Concurrent Non-Malleable Commitments*

Rafael Pass
CSAIL, MIT, Cambridge, MA
pass@csail.mit.edu

Alon Rosen†
DEAS, Harvard, Cambridge, MA
alon@eecs.harvard.edu

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 *collision resistant hash functions 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* [12]. 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 [12]. 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 [32].

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 [12]) 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 [12], 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 [12], Section 3.4.1 for an example in

*This paper appeared in the *46'th FOCS*, 2005.

†Work done while at CSAIL, MIT, Cambridge, MA.

the context of non-malleable encryption). Resolving this issue has been stated as a major open problem in [12].

Partially addressing this issue, some works have demonstrated the existence of commitment schemes that remain non-malleable under *bounded concurrent* composition [30, 3]. 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 solutions is that they require 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 these protocols has to grow linearly with the number of executions. Thus, from both a theoretical and a practical point of view, these protocols are 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 collision resistant hash functions. 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 (unbounded) concurrency without having to rely on set-up assumptions.

Additional contributions. Our proof also yields the first commitment scheme that is *strictly* non-malleable with respect to commitment.[2] 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 [12].

Our definitions of non-malleable commitments are somewhat different (stronger) than the ones appearing in the DDN paper [12]. 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 wish to 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 [32]. 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 [30, 32]. 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. [25]).)

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

[1] In a different variant, called non-malleable commitment *with respect to opening* [15], 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] This should not be confused with a previous result showing the existence of commitment schemes that are strictly non-malleability with respect to *opening* [32].

existence of a common reference string [15, 8], as well as non-malleable commitment schemes and non-interactive non-malleable $\mathcal{ZK}$ protocols assuming the existence of a common random string [11, 10, 9].

Several of the above works explicitly address the issue of multiple executions of non-malleable schemes [9, 8, 6] (also called *reusability* in the terminology of [8]). Perhaps most notable amongst the works addressing concurrency, is the one on Universally composable commitments [6]. Universal composability implies concurrent non-malleability. However, it is impossible to construct universally composable commitments without making set-up assumptions [6].

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 [18, 29, 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. [18]), but their treatment is limited to the case of sequential composition. A treatment of the full concurrent case appears in [23] (see also [7, 3]), but it relies on the existence of a common reference string.

2 Definitions

We assume familiarity with the standard definitions of zero-knowledge and commitment schemes (see [16]). Our definitions of non-malleability are somewhat stronger that the ones proposed by DDN [12]. Specifically, we formalize the notion of two values being unrelated through the concept of computational indistinguishability (rather than using polynomial time computable relations).

2.1 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).

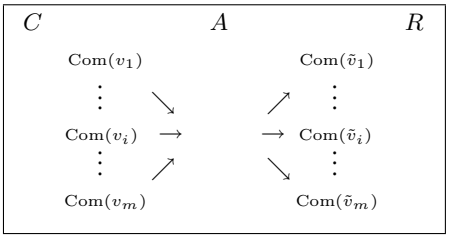

Figure 1. A concurrent man-in-the-middle adversary.

The traditional definition of non-malleable commitments [12] 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 [12], 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.

2.2 Non-Malleability via Indistinguishability

Following the simulation paradigm [21, 22, 19, 20], 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 simu-

lated 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 [12].

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 $\perp$.

2.3 Concurrent Non-Malleable Commitments

Let $\langle C, R \rangle$ be a commitment scheme, and let $n \in N$ be a security parameter. We consider man-in-the-middle adversaries that are simultaneously 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.[3] 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 $\perp$.[4]

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 output of S (which consists of a sequence of values $\tilde{v}_1, \ldots, \tilde{v}_m$).

Definition 2.1 *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:*

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

It can be seen that for $m = 1$ any protocol that satisfies Definition 2.1 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 High-level Structure of the Proof

As mentioned in the introduction, our construction follows the paradigm introduced by Pass and Rosen (PR) for obtaining (single execution) non-malleable commitments [32]. The commit phase of the PR protocol consists of having the sender engage in a (standard) statistically binding commitment with the receiver and thereafter also provide a *non-malleable* zero-knowledge proof of knowledge of the value committed to. The reveal phase consists of sending the de-commitment information of the statistically binding commitment used in the commit phase. Our analysis proceeds as follows.

A simple simulator. We first show that our non-malleable commitment scheme has the property that the

[3] Since we are dealing with statistically binding commitments, $\tilde{v}_1, \ldots, \tilde{v}_m$ are (almost always) well defined.

