# Efficient Non-Malleable Commitment Schemes

Marc Fischlin and Roger Fischlin

Fachbereich Mathematik (AG 7.2)
Johann Wolfgang Goethe-Universität Frankfurt am Main
Postfach 111932
D–60054 Frankfurt/Main, Germany

{marc,fischlin}@mi.informatik.uni-frankfurt.de
http://www.mi.informatik.uni-frankfurt.de/

**Abstract.** We present efficient non-malleable commitment schemes based on standard assumptions such as RSA and Discrete-Log, and under the condition that the network provides publicly available RSA or Discrete-Log parameters generated by a trusted party. Our protocols require only three rounds and a few modular exponentiations. We also discuss the difference between the notion of non-malleable commitment schemes used by Dolev, Dwork and Naor [DDN00] and the one given by Di Crescenzo, Ishai and Ostrovsky [DIO98].

## 1 Introduction

Loosely speaking, a commitment scheme is non-malleable if one cannot transform the commitment of another person's secret into one of a related secret. Such non-malleable schemes are for example important for auctions over the Internet: it is necessary that one cannot generate a valid commitment of a bid $b+1$ after seeing the commitment of an unknown bid $b$ of another participant. Unfortunately, this property is not achieved by commitment schemes in general, because ordinary schemes are only designated to hide the secret. Even worse, most known commitment schemes are in fact provably malleable.

The concept of non-malleability has been introduced by Dolev et al. [DDN00]. They present a non-malleable public-key encryption scheme (based on any trapdoor permutation) and a non-malleable commitment scheme with logarithmically many rounds based on any one-way function. Yet, their solutions involve cumbersome non-interactive and interactive zero-knowledge proofs, respectively. While efficient non-malleable encryption schemes under various assumptions have appeared since then [BR93,BR94,CS98], as far as we know more efficient non-malleable commitment protocols are still missing. Di Crescenzo et al. [DIO98] present a non-interactive and non-malleable commitment scheme based on any one-way function in the common random string model. Though being non-interactive, their system is rather theoretical as it excessively applies an ordinary commitment scheme to non-malleably commit to a single bit.

Here, we present efficient statistically-secret non-malleable schemes based on standard assumptions, such as the RSA assumption and the hardness of computing discrete logarithms. Our schemes are designed in the public parameter model (a.k.a. auxilary string model). That is, public parameters like a random prime $p$ and generators of some subgroup of $\mathbb{Z}_p^*$ are generated and published by a trusted party. We stress that, in contrast to public-key infrastructure, this model does not require the participants to put any trapdoor information into the parameters. The public parameter model relies on a slightly stronger assumption than the common random string model. Yet, the difference is minor as modern networks are likely to provide public parameters for standard crypto systems. Moreover, as for the example of the discrete logarithm, the public parameter model can be formally reduced to the common random string model if we let the participants map the random string via standard procedures to a prime and appropriate generators.

In our schemes the sender basically commits to his message using an ordinary, possibly malleable DLog- or RSA-based commitment scheme and performs a three-round witness-independent proof of knowledge, both times using the public parameters. While the straightforward solution of a standard proof of knowledge fails (because the adversary may in addition to the commitment also transform the proof of knowledge), we force the adversary to give his "own" proof of knowledge without being able to adapt the one of the original sender. Similar ideas have also been used in [DDN00,DIO98]. In our case, the proof of knowledge guarantees that the adversary already knows the message he has committed to. This means that he is aware of some information about the related message of the original sender, contradicting the secrecy property of the ordinary commitment scheme.

We also address definitional issues. We show that the notion of non-malleability used by Di Crescenzo et al. [DIO98] is weaker than the one presented in [DDN00]. According to the definition of [DIO98], a scheme is non-malleable if the adversary cannot construct a commitment from a given one, such that after having seen the opening of the original commitment, the adversary is able to correctly open his commitment with a related message. But the definition of Dolev et al. [DDN00] demands more: if there is a one-to-one correspondence between the commitment and the message (say, if the commitment binds unconditionally), then they define that such a scheme is non-malleable if one cannot even generate a commitment of a related message. We call schemes having the latter property *non-malleable with respect to commitment*. For these schemes to contradict non-malleability it suffices to come up with a commitment such that there exists a related opening. Schemes satisfying the former definition are called *non-malleable with respect to decommitment* or, for sake of distinctiveness, *with respect to opening*. In this case, the adversary must also be able to open the modified commitment correctly given the decommitment of the original commitment. The scheme in [DDN00] achieves the stronger notion, whereas we do not know if the scheme in [DIO98] is also non-malleable with respect to commitment.

