Constant Round Non-Malleable Protocols using One Way Functions

Vipul Goyal
Microsoft Research, India
Email: vipul@microsoft.com

1 Introduction

Since the work of Dolev, Dwork and Naor [DDN91], obtaining non-malleable protocols with small round complexity has been an important goal. The first constant round constructions of non-malleable commitments and zero-knowledge were given in the breakthrough work of Barak [Bar02] (building in turn on the techniques from [Bar01]). An improved construction was later obtained by Pass and Rosen [PR05b]. These construction require making use of so called non-black-box techniques which in turn build on expensive machinery like the PCP theorem. Since then, non-malleable commitments (and zero-knowledge) have been obtained based on sub-exponential or non-standard assumptions in constant rounds [PPV08, PW10] or under one way functions but in super constant number of rounds [LP09, Wee10]. To date there are no known constructions of constant round non-malleable commitments or zero-knowledge using black-box simulation under any standard polynomial time hardness assumption.

We resolve this open question in this work and provide constant round constructions for both non-malleable commitments as well zero-knowledge using only one way functions. Our first construction for non-malleable commitment schemes makes use of the one-way function in a non-black-box way and achieves the strong notion of non-malleability w.r.t. commitment. This construction can be easily modified to obtain a construction satisfying non-malleability w.r.t. opening while making only a black-box use of the one-way function. Our construction and the proof of security is relatively short and simple. Our primary technique is to have different "levels" of non-malleability in the left and right interaction by means of *parallel* repetition. A very rough intuition is as follows. Consider a man-in-the-middle adversary $\mathcal{M}$. In the right interaction, $\mathcal{M}$ is required commit to and then answer a "large" number of randomly generated "puzzles". However in the left interaction, $\mathcal{M}$ is getting a commitment and then an answer to only a "small" number of random puzzles. Thus, it must follow that in the right interaction, $\mathcal{M}$ must be able to compute the answer to a relatively large number of puzzles on its own (without any help from the left interaction). Our techniques involve the analysis of a basic protocol block which is used naturally in the design of larger cryptographic protocols; hence we believe that our techniques are of independent interest and might be useful elsewhere. More details about our construction and the intuition can be found in section 3.

2 Preliminaries

We roughly follow the Lin et al [LPV08] definition of non-malleable commitments. In the real interaction, there is a man-in-the-middle adversary $\mathcal{M}$ interacting with a committer $\mathcal{C}$ (such that the value $\mathcal{C}$ is committing to is ν) in the left session and interacting with a receiver $\mathcal{R}$ in the right session. Let $mim^{\mathcal{M}}_{\langle C,R\rangle}(v,z)$ denote a random variable that describes the value $\tilde{\nu}$ the $\mathcal{M}$ commits to the right execution and the view of $\mathcal{M}$ in the full experiment.

In the simulated experiment, a simulator $\mathcal{S}$ directly interacts with $\mathcal{R}$. Let $sim^{S}_{\langle C,R\rangle}(1^k, z)$ denote the random variable describing the value $\tilde{\nu}$ committed to by $\mathcal{S}$ and the output view of $\mathcal{S}$.

If the tag tag for the left interaction is equal to the tag $\tilde{tag}$ for the right interaction, the value $\tilde{\nu}$ committed to in the right interaction is defined to be $\perp$ in both experiments.

Definition 1 (Non-Malleable Commitments) *[LPV08] A commitment scheme $\langle C, R\rangle$ is said to be non-malleable if for every* PPT *man-in-the-middle adversary $\mathcal{M}$, there exists a* EPPT *simulator $\mathcal{S}$ such that the following ensembles are computationally indistinguishable:*

$$\{mim^{\mathcal{M}}_{\langle C,R\rangle}(v, z)\}_{\nu\in\{0,1\}^k,k\in N,z\in\{0,1\}^*}$$

$$\{sim^{S}_{\langle C,R\rangle}(1^k, z)\}_{\nu\in\{0,1\}^k,k\in N,z\in\{0,1\}^*}$$

We also define the notion of one sided non-malleable commitments where we only consider interactions where the tag tag for the left interaction is smaller than the tag $\tilde{tag}$ for the right interaction (if $tag \geq \tilde{tag}$, the value $\tilde{\nu}$ committed to in the right interaction is defined to be $\perp$ in both experiments).

Building Blocks. We shall make use of Naor's statistically binding commitment scheme and denote it by com_σ. In addition, we shall make use of a constant round (computational) zero-knowledge argument based on any OWF. We denote such a protocol by ZK . Let k be the security parameter.

3 Construction of Non-Malleable Commitments

3.1 Basic Construction

In this section, we describe our basic protocol for "small" tags with one sided non-malleability. They can be extended to the general case by relying on techniques from [PR05b, PR05a]. We assume that each execution has a tag $tag \in [2n]$. Denote by ℓ the value $k \cdot tag$. Let $\mathrm{com}_\sigma(m)$ denote a commitment to the message m with the first message σ under the statistically binding commitment scheme of Naor. Whenever we need to be explicit about the randomness used to generate the commitment, we denote it as $\mathrm{com}_\sigma(m; r)$ where r is the said randomness.

The commitment scheme $\langle C, R\rangle$ between a committer $\mathcal{C}$ trying to commit to ν and a receiver $\mathcal{R}$ proceeds as follows.

Commitment Phase.

0. **Initialization Message.** The receiver $\mathcal{R}$ generates the first message σ of the Naor commitment scheme and sends it to $\mathcal{C}$.

   Primary Slot

1. The committer $\mathcal{C}$ generates ℓ pairs of random strings $\{\alpha^0_i, \alpha^1_i\}_{i\in[\ell]}$ (with length of each string determined by the security parameter). $\mathcal{C}$ further generates commitments of these strings $\{A^0_i = \mathrm{com}_\sigma(\alpha^0_i), A^1_i = \mathrm{com}_\sigma(\alpha^1_i)\}_{i\in[\ell]}$ and sends them to $\mathcal{R}$ ($\mathcal{C}$ uses fresh randomness to generate each commitment).

2. The receiver $\mathcal{R}$ generates and sends to $\mathcal{C}$ a random ℓ-bit challenge string $ch = (ch_1, \ldots, ch_\ell)$.

3. The committer $\mathcal{C}$ sends to $\mathcal{R}$ the values $\alpha^{ch_1}_1, \ldots, \alpha^{ch_\ell}_\ell$. Note that $\mathcal{C}$ *does not* send the openings associated with the corresponding commitments.

4. **Verification Message.** Define ℓ strings $\{\alpha_i\}_{i\in[\ell]}$ such that $\alpha_i = \alpha_i^0 \oplus \alpha_i^1$ for all $i \in [\ell]$. $\mathcal{C}$ generates ℓ commitments $B_i = \mathrm{com}_\sigma(\nu; \alpha_i)$ for $i \in [\ell]$ and sends them to $\mathcal{R}$. (That is, randomness α_i is used to generate the i-th commitment to ν).