[4] 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.

left interaction can be simulated by simply (honestly) committing to 0^n. This is in contrast to previous simulators for non-malleable commitments, which invariably relied on the invocation of a *zero-knowledge* simulator. Our new, "simple" simulator, naturally extends to the concurrent setting (whereas simulators that relied on zero-knowledge simulations encountered difficulties). This observation alone allows us to obtain a non-malleable commitment that is secure if there are an *unbounded* number of concurrent *left* interactions, but only one right interaction.

Simulation-extractability. Next, we show that the commitment scheme is non-malleable when there are an *unbounded* number of *right* interactions, but only one left interaction. This is obtained by proving that the zero-knowledge protocols that we use satisfy a, so called, *simulation-extractability* property in the man-in-the-middle setting. By simulation-extractability we mean that there exists a combined simulator-extractor that can simulate both the left and the right interaction for the man-in-the middle adversary, while simultaneously outputting a witness for the statement proved by the adversary in the right interaction. Furthermore, we show that these zero-knowledge protocols also satisfy a "one-many" simulation-extractability property, namely that the simulation-extractability property holds even if there are an *unbounded* number of concurrent *right* interactions (while only one left interaction). The proof of this result relies on a novel *non-black box* simulation argument, which is delicately combined with a *black-box* extraction technique. (We note that our method *inherently* makes use of the "mix" of non-black box and black-box techniques.)

Combining the above ideas. Finally, using a hybrid argument, the above two techniques are combined in order to show that the commitment scheme is non-malleable even when there are an unbounded number of left and right interactions.

4 A Simulation-Extractable $\mathcal{ZK}$ Protocol

The $\mathcal{ZK}$ protocols used in order to enable the above analysis (and ultimately used in order to construct the non-malleable commitment protocol) are constructed in two steps: (1) Construct a family of n zero-knowledge protocols ZK_{tag} that satisfy "one-many" simulation-extractability. It turns out that the protocols of Pass [30] in fact satisfy this property. Our main technical contribution consists of proving this fact. (2) Relying on the technique of Pass and Rosen [32], obtain a family of 2^n protocols which satisfy the same property.

Constructing a family of n protocols. We start by reviewing Pass' protocols, ZK_{tag}, which rely on the zero-knowledge protocol of Barak [1]. (See [1, 30, 32] for a more detailed treatment.)

Let $n \in N$, and let $T : N \rightarrow N$ be a "nice" function that satisfies $T(n) = n^{\omega(1)}$. ZK_{tag} relies on a "special" $\mathbf{NTIME}(T(n))$ relation, which is a a variant of Barak's relation. We start by describing this 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. R_{sim} - A variant of Barak's relation.

Similarly to [1], ZK_{tag} makes use of a witness-indistinguishable universal argument ($WIUARG$) [14, 13, 24, 26, 4]. 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 $\mathsf{tag} \in [n]$. ZK_{tag} is described in Figure 3.

What differentiates between two protocols ZK_{tag} and $ZK_{\tilde{\mathsf{tag}}}$ is the fact that the length of the verifier's messages in ZK_{tag} is a parameter that depends on tag. The length of these verifier messages is also dictated by the parameter $\ell(n)$. In our case $\ell(n) = \ell'(n) + n$, where $\ell'(n)$ upper bounds the total length of all prover and verifier messages in ZK_{tag}, except for the "challenges" r_1, r_2.[5] (Note that although the statement proved in Stage 2 of the protocol actually depends on the challenges r_1, r_2 such an upper bound exists due to the "efficiency" of the $UARG$ used.)

Stand-alone zero-knowledge. Our use of ZK_{tag} relies on the fact that ZK_{tag} is a stand-alone zero-knowledge proof of knowledge.[6] The stand-alone $\mathcal{ZK}$ property of

[5] In [30], $\ell(n)$ was instantiated to a slightly different value in order to guarantee composition with other protocols. In this paper we do not need this additional feature.

[6] We mention that ZK_{tag} 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 [4], it is possible to show how by slightly modifying the relation R_{sim}, one can guarantee soundness under "standard" collision resistance.

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):

$V \rightarrow P$: Send $h \stackrel{R}{\leftarrow} \mathcal{H}_n$.

Stage 1 (Slot 1):

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

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

Stage 1 (Slot 2):

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

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

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

ZK_{tag} follows using essentially the same simulation techniques as Barak's one [1]. The main difference to be taken into consideration is the existence of multiple slots in Stage 1.