Clearly, a commitment scheme which is non-malleable in the strong sense is non-malleable with respect to opening, too. We stress that the other direction does not hold in general. That is, given a statistically-secret commitment scheme which is secure with respect to opening, we can devise a commitment scheme satisfying the weak notion, but not the strong definition. Since our statistically-secret schemes based on standard assumptions like RSA or Discrete-Log achieve non-malleability with respect to opening, both notions are *not* equivalent under these assumptions. The proof of this claim is deferred from this abstract.

We believe that non-malleability with respect to opening is the appropriate notion for statistically-secret schemes like ours. The reason is that for such schemes virtually any commitment can be opened with any message. Hence, finding a commitment of a related message to a given commitment is easy: any valid commitment works with very high probability. Recently, Yehuda Lindell informed us about an application of non-malleable commitment schemes to authenticated key-exchange where non-malleability with respect to commitment is necessary [L00]. Yet, non-malleability with respect to opening still seems to be adequate for most applications. For instance, recall the example of Internet auctions. The commitments of the bids are collected and then, after a deadline has passed, are requested to be opened. Any secret which is not correctly revealed is banned. Therefore, security with respect to opening suffices in this setting.

Our schemes as well as the one by [DDN00] use proof-of-knowledge techniques. But since we are merely interested in non-malleability with respect to opening, we do not need proofs of knowledge to the full extent. Namely, it suffices that the proof of knowledge is verifiable by the receiver after the sender has decommitted. Since the adversary must be able to open his commitment correctly, we can presume in the commitment phase that the proof of knowledge is indeed valid. This enables us to speed up our proofs of knowledge, i.e., we introduce new techniques for such *a-posteriori verifiable* proofs of knowledge based on the Chinese Remainder Theorem. As a side effect, this proof of knowledge allows to hash longer messages before committing and the resulting scheme still achieves non-malleability. In contrast to this, non-malleable schemes based on well-known proofs of knowledge do not seem to support the hash-and-commit paradigm in general.

The paper is organized as follows. In Section 2 we introduce basic notations and definitions of commitment schemes as well as the notions of non-malleability. In Section 3 we present efficient schemes in the public parameter model based on the discrete-log assumption, and, finally, in Section 4 we show how to speed up the proof of knowledge.

## 2 Preliminaries

Unless stated otherwise all parties and algorithms are probabilistic polynomial-time. Throughout this paper, we use the notion of uniform algorithms; all results transfer to the non-uniform model of computation. A function $\delta(n)$ is said to be *negligible* if $\delta(n) < 1/p(n)$ for every polynomial $p(n)$ and sufficiently large $n$. A function $\delta(n)$ is called *overwhelming* if $1 - \delta(n)$ is negligible. A function is *noticeable* if it is not negligible.

Two sequences $(X_n)_{n\in\mathbb{N}}$ and $(Y_n)_{n\in\mathbb{N}}$ of random variables are called *computationally indistinguishable* if for any probabilistic polynomial-time algorithm $\mathcal{D}$ the advantage

$$\left|\text{Prob}\left[\mathcal{D}(1^n, X_n) = 1\right] - \text{Prob}\left[\mathcal{D}(1^n, Y_n) = 1\right]\right|$$

of $\mathcal{D}$ is negligible, where the probabilities are taken over the coin tosses of $\mathcal{D}$ and the random choice of $X_n$ and $Y_n$, respectively. The sequences are called *statistically close* or *statistically indistinguishable* if

$$\tfrac{1}{2} \cdot \sum_{s \in S_n} \left|\text{Prob}\left[X_n = s\right] - \text{Prob}\left[Y_n = s\right]\right|$$

is negligible, where $S_n$ is the union of the supports of $X_n$ and $Y_n$.

## 2.1 Commitment Schemes

We give a rather informal definition of commitment schemes. For a formalization we refer the reader to [G98]. A commitment scheme is a two-phase interactive protocol between two parties, the sender $\mathcal{S}$ holding a message $m$ and a random string $r$, and the receiver $\mathcal{R}$.

In the first phase, called the commitment phase, $\mathcal{S}$ gives some information derived from $m, r$ to $\mathcal{R}$ such that, on one hand, $\mathcal{R}$ does not gain any information about $m$, and on the other hand, $\mathcal{S}$ cannot later change his mind about $m$. We call the whole communication in this phase the commitment of $\mathcal{S}$. Of course, both parties should check (if possible) that the values of the other party satisfy structural properties, e.g., that a value belongs to a subgroup of $\mathbb{Z}_p^*$, and should reject immediately if not. In the following, we do not mention such verification steps explicitely. We say that a commitment, i.e., the communication, is *valid* if the honest receiver does not reject during the commitment phase.