5. **Consistency Proof.** The committer $\mathcal{C}$ and the receiver $\mathcal{R}$ now engage in a zero-knowledge argument protocol ZK where $\mathcal{C}$ proves to $\mathcal{R}$ that the above commit phase is "valid". That is, there exist values $\hat{\nu}$, $\{\hat{\alpha}_i, \hat{\alpha}_i^0, \hat{\alpha}_i^1\}_{i\in[\ell]}$ such that for all i:
   - $\hat{\alpha}_i^0 \oplus \hat{\alpha}_i^1 = \hat{\alpha}_i$, and,
   - commitments A_i^0 and A_i^1 are valid commitments to the strings $\hat{\alpha}_i^0$ and $\hat{\alpha}_i^1$ respectively under some random tape, and,
   - commitment B_i is a valid commitment to $\hat{\nu}$ under the random tape $\hat{\alpha}_i$.

Decommitment Phase. The committer $\mathcal{C}$ simply reveals the committed value ν and the randomness used in the commitment phase. The receiver $\mathcal{R}$ checks if the commitment phase was run honestly using the above randomness (including making sure its a "valid" commit phase). If so, $\mathcal{R}$ takes the value committed to be ν and $\perp$ otherwise.

Lemma 1 *The commitment scheme $\langle C, R\rangle$ is computationally hiding and statistically binding (in the stand alone setting).*

The proof is this lemma is straightforward and we only provide a sketch. To prove computational hiding, we consider the following hybrid experiments. We first start simulating the protocol ZK in the final step of the commitment phase. Next, for each $i \in [\ell]$, we replace the commitments $\{A_i^0, A_i^1\}$ to be commitments to random strings (as opposed to shares of the string α_i used later as randomness to generate B_i). Finally, for each $i \in [\ell]$, we change the commitment B_i to be a commitment to a random string (as opposed to a commitment to ν). Hence in the final hybrid, the transcript of the commitment stage contains no information about the value ν being committed to. Statistical binding follows from the statistical binding property of the commitment scheme com.

Theorem 1 *The commitment scheme $\langle C, R\rangle$ is a one sided non-malleable commitment scheme against a synchronizing adversary.*

PROOF. To prove the above theorem, we construct a standalone machine $\mathcal{S}$ such that the ensembles $\{mim^{\mathcal{M}}_{\langle C,R\rangle}(v, z)\}_{\nu\in\{0,1\}^k,k\in N,z\in\{0,1\}^*}$ and $\{sim^{S}_{\langle C,R\rangle}(1^k, z)\}_{\nu\in\{0,1\}^k,k\in N,z\in\{0,1\}^*}$ (call these $\mathsf{dist1}$ and $\mathsf{dist2}$ respectively) are computationally indistinguishable. Our $\mathcal{S}$ works as follows. It starts an interaction with $\mathcal{M}$ by simply honestly committing to the value 0 in the left interaction and relaying messages between $\mathcal{M}$ and $\mathcal{R}$ in the right interaction. We claim that such a machine $\mathcal{S}$ satisfies the required property.

Towards contraction, assume that there exists a distinguisher $\mathcal{D}$ which can distinguish between these two distribution with an advantage at least $r(k) \geq \frac{1}{\mathrm{poly}(k)}$ for infinitely many value of k. That is, $Pr[\mathcal{D}(\mathsf{dist1}) = 1] \geq \frac{1}{2} + r(k)$, or in other words,

$$Pr[\mathcal{D}(\mathsf{dist1}) = 1] - Pr[\mathcal{D}(\mathsf{dist2}) = 1] \geq 2r(k) \tag{1}$$

Fix any such generic k. Now consider the real experiment in which the adversary $\mathcal{M}$ interacts with a committer $\mathcal{C}$ in the left interaction and with a receiver $\mathcal{R}$ in the right one. We shall now show how to construct an extractor $\mathcal{E}$ which takes as input the view of $\mathcal{M}$ in such an experiment and outputs the value $\tilde{\nu}$ committed by $\mathcal{M}$ in the right interaction with probability at least $1 - r(k)$ without rewinding $\mathcal{C}$ (i.e., having access to the value and the random coins used by $\mathcal{C}$ in the left interaction). It is easy to show that this violates the (standalone) computational hiding of the commitment scheme $\langle C, R\rangle$. This is because by equation 1, the value $\tilde{\nu}$ committed by $\mathcal{M}$ on the right in the real experiment is distinguishable with noticeable probability from when the value 0 is being committed to in the left interaction *even* conditioned on the extractor succeeding in outputting $\tilde{\nu}$. Thus, all that remains to show is an extractor that succeeds with probability at least $1 - r(k)$. For simplicity, the analysis of our extractor is conditioned on the event that given the completed main thread, there is exactly one value $\tilde{\nu}$ ($\neq \perp$) consistent with the transcript of the right interaction. By the soundness of the protocol ZK , the failure probability of the extractor increases by at most an additive negligible term in the general case. We assume that the adversary $\mathcal{M}$ is deterministic without loss of generality.

Extractor Description and Analysis Let $\ell(k) = k \cdot tag$ and $\tilde{\ell}(k) = k \cdot \tilde{tag}$. Observe that $\tilde{\ell}(k) - \ell(k) \geq k$. Throughout the description below, we denote the messages of the left execution as in the above protocol description; the notation in the right execution is augmented with "tildes" (e.g., A_i^b would refer to a commitment on the left while $\tilde{A}_i^b$ denotes a commitment in the right interaction).

The extractor $\mathcal{E}$ works as follows. It gets as input a transcript of the honestly executed left and right interactions; we refer to this collective interaction transcript as the "main thread". If $\mathcal{M}$ aborts before the main thread was complete, $\mathcal{E}$ does not need to extract and simply halts[1]. Otherwise, $\mathcal{E}$ rewinds $\mathcal{M}$ up to $\frac{k\tilde{\ell}(k)}{r(k)^3}$. For $j \in [\frac{k\tilde{\ell}(k)}{r(k)^3}]$, do the following.

- $\mathcal{E}$ rewinds the right interaction to the beginning of the step 2 of the protocol. It generates a new random challenge $\tilde{ch}[j] \in \{0,1\}^{\tilde{\ell}(k)}$, sends it to $\mathcal{M}$ and receives the challenge $ch[j] \in \{0,1\}^{\ell(k)}$ for the left interaction from $\mathcal{M}$.

- $\mathcal{E}$ now prepares a response to the challenge $ch[j]$ on its own since it is not allowed to rewind the committer $\mathcal{C}$ and make additional queries. Consider the set of commitments on the left already "recovered" in the main thread (i.e., the ℓ commitments A_i^b such that their value was asked by $\mathcal{M}$ and given by $\mathcal{C}$ in the main thread itself). Now, the challenge $ch[j]$ induces a selection of ℓ commitments on the left. Consider any such selected commitment A_i^b. If its value was recovered in the main thread, $\mathcal{E}$ uses that value to prepare the response. Otherwise, $\mathcal{E}$ simply chooses a random string and uses that in the response in place of its value. Such a *simulated response* is sent to $\mathcal{M}$.

