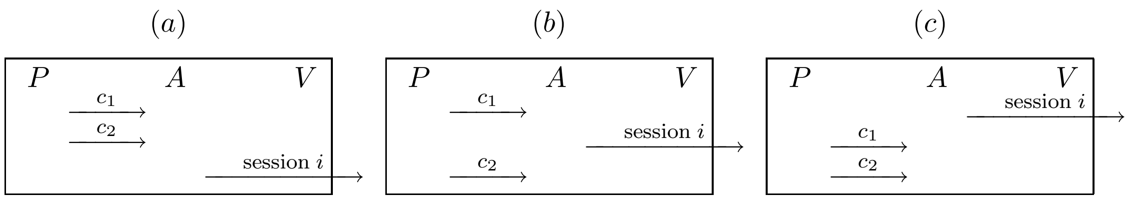

Figure 9: Three possible “starting points” for session i.

In each corresponding case, EXT acts as follows:

Both c_1 and c_2 have been sent before session i begins (Figure 9.a). In this event Slot 1 has no external messages and the basic simulation can be performed, i.e., EXT can use Π' as a witness for $\langle h, c_1, r_1 \rangle \in L_{\mathsf{sim}}$ in Stage 2 (just as in SIM).

c_1 has been sent but not c_2 (Figure 9.b). Let M_1, M_2 denote the “external” messages A receives on the right hand side in Slot 1 and Slot 2 of the left interaction, respectively (see Figure 10 for two ”representative” schedulings). In this case, we define $\Pi'_2(b, \cdot) = \Pi'(b, M_1, \cdot)$ and let EXT send $c_2 = \text{Com}(\Pi'_2; s)$ (whereas c_1 is defined just as in SIM).

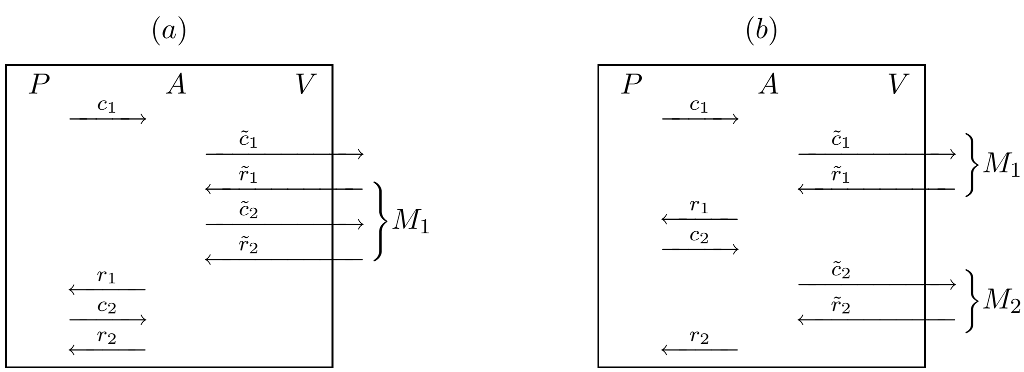

Figure 10: Two “representative” schedulings.

Consider a Stage 1 transcript $\tau_1 = \langle h, c_1, r_1, c_2, r_2 \rangle$ of the left interaction. By the construction of $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}} \rangle$, and from the fact that $\tilde{\mathsf{tag}}_i$ is different from tag, it must be the case that either $|M_1| \leq |r_1| - \ell(n)$ or $|M_2| \leq |r_2| - \ell(n)$. Thus, either $|M_1| + |h| + |c_1| + n \leq |r_1| - n$ or $|M_2| + |h| + |c_1| + n \leq |r_2| - n$ (see [36, 38] for more details). Furthermore, c_1 is a commitment to Π' and c_2 is a commitment to Π'_2, and $r_1 = \Pi'(1, (h, c_1, M_1))$ and $r_2 = \Pi'_2(1, (h, c_1, M_2))$.