In the decommitment stage, the sender communicates the message $m$ and the randomness $r$ to the receiver, who verifies that $m, r$ match the communication of the first phase. If the sender obeys the protocol description, then the commitment is valid and $\mathcal{R}$ always accepts the decommitment.
There are two fundamental kinds of commitment schemes:

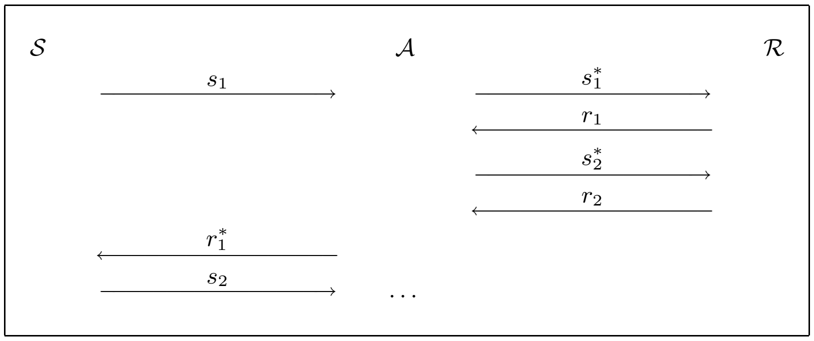
- A scheme is *statistically-binding (and computationally-secret)* if any arbitrary powerful malicious $\mathcal{S}^*$ cannot open a valid commitment ambiguously except with negligible probability (over the coin tosses of $\mathcal{R}$), and two commitments are computationally indistinguishable for every probabilistic polynomial-time (possibly malicious) $\mathcal{R}^*$. If the binding property holds unconditionally and not only with high probability, then we call the scheme unconditionally-binding.
- A scheme is *(computationally-binding and) statistically-secret* if it satisfies the “dual” properties, that is, if the distribution of the commitments are statistically close for any arbitrary powerful $\mathcal{R}^*$, and yet opening a valid commitment ambiguously contradicts the hardness of some cryptographic assumption. If the distribution of the commitments of any messages are identical, then a statistically-secret schemes is called perfectly-secret.

## 2.2 Non-Malleability

As mentioned in the introduction, different notions of non-malleability have been used implicitely in the literature. To highlight the difference we give a formal definition of non-malleable commitment schemes, following the approach of [DDN00]. For non-interactive commitment schemes, all the adversary can do is modify a given commitment. In the interactive case, though, the adversary might gain advantage from the interaction. We adopt this worst-case scenario and assume that the adversary interacts with the original sender, while at the same time he is trying to commit to a related message to the original receiver.

A pictorial description of a so-called *person-in-the-middle attack* (PIM attack) on an interactive protocol is given in Figure 1. The adversary $\mathcal{A}$ intercepts the messages of the sender $\mathcal{S}$. Then $\mathcal{A}$ may modify the messages before passing them to the receiver $\mathcal{R}$ and proceeds accordingly with the answers. In particular, $\mathcal{A}$ decides to whom he sends the next message, i.e., to the sender or to the receiver. This is the setting where $\mathcal{A}$ has full control over the parties $\mathcal{R}_1$ and $\mathcal{S}_2$ in two supposedly independent executions $\langle \mathcal{S}_1, \mathcal{R}_1 \rangle(m)$, $\langle \mathcal{S}_2, \mathcal{R}_2 \rangle(m^*)$ of the same interactive protocol. Here and in the rest of this paper, we usually mark values sent by the adversary with an asterisk.

**Fig. 1.** Person-In-The-Middle Attack on Interactive Protocols



Apparently, the adversary can always commit to the same message by forwarding the communication. In many applications, this can be prevented by letting the sender append his identity to the committed message. The messages of the sender and the adversary are taken from a space $\mathbb{M}$ chosen by the adversary. Abusing notations, we view $\mathbb{M}$ also as a distribution, and write $m \in_{\mathrm{R}} \mathbb{M}$ for a randomly drawn message according to $\mathbb{M}$. The adversary is deemed to be successful if he commits to a related message, where related messages are identified by so-called interesting relations: a relation $R \subseteq \mathbb{M} \times \mathbb{M}$ is called *interesting* if it is non-reflexive (to exclude copying) and efficiently computable. Let $\mathrm{hist}(\cdot)$ be a polynomial-time computable function, representing the a-priori information that the adversary has about the sender's message. In the sequel, we view $\text{hist}(\cdot)$ as a part of the adversary's description, and usually omit mentioning it explicitely.

