Concurrent Non-malleable Commitments from Any One-Way Function

Huijia Lin, Rafael Pass,
and Muthuramakrishnan Venkitasubramaniam

Cornell University
{huijia,rafael,vmuthu}@cs.cornell.edu

Abstract. We show the existence of concurrent non-malleable commitments based on the existence of one-way functions. Our proof of security only requires the use of black-box techniques, and additionally provides an arguably simplified proof of the existence of even stand-alone secure non-malleable commitments.

1 Introduction

Often described as the "digital" analogue of sealed envelopes, commitment schemes enable a *sender* to commit itself to a value while keeping it secret from the *receiver*. For some applications, however, the most basic security guarantees of commitments are not sufficient. For instance, the basic definition of commitments does not rule out an attack where 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. In order to address the above concerns, Dolev, Dwork and Naor (DDN) introduced the concept of *non-malleable commitments* [6]. Loosely speaking, a commitment scheme is said to be non-malleable if it is infeasible for an adversary to "maul" a commitment to a value v into a commitment of a related value $\tilde{v}$.

The first non-malleable commitment protocol was constructed by Dolev, Dwork and Naor [6]. 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 the length of party identifiers (or alternatively, a security parameter). A more recent result by Barak presents a constant-round protocol for non-malleable commitments 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, assuming only collision resistant hash function secure against polynomial sized circuits [12].

R. Canetti (Ed.): TCC 2008, LNCS 4948, pp. 571–588, 2008.

1.1 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 [6]) 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 [6], 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 creating *joint* dependencies between more than a single individual pair (see [6], 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 [6].

Partially addressing this issue, Pass demonstrated the existence of commitment schemes that remain non-malleable under *bounded concurrent* composition [10]. 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. More recently, Pass and Rosen [12] constructed a commitment scheme that remains non-malleable also under an unbounded number of concurrent executions. Their construction uses only a constant number of rounds and is based on the existence of (certified) claw-free permutations. The protocol—which is a variant of the protocol of [11]—relies on the message-length technique of [10], which in turn relies on the non-black box zero-knowledge protocol of Barak [1]. As such, it seems that practical implementations of this approach currently are not within reach.

In contrast, the original construction of Dolev, Dwork and Naor (which is only stand-alone secure) relied on the minimal assumption of one-way functions and had a black-box security proof. Natural questions left open are thus:

> *Can concurrent non-malleable commitments be based solely on the existence of one-way functions?*
>
> *Does there exist concurrent non-malleable commitments with black-box proofs of security?*

A partial answer to the second question was provided by Pass and Vaikuntanathan [13], demonstrating the existence of concurrent non-malleable commitments with black-box security proofs; their construction, however, relies on a new (and non-standard) hardness assumption with a strong non-malleability flavor.[1]

1.2 Our Results

In this work, we fully resolve both of the above questions. Namely, we show the following theorem using only black-box techniques.

[1] More precisely, they assume the existence of, so called, *adaptive one-way permutations*—namely permutations which remain one-way even when the adversary has access to an inversion oracle.

Main Theorem If one-way functions exist, then there exists a statistically-binding commitment scheme that is concurrent non-malleable.

Our protocol, which is a variant of the protocol of [6] (and in particular relies on the same scheduling techniques as in [6]), uses $O(n)$ number of communication rounds. Moreover, it seems that by relying on specific (number theoretic) hardness assumptions (and appropriate Σ-protocols [4]), one can obtain an "implementable" instantiation of our protocol (without going through Cook's reductions).

Additional results. All previous constructions of non-malleable commitments require complex and subtle proofs. As an additional contribution, our protocol and its proof provide the arguably simplest proof of existence of non-malleable commitments (let alone the question of concurrency); more precisely, it provides a new (and arguably simpler) proof of the feasibility result of [6].

Furthermore, by relying on the concurrent security of our protocol, we also obtain a simple (and self-contained) proof of the existence of $\log n$-round (stand-alone secure) non-malleable commitment schemes based on only the existence of one-way functions. As far as we know, a complete proof of this statement (which appeared only with a proof sketch in [6]) has never appeared before.

Finally, we mention that our protocols satisfy a notion of non-malleability called *strict* (as opposed to *liberal*) non-malleability—this notion, which was defined (but not achieved) in [6], requires simulation to be performed by a strict polynomial-time machine (as opposed to an expected polynomial-time machine). Our results provide the first construction of strictly non-malleable commitments based on one-way functions, or using a black-box security proof.

1.3 Overview

Section 2 contains basic notation and definitions of commitment schemes and concurrent non-malleability. In Section 3, we present our $O(n)$-round commitment scheme, and in Section 4, we prove that the commitment scheme is concurrent non-malleable. In Section 5, we additionally provide the construction of a $O(\log n)$-round (stand-alone secure) non-malleable commitment scheme based on any $O(n)$-round concurrent non-malleable commitment scheme.

2 Definitions and Notations

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 $\nu(n) < n^{-c}$.

2.1 Witness Relations

We recall the definition of a witness relation for an $\mathcal{NP}$ language [8].

Definition 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 $L \in \mathcal{NP}$.

2.2 Interactive Proofs

We use the standard definitions of interactive proofs (and interactive Turing machines) [9] and arguments (a.k.a computationally-sound proofs) [3]. 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 (Interactive Proof System). *A pair of interactive machines $\langle P, V \rangle$ is called an interactive proof system for a language L if for every probabilistic polynomial time machine (PPT) V there is a negligible function $\nu(\cdot)$ such that the following two conditions hold :*

- Completeness: *For every $x \in L$,* $\Pr[\langle P, V \rangle(x) = 1] = 1$
- Soundness: *For every $x \notin L$, and every interactive machine B,* $\Pr[\langle B, V \rangle(x) = 1] \leq \frac{1}{\nu(|x|)}$

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.*

Special-sound proofs. A 3-round public-coin interactive proof for the language $L \in \mathcal{NP}$ with witness relation R_L is special-sound with respect to R_L, if for any two transcripts (α, β, γ) and $(\alpha', \beta', \gamma')$ such that the initial messages α, α' are the same but the challenges β, β' are different, there is a deterministic procedure to extract the witness from the two transcripts and runs in polynomial time. Special-sound $\mathcal{WI}$ proofs for languages in $\mathcal{NP}$ can be based on the existence of non-interactive commitment schemes, which in turn can be based on one-way permutations. Assuming only one-way functions, 4-round special-sound $\mathcal{WI}$ proofs for $\mathcal{NP}$ exist.[2] For simplicity, we use 3-round special-sound proofs in our protocol though our proof works also with 4-round proofs.