Thus, either $w_1 = \Pi', 1, (h, c_1, M_1), s_1$ is a valid witness for $\langle h, c_1, r_1 \rangle \in L_{\mathsf{sim}}$ or $w_2 = \Pi'_2, 1, (h, c_2, M_2), s_2$ is a valid witness for $\langle h, c_2, r_2 \rangle \in L_{\mathsf{sim}}$. If the former is true, EXT follows the prescribed prover using w_1 as witness, and otherwise uses w_2 as witness.

Neither of c_1 or c_2 have been sent (Figure 9.c). In this case EXT first generates a commitment c_1 just as SIM would, i.e., lets c_1 be a commitment to Π', and then performs the same operations as in the previous case.

It follows from the description above that the simulation employed by P_i on the left interaction is always able to convince A in the validity of the statement proved on the left interaction (a very similar analysis appears in [38]). Moreover, this will hold even in case that the simulator is "rewound" and new external messages are sent over in the right hand side interaction (the ability to simulate while rewinding is necessary for the extraction to succeed).

Once P_i is constructed, EXT can apply the (stand-alone) extractor, guaranteed by the proof of knowledge property of $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}} \rangle$, to P_i and extract a witness to the statement x_i. In the unlikely event that the extraction failed in any of the m executions, EXT outputs `fail`, and otherwise it outputs all the extracted witnesses.

Remark 2 *It is important to have a P_i for the* entire *protocol $\langle P_{\tilde{\mathsf{tag}}_i}, V_{\tilde{\mathsf{tag}}_i} \rangle$ (and not just for $\langle P_{UA}, V_{UA} \rangle$). This is because in order to argue that the witness extracted is a witness for x_i and not a witness to $\langle h, c_1, r_1 \rangle \in L_{\mathsf{sim}}$ or to $\langle h, c_2, r_2 \rangle \in L_{\mathsf{sim}}$ (which could indeed be the case if we fixed the messages $\langle h, c_1, r_1, c_2, r_2 \rangle$ in advance).*

The output of S. Finally the combined simulator-extractor S outputs `fail` whenever EXT does. Otherwise, S outputs whatever SIM outputs, followed by whatever EXT outputs.

5.4 Correctness of the simulation-extraction

We proceed to show the correctness of the combined simulator-extractor $S = (\mathsf{SIM}, \mathsf{EXT})$. We start by showing the correctness of simulation part of the above-described simulator-extractor.

Claim 5.1 *The view of A in the simulation by* SIM *is identically distributed to its view in an actual interaction with P and V.*

Proof: The claim follows from 1) the perfect zero-knowledge property of $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}} \rangle$, and 2) the fact that the emulation of the right interactions by SIM is perfect. More precisely, consider the following hybrid experiments.

1. Let H_0 denote the view of A in the simulated execution.

2. Let H_1 denote the view of A in a simulated execution when letting the simulator use the true witness w for x in the $WIUARG$ in Stage 2 (instead of using the "fake" witness"). Thus the only difference between H_0 and H_1 is the choice of the witness used in the $WIUARG$.

3. Let H_2 denote the real execution. Note that the only difference between H_1 and H_2 is that in H_1 A receives commitments c_1, c_2 to a program Π', whereas in H_2 it receives a commitments to the string 0^k.

Sub Claim 5.2 *H_0 is identically distributed to H_1*

Proof: The claim follows from the witness independent property of the $WIUARG$ used in Stage 2. More precisely, assume for contradiction that H_0 is not identically distributed to H_1. Then there must exists some Stage 1 transcript, such that the proofs generated in Stage 2 in H_0 and H_1 are not identically distributed, in contradiction to the witness independent property of stage 2. ■

Sub Claim 5.3 *H_1 is identically distributed to H_2*

Proof: The claim directly follows from the perfect hiding property of the commitments used to generate c_1 and c_2. ■

This completes the proof of Claim 5.1. ■

Claim 5.4 EXT *outputs fail with negligible probability.*

Proof: Note that EXT outputs `fail` only in the event that extraction from one of the left interactions $i \in [m]$ fails. It follows from the POK property of $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}} \rangle$ that for each i the extraction fails with negligible probability. Since the extraction procedure is repeated at most m time (at most once per left-interaction), we conclude (by the Union Bound) that the probability that the extraction fails for any of the left interactions is negligible. ■

By combining Claim 5.1 and 5.4, we conclude that,

Claim 5.5 *The first output of S is statistically close to A's view in a "real" interaction.*

Proof: By claim 5.1 if follows that the first output of S is identically distributed to a "real" interaction, conditioned on the event that S does not output `fail`. However, since this event only happens when EXT outputs `fail`, which by claim 5.4 only happens with negligible probability, the claim follows. ■

We proceed to show the correctness of the extraction.