We describe the attack in detail. First, the adversary $\mathcal{A}$ generates a description of $\mathbb{M}$. Then the public parameters are generated by a trusted party according to a publicly known distribution (if a protocol does not need public information then this step is skipped).[1] The sender $\mathcal{S}$ is initialized with $m \in_{\mathrm{R}} \mathbb{M}$. Now $\mathcal{A}$, given $\text{hist}(m)$, mounts a PIM attack with $\mathcal{S}(m)$ and $\mathcal{R}$. Let $\pi_{\mathsf{com}}(\mathcal{A}, R)$ denote the probability that, at the end of the commitment phase, the protocol execution between $\mathcal{A}$ and $\mathcal{R}$ constitutes a valid commitment for a message $m^*$ satisfying $(m, m^*) \in R$. Let $\pi_{\mathsf{open}}(\mathcal{A}, R)$ denote the probability that $\mathcal{A}$ is also able to successfully open the commitment after $\mathcal{S}$ has decommitted.

In a second experiment, a simulator $\mathcal{A}'$ tries to commit to a related message without the help of the sender. That is, $\mathcal{A}'$ first generates a description of $\mathbb{M}'$ and the public parameters and then, given $\text{hist}(m)$ for some $m \in_{\mathrm{R}} \mathbb{M}'$, it outputs a commitment communication without interacting with $\mathcal{S}(m)$. Let $\pi'_{\mathsf{com}}(\mathcal{A}', R)$ denote the probability that this communication is a valid commitment of a related message $m'$. By $\pi'_{\mathsf{open}}(\mathcal{A}', R)$ we denote the probability that $\mathcal{A}'$ additionally reveals a correct decommitment.

Note that all probabilities are implicit functions of a security parameter. For the definition we assume that messages contain a description of the distributions $\mathbb{M}$ and $\mathbb{M}'$, respectively, as prefix. This prevents $\mathcal{A}'$ from taking a trivial set $\mathbb{M}'$, since these sets can be ruled out by $R$. If the simulator $\mathcal{A}'$ sets $\mathbb{M}' = \mathbb{M}$ —which is the case in all the schemes we know of— we can omit the description portion.

**Definition 1.** *A commitment scheme is called*

*b) non-malleable with respect to commitment if for every adversary $\mathcal{A}$ there exists a simulator $\mathcal{A}'$ such that for all interesting relations $R$ the difference $|\pi_{\mathsf{com}}(\mathcal{A}, R) - \pi'_{\mathsf{com}}(\mathcal{A}', R)|$ is negligible.*

*a) non-malleable with respect to opening if for every adversary $\mathcal{A}$ there exists a simulator $\mathcal{A}'$ such that for all interesting relations $R$ the difference $\left|\pi_{\mathsf{open}}(\mathcal{A}, R) - \pi'_{\mathsf{open}}(\mathcal{A}', R)\right|$ is negligible.*

Slightly relaxing the definition, we admit an *expected* polynomial-time simulator $\mathcal{A}'$. In fact, we are only able to prove our schemes non-malleable with this deviation. The reason for this is that we apply proofs of knowledge, so in order to make the success probability of $\mathcal{A}'$ negligibly close to the adversary's success probability, we run a knowledge extractor taking expected polynomial-time.[2] Following the terminology in [DDN00], we call such schemes *liberal non-malleable* with respect to commitment and opening, respectively.

[1] In a stronger requirement the order of these steps is swapped, i.e., the adversary chooses the message space in dependence of the public parameters. Although our scheme achieves this stronger notion, we defer this from this abstract.

[2] The same problem occurs in [DDN00]. Alternatively, the authors of [DDN00] also propose a definition of $\epsilon$-malleability, which basically says that for given $\epsilon$ there is a strict polynomial-time simulator (polynomial in the security parameter $n$ and $\epsilon^{-1}(n)$) whose success probability is only $\epsilon$-far from the adversary's probability.