- $\mathcal{E}$ receives the response corresponding to $\tilde{ch}[j]$ from $\mathcal{M}$ in the right interaction. If there is an index i s.t. $\tilde{\alpha_i^0} \oplus \tilde{\alpha_i^1}$ (where one of $(\tilde{\alpha_i^0}, \tilde{\alpha_i^1})$ was recovered during the main thread while the other received as part of the current response in rewinding j) allows for opening of the commitment $\tilde{B}_i$, $\mathcal{E}$ recovers the committed value $\tilde{\nu}$ from $\tilde{B}_i$. If not, $\mathcal{E}$ goes to the beginning of this loop.

If at the end of $\frac{k\tilde{\ell}(k)}{r(k)^3}$ rewindings, $\mathcal{E}$ still was not successful in outputting the value $\tilde{\nu}$ (due to $\mathcal{M}$ aborting or not revealing the correct values for the commitments), it aborts and outputs $\mathsf{Ext\_Fail}$. The fact that $\mathcal{E}$ runs in probabilistic polynomial time is straightforward to prove since $r(k) \geq \frac{1}{\text{poly}(k)}$. We now analyze the probability of $\mathcal{E}$ outputting $\mathsf{Ext\_Fail}$.

Lemma 2 *The probability that the extractor $\mathcal{E}$ outputs $\mathsf{Ext\_Fail}$ is bounded by $r(k)$ for large enough k.*

PROOF. We first establish three different categories of the main thread (each satisfying a different property) for which the probability (over the random coins used in the rewinds) of $\mathcal{E}$ outputting $\mathsf{Ext\_Fail}$ is noticeable. We will call them "bad" main threads. Keep in mind that $\mathcal{E}$ never output $\mathsf{Ext\_Fail}$ for main

[1]If in an interaction, the parties $\mathcal{C}$ or $\mathcal{R}$ terminate the protocol due to an obvious cheating by $\mathcal{M}$, we also consider it as $\mathcal{M}$ aborting.

threads which are incomplete (i.e., in which $\mathcal{M}$ already aborted). We define a prefix of the main thread as the transcript of the steps 0 and 1 of the left and the right interaction (i.e., up to the stage where $\mathcal{M}$ is waiting for a challenge from the right). For a particular prefix, let p denote the probability that $\mathcal{M}$ completes the real experiment without aborting (i.e., the probability is taken over the random coins used by $\mathcal{C}$ and $\mathcal{R}$ after step 1 of the interaction).

It is convenient to introduce the notion of a *fraction* of main threads. By the fraction of main threads with a particular property being f, we mean the probability that $\mathcal{E}$ receives a *completed* main thread with that property is f (over the random coins of the entire experiment). We choose three arbitrary constants C_1, C_2, C_3 such that $\frac{1}{C_1} + \frac{1}{C_2} + \frac{1}{C_3} \leq \frac{3}{4}$. Note that these constants could in fact be the same and arbitrarily big. However we choose to use three constants for the sake of making the connections between the different parts of proof more clear.

Lemma 3 *The fraction of main threads for which $p < \frac{r(k)}{C_1}$ is bounded by $\frac{r(k)}{C_1}$. We call these threads as main threads of type* $\mathsf{bad1}$.

PROOF. The proof of this lemma is straight forward. Pr[main thread is of type $\mathsf{bad1}$] $\leq$ Pr[main thread has a prefix with $p < \frac{r(k)}{C_1}$]$\cdot$Pr[main thread is completed $\mid p < \frac{r(k)}{C_1}$] $\leq 1 \cdot \frac{r(k)}{C_1}$

Now for a given main thread, we define the *dependent set of commitment S* as the following subset of commitments in the right interaction. Intuitively, the dependent set of commitments can be thought of as the commitments in the right interaction which were constructed by mauling one of the commitment from the *unrecovered* set of commitments in the left interaction (and hence, to reveal their value correctly with "good" probability, $\mathcal{M}$ has to get the correct value from a commitment in the unrecovered set of commitments).

Definition 2 (Dependent Set of Commitments) *S is the* dependent set of commitments *of a main thread iff the following two conditions are satisfied. Denote by ch the challenge by $\mathcal{M}$ in the left interaction in this main thread. The probabilities below are over the random coins of the experiment* after *the prefix completion.*

1. *For every commitment $\tilde{A}_i^b$ in S, the probability that the commitment is selected by $\mathcal{R}$ AND its value is revealed correctly by $\mathcal{M}$ is at least $\frac{r(k)}{3C_1}$ (for this prefix).*

2. *The probability that the commitment $\tilde{A}_i^b$ is selected by $\mathcal{R}$ AND its value is revealed correctly by $\mathcal{M}$ on the right is less than $\frac{r(k)}{2C_2\tilde{\ell}(k)}$* conditioned *on the event that the challenge by $\mathcal{M}$ in the left interaction is ch.*

Observe that the first probability in the above definition is dependent only on what the prefix in the main thread is, while, the second one depends on the prefix as well as what the left challenge ch appearing in the main thread is. Both these probabilities values are well defined for a given main thread. Let $|S|$ denote the number of commitments in the set S.

Lemma 4 *$|S| > \ell + \log^2 k$ for at most a $\frac{r(k)}{C_2} + negl(k)$ fraction of the main threads. Call these threads as main threads of type* $\mathsf{bad2}$.

Intuition. The above lemma constitutes the "core of the intuition" regarding why the protocol (and the extractor) works. To understand this lemma, consider the following "explicit" attack by the adversary. Each commitment on the right is "dependent" on a set (or possibly just one) of commitments on the left. This means that the commitment on the right was constructed by $\mathcal{M}$ by mauling this set of commitments on the left. The probability that a correct value is revealed by $\mathcal{M}$ for this commitment on the right is "negligible" if all of these left commitment values are not asked by $\mathcal{M}$ (and is "noticeable" otherwise)[2]. Now lets look at commitments in the unrecovered set on the left in the main thread. *Could it be that a large number of commitments on the right are "dependent" on the commitments in the unrecovered set?.* A simple combinatorial argument shows that this cannot be the case. To start with, observe that if a large number of commitments on the right are dependent on a particular commitment on the left, with high probability, that commitment has to be selected by $\mathcal{M}$ in the main thread (and hence it will not be in the unrecovered set). On the other hand, if only a small number of commitments on the right are dependent on a particular commitment on the left, overall the number of commitments dependent on the commitments in the unrecovered set may remain "small".

PROOF. For a given prefix, consider a set S for a challenge ch such that $|S| > \ell + \log^2 k$. Now consider the random challenge $\tilde{ch}$ given by $\mathcal{R}$ in the right interaction. Probability (over choice of $\tilde{ch}$) that the set of commitments selected by $\tilde{ch}$ and the set S are disjoint is at most $\frac{1}{2^{\ell+\log^2 k}}$. In more detail, this probability is either 0 or $\frac{1}{2^{|S|}}$; 0 when S has a pair of "conflicting" commitments $(\tilde{A}_i^0, \tilde{A}_i^1)$ and $\frac{1}{2^{|S|}}$ otherwise since each commitment in S is selected independently with probability $\frac{1}{2}$.