Claim 5.6 *For each $i \in [m]$, it holds that if the i^{th} right interaction was accepting in first output of S, and if the tag of the i^{th} interaction is different from the tag of the left interaction, then* EXT *outputs a witness to the statement proved in the i^{th} right interaction.*

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

Secondly, note that EXT performs extraction for all right-executions which satisfy the properties described in the hypothesis (i.e., accepting proofs and different tags). Finally, note that for each such interaction i, the stand-alone prover P_i constructed by EXT uses the same random coins as SIM in order to emulate all the interactions before session i begins. In addition, the prescribed actions for the simulation of EXT are *identical* to the prescribed actions for the simulation of SIM.

This means that the statement proved by P_i will be identical to the statement proved in the view output by SIM. Since the extraction by EXT proceeds until a witness is extracted (or until the extraction fails in which case, we are already done), EXT always outputs a witness to the statement proved in the i^{th} right-interaction. ■

We conclude the proof by bounding the running time of the combined simulator-extractor S.

Claim 5.7 *S runs in expected polynomial time.*

Proof: We start by noting that the running time of SIM is polynomial. Recall that that the program Π' committed to by SIM is of size $\text{poly}(k)$. It thus directly follows that simulation of Stage 1 messages can be done in polynomial time. Furthermore, it follows that the verification time of R_{sim} on the instance $\langle h, c_1, r_1\rangle$ is polynomial in k. Finally, by the relative prover efficiency of $\langle P_{\mathsf{UA}}, V_{\mathsf{UA}}\rangle$ it holds that the simulator can generate also Stage 2 message in polynomial time.

It now only remains to show that the *expected* running time of EXT also is polynomial. Recall that EXT proceeds by first sampling a view using SIM and then proceeds to extract witnesses in all *accepting* right executions. We show that for every right execution i, the expected running-time needed to extract a witness from that execution is polynomially bounded. Since the number of right interactions is polynomially bounded, we conclude by linearity of expectations that the total expected running time of the combined simulator-extractor $\mathsf{SIM}, \mathsf{EXT}$ is polynomially bounded.

Let $\mathtt{view}_i$ denote the partial view for A in a emulation by SIM up until A is about to start its i^{th} right execution. Let $p_i(\mathtt{view}_i)$ denote the probability that A produces an accepting proof in i^{th} right execution in the simulation by SIM, given that SIM has fed to A the view $\mathtt{view}_i$. Let $p'_i(\mathtt{view}_i)$ denote the probability that A produces an accepting proof in the i^{th} right execution in the simulation by P_i (constructed in EXT), given that EXT has feed A the view $\mathtt{view}_i$.

Sub Claim 5.8 *Let* $\mathtt{view}_i$ *denote the partial view for A in a emulation by* SIM *up until A is about to start its i^{th} right execution. Then,* $p_i(\mathtt{view}_i) = p'_i(\mathtt{view}_i)$.

Proof: The claim follows from the *perfect* indistinguishability of the "standard" simulator used by SIM, and the alternative simulator used by EXT (this is proved similarly to Claim 5.1). ■

Note that if we only assume that $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$ is statistical zero-knowledge, we could only conclude that $p'_i(\mathtt{view_i})$ is *negligibly close* to $p_i(\mathtt{view_i})$. This would not be sufficient to bound the running-time of the simulator (as this would have introduced difficulties similar to the ones discussed in [20]).