[2] A 4-round protocol is special sound if a witness can be extracted from any two transcripts $(\tau, \alpha, \beta, \gamma)$ and $(\tau', \alpha', \beta', \gamma')$ such that $\tau = \tau$,$\alpha = \alpha'$ and $\beta \neq \beta'$.

2.3 Indistinguishability

Definition 3 ((Computational) Indistinguishability). *Let X and Y be countable sets. Two ensembles $\{A_{x,y}\}_{x \in X, y \in Y}$ and $\{B_{x,y}\}_{x \in X, y \in Y}$ are said to be* computationally indistinguishable over *$x \in X$, if for every probabilistic "distinguishing" machine D whose running time is polynomial in its first input, there exists a negligible function $\nu(\cdot)$ so that for every $x \in X, y \in Y$:*

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

2.4 Witness Indistinguishability

An interactive proof is said to be *witness indistinguishable* ($\mathcal{WI}$) if the verifier's output is "computationally independent" of the witness used by the prover for proving the statement. In this context, we focus on 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 output is computationally independent of the witness, we mean that for any two possible $\mathcal{NP}$-witnesses that could be used by the prover to prove the statement $x \in L$, the corresponding outputs are computationally indistinguishable.

Definition 4 (Witness-indistinguishability). *Let $\langle P, V\rangle$ be an interactive proof system for a language $L \in \mathcal{NP}$. We say that $\langle P, V\rangle$ is* witness-indistinguishable *for R_L, if for every probabilistic polynomial-time interactive machine V^* and for every two sequences $\{w_x^1\}_{x \in L}$ and $\{w_x^2\}_{x \in L}$, such that $w_x^1, w_x^2 \in R_L(x)$ for every $x \in L$, the probability ensembles $\{\langle P(w_x^1), V^*(z)\rangle(x)\}_{x \in L, z \in \{0,1\}^*}$ and $\{\langle P(w_x^2), V^*(z)\rangle(x)\}_{x \in L, z \in \{0,1\}^*}$ are computationally indistinguishable over $x \in L$.*

2.5 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. In this work, we consider commitment schemes that are statistically-binding, namely while the hiding property only holds against computationally bounded (non-uniform) adversaries, the binding property is required to hold against unbounded adversaries. More precisely, a pair of PPT machines $\langle C, R\rangle$ is said to be a commitment scheme if the following two properties hold.

Computational hiding: For every (expected) PPT machine R^*, it holds that, the following ensembles are computationally indistinguishable over $\{0,1\}^n$.
- $\{\mathsf{sta}^{R^*}_{\langle C,R\rangle}(v_1, z)\}_{v_1, v_2 \in \{0,1\}^n, n \in N, z \in \{0,1\}^*}$
- $\{\mathsf{sta}^{R^*}_{\langle C,R\rangle}(v_2, z)\}_{v_1, v_2 \in \{0,1\}^n, n \in N, z \in \{0,1\}^*}$

where $\mathsf{sta}^{R^*}_{\langle C,R\rangle}(v,z)$ denotes the random variable describing the output of R^* after receiving a commitment to v using $\langle C,R\rangle$.

Statistical binding: Informally, the statistical-binding property asserts that, with overwhelming probability over the coin-tosses of the receiver R, the transcript of the interaction fully determines the value committed to by the sender. We refer to [8] for more details.

2.6 Concurrent Non-Malleable Commitments

Our definition of concurrent non-malleable commitments is very similar to that of [11], but different in two aspects: first, our definition of non-malleability is w.r.t identities (in analogy with DDN [6])[3]; second, our definition considers not only the values the adversary commits to, but also the view of the adversary.[4] 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. 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 $v_1,\ldots,v_m$, using identities $\mathsf{id_1},\ldots,\mathsf{id_m}$ of its choice. In the right interaction A interacts with R attempting to commit to a sequence of related values $\tilde{v}_1,\ldots,\tilde{v}_m$, again using identities of its choice $\tilde{\mathsf{id}}_1,\ldots,\tilde{\mathsf{id}}_m$. If any of the right commitments are invalid, or undefined, its value is set to $\perp$. For any i such that $\tilde{\mathsf{id}}_i = \mathsf{id_j}$ for some j, set $\tilde{v}_i = \perp$—i.e., any commitment where the adversary uses the same identity as one of the honest committers is considered invalid. Let $\mathsf{mim}^{A}_{\langle C,R\rangle}(v_1,\ldots,v_m,z)$ denote a random variable that describes the values $\tilde{v}_1,\ldots,\tilde{v}_m$ and the view of A, in the above experiment.

In the simulated execution, a simulator S directly interacts with R. Let $\mathsf{sim}^{S}_{\langle C,R\rangle}(1^n,z)$ denote the random variable describing the values $\tilde{v}_1,\ldots,\tilde{v}_m$ committed to by S, and the output *view* of S; again, whenever *view* contains a right interaction i where the identity is the same as any of the left interactions, $\tilde{v}_i$ is set to $\perp$.

Definition 5. *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 over $\{0,1\}^n$:*

[3] That is, we disallow even copying of commitment as long as the adversary uses a different identity (than all the committers he receives commitments from). In contrast, [11] defined non-malleability w.r.t content; i.e., the adversary allowed copy commitments. This difference is inconsequential as any commitment non-malleable w.r.t content can be turned into one that is non-malleable w.r.t identities, and vice versa.