Now note that there are at most 2^ℓ possibilities for such a set S depending upon the choice of challenge $ch \in \{0,1\}^\ell$. Taking the union bound over all such sets, we get that the probability that the set of commitments selected by $\tilde{ch}$ is disjoint with *any* such set S (with $|S| > \ell + \log^2 k$) is at most $\frac{2^\ell}{2^{\ell+\log^2 k}} = negl(k)$.

Next observe that by the second condition of the definition 2, the probability that for *some* $\tilde{A}_i^b \in S$, $\mathcal{M}$ revealed the correct value in the right interaction in the main thread is bounded by $\frac{r(k)}{C_2}$ (by a union bound over all $\tilde{A}_i^b \in S$ given that $|S|$ cannot exceed $2\tilde{\ell}(k)$). Now we have the following:

Pr[main thread is of type bad2] $\leq$ Pr[$\tilde{ch}$ does not select any commitment in S] + Pr[main thread is completed $|$ $\tilde{ch}$ selects a commitment in S]

Pr[main thread is of type bad2] $\leq negl(k)$ + Pr[$\exists \tilde{A}_i^b \in S$ s.t. $\mathcal{M}$ revealed the correct value of $\tilde{A}_i^b$ in main thread]

Hence, the fraction of main threads of type bad2 is bounded by $\frac{r(k)}{C_2} + negl(k)$

Looking ahead, the intuition for the rest of the proof should be clear. This lemma shows that there are at most $\ell+\log^2 k$ commitments on the right which are "dependent" on the left commitments whose value $\mathcal{E}$ did not recover in the main thread. However the total number of commitments on the right is $2\cdot\tilde{\ell} > 2(\ell+\log^2 k)$ (since $\tilde{tag} > tag$). Hence, there should exists at least one pair of commitments on the right such that $\mathcal{M}$ can correctly compute both the committed values (without asking for values unrecovered in the main thread). If that is the case, $\mathcal{E}$ is successful is extracting the value $\tilde{\nu}$ committed on the right without any "additional queries" on the left.

Definition 3 (Strictly Dependent Set of Commitments) *G is the* strictly dependent set of commitments *for a main thread iff the following two conditions are satisfied.*

1. *For every commitment $\tilde{A}_i^b$ in G, the probability that the commitment is selected by $\mathcal{R}$ AND its value is revealed correctly by $\mathcal{M}$ is at least $\frac{r(k)}{3C_1}$ (for this prefix).*

[2]We are abusing the terms negligible and noticeable for the purpose of this intuition.

2. *The probability that the commitment $\tilde{A}_i^b$ is selected by $\mathcal{E}$ in a rewinding AND its value is revealed correctly by $\mathcal{M}$ on the right is less than $\frac{r(k)^3}{50\tilde{\ell}(k)^2 C_1 C_2 C_3}$ (i.e., the probability in the experiment where $\mathcal{M}$ gets random strings in places of left commitment values unrecovered in the main thread).*

Observe that the first probability in the above definition is dependent only on what the prefix in the main thread is, while, the second one depends on the prefix as well as the left challenge ch appearing in the main thread. We now prove the following lemma.

Lemma 5 *$G \not\subseteq S$ for at most $\frac{r(k)}{C_3}$ fraction of the main threads. Call these threads as main threads of type* bad3.

Intuition. Continuing the intuition from the last lemma, consider now the following scenario. The adversary $\mathcal{M}$ somehow is able to use two (or more) commitments on the left to construct a commitment on the right such that the following happens. The $\mathcal{M}$ asks for exactly one of these two values in the left interaction (as part of its challenge ch). If $\mathcal{M}$ gets access to *any one* of these two values, it is able to correctly reveal the value of the commitment on the right with noticeable probability. However, if the value asked on the left is provided at random, $\mathcal{M}$ correctly reveals the value on the right with only negligible probability. We show that such a scenario contradicts the hiding property of the commitment scheme. Getting the intuition closer to the statement we prove in the lemma, suppose there is a commitment on the right which is correctly revealed with noticeable probability in the presence of correct values from the unrecovered set of commitment *as well as* in the absence of values from unrecovered set of commitments. However if the values from the unrecovered set are given randomly, the value of this commitment on the right is correctly revealed only with negligible probability. Then, we can construct an adversary to contradict the hiding property of the commitment scheme.

PROOF. We prove the above by contradiction. Assume that for at least a fraction $\frac{r(k)}{C_3}$ of the main threads, there exists a commitment $\tilde{A}_i^b$ in G but not in S. This means the following 3 conditions are true for this main thread (where the probabilities are taken over the random coins of the experiment after the prefix completion):

1. If the values of the commitments in the unrecovered set are given correctly on the left, $\mathcal{M}$ reveals the correct value in $\tilde{A}_i^b$ on the right with "large" probability (i.e., at least $\frac{r(k)}{3C_1}$)

2. If the value of the commitments in the unrecovered set are given randomly on the left (i.e., the response is simulated), $\mathcal{M}$ reveals the correct value in $\tilde{A}_i^b$ on the right with "small" probability (i.e., smaller than $\frac{r(k)^3}{50\tilde{\ell}(k)^2 C_1 C_2 C_3}$)

3. Conditioned on the event that $\mathcal{M}$ does not ask any of the values from the unrecovered set of commitments (i.e., its challenge on the left is ch w.r.t. which S and G are defined), $\mathcal{M}$ reveals the correct value in $\tilde{A}_i^b$ on the right with "large" probability (i.e., at least $\frac{r(k)}{2C_2\tilde{\ell}(k)}$)

We now construct an adversary $\mathcal{A}$ to show that the above conditions violate the (computational) hiding property of the commitment scheme com. Consider the following experiment between the adversary $\mathcal{A}$ and an external challenger *Chal* .

1. $\mathcal{A}$ starts the execution of $\mathcal{M}$ and gives it honestly the messages in the right session. The messages received from $\mathcal{M}$ in the left session are forwarded to *Chal* and its reply is forwarded to $\mathcal{A}$ until the protocol is completed till step 3 (on both left and right interactions).

2. Now the *Chal* provides to $\mathcal{A}$ a total of $M = \frac{25\tilde{\ell}(k)^2 C_1 C_2 C_3}{r(k)^3}$ candidate tuples for the values in the unrecovered set of commitments on the left. Exactly one of the candidate tuples has correct values for all the commitments in the unrecovered set. All the values in the rest of the candidate tuples are generated by *Chal* randomly. The goal of $\mathcal{A}$ would be guess which of the M tuples is the correct one. $\mathcal{A}$ is not allowed any further interaction with *Chal* (including running the protocol beyond step 3).

3. $\mathcal{A}$ proceeds as follows. It selects a commitment $\tilde{A_i^b}$ from the right interaction as a guess for a commitment in $G - S$ (if one exists).