We provide a brief sketch of how this simulation, which we refer to as the *stand-alone simulation*, is performed. In Stage 1 of the protocol (i.e., in Slot 1 and 2), the simulator proceeds by committing to the program of the residual verifier. Let Π_1, Π_2 denote the respective programs, and 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 precisely, the simulator uses the tuple $\langle \Pi_1, c_1, s_1\rangle$ 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_1 it holds that $\Pi_1(c_1) = r_1$, and (2) by the choice of $\ell(n)$, for every $\mathsf{tag} \in [m]$, $|r_i| - |c_i| \geq \ell(n) - |c_i| \geq n$. Note that the simulator could have alternatively committed to the program of the verifier in *Slot 2*. This additional feature will be useful to us in Section 4.1.

Proof of knowledge. In this paper, just as in [32], we additionally require that ZK_{tag} is a proof of knowledge. That is, for any prover P^* and for any $x \in \{0,1\}^n$, if P^* convinces the honest verifier V that $x \in L$ with non-negligible probability then one can extract a witness w that satisfies $R_L(x,w) = 1$ in (expected) polynomial time.[7]

[7] We note that the weak proof of knowledge property of a $WIUARG$ is not sufficient for our purposes. To guarantee the "traditional" proof of knowledge property, we have to use a "specialized" version of $WIUARG$s which provides this guarantee (cf. [32]).

4.1 Simulation-Soundness

As shown in [30], the protocols ZK_{tag} are *simulation-sound* (c.f. Sahai [33]) with respect to each other, i.e., a man-in-the-middle adversary that receives a simulated proof of ZK_{tag} (as part of its left interaction) will not be able to break the soundness of the protocol $ZK_{\tilde{\mathsf{tag}}}$ (as part of its right interaction), where $\tilde{\mathsf{tag}} \neq \mathsf{tag}$. We briefly review how this is proved, as this technique will be useful to us. For more details, the reader is referred to [30, 32].

Suppose there exists a man-in-the middle adversary A that manages to violate the soundness of protocol $ZK_{\tilde{\mathsf{tag}}}$, while receiving a simulated proof of ZK_{tag}. We show how to construct a cheating prover P^* for a single instance of $ZK_{\tilde{\mathsf{tag}}}$ by forwarding A's messages in $ZK_{\tilde{\mathsf{tag}}}$ to an external honest verifier V and internally simulating the messages of ZK_{tag} for A. One problem that arises is that the code of the external verifier V is not available to us. This means that the straightforward simulation of the protocol ZK_{tag} cannot be completed as it is, since it explicitly requires possession of a "short" description of the corresponding verifier messages. To overcome this problem we resort to an alternative simulation technique.

The alternative simulator. Note that except for the "long" challenges r_1, r_2 sent by the verifier of $ZK_{\tilde{\mathsf{tag}}}$ we do have a description of all messages sent to the adversary 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 ZK_{tag}. As in [30], we separate between two different schedulings:

There exist one "free" slot in ZK_{tag} in which neither of r_1, r_2 are contained. In this case the "free" slot in ZK_{tag} can be used to perform the straightforward simulation.

The messages r_1, r_2 in $ZK_{\tilde{\mathsf{tag}}}$ occur in slot 1, 2 respectively in ZK_{tag}. By the construction of the protocols it follows that the length of either the first or the second challenge in $ZK_{\tilde{\mathsf{tag}}}$ is at least $\ell(n)$ bits longer than the corresponding challenge in ZK_{tag}. Thus there exist a slot in ZK_{tag} such that even if we include the verifier's challenge from

the protocol $ZK_{\tilde{\mathsf{tag}}}$ in the description, we still have $\ell(n) - n$ bits to describe the other messages.

Bounded-concurrent simulation-soundness. As is shown in [30], the protocols ZK_{tag} 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.[8] As in [32], this additional property is useful to us in order to construct a family of 2^n protocols (see Section 4.3 and [32] for more details).

4.2 Simulation-Extractability

Recall that simulation-extractability means that there exists a combined "simulator-extractor" that is able to simulate both the left and the right interaction for a man-in-the-middle adversary, while simultaneously extracting a witness to the statement proved in the right interaction. It has already been shown in [32] that the family ZK_{tag} is simulation extractable. We show that this family satisfies an even stronger property: It is simulation-extractable even when the adversary participates in an *unbounded* number of concurrent *right*-interactions (while only receiving one left interaction).