Consider a computationally-binding and perfectly-secret commitment scheme. There, every valid commitment is correctly openable with every message (it is, however, infeasible to find different messages that work). Thus, we believe that non-malleability with respect to opening is the interesting property in this case. On the other hand, non-malleability with respect to commitment is also a concern for statistically-binding commitment schemes: with overwhelming probability there do not exist distinct messages that allow to decommit correctly. This holds for *any dishonest* sender and, in particular, for the person-in-the-middle adversary. We can therefore admit this negligible error and still demand non-malleability with respect to commitment.

## 3 Efficient Non-Malleable Commitment Schemes

In this section we introduce our commitment schemes which are non-malleable with respect to opening. For lack of space, we only present the discrete-log scheme; the RSA-based protocol is omitted. In Section 3.1 we start with an instructive attempt to achieve non-malleability by standard proof-of-knowledge techniques. We show that this approach yields a scheme which is only non-malleable with respect to opening against static adversaries, i.e., adversaries that try to find a commitment after passively observing a commitment between the original sender and receiver and such that the adversary can later correctly open the commitment after learning the decommitment of the sender. In Section 3.2 we develop out of this our scheme which is non-malleable against the stronger PIM adversaries.

### 3.1 Non-Malleability with Respect to Static Adversaries

Consider Pedersen's well-known discrete-log-based perfectly-secret scheme [P91]. Let $G_q \subseteq \mathbb{Z}_p^*$ be a group of prime order $q$ and $g_0, h_0$ two random generators of $G_q$. Assume that computing the discrete logarithm $\log_{g_0} h_0$ is intractable. To commit to a message $m \in \mathbb{Z}_q$, choose $r \in_{\mathrm{R}} \mathbb{Z}_q$ and set $M := g_0^m h_0^r$. To open this commitment, reveal $m$ and $r$. Obviously, the scheme is perfectly-secret as $M$ is uniformly distributed in $G_q$, independently of the message. It is computationally-binding because opening a commitment with distinct messages requires computing $\log_{g_0} h_0$.

Unfortunately, Pedersen's scheme is malleable: given a commitment $M$ of some message $m$ an adversary obtains a commitment for $m+1 \bmod q$ by multiplying $M$ with $g$. Later, the adversary reveals $m+1 \bmod q$ and $r$ after learning the original decommitment $m, r$. This holds even for static adversaries. Such adversaries do not try to inject messages in executions, but rather learn a protocol execution of $\mathcal{S}$ and $\mathcal{R}$ —which they cannot influence— and afterwards try to commit to a related message to $\mathcal{R}$. As for non-malleability with respect to opening, the adversary must also be able to open the commitment after the sender has decommitted.

A possible fix that might come to one's mind are proofs of knowledge showing that the sender actually knows the message encapsulated in the commitment. For the discrete-log case such a proof of knowledge consists of the following steps [O92]: the sender transmits a commitment $S := g_0^s h_0^t$ of a random value $s \in_{\mathrm{R}} \mathbb{Z}_q$, the receiver replies with a random challenge $c \in_{\mathrm{R}} \mathbb{Z}_q$ and the sender answers with $y := s + cm \bmod q$ and $z := t + cr \bmod q$. The receiver finally checks that $SM^c = g_0^y h_0^z$.

If we add a proof of knowledge to Pedersen's scheme we obtain a protocol which is non-malleable with respect to opening against static adversaries. This follows from the fact that any static adversary merely sees a commitment of an unknown message before trying to find an appropriate commitment of a related message. Since the proof of knowledge between $\mathcal{S}$ and $\mathcal{R}$ is already finished at this time, the static adversary cannot rely on the help of $\mathcal{S}$ and transfer the proof of knowledge. We leave further details to the reader and focus instead on the non-malleable protocol against PIM adversaries in the next section.

## 3.2 Non-Malleability with Respect to PIM Adversaries

The technique of assimilating a proof of knowledge as in the previous section does not thwart PIM attacks. Consider again the PIM adversary committing to $m + 1 \bmod q$ by multiplying $M$ with $g$. First, this adversary forwards the sender's commitment $S$ for the proof of knowledge to the receiver and hands the challenge $c$ of the receiver to the sender. Conclusively, he modifies the answer $y, z$ of the sender to $y^* := y + c \bmod q$ and $z^* := z$. See Figure 2. Clearly, this is a valid proof of knowledge for $m + 1 \bmod q$ and this PIM adversary successfully commits and later decommits to a related message.

Coin-flipping comes to rescue. In a coin flipping protocol one party commits to a random value $a$, then the other party publishes a random value $b$, and finally the first party decommits to $a$. The result of this coin flipping protocol is set to $c := a \oplus b$ or, in our case, to $c := a + b \bmod q$ for $a, b \in \mathbb{Z}_q$. If at least one party is honest, then the outcome $c$ is uniformly distributed (if the commitment scheme is "secure").