4. $\mathcal{A}$ now rewinds $\mathcal{M}$ exactly M times. In the i-th rewind, $\mathcal{M}$ gives a challenge $ch[i]$ on the left (if it aborts at any point, we move on the next rewinding). To construct the response, for the commitments in the unrecovered set picked by $ch[i]$, $\mathcal{A}$ uses the values in the i-th candidate tuple. Observe that for exactly one rewind, the response given by $\mathcal{A}$ would be correct and in all other cases, it would be the *simulated* response as given by the extractor $\mathcal{E}$ when it rewinds.

5. Now we consider the case where the following happens. In the main thread, the commitment $\tilde{A_i^b}$ was selected by $\mathcal{A}$ and a value $\tilde{\alpha_i^b}$ was received. There is exactly one rewind (say index ind), such that the commitment $\tilde{A_i^b}$ was selected by $\mathcal{A}$ AND a value $\tilde{\alpha_i^b}[ind] = \tilde{\alpha_i^b}$ was received. If that is the case, $\mathcal{A}$ outputs the index ind to *Chal* as its guess for the correct value tuple. In all other cases, $\mathcal{A}$ aborts and outputs $\perp$.

We now analyze the success probability of $\mathcal{A}$. Let E denote the event that main thread is of type bad3 and $E1$ denote the event (E AND $\tilde{A_i^b} \in (G - S)$).

Pr[$\mathcal{A}$ outputs the correct guess] $\geq$ Pr[E] $\cdot$ Pr[$E1|E$] $\cdot$ Pr[correct value $\tilde{\alpha_i^b}$ for $\tilde{A_i^b}$ appears in the main thread $|E1$] $\cdot$ Pr[correct value $\tilde{\alpha_i^b}$ appears in the rewind with correct response $|E1$] $\cdot$ Pr[correct value $\tilde{\alpha_i^b}$ does not appear in any rewind with simulated response $|E1$]

(Note that the last 3 probability terms are results of experiments run with independent random coins and hence are independent)

$$Pr[\mathcal{A} \text{ outputs the correct guess}] \geq \frac{r(k)}{C_3} \cdot \frac{1}{2\tilde{\ell}(k)} \cdot \frac{r(k)}{2C_2\tilde{\ell}(k)} \cdot \frac{r(k)}{3C_1} \cdot \frac{1}{2}$$

(Note that the expected number of times correct value $\tilde{\alpha_i^b}$ appears in simulated responses is $\frac{r(k)^3}{50\tilde{\ell}(k)^2 C_1 C_2 C_3} \cdot (\frac{25\tilde{\ell}(k)^2 C_1 C_2 C_3}{r(k)^3} - 1) < \frac{1}{2}$, hence at least with probability $\frac{1}{2}$, there are 0 such appearances)

$$Pr[\mathcal{A} \text{ outputs the correct guess}] \geq \frac{r(k)^3}{24\tilde{\ell}(k)^2 C_1 C_2 C_3} \tag{2}$$

Now we have the following claim:

Claim 1 *In the above experiment, assuming the commitment scheme com is computationally hiding, the probability of any PPT $\mathcal{A}$ outputting the correct guess is bounded by $\frac{r(k)^3}{25\tilde{\ell}(k)^2 C_1 C_2 C_3} + negl(k)$.*

Proof Sketch. The proof of this claim relies on a straight forward hybrid argument[3]. In the i-th hybrid experiment, in the chosen tuple (out of M tuples) *Chal* keeps the values for the first i unrecovered commitments to be random and the rest correct. In the $\ell(k)$-th hybrid, clearly the probability of $\mathcal{A}$ winning is

[3]since *Chal* provides just the committed values and not any opening to the commitments, there are no issues related to "selected opening attacks" etc (see [BHY09] and the reference therein)

exactly $\frac{1}{M}$ since the chosen tuple distribution is identical to the rest. Hence, there should exists a hybrid i in which the probability of $\mathcal{A}$ winning changes by a noticeable amount from the last hybrid. Then it can shown that the hiding property of the commitment scheme com can be broken with a noticeable advantage.

The above claim is in contradiction to the equation 2. This concludes the proof of lemma 5.

Concluding the Analysis. We now conclude the proof of lemma 2. The rest of the proof is quite straightforward. Very roughly, we have already established that there are only a "small" number of commitments on the right (i.e., commitments in set G) which go from being correct with "large" probability (given a correct response on the left) to being correct only with "small" probability (given a simulated response on the left). Thus, there are sufficiently large number of commitments on the right such that given a simulated response, they are revealed correctly by $\mathcal{M}$ (thus implying success for the extractor $\mathcal{E}$). Details follow.

As earlier, for the prefix of the given main thread, let p denote the probability that $\mathcal{M}$ completes the main thread (i.e., the real experiment) without aborting (i.e., the probability is taken over the random coins after step 1). For the given main thread, let q denote the probability of $\mathcal{E}$ succeeding in extracting in a rewinding using a simulated response. Since $\mathcal{E}$ rewinds $\mathcal{M}$ $\frac{k\tilde{\ell}(k)}{r(k)^3}$ times,

$$Pr[\mathcal{E} \text{ aborts}] \leq p \cdot (1-q)^{\frac{k\tilde{\ell}(k)}{r(k)^3}}$$

(Exact equality may not be satisfied because $\mathcal{M}$ may abort even before prefix completion.) Now this value is noticeable only if $q = o(\frac{r(k)^3}{\tilde{\ell}(k)})$, or, in other words, $q < \frac{r(k)^3}{50\tilde{\ell}(k)} C_1 C_2 C_3$. Now, Pr[$\mathcal{E}$ aborts] $\leq$ Pr[main thread is of type bad1 or bad2 or bad3] + Pr[$\mathcal{E}$ aborts | main thread is neither of these 3 types].

To compute the second term, we first compute q for the main thread. Note that the main thread being not of type bad2 or bad3 implies that $|G| \leq \ell + \log^2 k$ (since $|S| \leq \ell + \log^2 k$ and $G \subseteq S$). Also, since the main thread is not of type bad1, there are at most $O(log(k))$ commitments in the right interaction for which the probability of getting asked on the right (which happens with probability $\frac{1}{2}$) AND revealed correctly by $\mathcal{M}$ is less than $\frac{r(k)}{3C_1}$ (otherwise, it is easy to show that $p < \frac{r(k)}{C_1}$). Or in other words, there are at least $2\tilde{\ell} - log(k)^2$ commitments on the right with probability of getting asked and revealed correctly is at least $\frac{r(k)}{3C_1}$. Out of these, at most $\ell + \log^2 k$ are in G. Hence, (for large enough k) there are at least $\tilde{\ell} + 1$ commitments, or in other words at least one pair of commitments on the right, such that the probability that such a commitment is selected by $\mathcal{E}$ in a rewinding and $\mathcal{M}$ reveals the correct value is at least $\frac{r(k)^3}{50\tilde{\ell}(k)} C_1 C_2 C_3$. This means for such a main thread, $q \geq \frac{r(k)^3}{50\tilde{\ell}(k)} C_1 C_2 C_3$. Thus,

$$Pr[\mathcal{E} \text{ aborts}] \leq \frac{r(k)}{C_1} + \frac{r(k)}{C_2} + \frac{r(k)}{C_3} + \text{negl}(k)$$

$$Pr[\mathcal{E} \text{ aborts}] \leq \frac{3}{4} r(k) + \text{negl}(k)$$

This completes the proof.