Let A be a man-in-the middle adversary that is simultaneously participating in one left interaction of ZK_{tag}, acting as verifier, and an (unbounded) polynomial number of right-interactions of ZK_{tag}, acting as prover. Let $\mathsf{view}_A(x, z, \mathsf{tag})$ denote the view of $A(x, z)$ when receiving a left-proof of the statement x, using tag tag, and giving right-proofs of statements of its choice and using tags of its choice.

Lemma 4.1 ("One-many" Simulation-extractability) *Let A be defined as above. Then, there exists a combined "simulator-extractor", S such that for every $\mathsf{tag} \in [n]$, every $x \in L$ and every auxiliary input $z \in \{0,1\}^*$, the following holds:*

1. *The first output of $S(x, z)$ is statistically indistinguishable from $\mathsf{view}_A(x, z, \mathsf{tag})$.*
2. *The second output of $S(x, z)$ contains witnesses for all statements proved in accepting right interactions in the view described by the first output of $S(x, z)$, which use tags $\tilde{\mathsf{tag}} \neq \mathsf{tag}$ (i.e., which use a different tag than the left interaction).*

Proof Sketch: To simplify the proof we assume that ZK_{tag} is *perfect* zero-knowledge. This extra assumption will be lifted at the end of the proof.

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

Consider a man-in-the-middle adversary A. Let k denote the number of rounds in ZK_{tag}, and let m be an upper-bound on the number of right interactions that A participates in. We describe a combined simulator-extractor S which proceeds as follows:

Simulation of view. We start by describing a machine SIM that simulates the view of A. This implies simulating all the left and the right interactions for A. Note that in the right interactions SIM is supposed to act as a verifier and can therefore "emulate" those executions by acting as the honest verifier. In the left interaction, on the other hand, SIM is supposed to act as a prover and has to perform a "real" simulation. Towards this goal, SIM performs the following modified version of the stand-alone simulator.

(1) Pick random coins $\bar{r} = (r_{1,1}, r_{1,2}, .., r_{1,k})$, .., $(r_{m,1}, r_{m,2}, .., r_{m,k})$ for emulating the honest verifiers in the right interactions. The coins $r_{i,j}$ are used in the j^{th} message in the i^{th} interaction.

(2) Messages in the right interactions are then emulated by playing the role of the honest verifier with the fixed random coins $\bar{r}$. 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.

(3) The left interaction is simulated as follows. SIM considers the execution of A and the emulation of the right interactions (with the fixed random coins $\bar{r}$) as a *stand-alone* verifier and applies a close variant of the stand-alone simulator in order to simulate the left interaction. Let $\Pi(\cdot)$ denote the joint code of A and the emulation of the right interactions (including the coins $\bar{r}$). Whereas the stand-alone simulator would have committed to $\Pi(\cdot)$, we instead let SIM commit to the program $\Pi'(i, \cdot)$ defined as follows: it executes $\Pi(\cdot)$ if $i = 0$ and otherwise executes the same code as $\Pi(\cdot)$ with the exception that messages in the i^{th} right interaction are not emulated, rather $\Pi(i, \cdot)$ will expect to receive these (external) messages as input.

Thereafter, SIM proceeds exactly as the stand-alone simulator, by additionally letting the input i to Π' be set to 0, in stage 2 of the protocol. Note that the additional input i to Π' is thus not used in the simulation. However, the possibility of using this additional input will facilitate the task of extracting witnesses.

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 [25], combined with the alternative simulator technique described in Section 4.1. In a sense, this can be seen as a (non-trivial) extension of the method of Pass and Rosen [32] (which was used to show

a similar property for the significantly simpler case of only one right interaction).

More precisely, we describe a machine EXT that proceeds as follows. EXT fixes the random coins of the simulator SIM and iteratively extracts witnesses for each of the right interactions. More precisely, for each $i \in [m]$ such that the i^{th} right interaction was accepting in the simulation by SIM (with fixed random coins), and the tag of the i^{th} interaction is different from the tag of the left interaction, perform the following steps (note that otherwise we do not need to extract a witness):

Construct a stand-alone prover P_i for the i^{th} right interaction as follows. Perform the same simulation of the left and the right interactions for A as was done by SIM (using the same fixed random coins), except for the following differences:

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

(2) In the simulation of the left protocol, use the *alternative* simulator from Section 4.1 in order to complete stage 2 of the protocol. Note that the *stand-alone* simulator no longer can be used since the simulator has not (and cannot) commit to the external messages for the i^{th} left interaction. On the other hand, since there is only *one* external interaction the alternative simulator will succeed. The only change needed to the alternative simulator is that we additionally provide the input i to Π', in order to let Π' depend on the external messages in interaction i (since this input is "short" the simulation still succeeds).