The idea is now to let the challenge in our proof of knowledge be determined by such a coin-flipping protocol. But if we too use Pedersen's commitment scheme with the public generators $g_0, h_0$ to commit to value $a$ in this coin-flipping protocol, we do not achieve any progress: the adversary might be able to commit to a related $a^*$ and thus bias the outcome of the coin-flipping to a suitable challenge $c^*$. The solution is to apply Pedersen's scheme in this sub protocol with the commitment $M$ as one of the generators, together with an independent generator $h_1$ instead of $g_0, h_0$; for technical reasons we rather use $(g_1 M)$ and $h_1$ for another generator $g_1$. As we will show, since the coin-flipping in the proof of knowledge between $\mathcal{A}$ and $\mathcal{R}$ is based on generators $g_1 M^*$ and $h_1$ instead of $g_1 M, h_1$ as in the sender's proof of knowledge, this prevents the adversary from adopting the sender's and receiver's values and therefore to transfer the proof of knowledge. Details follow.

**Fig. 2.** PIM Attack on Pedersen's Commitment Scheme with Proof of Knowledge

| sender $\mathcal{S}$ | | adversary $\mathcal{A}$ | | receiver $\mathcal{R}$ |
|---|---|---|---|---|
| message $m \in \mathbb{Z}_q$ | | public: $p, q, g_0, h_0$ | | |
| *a) commitment phase*: | | | | |
| choose $r, s, t \in_{\mathrm{R}} \mathbb{Z}_q$<br>set $M := g_0^m h_0^r$<br>set $S := g_0^s h_0^t$ | $\xrightarrow{M,S}$ | $S^* := S$<br>$M^* := gM$ | $\xrightarrow{M^*,S^*}$ | choose $c \in_{\mathrm{R}} \mathbb{Z}_q$ |
| | | | $\xleftarrow{c}$ | |
| | | $c^* := c$ | | |
| | $\xleftarrow{c^*}$ | | | |
| $y := s + c^* m \ (q)$<br>$z := t + c^* r \ (q)$ | $\xrightarrow{y,z}$ | $z^* := z$<br>$y^* := y + c \ (q)$ | $\xrightarrow{y^*,z^*}$ | verify that<br>$S^*(M^*)^c \overset{!}{=} g_0^{y^*} h_0^{z^*}$ |
| *b) decommitment phase*: | | | | |
| | $\xrightarrow{m,r}$ | $r^* := r$<br>$m^* := m + 1 \ (q)$ | $\xrightarrow{m^*,r^*}$ | verify that<br>$M^* \overset{!}{=} g_0^{m^*} h_0^{r^*}$ |

We describe the protocol given in Figure 3 which combines the aforementioned ideas. The public parameters are primes $p, q$ with $q|(p-1)$ together with four random generators $g_0, g_1, h_0, h_1$ of a subgroup $G_q \subseteq \mathbb{Z}_p^*$ of prime order $q$. Our protocol also works for other cyclic groups of prime order $q$ like elliptic curves, but we explain for the case $G_q \subseteq \mathbb{Z}_p^*$ only. Basically, the sender $\mathcal{S}$ commits to his message $m \in \mathbb{Z}_q^*$ with Pedersen's scheme[3] by computing $M = g_0^m h_0^r$ and proves by a proof of knowledge (values $S, c, y, z$ in Figure 3) that he is aware of a valid opening of the commitment. The challenge $c$ in this proof of knowledge is determined by a coin-flipping protocol with values $A, a, u, b$.

It is clear that our protocol is computationally-binding under the discrete-log assumption, and perfectly-secret as the additional proof of knowledge for $m$ is *witness-independent* (a.k.a. perfectly witness-indistinguishable) [FS90], i.e., for any challenge $c$ the transmitted values $S, y, z$ are distributed independently of the actual message [O92].

[3] Note that as opposed to Pedersen's scheme we require that $m \neq 0$; the technical reason is that in the security proof we need to invert the message modulo $q$.