[4] This point is particularly important when considering our definition w.r.t composability; see Proposition 1 and Section 5.

$$\left\{\mathsf{mim}^A_{\langle C,R\rangle}(v_1,\ldots,v_m,z)\right\}_{v_1,\ldots,v_m\in\{0,1\}^n,n\in N,z\in\{0,1\}^*}$$
$$\left\{\mathsf{sim}^S_{\langle C,R\rangle}(1^n,z)\right\}_{v_1,\ldots,v_m\in\{0,1\}^n,n\in N,z\in\{0,1\}^*}$$

We also consider relaxed notions of concurrent non-malleability: one-many, many-one and one-one secure non-malleable commitments. In a one-one (i.e., a stand-alone secure) non-malleable commitment, we consider only adversaries A that participate in one left and one right interaction; in one-many, A participates in one left and many right, and in many-one, A participates in many left and one right.

Dolev, Dwork and Naor [6] argued that one-one commitments are also many-one secure. Pass and Rosen [11] additionally showed that one-many non-malleability implies (many-many) concurrent non-malleability if the commitment protocol is “natural”. Given our stronger definition, which also considers the view of the adversary, we prove that *any* protocol that is one-many non-malleable is also concurrent non-malleable. Namely,

Proposition 1. *Let $\langle C,R\rangle$ be a one-many concurrent non-malleable commitment. Then, $\langle C,R\rangle$ is also a concurrent non-malleable commitment.*

Proof. Let A be a man-in-the-middle adversary that participates in at most $m = p(n)$ concurrent executions. Below, we provide a simulator S for A. S proceeds as follows on input 1^n and z. S incorporates $A(z)$ and internally emulates all the left interactions for A by simply honestly committing to the string 0^n. Messages from the right interactions are instead forwarded externally. Finally S outputs the view of A.

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. Then, 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 D distinguishes $\mathsf{mim}^A_{\langle C,R\rangle}(v_1,\ldots,v_m,z)$ and $\mathsf{sta}^S_{\langle C,R\rangle}(1^n,z)$ with probability $\frac{1}{p(n)}$. Fix a generic n for which this happens. Consider the hybrid simulator S_i that on input $1^n, z' = v_1,\ldots,v_m,z$, proceeds just as S, with the exception that in left interactions $j \leq i$, it instead commits to v_j. It directly follows that $\mathsf{mim}^A_{\langle C,R\rangle}(v_1,\ldots,v_m,z) = \mathsf{sta}^{S_m}_{\langle C,R\rangle}(1^n,z')$ and $\mathsf{sta}^S_{\langle C,R\rangle}(1^n,z) = \mathsf{sta}^{S_0}_{\langle C,R\rangle}(1^n,z')$. By a standard hybrid argument there exists an $i \in [m]$ such that

$$\left|\Pr\left[a \leftarrow \mathsf{sta}^{S_{i-1}}_{\langle C,R\rangle}(1^n,z') : D(1^n,z',a)=1\right] - \Pr\left[b \leftarrow \mathsf{sta}^{S_i}_{\langle C,R\rangle}(1^n,z') : D(1^n,z',b)=1\right]\right| \geq \frac{1}{p(n)m}$$

Note that the only difference between the executions by $S_{i-1}(1^n,z')$ and $S_i(1^n,z')$ is that in the former A receives a commitment to 0^n in session i, whereas in the latter it receives a commitment to v_i. Consider the one-many adversary $\tilde{A}$ that on input $\tilde{z} = z', n, i$ executes $S_{i-1}(1^n,z')$ with the exception that the i'th left interaction is forwarded externally. Consider, the function

$\mathsf{reconstruct}$ that on input $\mathsf{mim}_{\mathsf{com}}^{\tilde{A}}(0^n, \tilde{z})$, i.e. values $v'_1, \ldots, v'_m$, and the view of $\tilde{A}$, reconstructs the view *view* of A in the emulation by $\tilde{A}$, and sets $\tilde{v}_i = v'_1$ if A did not copy the identity of any of the left interactions, and $\perp$ otherwise, and finally outputs $\tilde{v}_1, \ldots, \tilde{v}_m, view$. By construction, it follows that

$$\mathsf{reconstruct}(\mathsf{mim}_{\langle C,R\rangle}^{\tilde{A}}(0^n, \tilde{z})) = \mathsf{sta}_{\langle C,R\rangle}^{S_{i-1}}(1^n, z')$$
$$\mathsf{reconstruct}(\mathsf{mim}_{\langle C,R\rangle}^{\tilde{A}}(v_i, \tilde{z})) = \mathsf{sta}_{\langle C,R\rangle}^{S_i}(1^n, z')$$

Since $\mathsf{reconstruct}$ is polynomial-time computable, this contradicts the one-many non-malleability of $\langle C, R\rangle$.

3 The Protocol

Our protocol is based on Feige-Shamir's zero-knowledge protocol [7] while relying on the *message scheduling technique* of Dolev, Dwork and Naor[6]. For simplicity of exposition, our description below relies on the existence of one-way functions with efficiently recognizable range, but the protocol can be easily modified to work with any arbitrary one-way function (by simply providing a witness hiding proof that an element is in the range of the one-way function). The protocol proceeds in the following three stages on common input the identity $\mathsf{id} \in \{0,1\}^l$ of the committer, and security parameter n.

1. In Stage 1, the Receiver picks a random string $r \in \{0,1\}^n$, and sends its image $s = f(r)$ through a one-way function f with an efficiently recognizable range to the Committer. The Committer checks that s is in the range of f and aborts otherwise.
2. In Stage 2, the Committer sends $c = \mathsf{com}(v)$, where $\mathsf{com}(\cdot)$ is any commitment scheme that is statistically-binding.
3. In Stage 3, the Committer proves that c is a valid commitment for v or s is in the image set of f. This is proved by $4l$ invocations of a special-sound $\mathcal{WI}$ proof where the messages are scheduled based on the id (very similar to the scheduling presented in [6]). More precisely, there are l rounds, where in round i, the schedule $\mathsf{design}_{\mathsf{id}_i}$ is followed by $\mathsf{design}_{1-\mathsf{id}_i}$ (See Figure 1).

We remark that the scheduling (essentially identical to [6]) in Stage 3 of the protocol is the key in achieving concurrent non-malleability. Loosely speaking, the purpose of the scheduling is to guarantee that for each of the commitments that a man-in-the-middle adversary gives, there exists a point at which the adversary cannot answer the challenge from the receiver simply by "mauling" the commitments on the left (provided that the identity of the commitment is different from any of the commitments on the left).