We can now apply the (stand-alone) extractor, guaranteed by the proof of knowledge property of ZK_{tag}, to P_i in order to extract a witness. In order to simplify the analysis, we consider an extractor that proceeds by feeding (truly) random coins to P_i until it obtains another accepting transcript using the same prefix (note that we only start the extraction in the case that the simulation by SIM resulted in an accepting transcript).

In the unlikely event when we obtain two accepting transcripts using the *same* random challenges (in Stage 2 of the protocol), the extractor outputs `fail` (note that this only happens if the coins sent by the extractor are the same as the coins sent by SIM).

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

Correctness of the simulation-extraction. A proof of the correctness of the above simulator S can be found in the full version. There it is also shown how the above (simplified) combined simulator-extractor can be modified to work even if only assuming that ZK_{tag} is statistical zero-knowledge and not perfect zero-knowledge. We hint that this is done by relying on a variant of the sampling technique from [17]. ■

4.3 Constructing a Family of 2^n Protocols

We show how to extend the family of n protocols $\{ZK_{\mathsf{tag}}\}_{\mathsf{tag}\in[n]}$ into a family of 2^n protocols that satisfy the one-many simulation-extractability property. One approach for obtaining this is to rely on an "n-slot" version of $\{ZK_{\mathsf{tag}}\}$ (i.e., using n "slots" instead of only 2, see [30] for more details). The proof of one-many simulation-extractability from Section 4.2 directly extends to this new (non-constant round) protocol.

Pass and Rosen [32], instead show how to obtain a family of 2^n *constant-round* protocols by running n parallel executions of ZK_{tag}, using appropriately chosen tags. This new family of protocols is denoted $\{nmZK_{tag}\}_{tag\in\{0,1\}^n}$ and is depicted in Figure 4. In the full version of the paper, we show how to modify the combined simulator-extractor from section 4.2 to prove that $nmZK_{tag}$ is one-many simulation-extractable. Just as in [32], we here rely on the bounded-concurrent security of $\{ZK_{\mathsf{tag}}\}$.[9]

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

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

Tag String: $tag \in \{0,1\}^n$. Let $tag = tag_1, \ldots, tag_n$.

The protocol:

$P \leftrightarrow V$: For all $i \in \{1, \ldots, n\}$ (in parallel):
Run $ZK_{(i,\, tag_i)}$ 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 – $nmZK_{tag}$.

5 Non-Malleable Commitments

Using $nmZK_{tag}$ as a subroutine, we present a construction of concurrent non-malleable commitments. Let Com be a statistically binding commitment scheme (for simplicity, assume that Com is non-interactive), and let $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ be a one-time signature scheme secure against a chosen-message attack. Consider the following protocol for non-malleable commitments (which is a variant of the non-malleable commitment of Pass and Rosen [32]).[10]

[9] In fact, since the protocols ZK_{tag} are all run in parallel in $nmZK_{tag}$, "bounded-parallel" security is here sufficient.

[10] The difference between the this protocol and the protocol of [32] is that here we also employ a signature scheme. We note that the important difference, nevertheless, lies in the analysis of the protocol.

Security Parameter: 1^k.

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

Commit Phase:

$C \rightarrow R$: Let $vk, sk \leftarrow \mathsf{Gen}(1^k)$. Pick $s \in \{0,1\}^n$. Send $c = \mathsf{Com}(v;s), vk$.

$C \leftrightarrow R$: Prove using $nmZK_{vk}$ that there exist $v, s \in \{0,1\}^k$ so that $c = \mathsf{Com}(v;s)$.

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

R: Verify that $nmZK_{vk}$ is accepting and that $\mathsf{Verify}(vk, T, \sigma) = 1$.

Reveal Phase:

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

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

Figure 5. Conc. non-malleable commitment - $nm\mathcal{C}$.

The statistical binding property of $nm\mathcal{C}$ follows 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 $nmZK_{tag}$ (see [32] of more details). It remains to show that $nm\mathcal{C}$ is concurrently non-malleable.