3.2 Getting Full-Fledged Non-Malleable Commitments

The construction in the previous section can now be extended to get constant round full-fledged non-malleable commitments based only one a one way function. This can be done by simple application of known techniques. We provide more details here.

We first construct a full-fledged (i.e., "two" sided) non-malleable commitment scheme for small tags (i.e., $tag \in [2n]$) against a synchronizing adversary. This can be done very similar to the construction by Pass and Rosen [PR05b]. Denote by $\ell[a]$ the value $k \cdot tag$ and by $\ell[b]$ the value $k \cdot (2n - tag)$. The idea is to have two slots (each representing a rewinding opportunity) such that for exactly one of these slots, the "tag being used on the right" is larger than the one on the left. The extractor will now rewind this slot and extract the value ν. The protocol $\langle C_1, R_1 \rangle$ is as follows.

0. **Initialization Message.** The receiver $\mathcal{R}$ generates the first message σ of the Naor commitment scheme and sends it to $\mathcal{C}$.

1. **Primary Slot** a

   (a) The committer $\mathcal{C}$ generates $\ell[a]$ pairs of random strings $\{\alpha_i^0[a], \alpha_i^1[a]\}_{i\in[\ell[a]]}$ (with length of each string determined by the security parameter). $\mathcal{C}$ further generates commitments of these strings $\{A_i^0[a] = \mathrm{com}_\sigma(\alpha_i^0[a]), A_i^1[a] = \mathrm{com}_\sigma(\alpha_i^1[a])\}_{i\in[\ell[a]]}$ and sends them to $\mathcal{R}$ ($\mathcal{C}$ uses fresh randomness to generate each commitment).

   (b) The receiver $\mathcal{R}$ generates and sends to $\mathcal{C}$ a random $\ell[a]$-bit challenge string $ch[a] = (ch_1[a], \ldots, ch_{\ell[a]}[a])$.

   (c) The committer $\mathcal{C}$ sends to $\mathcal{R}$ the values $\alpha_1^{ch_1[a]}[a], \ldots, \alpha_{\ell[a]}^{ch_\ell[a]}[a]$. Note that $\mathcal{C}$ *does not* send the openings associated with the corresponding commitments.

2. **Primary Slot** b

   (a) The committer $\mathcal{C}$ generates $\ell[b]$ pairs of random strings $\{\alpha_i^0[b], \alpha_i^1[b]\}_{i\in[\ell[b]]}$ (with length of each string determined by the security parameter). $\mathcal{C}$ further generates commitments of these strings $\{A_i^0[b] = \mathrm{com}_\sigma(\alpha_i^0[b]), A_i^1[b] = \mathrm{com}_\sigma(\alpha_i^1[b])\}_{i\in[\ell[b]]}$ and sends them to $\mathcal{R}$ ($\mathcal{C}$ uses fresh randomness to generate each commitment).

   (b) The receiver $\mathcal{R}$ generates and sends to $\mathcal{C}$ a random $\ell[b]$-bit challenge string $ch[b] = (ch_1[b], \ldots, ch_{\ell[b]}[b])$.

   (c) The committer $\mathcal{C}$ sends to $\mathcal{R}$ the values $\alpha_1^{ch_1[b]}[b], \ldots, \alpha_{\ell[b]}^{ch_\ell[b]}[b]$. Note that $\mathcal{C}$ *does not* send the openings associated with the corresponding commitments.

3. **Verification Message.** Define $\ell[a]$ strings $\{\alpha_i[a]\}_{i\in[\ell[a]]}$ such that $\alpha_i[a] = \alpha_i^0[a] \oplus \alpha_i^1[a]$ for all $i \in [\ell[a]]$. $\mathcal{C}$ generates $\ell[a]$ commitments $B_i[a] = \mathrm{com}_\sigma(\nu; \alpha_i[a])$ for $i \in [\ell[a]]$ and sends them to $\mathcal{R}$. (That is, randomness $\alpha_i[a]$ is used to generate the i-th commitment to ν). Similarly compute commitments $B_i[b]$, $i \in [\ell[b]]$ in an analogous way and send them to $\mathcal{R}$.

4. **Consistency Proof.** The committer $\mathcal{C}$ and the receiver $\mathcal{R}$ now engage in a zero-knowledge argument protocol ZK where $\mathcal{C}$ proves to $\mathcal{R}$ that the above commit phase is "valid". That is, both the above primary slots and the verification message are correctly executed with the same value ν.

Decommitment Phase. The committer $\mathcal{C}$ simply reveals the committed value ν and the randomness used in the commitment phase. The receiver $\mathcal{R}$ checks if the commitment phase was run honestly using the above randomness (including making sure its a "valid" commit phase). If so, $\mathcal{R}$ takes the value committed to be ν and $\perp$ otherwise.

Proof Sketch. The proof of security of the above construction remains essentially identical to that of our basic construction. Keep in mind that $\mathcal{M}$ is a synchronizing adversary. Assume that $tag \neq \tilde{tag}$. This means that either $\ell[a] < \tilde{\ell}[a]$ or $\ell[b] < \tilde{\ell}[b]$. In the former case, the extractor $\mathcal{E}$ performs its rewindings for the primary slot a (by giving simulated responses for the challenges of $\mathcal{M}$ on the left). In the latter case, $\mathcal{E}$ rewinds the primary slot b assuming the messages before start of primary slot b as the prefix of the protocol. In both cases, the proof of security (and in particular the proof of all of our 3 key lemmas bounding the fraction of bad main threads) remains essentially identical.

Proving many-many security of the above non-malleable commitment scheme. To prove that our scheme is a many-many or concurrent non-malleable commitment scheme (for tags of length $\log(n)+1$), we first focus on proving one-many security. There are several right executions with tags $\tilde{tag}_1, \dots, \tilde{tag}_m$ and a left execution with tag tag. The interesting case is when $\tilde{tag}_i \neq tag$ for all $i \in [m]$. Our idea is to simply apply the extractor $\mathcal{E}$ one by one for all m sessions. More precisely, $\forall i \in [m]$:

- Define a machine $\mathcal{M}_i$ which "emulates" all the right sessions except session i on its own and exposes the i-th session to an outside receiver $\mathcal{R}_i$.
- Run the extractor on the machine $\mathcal{M}_i$ giving it as input the left view as in the main thread and the right view of the i-th session in the main thread.