One important difference between our protocol and the protocol of [6] is that the designs we use consist of two *three-round* protocols, whereas the protocol in [6] uses more rounds; this makes the analysis clearer. An additional simplification is the use of only $\mathcal{WI}$ proofs (instead of zero-knowledge proofs as in [6]).

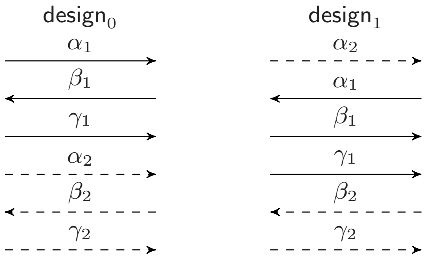

Fig. 1. Description of the schedules used in Stage 3 of the protocol

Claim 1. *$\langle C, R\rangle$ is a statistically-binding commitment scheme.*

Proof. We show that the $\langle C, R\rangle$ scheme satisfies the binding and hiding properties.

Protocol ConcNMCom

Common Input: An identifier $\mathsf{id} \in \{0,1\}^l$.
Auxiliary Input for Committer: A string $v \in \{0,1\}^n$.
Stage 1:
R uniformly chooses $r \in \{0,1\}^n$.
R → C: $s = f(r)$.
C aborts if s not in the range of f.
Stage 2:
C uniformly chooses $r' \in \{0,1\}^{poly(n)}$.
C → R: $c = \mathsf{com}(v, r')$.
Stage 3:
C → R: $4l$ special-sound $\mathcal{WI}$ proofs of the statement
either there exists values v, r' s.t $c = \mathsf{com}(v, r')$
or there exists a value r s.t $s = f(r)$
with verifier query of length $2n$, in the following schedule:
For $j = 1$ to l do: Execute $\mathsf{design}_{\mathsf{id_j}}$ followed by Execute $\mathsf{design}_{1-\mathsf{id_j}}$

Fig. 2. Non-Malleable String Commitment Scheme $\langle C, R\rangle$

Binding: The binding property follows directly from the binding property of com.

Hiding: The hiding property essentially follows from the hiding property of com and the fact that Stage 3 of the protocol is $\mathcal{WI}$ (since $\mathcal{WI}$ proofs are closed under concurrent composition [7]). For completeness, we provide the proof. We show that any adversary R^* that violates the hiding property of $\langle C, R\rangle$ can be used to violate the hiding property of com. More precisely, given any adversary R^* (without loss of generality, deterministic) that distinguishes a commitment made using $\langle C, R\rangle$, we construct a machine R' that distinguishes a commitment made using com. Let s be the first message sent by R^*. R' on auxiliary-input a "fake" witness r such that $s = f(r)$, proceeds as follows. It internally incorporates R^* and forwards the external commitment made using com to R^* in Stage 2. In Stage 3, R' gives $\mathcal{WI}$ proofs using the "fake witness" r. Finally, it outputs whatever R^* outputs. From the $\mathcal{WI}$ property of Stage 3, it follows that R' distinguishes the commitment made using com, if R^* distinguishes the commitment made using $\langle C, R\rangle$.

4 Proof of Security

Theorem 1. *$\langle C, R\rangle$ is one-many concurrent non-malleable.*

Proof: Let A be a man-in-the-middle adversary that participates in one execution in the left and many executions in the right. We construct a simulator S such that the following ensembles are computationally indistinguishable over $\{0,1\}^*$.

$$\left\{\mathsf{mim}^{A}_{\langle C,R\rangle}(v, z)\right\}_{v\in\{0,1\}^n, n\in N, z\in\{0,1\}^*}$$
$$\left\{\mathsf{sim}^{S}_{\langle C,R\rangle}(1^n, z)\right\}_{v\in\{0,1\}^n, n\in N, z\in\{0,1\}^*}$$

The simulator S on input $(1^n, z)$ proceeds as follows. S incorporates $A(z)$ and internally emulates the left interaction by *honestly* committing to the string 0^n. Messages in the right interactions are instead forwarded externally. Finally, S outputs the view of A. We show that the values that S commits to combined with the output view are indistinguishable from the values that A commits to combined with its view. Since S emulates the left interaction by *honestly* committing to 0^n, this is equivalent to showing that

$$\left\{\mathsf{mim}^{A}_{\langle C,R\rangle}(v, z)\right\}_{v\in\{0,1\}^n, n\in N, z\in\{0,1\}^*} \approx \left\{\mathsf{mim}^{A}_{\langle C,R\rangle}(0^n, z)\right\}_{v\in\{0,1\}^n, n\in N, z\in\{0,1\}^*}$$

Towards this goal, we define a new commitment scheme $\langle \hat{C}, \hat{R}\rangle$ (much like the adaptor scheme in DDN [6]), which is a variant of $\langle C, R\rangle$ where the receiver can ask for an arbitrary number of special-sound $\mathcal{WI}$ designs in Stage 3. Furthermore, $\langle \hat{C}, \hat{R}\rangle$ does not have a fixed scheduling in Stage 3; the receiver instead gets to choose which design to execute in each iteration (by sending bit i to select design_i). Note that, clearly, any execution of $\langle C, R\rangle$ can be emulated by an execution of $\langle \hat{C}, \hat{R}\rangle$ by simply requesting the appropriate designs.

Using the same proof as in Claim 1, it follows that $\langle \hat{C}, \hat{R}\rangle$ is hiding, i.e.