**Fig. 3.** DLog-Based Non-Malleable Commitment Scheme

| **Sender $\mathcal{S}$** | $p, q, g_0, g_1, h_0, h_1$ | **Receiver $\mathcal{R}$** |
|---|---|---|
| message $m \in \mathbb{Z}_q^*$ | | |
| *a) commitment phase*: | | |
| choose $a, r, s, t, u \in_{\mathrm{R}} \mathbb{Z}_q$<br>set $M := g_0^m h_0^r$<br>set $A := (g_1 M)^a h_1^u$<br>set $S := g_0^s h_0^t$ | $\xrightarrow{M, A, S}$ | choose $b \in_{\mathrm{R}} \mathbb{Z}_q$ |
| | $\xleftarrow{b}$ | |
| set $c := a + b \bmod q$<br>set $y := s + cm \bmod q$<br>set $z := t + cr \bmod q$ | $\xrightarrow{a, u, y, z}$ | set $c := a + b \bmod q$<br>check $A \overset{!}{=} (g_1 M)^a h_1^u$<br>check $SM^c \overset{!}{=} g_0^y h_0^z$ |
| *b) decommitment phase*: | | |
| | $\xrightarrow{m, r}$ | check $M \overset{!}{=} g_0^m h_0^r$ |

**Proposition 1.** *The commitment scheme in Figure 3 is perfectly-secret and, under the discrete-log assumption, computationally-binding.*

It remains to show that our scheme is non-malleable. We present the proof from a bird's eye view and fill in more details in Appendix A, yet remain sketchily in this version. By now, we already remark that the non-malleability property of our scheme also relies on the hardness of computing discrete logarithms. This dependency is not surprising: after all, any adversary being able to compute discrete logarithms with noticeable probability also refutes the binding property of Pedersen's scheme and can thus decommit for any related message with this probability.

The idea of the proof is as follows. Given a commitment $M$ of some unknown message $m$ (together with a witness-independent proof of knowledge described by $S, c, y, z$) with respect to parameters $p, q, g_0, h_0$ we show how to employ the PIM adversary $\mathcal{A}$ to derive some information about $m$. Namely, if we are able to learn the related message $m^*$ of the adversary, then we know that $m$ satisfies $(m, m^*) \in R$ for the relation $R$. This, of course, contradicts the perfect secrecy of the commitment scheme.

In this discussion here, we make two simplifications concerning the adversary: first, we assume that the PIM adversary always catches up concerning the order of the transmissions, i.e., sends his first message after learning the first message of

$\mathcal{S}$ and answers to $\mathcal{S}$ after having seen $\mathcal{R}$'s response etc. Second, let the adversary *always* successfully commit and decommit to a related message, rather than with small probability. Both restrictions can be removed.

The fact that we learn the adversary's message $m^*$ follows from the proof of knowledge. Intuitively, a proof of knowledge guarantees that the prover knows the message, i.e., one can extract the message by running experiments with the prover. Specifically, we inject values $p, q, g_0, h_0, M, S, c, y, z$ into a simulated PIM attack with $\mathcal{A}$ and impersonate $\mathcal{S}$ and $\mathcal{R}$. Additionally, we choose $g_1$ at random and set $h_1 := (g_1 M)^w$ for a random $w \in_{\mathrm{R}} \mathbb{Z}_q$. We also compute random $a_0, u_0 \in_{\mathrm{R}} \mathbb{Z}_q$ and insert $g_1, h_1$ and $A := (g_1 M)^{a_0} h_1^{u_0}$ into the experiment with $\mathcal{A}$. We start with the extraction procedure by committing to $m, s, a_0$ via $M, S, A$ on behalf of the sender. Then, by the predetermination about the order of the transmissions, the adversary sends $M^*, S^*, A^*$ (possibly by changing $M, S, A$ and without knowing explicitely the corresponding values $m^*, r^*$ etc.). See Figure 5 on page 431 for a pictorial description.

We play the rest of the commitment phase twice by rewinding it to the step where the receiver chooses $b$ and sends it to the adversary $\mathcal{A}$. To distinguish the values in both repetitions we add the number of the loop as subscript and write $a_1, a_1^*, a_2, a_2^*$ etc. The first time, the adversary, upon receiving $b_1$, passes some $b_1^*$ to the (simulated) sender $\mathcal{S}$, and expects $\mathcal{S}$ to open the commitment for $a$ and supplement the proof of knowledge for $M$ with respect to the challenge $a_1 + b_1^* \bmod q$. By the trapdoor property of Pedersen's commitment scheme [BCC88] we are able to open $A$ with any value for $a_1$ since we know $\log_{(g_1 M)} h_1$. That is, to decommit $A$ with some $a_1$ reveal $a_1$ and $u_1 = u_0 + (a_0 - a_1)/\log_{(g_1 M)} h_1 \bmod q$; it is easy to verify that indeed $A = (g_1 M)^{a_1} h_1^{u_1}$. In particular, we choose $a_1$ such that $a_1 + b_1^* \bmod q$ equals the given value $c$. Hence, $y$ and $z$ are proper values to complement the proof of knowledge for $M$. Finally, the adversary answers with the decommitment $a_1^*, u_1^*$ for $A^*$ and the rest of the proof of knowledge for $M^*$ with respect to challenge $a_1^* + b_1 \bmod q$. Now we rewind the execution and select another random challenge $b_2$. The adversary then decides upon his value $b_2^*$ (possibly different from his previous choice $b_1^*$) and hands it to $\mathcal{S}$. Again, we open $A$ with $a_2$ such that $c = a_2 + b_2^* \bmod q$. The adversary finishes his commitment with $a_2^*, u_2^*$ as opening for $A^*$ and the missing values for the proof of knowledge.