The probability that the extractor fails can be computed by a union bound over the m right sessions (and can be made smaller than $\frac{1}{\text{poly}(k)}$ for any polynomial function $\text{poly}(k)$ as in the previous section). Following [LPV08], we get that the above construction is also a many-many non-malleable commitment scheme. Hence we get the following lemma.

Lemma 6 *The commitment scheme $\langle C_1, R_1 \rangle$ is a many-many non-malleable commitment scheme against synchronizing adversaries for tags of length* $\log(n)+1$ *(i.e., $tag \in [2n]$).*

Handing tags of length n. A many-many non-malleable commitment scheme for tags of length $\log(n)+1$ directly leads to a one-one non-malleable commitment scheme for tags of length n using the so called "DDN LOG N trick" [DDN91, LP09]. A construction for many-many non-malleable commitment scheme for tags of length n can also be directly obtained by a single step of non-malleability amplification from [LP09, Wee10]. In particular, we make a direct use of the following result from [Wee10].

Proposition 1 *(Proposition 3.1 in [Wee10]) Given a one-many commitment scheme $\langle C_1, R_1 \rangle$ for tags of length* $\log(n)+1$ *w.r.t. synchronizing adversaries, there exists another one-many (and hence many-many) commitment scheme $\langle C_2, R_2 \rangle$ for tags of length n w.r.t. synchronizing adversaries with only an additive constant increase in the round complexity.*

Security against Non-Synchronizing Adversaries. As is generally the case, once security against synchronizing adversaries is obtained, it is easy to extend it to obtain security even against a non-synchronizing adversary. A general result along these lines has been claimed by Wee [Wee10]. That is, [Wee10] presents a simple and general transformation of non-malleable commitment schemes that are secure against synchronizing adversaries into one that are secure against arbitrary scheduling strategies using one way functions with only an additive constant increase in round complexity. Applying this transformation to the commitment scheme $\langle C_2, R_2 \rangle$ (from proposition 1 yields a constant round non-malleable commitment scheme using only one way functions.

However since the details of this transformation are not yet available to us, we also provide an alternative protocol to get security against non-synchronizing adversaries.The protocol is a modification of the commitment scheme $\langle C_1, R_1 \rangle$ (for tags of length $\log(n)+1$). We first provide some intuition behind the modified protocol. Consider a non-synchronizing adversary $\mathcal{M}$. Our earlier proof (for synchronizing adversaries) runs into problems only when in the left interaction, $\mathcal{M}$ asks for the verification message *before* finishing the two primary slots in the right interaction. In this case, the proof of lemma 5 does not go through. This is since it relies on the inability of an adversary to distinguish between a correct value tuple from an incorrect value tuple for the unrecovered set of commitments. However given the verification message, indeed it is easy to explicitly distinguish the correct value tuple from an incorrect one. Thus to make our proof of security go through, we add additional "secondary slots" each of which represents a rewinding opportunity. If $\mathcal{M}$ asks for the verification message on the left *before* finishing the two primary slots on the right (in the main thread), it will be possible to exploit these additional rewinding opportunities on the right (such that $\mathcal{M}$ does not ask for messages in the left interaction while $\mathcal{E}$ is rewinding such slots).

Assume that the zero-knowledge protocol ZK has c_{zk} rounds of interaction between the prover and the verifier. The protocol $\langle C_3, R_3 \rangle$ proceeds as follows.

- **Initialization Message:** Identical to protocol $\langle C_1, R_1 \rangle$.
- **Primary Slot** a**:** Identical to protocol $\langle C_1, R_1 \rangle$.
- **Primary Slot** b**:** Identical to protocol $\langle C_1, R_1 \rangle$.
- $c_{zk} + 1$ **Secondary Slots:** For all $j \in [c_{zk} + 1]$, do the following.
  1. The committer $\mathcal{C}$ generates k pairs of random shares $\{\nu_i^0[j], \nu_i^1[j]\}_{i\in[k]}$ of the string ν (i.e., $\nu = \nu_i^0[j] \oplus \nu_i^1[j]$ for all i). $\mathcal{C}$ further generates commitments of these strings $\{C_i^0[j] = \mathrm{com}_\sigma(\nu_i^0[j]), C_i^1[j] = \mathrm{com}_\sigma(\nu_i^1[j])\}_{i\in[k]}$ and sends them to $\mathcal{R}$.
  2. The receiver $\mathcal{R}$ generates and sends to $\mathcal{C}$ a random k-bit challenge string $ch[j] = (ch_1[j], \ldots, ch_k[j])$.
  3. The committer $\mathcal{C}$ sends to $\mathcal{R}$ the committed shares $\nu_1^{ch_1[j]}[j], \ldots, \nu_k^{ch_k}[j]$ along with the corresponding openings.
- **Verification Message:** Identical to protocol $\langle C_1, R_1 \rangle$.
- **Consistency Proof:** The committer $\mathcal{C}$ and the receiver $\mathcal{R}$ now engage in a zero-knowledge argument protocol ZK where $\mathcal{C}$ proves to $\mathcal{R}$ that the entire commit phase above is "valid". That is, both the primary slots, the $c_{zk} + 1$ secondary slots and the verification messages are correctly executed with the same value ν.

PROOF. We consider following two different interleavings in the main thread:

- **Case 1: The verification message in the left interaction appears *before* the end of two primary slots in the right interaction.** This case constitutes the new part of our proof where the secondary slots will be useful. Observe that when this case happens:
  - Since the verification message appears after the secondary slots, *all* the secondary slots in the left interaction are executed (along with the verification message) before the primary slot b finishes on the right.
  - Consider the point where the primary slot b in the right interaction finishes. There are at most c_{zk} message remaining in the left interaction (i.e., message of the ZK protocol) and $c_{zk} + 1$ secondary slots remaining in the right interaction.
  - Hence, there exists at least one secondary slot in the right interaction such that during its execution, there are no message in the left interaction (pigeon-hole principle). Call this the secondary slot j.

  Now our extractor $\mathcal{E}$ will rewind the secondary slot j in the right interaction and extract the value ν by giving a different challenge. If during rewinding, $\mathcal{M}$ aborts or changes the scheduling to ask for a message of the ZK protocol (as opposed to its strategy in the main thread), $\mathcal{E}$ simply rewinds and tries with a different challenge. It is easy to see that the expected number of rewindings required is a constant (observe that $c_{zk}+1$ is a constant). Alternatively, given any $r(k) = \frac{1}{\text{poly}(k)}$, one can construct an extractor which performs a strict polynomial number of rewinds and succeeds with probability at least $(1 - r(k))$.