Lemma 1. *For every (expected) PPT machine M,*

$$\left\{\mathsf{sta}^{M}_{\langle \hat{C},\hat{R}\rangle}(v, z)\right\}_{v\in\{0,1\}^n, n\in N, z\in\{0,1\}^*} \approx \left\{\mathsf{sta}^{M}_{\langle \hat{C},\hat{R}\rangle}(0^n, z)\right\}_{v\in\{0,1\}^n, n\in N, z\in\{0,1\}^*}$$

Below, in Lemma 2, we show that for every adversary A, there exists an expected *non-uniform* PPT machine R^* whose output, upon receiving a commitment using $\langle \hat{C}, \hat{R}\rangle$ to v, is indistinguishable from the view and the *values committed to* by $A(z)$ when receiving a commitment to v using $\langle C, R\rangle$; by the hiding property of $\langle \hat{C}, \hat{R}\rangle$ we then conclude that $\mathsf{mim}^A_{\langle C,R\rangle}(v,z)$ and $\mathsf{mim}^A_{\langle C,R\rangle}(0^n,z)$ are indistinguishable. On a high-level, R^* will emulate an execution of $\langle C, R\rangle$ for A (by requesting the appropriate design in $\langle \hat{C}, \hat{R}\rangle$) and then will attempt to extract the values committed to by A. In fact, it suffices for R^* to extract only the values committed to *after* the left execution starts (as all values committed to before-hand can be non-uniformly given to R^*).

Let $\Gamma(A,z)$ denote the distribution of all joint views τ of A and the receivers in the right, such that A sends its first message in the left interaction directly after receiving the messages in τ. Let the function $\mathcal{Z} : \{0,1\}^* \times \{0,1\}^* \rightarrow \{0,1\}^*$ be such that, $\mathcal{Z}(z,\tau) = z\|\tau\|\tilde{v}_1\|\ldots\|\tilde{v}_\ell$ where $\tilde{v}_1 \ldots \tilde{v}_\ell$, $\ell \in [m]$ are the values committed to by $A(z)$ in τ (using com).

The main technical content of Theorem 1 is in proving the following lemma.

Lemma 2. *For every PPT adversary A, there exists an expected PPT adversary R^* such that the following ensembles are indistinguishable over $\{0,1\}^*$.*

- $\left\{\tau \leftarrow \Gamma(A,z),\ z' \leftarrow \mathcal{Z}(z,\tau) : \mathsf{sta}^{R^*}_{\langle \hat{C},\hat{R}\rangle}(v,z')\right\}_{v\in\{0,1\}^n, n\in N, z\in\{0,1\}^*}$
- $\left\{\mathsf{mim}^A_{\langle C,R\rangle}(v,z)\right\}_{v\in\{0,1\}^n, n\in N, z\in\{0,1\}^*}$

Before proceeding to the proof of lemma 2, note that by lemma 1, it holds that the following ensembles are indistinguishable

- $\left\{\mathsf{sta}^{R^*}_{\langle \hat{C},\hat{R}\rangle}(v,z')\right\}_{v\in\{0,1\}^n, n\in N, z,\tau,z'\in\{0,1\}^*}$
- $\left\{\mathsf{sta}^{R^*}_{\langle \hat{C},\hat{R}\rangle}(0^n,z')\right\}_{v\in\{0,1\}^n, n\in N, z,\tau,z'\in\{0,1\}^*}$

It thus follows that the following ensembles also are indistinguishable

- $\left\{\tau \leftarrow \Gamma(A,z),\ z' \leftarrow \mathcal{Z}(z,\tau) : \mathsf{sta}^{R^*}_{\langle \hat{C},\hat{R}\rangle}(v,z')\right\}_{v\in\{0,1\}^n, n\in N, z\in\{0,1\}^*}$
- $\left\{\tau \leftarrow \Gamma(A,z),\ z' \leftarrow \mathcal{Z}(z,\tau) : \mathsf{sta}^{R^*}_{\langle \hat{C},\hat{R}\rangle}(0^n,z')\right\}_{v\in\{0,1\}^n, n\in N, z\in\{0,1\}^*}$

By lemma 2, we thus conclude that the following ensembles are indistinguishable,

- $\left\{\mathsf{mim}^A_{\langle C,R\rangle}(v,z)\right\}_{v\in\{0,1\}^n, n\in N, z\in\{0,1\}^*}$
- $\left\{\mathsf{mim}^A_{\langle C,R\rangle}(0^n,z)\right\}_{v\in\{0,1\}^n, n\in N, z\in\{0,1\}^*}$

which concludes the proof of theorem 1.

Proof (of lemma 2). Recall that by the definition of $\mathcal{Z}$ it holds that $z' = z\|\tau\|\tilde{v}_1\|\ldots\|\tilde{v}_\ell$ where $\tilde{v}_1 \ldots \tilde{v}_\ell$, $\ell \in [m]$, are the values committed to by $A(z)$ using com in the view τ. On a high-level, R^* on auxiliary input z', internally incorporates $A(z)$ and emulates the left and the right executions for A. First, however, it starts by feeding A its part of the joint view τ. It, then, emulates the left interaction for A by externally forwarding messages using $\langle \hat{C}, \hat{R}\rangle$ (by

Description of R^*
Input: R^* receives auxiliary input $z' = z\|\tau\|\tilde{v}_1\|\ldots\|\tilde{v}_\ell$.
Procedure: R^* interacts externally as a receiver using $\langle\hat{C},\hat{R}\rangle$. Internally it incorporates $A(z)$ and emulates a one-many man-in-the-middle execution by simulating all right receivers and emulating the left $\langle C,R\rangle$ interaction by requesting the appropriate designs expected by $A(z)$ using $\langle\hat{C},\hat{R}\rangle$ from outside.

Main Execution Phase: Feed the view in τ to A and all right receivers. Emulate all the interactions from τ and complete the execution with A. Let Δ be the transcript of messages obtained.

Rewinding Phase: For $k = \ell+1$ to m, if interaction k is convincing and its identity is different from the left interaction, do:

- In Δ, find the first point ρ that is a safe-point for interaction k; let the associated proof be $(\alpha_\rho,\beta_\rho,\gamma_\rho)$.
- Repeat until a second-proof transcript $(\alpha_\rho,\beta'_\rho,\gamma'_\rho)$ is obtained:
  Emulate the left interaction as in the Main-Execution Phase. For the left interaction:
  - If A expects to get a new proof from the external committer (case (i) in Figure 5): Emulate the proof, by requesting for design_0 from outside committer. Forward one of the two proofs internally.
  - If A sends a challenge for a proof whose first message occurs in ρ: Cancel the execution, rewind to ρ and continue.
- If $\beta_\rho \neq \beta'_\rho$ extract witness w from $(\alpha_\rho,\beta_\rho,\gamma_\rho)$ and $(\alpha_\rho,\beta'_\rho,\gamma'_\rho)$. Otherwise halt and output fail.
- If $w = (v,r)$ is valid commitment for interaction k, i.e. $\mathsf{com}(v,r) = c_k$, where c_k is the Stage 2 message in interaction k, then set $\hat{v}_k = v$. Otherwise halt and output fail.