By the POK property of $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$ it holds that for any partial view $\mathtt{view}_i$ up until A is about to start its i^{th} right execution, the expected running-time of the extractor is bounded by

$$\frac{\text{poly}(n)}{p'_i(\mathtt{view}_i)}$$

Since the probability of invoking the extraction procedure given this partial view is $p_i(\mathtt{view}_i)$, the expected number of steps used to extract a witness is[12]

$$p_i(\mathtt{view_i})\frac{\text{poly}(n)}{p'_i(\mathtt{view_i})} = p_i(\mathtt{view_i})\frac{\text{poly}(n)}{p_i(\mathtt{view_i})} = \text{poly}(n)$$

We conclude that the expected time needed to extract the witness in the i^{th} right execution is polynomially bounded. The claim follows. ■

5.5 Constructing a Family of 2^n Protocols

Relying on the proof from Section 5.3 we now argue that the family $\{\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle\}_{\text{TAG}\in\{0,1\}^n}$ is also one-many simulation extractable. The key for demonstrating this is to show that the protocols $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$ are simulation-extractable as long as there is at most a *bounded* number of left interactions (as opposed to one as in Def. 4.1), and an unbounded number of right interactions.

[12] It is here that complications arise in the case when $p'_i \neq p_i$. Note that the expected number of steps is no longer guaranteed to be polynomial in this case, *even* if p'_i is negligibly close to p_i.

Specifically, consider a man-in-the middle adversary A that is simultaneously participating in k left interactions of $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$, acting as verifier, and an (unbounded) polynomial number of right-interactions of $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$, acting as prover. Let $\mathsf{view}_A(x, z, \mathsf{tag})$ denote the view of $A(x, z)$ when receiving left-proofs of statements $\bar{x} = x_1, \ldots, x_k$, using identity strings $\bar{\mathsf{tag}} = \mathsf{tag}_1, \ldots, \mathsf{tag}_k$, and proving statements of its choice in the right interaction (using tags of its choice).

Definition 5.9 (Bounded-many Simulation-extractability) *A family $\{\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle\}_{\text{TAG}\in[n]}$, is said to be k-*bounded-many simulation extractable *if for any polynomial $p(\cdot)$ and any man-in-the-middle adversary A that participates in k left interactions and at most $m = p(n)$ right interactions, there exists a probabilistic polynomial time, $S = (\mathsf{SIM}, \mathsf{EXT})$ such that for every $\bar{\mathsf{tag}} \in [n]^k$, every $\bar{x} \in L^k$ and every auxiliary input $z \in \{0,1\}^*$, the following holds:*

1. $\mathsf{SIM}(\bar{x}, z, \bar{\mathsf{tag}})$ *is statistically indistinguishable from* $\mathsf{view}_A(\bar{x}, z, \bar{\mathsf{tag}})$.
2. $\mathsf{EXT}(\bar{x}, z, \bar{\mathsf{tag}})$ *contains witnesses w_i for all statements x_i proved in accepting right sessions in the view $\mathsf{SIM}(\bar{x}, z, \bar{\mathsf{tag}})$, for which $\tilde{\mathsf{tag}}_i \neq \mathsf{tag}_j$ for all j (where $\tilde{\mathsf{tag}}_i$ is the identity string used in the i^{th} session in the right hand side view of $\mathsf{SIM}(\bar{x}, z, \bar{\mathsf{tag}})$).*

Lemma 5.10 *Suppose that* **Com** *are perfectly 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 witness independent UARG, and that $\ell(n) \geq 2n^3 + n$. Then, $\{\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle\}_{\mathsf{tag}\in[n]}$ is n-bounded-many simulation extractable.*

Proof: The proof is essentially identical to the proof of one-many simulation-extractability of $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$. The only difference is that in the simulation by SIM (and EXT), the message r_1 in the i^{th} left execution can no longer be computed as $\Pi(h, c_1)$, but in fact it is defined as $\Pi(M)$ where M denotes all left-hand side prover messages that have occurred before r_1. This creates a potential problem when simulating the Stage 2 messages in the j^{th} left protocol.