- **Case 2: The verification message in the left interaction appears *after* the end of two primary slots in the right interaction.** Our proof for this case is similar to the case for synchronizing adversaries. The only different is the consideration that the secondary slots (but not the verification message) in the left interaction might now appear before the primary slots in the right interaction finish. However during the rewinds, $\mathcal{E}$ does not have to provide the verification message or the final ZK protocol for consistency (if upon rewinding, $\mathcal{M}$ changes its scheduling to ask for such messages, $\mathcal{E}$ simply rewinds again). Hence during the rewinds, $\mathcal{E}$ simply runs the required secondary slot with the value 0 (as opposed to the real value ν being committed to in the left interaction). If the probability of $\mathcal{E}$ outputting $\mathsf{Ext\_Fail}$ changes by a noticeable amount, one can construct an adversary $\mathcal{A}$ to contradict the computational hiding property of the commitment scheme com_σ. In more detail, in the proof of lemma 5, the challenge *Chal* and $\mathcal{A}$ interact as follows. In addition to interacting with $\mathcal{A}$ to complete a primary slot (and giving candidate tuples), *Chal* now additionally allows interaction in *any polynomial* number of secondary slots as well. Thus, $\mathcal{A}$ can rewind $\mathcal{M}$ successfully M times; each time interacting with *Chal* to complete the secondary slots. Now consider the following two hybrid experiments. In the first hybrid, *Chal* has access to the correct value ν and executes the secondary slots honestly. In that case, essentially the same proof of success of $\mathcal{A}$ goes through. In the second hybrid, *Chal* uses the value 0 (as opposed to ν). By the computational hiding property of scheme com_σ, the success probability of $\mathcal{A}$ cannot change by a noticeable amount in the two hybrid.

Thus, the above gives us a non-malleable commitment scheme $\langle C_3, R_3 \rangle$ for non-synchronizing adversaries for tags of length $\log(n) + 1$. Similar to before, it can be shown that $\langle C_3, R_3 \rangle$ is also many-many non-malleable by applying the extractor on each right session one by one. By applying to DDN LOG N trick, we get a one-one non-malleable commitment scheme against non-synchronizing adversaries for tags of length n. A many-many non-malleable commitment scheme for tags of length n against such adversaries can be obtained by applying one step of the non-malleability amplification [LP09]. This gives us the following theorem.

Theorem 2 *There exists a constant round many-many non-malleable commitment scheme using only one way functions.*

3.3 Additional Results

Our techniques can be extended to get the following additional results.

Constant Round Multi-Party Computation. Our commitment scheme $\langle C_3, R_3 \rangle$ can be modified in a straightforward way so as to use the underlying one way function (i.e., the commitment scheme com_σ) only in a black-box way. This can be done by essentially removing the step where the committer gives a proof of consistency of the commitment to the receiver. This allows us to construct commitment scheme for tags of length n non-malleable *w.r.t. opening* and making only a black-box use of a one way function. The weaker security notion so obtained still turns out to be sufficient for application to secure multi-party computation. By relying on techniques from [LPV09, Wee10], this allows us to obtain the first construction of constant round secure multi-party computation making only a black-box use of standard cryptographic primitives.

Constant Round Non-Malleable Zero-Knowledge. The construction of Lin et al [LPTV10] can be instantiated based on any non-malleable commitment scheme and gives rise to a constant round non-malleable zero-knowledge assuming only one way functions (if one requires only one-one non-malleable zero-knowledge as opposed to concurrent). This immediately gives us a construction of constant round non-malleable zero-knowledge based only on one way functions. Constant round non-malleable zero-knowledge constructions can also be obtained by instantiating our commitment scheme using the protocols in [BPS06, GJO10] at the cost of requiring stronger computational assumptions.

Acknowledgements. We wish to thank Amit Sahai for many useful discussion and suggestions about the presentation of the proof of security. Thanks also to Abhishek Jain, Rafail Ostrovsky, Omkant Pandey and Ivan Visconti for useful discussions.

References

[Bar01] Boaz Barak. How to go beyond the black-box simulation barrier. In *FOCS*, pages 106–115, 2001.

[Bar02] Boaz Barak. Constant-round coin-tossing with a man in the middle or realizing the shared random string model. In *FOCS*, pages 345–355. IEEE Computer Society, 2002.

[BHY09] Mihir Bellare, Dennis Hofheinz, and Scott Yilek. Possibility and impossibility results for encryption and commitment secure under selective opening. In Antoine Joux, editor, *EUROCRYPT*, volume 5479 of *Lecture Notes in Computer Science*, pages 1–35. Springer, 2009.

[BPS06] Boaz Barak, Manoj Prabhakaran, and Amit Sahai. Concurrent non-malleable zero knowledge. In *FOCS*, pages 345–354, 2006.

[DDN91] Danny Dolev, Cynthia Dwork, and Moni Naor. Non-malleable cryptography (extended abstract). In *STOC*, pages 542–552. ACM, 1991.

[GJO10] Vipul Goyal, Abhishek Jain, and Rafail Ostrovsky. Password-authenticated session-key generation on the internet in the plain model. In Rabin [Rab10], pages 277–294.

[LP09] Huijia Lin and Rafael Pass. Non-malleability amplification. In Mitzenmacher [Mit09], pages 189–198.

[LPTV10] Huijia Lin, Rafael Pass, Wei-Lung Dustin Tseng, and Muthuramakrishnan Venkitasubramaniam. Concurrent non-malleable zero knowledge proofs. In Rabin [Rab10], pages 429–446.

[LPV08] Huijia Lin, Rafael Pass, and Muthuramakrishnan Venkitasubramaniam. Concurrent non-malleable commitments from any one-way function. In Ran Canetti, editor, *TCC*, volume 4948 of *Lecture Notes in Computer Science*, pages 571–588. Springer, 2008.

[LPV09] Huijia Lin, Rafael Pass, and Muthuramakrishnan Venkitasubramaniam. A unified framework for concurrent security: universal composability from stand-alone non-malleability. In Mitzenmacher [Mit09], pages 179–188.

[Mit09] Michael Mitzenmacher, editor. *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31 - June 2, 2009*. ACM, 2009.

[PPV08] Omkant Pandey, Rafael Pass, and Vinod Vaikuntanathan. Adaptive one-way functions and applications. In David Wagner, editor, *CRYPTO*, volume 5157 of *Lecture Notes in Computer Science*, pages 57–74. Springer, 2008.

[PR05a] Rafael Pass and Alon Rosen. Concurrent non-malleable commitments. In *FOCS*, pages 563–572, 2005.

[PR05b] Rafael Pass and Alon Rosen. New and improved constructions of non-malleable cryptographic protocols. In Harold N. Gabow and Ronald Fagin, editors, *STOC*, pages 533–542. ACM, 2005.

[PW10] Rafael Pass and Hoeteck Wee. Constant-round non-malleable commitments from sub-exponential one-way functions. In Henri Gilbert, editor, *EUROCRYPT*, volume 6110 of *Lecture Notes in Computer Science*, pages 638–655. Springer, 2010.

[Rab10] Tal Rabin, editor. *Advances in Cryptology - CRYPTO 2010, 30th Annual Cryptology Conference, Santa Barbara, CA, USA, August 15-19, 2010. Proceedings*, volume 6223 of *Lecture Notes in Computer Science*. Springer, 2010.

[Wee10] Hoeteck Wee. Black-box, round-efficient secure computation via non-malleability amplification. In *FOCS (to appear)*, 2010.