Note that, since right interactions $\ell+1$ to n all have their Stage 2 and 3 occurring after τ, none of the rewinding can make A request a new commitment from the external committer.

Output Phase: For every interaction k that is not convincing or if the identity of the right interaction is the same as the left interaction, set $\hat{v}_k = \perp$. Output $(\hat{v}_1,\ldots,\hat{v}_m)$ and the view from the Main Execution Phase.

Finally, if it runs for more than 2^n steps, halt and output fail.

Fig. 3. The construction of R^*

appropriately choosing the "right" designs); the right interactions are instead dealt with internally by first honestly emulating the receivers on the right, from the view in τ—this is called the *main execution*. In a second phase, it then attempts to extract all the values committed to on the right—this is called the *rewinding phase*. Finally, in the *output phase*, it outputs the view of A and all the values extracted, including the ones received as auxiliary input (additionally, if A fails in completing one of the commitments that started in τ, or if it uses the same identity as the left interaction, that value is replaced by $\perp$). The core of the proof is to show that extraction during the rewinding phase is successful. Towards this goal, we need to ensure that there exist some point where we can rewind A on the right interaction, *without rewinding on the left*; this is possible in two cases: (1) if rewinding on the right does not cause A to request any new messages on the left, or (2) if rewinding on the right causes A to only request a new special-sound proof—in this case R^* can perfectly emulate this new proof by simply requesting another design from $\langle\hat{C}, \hat{R}\rangle$.

We show below that there exist certain points—called safe-points—in each execution, from which it will be possible to perform extraction by simply rewinding until we obtain a second proof transcript, without rewinding on the left (and aborting all rewindings where A requests a message on the left which would require us to rewind also the left execution). (Actually, to simplify our analysis this extraction procedure is cut-off if it runs "too long" (2^n steps) in which case R^* halts and outputs fail.)

Below we provide a definition of safe-points. A formal description of R^* (which relies on the notion of safe-points) is found in Figure 3.

Intuitively, a safe point ρ is a prefix of some transcript Δ which has the property that if during a rewinding from ρ, A uses the same "scheduling" of messages as in Δ, then the left execution can be perfectly emulated without rewinding (on the left). As we show later, if we rewind only from such points we ensure that the expected running time is polynomially bounded (even if A adaptively schedule the messages on the left).

Definition 6. *A prefix ρ of a transcript Δ is called a* safe-point *for interaction k, if there exists an accepting proof $(\alpha_r, \beta_r, \gamma_r)$ in the k^{th}right interaction, such that:*

1. *α_r occurs in ρ, but not β_r (and γ_r).*
2. *for any proof $(\alpha_l, \beta_l, \gamma_l)$ in the left interaction, if α_l occurs in ρ, then β_l occurs after γ_r.*

If ρ is a safe-point, let $(\alpha_\rho, \beta_\rho, \gamma_\rho)$ denote the canonical "safe" right proof.[5]

Note that the only case a right-interaction proof does not have a safe-point is if it is "aligned" with a left execution proof (such that A can forward messages between the left and the right interactions); see Figure 4. In contrast, in all other cases, a right-interaction proof has a safe-point. In Figure 5, we present the three characteristic types of safe-point. Note that in the first case (see Figure 5 (i)), when rewinding from ρ, R^* can emulate the left proof by requesting a new design from $\langle\hat{C}, \hat{R}\rangle$; in the second case (Figure 5 (ii)), R^* can simply re-send the third message of the left proof (since it is determined by the first two messages in the proof); and in the last case (Figure 5 (ii)), no new message is requested by A, so the left interaction can be "trivially" emulated (by doing nothing).

[5] We remark that our definition of safe-points is analogous to the "safe" rewinding points inside *exposed triplets* defined in DDN [6]. Loosely speaking, for every *exposed triplet*, there is a "safe" rewinding point that one can rewind to extract the committed value on the right without "affecting" the left interaction. Defining safe-point this way, avoids the complication of finding the "safe" rewinding point in each type of the *exposed triplet*.

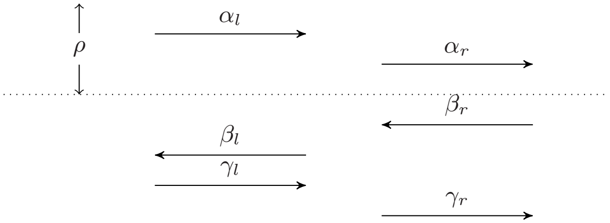

Fig. 4. Prefix ρ that is not a safe point

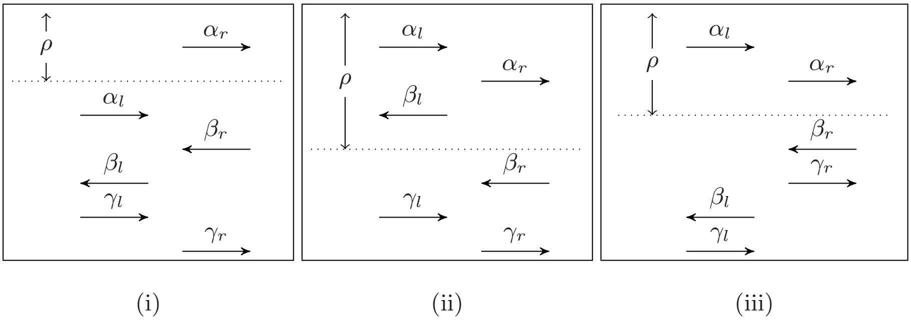

Fig. 5. Three characteristic safe-points

Running-time analysis of R^*. We show that R^* is expected PPT. Note that the time spent by R^* in the Main Execution Phase is $poly(n)$ (where n is the security parameter), since A is a strict polynomial time machine. We show below that the expected time spent by R^* in the Rewinding Phase is $poly(n)$. To bound the expected running time, we assume for simplicity that R^* does not check the fail conditions and may run for more than 2^n steps (since this only increases the running time).