The key observation is that the *total* length of all prover messages on the left interaction does not exceed $2n^3$ (here we assume w.l.o.g. that the length of all prover messages in a session is upper bounded by n^2). Thus SIM (as well as EXT) can include *all* left-hand side prover messages sent to A before the message r_1 (or r_2 depending on what "slot" the simulator uses) as part of the witness for either $\langle h, c_1, r_1\rangle \in L_{\mathsf{sim}}$ or $\langle h, c_2, r_2\rangle \in L_{\mathsf{sim}}$. ■

Corollary 5.11 *Suppose that* **Com** *are perfectly 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 witness independent UARG, and that $\ell(n) \geq 2n^3 + n$. Then, $\{\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle\}_{\text{TAG}\in\{0,1\}^n}$ is one-many simulation extractable.*

Proof: Consider a man-in-the-middle adversary that is verifying a statement x with identity string $\text{TAG} = \text{TAG}_1, \ldots, \text{TAG}_n$ in the left interaction while proving m statements $\tilde{x}_1, \ldots, \tilde{x}_m$ in the right interaction, where for $i \in [m]$ the i^{th} right session has identity string $\tilde{\text{TAG}}^i = \tilde{\text{TAG}}^i_1, \ldots, \tilde{\text{TAG}}^i_n$. We show how to construct a simulator-extractor $S = (\mathsf{SIM}, \mathsf{EXT})$ that simulates the view of A while extracting all the witnesses for statements $\tilde{x}_i$ for which $\tilde{\text{TAG}}^i \neq \text{TAG}$.

Observe that for any $i \in [m]$ so that $\tilde{\text{TAG}}^i \neq \text{TAG}$, there exist $i_0 \in [n]$ for which $(i_0, \tilde{\text{TAG}}^i_{i_0}) \neq (j, \text{TAG}_j)$ for all $j \in [n]$ (just take the i_0 for which $\tilde{\text{TAG}}^i_{i_0} \neq \text{TAG}_{i_0}$). Let $\tilde{\mathsf{tag}}_i = (i_0, \tilde{\text{TAG}}^i_{i_0})$.

Given a one-many adversary A for $\langle P_{\text{TAG}}, V_{\text{TAG}}\rangle$, it is then possible to construct an n-many adversary A' for $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$ that runs n parallel sessions in the left interaction and mn concurrent sessions in the right interaction. The inputs and identity strings for the various sessions are defined as follows:

Left sessions. For $j \in [n]$ the common input of the j^{th} left session is $x_j = x$ and the identity string is $\bar{\mathsf{tag}} = (j, \text{TAG}_j)$.

Right sessions. For $(i, j) \in [m] \times [n]$, the input to the $(i, j)^{\text{th}}$ right session is $\tilde{x}_j$ and the identity string is $\tilde{\mathsf{tag}}^i_j = (j, \tilde{\text{TAG}}^i_j)$.

By Lemma 5.10 there exists a simulator $S' = (\mathsf{SIM}', \mathsf{EXT}')$ that produces a view that is statistically close to the real view of A', and outputs witnesses to all right executions for which the tag is different from all of $(1, \text{TAG}_1), \ldots, (n, \text{TAG}_n)$. As observed above, and by construction of A', for any $i \in [m]$ so that $\text{TAG}^i \neq \text{TAG}$, the identity tag_i that A' uses in the proof of the i^{th} right interaction is different than *all* identities $(1, \text{TAG}_1), \ldots, (n, \text{TAG}_n)$ used in the n left interactions. Thus, for every $i \in [m]$ so that $\text{TAG}^i \neq \text{TAG}$, the procedure EXT will successfully extract a witness for the statement x_i. The extractor EXT will output the extracted witnesses for all such i's. Finally, observe that the output of SIM' actually consists of a simulated view of A. Hence, we can define the output of SIM to simply the messages generated by SIM' and output them as the view of A. ■

6 Acknowledgments

We are grateful to Silvio Micali and Moni Naor for many helpful discussions and encouragement.

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*, pages 345-355, 2002.

[3] B. Barak, R. Canetti, Y. Lindell, R. Pass and T. Rabin. Secure Computation Without Authentication. In *CRYPTO 2005*.