Theorem 5.1 *$nm\mathcal{C}$ is concurrently non-malleable.*

Proof: Consider a man-in-the-middle adversary A that, given access to m commitments to the values $v_1, .., v_m$, succeeds in committing to values $\tilde{v}_1, ..., \tilde{v}_m$. We show that for every such adversary A, there exists a simulator S (which only participates in right interactions), that manages to commit to values that are indistinguishable from $\tilde{v}_1, ..., \tilde{v}_m$. The simulator S incorporates A, and internally emulates the left interactions for A by simply *honestly* committing to the string 0^n. We show that the values that S commits to[11] are indistinguishable from the values that A commits to.[12]

Suppose, for contradiction, that this is not the case. Using a hybrid argument, this means that there exists an i such that the values that the machines H_i and H_{i+1} commit to are distinguishable, where the machines H_k for $k \in [m]$ are defined as follows: H_k proceeds as S except that it emulates the j^{th} left interaction for A by committing to v_j, if $j \leq k$, and 0^n otherwise. We show that this contradicts the "one-many" simulation-extractability property of $nmZK_{tag}$.

[11] In case that the commitment Com can be opened to two different strings, we set the value of this commitment to $\perp$.

[12] What we actually show is a somewhat stronger property than non-malleability. Namely, that the values that the man-in-the-middle adversary commits to are (computationally) independent of the values he receives commitments to.

Towards this goal, we define two intermediate hybrid machines $\tilde{H}_i, \tilde{H}_{i+1}$. First, consider $\tilde{H}_i$, which proceeds just as H_i except that it executes the combined simulator-extractor in order to simulate $nmZK_{tag}$ in the i^{th} left interaction. (Note that this is possible since we are only simulating *one* left interaction. All the other left interactions are emulated internally by committing honestly.) We show the following two claims.

Claim 5.2 *The values committed to by $\tilde{H}_i$ are statistically close to the values committed to by H_i.*

Proof Sketch: The claim follows from the *statistical* indistinguishability property of the combined simulator-extractor which is used by $\tilde{H}_i$ (see [32] for a more formal proof of a similar statement). Note that since the values committed to are not efficiently computable from the transcript, we need to rely on the *statistical* indistinguishability of the simulation, even to guarantee computational indistinguishability of these values. (Indeed it would have been sufficient for the rest of the proof to simply guarantee computational indistinguishability of the values committed to.) ■

Claim 5.3 *Except with negligible probability, $\tilde{H}_i$ also outputs the values committed to (instead of only the commitments to these).*

Proof Sketch: The claim, roughly speaking, follows from the correctness of the combined simulation-extraction used by $\tilde{H}_i$. More precisely, 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 $\perp$).

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.[13] ■

Now, define the machine $\tilde{H}_{i+1}$ which proceeds just as $\tilde{H}_i$ except that $\tilde{H}_{i+1}$ internally emulates the i^{th} left-interaction by sending a commitment (using Com) to 0^n in the first message of $nm\mathcal{C}$ (instead of v_i as $\tilde{H}_i$ would), before executing the combined simulator-extractor. It

[13] 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).

follows from the computationally hiding property of Com that the values output by $\tilde{H}_i$ and $\tilde{H}_{i+1}$ are computationally indistinguishable.[14] (Recall that both $\tilde{H}_i$ and $\tilde{H}_{i+1}$, not only output commitments, but also the actual values that these are commitments to.)

Finally, it follows from the statistical indistinguishability property of (the combined simulator-extractor for) $nmZK_{tag}$ that the values committed to by $\tilde{H}_{i+1}$ (which in turn are statistically close to the values output by $\tilde{H}_{i+1}$) are indistinguishable from the values committed to by H_{i+1}. We conclude that the values committed to by H_i and H_{i+1} are indistinguishable, which contradicts our assumptions. ■

Acknowledgments. We are grateful to Silvio Micali 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] B. Barak and O. Goldreich. Universal Arguments and their Applications. *17th CCC*, pages 194–203, 2002.

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

[6] R. Canetti and M. Fischlin. Universally Composable Commitments. In *CRYPTO 2001*, pages 19–40, 2001.

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

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

[9] 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.

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

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

[12] 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

[13] 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.

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

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

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

[17] 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.

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

[19] 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.

[20] 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.

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

[22] 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.

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

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

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

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

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

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

[14] Otherwise we could distinguish between commitments (using Com) to 0^n and v_i.

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

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

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

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

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