Recall that in the Rewinding Phase, R^* rewinds A from all safe points. Let $T_k(i)$ be the random variable that describes the time spent in rewinding a proof in interaction k after i messages have been exchanged. We show that $E[T_k(i)] \leq poly(n)$ and then by linearity of expectation, we conclude that the expected time spent by R^* in the Rewinding phase is

$$\sum_{k=1}^{m} \sum_{i} E[T_k(i)] \leq \sum_{k=1}^{m} \sum_{i} poly(n) \leq poly(n),$$

where the total number of messages exchanged and m is $poly(n)$.

Bounding $E[T_k(i)]$. Given a (partial) transcript of messages ρ, let $\Pr[\rho]$ denote the probability that ρ occurs as a prefix of the execution emulated in the Main Execution phase. Furthermore, let p_ρ denote the probability that ρ is a safe-point and is rewound—i.e. p_ρ is the probability that, conditioned on the prefix ρ occurring, the right interaction k is convincing and ρ is a safe-point for interaction k. Recall that R^* rewinds until it finds another transcript for the proof $(\alpha_\rho, \beta_\rho, \gamma_\rho)$ associated with ρ, cancelling each rewinding for which A requests the second message of a proof in the left-interaction whose first message occurs in ρ. We claim that the probability of cancelling a rewinding from ρ, is at most $1 - p_\rho$ since ρ is not a safe-point for every rewinding that is cancelled, and conditioned on ρ, the probability of a view occurring in a rewinding from ρ is same as occurring in the Main Execution phase (as the emulated receiver picks uniformly random messages in Stage 3 of the protocol). Thus, the expected number of rewindings is at most $\frac{1}{p_\rho}$ Therefore, the expected number of rewindings from ρ is at most $p_\rho \cdot \frac{1}{p_\rho} = 1$ and each rewinding takes at most $poly(n)$ steps, i.e.

$$E[T_k(i)|\rho] \leq poly(n)$$

Thus,

$$E[T_k(i)] = \sum_{\rho \text{ of length } i} E[T_k(i)|\rho] \Pr[\rho] \ \leq \ poly(n) \times \sum_{\rho \text{ of length } i} \Pr[\rho] \leq \ poly(n)$$

Output distribution of R^* is correct. We proceed to show that the output distribution of R^* is correct. This follows from the following two claims:

Claim 2. *Assume that R^* does not output* fail*, then except with negligible probability, its output is identical to the values committed to by A in the right interactions combined with its view.*

Proof. We first note that since in the Main Execution Phase, R^* feeds A messages according to the correct distribution, the view of A in the simulation by R^* is identical to the view of A in a real interaction. We show in Lemma 3 that there is a safe point for every right interaction that has an identity different from the left interaction. Hence, for every convincing right interaction $k > \ell$ that has a different identity, R^* rewinds that interaction and eventually will either output fail or a witness is extracted from the rewinding phase of R^*. Conditioned on R^* not outputting fail, by the statistical-binding property of com, except with negligible probability the witnesses extracted by R^* are the values committed to by A.

Lemma 3 (Safe-point Lemma). *In any one-many man-in-the-middle execution with m right interactions, for any right interaction k, $k \in [m]$, such that it has a different identity from the identity of the left interaction, there exists a* safe-point *for interaction k.*

Proof. Consider a one-many man-in-the-middle execution Δ, where the identities in the left and right interaction are different. Assume for contradiction, that there is some right interaction k which does not have a safe-point, i.e. every prefix of Δ is not a safe-point for interaction k.

Consider any proof $(\alpha_r, \beta_r, \gamma_r)$ in the right interaction k. Let ρ be the prefix after which β_r is sent immediately. By assumption, ρ is not a safe-point. This means there exists a proof $(\alpha_l, \beta_l, \gamma_l)$ in the left interaction, such that α_l occurs before ρ, β_l occurs after ρ and before γ_r, as depicted in Figure 4. That is, β_l occurs in between β_r and γ_r; we say a left proof is associated with a right proof in this case. Note that each left proof can be associated with at most one right proof. For the interaction k to not have a safe-point, the proofs in the left and right interactions must match up each other one by one: the ith proof in the left is associated with the ith proof in the right.

Since the identities in the left and right interactions are different, there must be a position j they differ at. Let the jth bit in the left be b and that in the right be $1 - b$. Recall that, in the j^{th}round of Stage 3 of the protocol, the left interaction has design_b followed by design_{1-b}; and the right interaction has design_{1-b} followed by design_b. Since all the proofs are "matched up", it must be the case that there is a design_0 on the left that is matched with a design_1 on the right, as depicted in Figure 6. Let $(\alpha_i^l, \beta_i^l, \gamma_i^l)$, $i = 1, 2$, be the two proofs in design_0, and $(\alpha_i^r, \beta_i^r, \gamma_i^r)$, $i = 1, 2$, be the ones on the right in design_1. In this case, consider ρ to be the prefix that includes all the message up until β_1^l. Consider the second proof $(\alpha_2^r, \beta_2^r, \gamma_2^r)$; there is no proof on the left having its first message before ρ and its challenge before γ_2^r at the same time. Hence, we arrive at a contradiction to our assumption that there is no safe-point for that right interaction.

Claim 3. *R^* outputs* fail *with negligible probability.*

Proof. R^* outputs fail only in the following cases:

R^* runs for more than 2^n steps: We know that the expected running time of R^* is $poly(n)$. Using Markov inequality, we conclude that the probability that R^* runs more than 2^n steps is at most $\frac{poly(n)}{2^n}$.