[4] 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.

[5] B. Barak and O. Goldreich. Universal Arguments and their Applications. *17th CCC*, pages 194–203, 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] R. Canetti and M. Fischlin. Universally Composable Commitments. In *CRYPTO 2001*, pages 19–40, 2001.

[9] R. Canetti, S. Halevi, J. Katz, Y. Lindell, P. D. MacKenzie. Universally Composable Password-Based Key Exchange. In *EUROCRYPT 2005*, pages 404-421, 2005.

[10] I. Damgård and J. Groth. Non-interactive and Reusable Non-Malleable Commitment Schemes. In *35th STOC*, pages 426-437, 2003.

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

[12] 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.

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

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

[15] 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

[16] 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.

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

[18] M. Fischlin and R. Fischlin. Efficient Non-malleable Commitment Schemes. *In CRYPTO 2000*, pages 413-431, 2000.

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

[20] 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.

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

[22] 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.

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

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

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

[26] 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.

[27] 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.

[28] J. Katz, R. Ostrovsky, M. Yung. Efficient Password-Authenticated Key Exchange Using Human-Memorable Passwords. In *EUROCRYPT 2001*, pages 475-494, 2001.

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

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

[33] 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.

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

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

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

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

[38] R. Pass and A. Rosen. New and Improved Constructions of Non-Malleable Cryptographic Protocols. In *37th STOC*, 2004, pages 533-542, 2005.

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

Appendix

A The "Special-Purpose" Universal Argument

We construct a special-purpose $WIUARG$. Using this $WIUARG$ in the protocol $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$, will allow us to obtain a protocol $\langle \widehat{P}_{\mathsf{tag}}, \widehat{V}_{\mathsf{tag}}\rangle$ with the following two extra properties:

- $\langle \widehat{P}_{\mathsf{tag}}, \widehat{V}_{\mathsf{tag}}\rangle$ is a proof of knowledge (while $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$ only is a weak proof of knowledge).[13]
- $\langle \widehat{P}_{\mathsf{tag}}, \widehat{V}_{\mathsf{tag}}\rangle$ is perfect $\mathcal{ZK}$ (while $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$ only is computational $\mathcal{ZK}$). We note that in order to achieve this we also need to instantiate the commitment scheme used in $\langle P_{\mathsf{tag}}, V_{\mathsf{tag}}\rangle$ with a *perfectly hiding* commitment scheme.

Both these properties will be useful to us when using the zero-knowledge protocol as part of other protocols. Let **Com** be a perfectly hiding commitment scheme. The specialized $WIUARG$ is depicted in Figure 11.

Protocol - Specialized-$WIUARG$

Parameters: Security parameter 1^k.

Common Input: $x \in \{0,1\}^n$, and $\langle h, c_1, c_2, r_1, r_2\rangle \in \{0,1\}^{\mathrm{poly}(k)} \times \{0,1\}^{\mathrm{poly}(k)} \times \{0,1\}^{\mathrm{poly}(k)}$.

Stage 1 (Encrypted UARG):

$V \leftrightarrow P$: Send $\alpha \xleftarrow{\mathrm{R}} \{0,1\}^k$.

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

$V \leftrightarrow P$: Send $\gamma \xleftarrow{\mathrm{R}} \{0,1\}^k$.

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

Stage 2 (Body of the proof):

$P \leftrightarrow V$: A witness independent POK proving the OR of the following two statements:

1. There exists $w \in \{0,1\}^{\mathrm{poly}(|x|)}$ so that $R_L(x, w) = 1$.
2. There 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 the $UARG$ proving the statement:
     − there exists a tuple $\langle i, \Pi, y, s\rangle$ so that $R_{\mathsf{Sim}}(\langle h, c_i, r_i\rangle, \langle \Pi, y, s\rangle) = 1$

Figure 11: The "special-purpose"-$WIUARG$

[13] In a proof of knowledge the extractor is supposed to succeed with probability negligibly close to the success probability of the prover. In a weak proof of knowledge, on the other hand, it is sufficient that the extractor succeeds with polynomially related probability