The same proof transcript is obtained from some safe-point: This case occurs if R^* picks some challenge β in the Rewinding Phase that appeared as a challenge in the Main Execution Phase. As R^* runs for at most 2^n steps, it picks at most 2^n challenges. Furthermore, the length of each challenge is

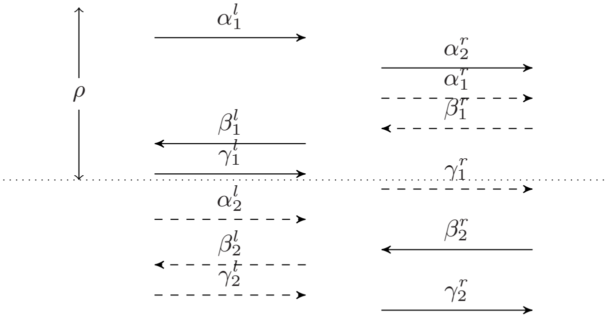

Fig. 6. A design_0 matches up with design_1

$2n$. By applying the union bound, we obtain that the probability that a β is picked twice is at most $\frac{2^n}{2^{2n}}$. Since there are at most polynomially many challenges picked in the Main Execution Phase, using the union bound again, we conclude that the probability that it outputs fail in this case is negligible.

The witness extracted is not a valid decommitment: Suppose, the witness extracted is not the decommitment information, then by the special-sound property it follows that it must be a value r such that $f(r) = s$. We show that if this happens with non-negligible probability, then we can invert the one-way function f. More precisely, given A, z and v, we construct A^* that inverts f; A^* on input y, picks τ uniformly at random from $\Gamma(A, z)$ (by emulating an execution of $A(z)$ internally) and proceeds identically as R^* with inputs τ, z' where $z' = z\|\tau\|\bot\|\bot\| \ldots$ with the exception that it picks a random right interaction, say k, and feeds y as the Stage 1 message in that interaction. On the left interaction it honestly commits to the string v using $\langle \hat{C}, \hat{R} \rangle$. Finally, if the value r' output for interaction k is the inverse image of y w.r.t f (*i.e.* $f(r') = y$), then A^* outputs r'. (Notice that it is not necessary to compute z' according to the definition of $\mathcal{Z}$, since R^* uses the values in z' only in the output phase and not in its extraction procedure). Therefore, the probability that A^* inverts f is identical to the probability that R^* inverts f which is non-negligible; this contradicts the one-wayness of f.

Since each of the above cases occur with negligible probability, using the union bound, we conclude that R^* outputs fail with negligible probability.

5 A $\log n$-Round Non-malleable Commitment Scheme

In this section, we show how to construct a $O(\log n)$-round commitment scheme that is stand-alone non-malleable using any $O(n)$-round commitment scheme that is one-many non-malleable. In particular, using the scheme $\langle C, R \rangle$ described in the previous section, we obtain a $O(\log n)$-round commitment scheme that is stand-alone non-malleable. The idea for this construction is almost identical to the $O(\log n)$-round protocol constructed in [6], except that our construction is more general, as it can be applied to *any* commitment scheme that satisfies our notion of one-many non-malleability; we here rely on the fact that our definition considers not only the values committed to by the adversary but also its view.

Description of the Protocol $\langle \tilde{C}, \tilde{R} \rangle$*:* To commit to value $v \in \{0,1\}^n$, choose random shares $r_1, \ldots, r_n \in \{0,1\}^n$, such that $v = r_1 \oplus \ldots \oplus r_l$. If $\mathsf{id} \in \{0,1\}^{\mathsf{l}}$ is the identity of the $\langle \tilde{C}, \tilde{R} \rangle$ interaction, then for each i, commit to r_i (in parallel) using $\langle C, R \rangle$ with identity (i, id_i), where id_i is the ith bit of id.

In the full version of the paper, we show the following claim.

Claim 4. *$\langle \tilde{C}, \tilde{R} \rangle$ is stand-alone non-malleable.*

Acknowledgement

We are very grateful to Danny Dolev for helpful conversations and for his contagious enthusiasm. We are also grateful to Cynthia Dwork and Moni Naor for helpful clarifications.

References

1. Barak, B.: How to go Beyond the Black-Box Simulation Barrier. In: 42nd FOCS, pp. 106–115 (2001)
2. Barak, B.: Constant-Round Coin-Tossing or Realizing the Shared-Random String Model. In: 43rd FOCS, pp. 345–355 (2002)
3. Brassard, G., Chaum, D., Crépeau, C.: Minimum Disclosure Proofs of Knowledge. JCSS 37(2), 156–189 (1988); Preliminary version by Brassard and Crépeau. In: 27th FOCS (1986)
4. Cramer, R., Damgård, I., Schoenmakers, B.: Proofs of Partial Knowledge and Simplified Design of Witness Hiding Protocols. In: Desmedt, Y.G. (ed.) CRYPTO 1994. LNCS, vol. 839, pp. 174–187. Springer, Heidelberg (1994)
5. di Crescenzo, G., Persiano, G., Visconti, I.: Constant-Round Resettable Zero Knowledge with Concurrent Soundness in the Bare Public-Key Model. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 237–253. Springer, Heidelberg (2004)
6. Dolev, D., Dwork, C., Naor, M.: Non-Malleable Cryptography. SIAM Journal on Computing 30(2), 391–437 (2000)
7. Feige, A., Shamir, A.: How to Prove Yourself: Practical Solutions to Identification and Signature Problems. In: Odlyzko, A.M. (ed.) CRYPTO 1986. LNCS, vol. 263, pp. 181–187. Springer, Heidelberg (1987)
8. Goldreich, O.: Foundations of Cryptography – Basic Tools. Cambridge University Press, Cambridge (2001)
9. Goldwasser, S., Micali, S., Rackoff, C.: The Knowledge Complexity of Interactive Proof Systems. SIAM Jour. on Computing 18(1), 186–208 (1989)
10. Pass, R.: Bounded-Concurrent Secure Multi-Party Computation with a Dishonest Majority. In: 36th STOC, pp. 232–241 (2004)
11. Pass, R., Rosen, A.: Bounded-Concurrent Secure Two-Party Computation in a Constant Number of Rounds. In: 44th FOCS (2003)
12. Pass, R., Rosen, A.: New and Improved Constructions of Non-Malleable Cryptographic Protocols. In: 37th STOC, pp. 533–542 (2005)
13. Pass, R., Vaikuntanathan, V.: New-Age Cryptography (manuscript)