The fundamental proof-of-knowledge paradigm [FFS88] says that we can extract the message $m^*$ if we learn two valid executions between $\mathcal{A}$ and $\mathcal{R}$ with the same commitment $M^*, S^*, A^*$ but different challenges. Hence, if the adversary's decommitments satisfy $a_1^* = a_2^*$ and we have $b_1 \neq b_2$ (which happens with probability $1 - 1/q$), then this yields different challenges $a_1^* + b_1$, $a_2^* + b_2$ in the executions between $\mathcal{A}$ and $\mathcal{R}$ and we get to know the message $m^*$. We are therefore interested in the event that the adversary is able to "cheat" by presenting different openings $a_1^* \neq a_2^*$. In Appendix A we prove that if the adversary finds different openings for commitment $A^*$ with noticeable probability, then we derive a contradiction to the intractability of the discrete-log problem. Hence, under the discrete-log assumption the probability that this event occurs is negligible and we extract $m^*$ with overwhelming probability.

Note that that in the repetitions we force the coin-flipping protocol between $\mathcal{S}$ and $\mathcal{A}$ to result in the same challenge both times. The latter is necessary because if we were able to answer a different challenge than $c$ then we could extract the unknown message $m$ and would thus know $m$ (which is of course not the case).

In conclusion, if the adversary is able to commit and decommit to a related message $m^*$, then we can extract $m^*$ and learn something about $m$. For details and further discussion we refer to Appendix A. Altogether,

**Theorem 1.** *Under the discrete-logarithm assumption, the scheme in Figure 3 is a perfectly-secret commitment scheme which is liberal non-malleable with respect to opening.*

It is worthwhile to point out that we cannot hash longer messages to $\mathbb{Z}_q^*$ before applying our non-malleable commitment scheme. Because then we extract the hash value and not the message $m^*$ itself. But this could be insufficient, since it might be impossible to deduce anything about $m$ via $R(m, m^*)$ given solely the hash value of $m^*$. We stress that the schemes in Section 4 with the faster a-posteriori verifiable proofs of knowledge do not suffer from this problem. There, one can first hash the message as the proof of knowledge operates on the original message instead of the hash value.

## 4 Speeding Up the Proof of Knowledge

The DLog-based scheme in the previous section (as well as the RSA-based one) uses Okamoto's witness-independent proof of knowledge. But since we are interested in non-malleability with respect to opening, the proof of knowledge need not be verifiable immediately in the commitment phase. It suffices that the sender convinces the receiver of the proof's validitiy in the decommitment stage. To refute non-malleability, the adversary must open his commitment correctly, and particularly, the proof must be shown to be right then. Therefore, the simulator can already in the commitment phase assume that the proof is indeed valid. We call such a proof of knowledge *a-posteriori verifiable.*

Using the Chinese Remainder Theorem, we present a very fast a-posteriori verifiable proof of knowledge and thus a faster variant of the non-malleable commitment scheme given in Section 3. Assume that we hash messages of polynomial length to $\mathbb{Z}_q^*$ with a collision-intractable hash function $\mathcal{H}$ (whose description is part of the public parameters). Given the DLog commitment $M = g_0^{\mathcal{H}(m)} h_0^r$ of the hash value $\mathcal{H}(m)$, the proof of knowledge consists of two steps: the receiver selects a small random prime $P$ as challenge and the sender answers with $(m, r)$ —viewed as a natural number— reduced $\bmod P$. If we repeat this for several primes, then we reconstruct the number $(m, r)$ using the Chinese Remainder Theorem. This corresponds to the case of RSA or DLog proofs of knowledge, where distinct challenges yield a representation. Yet, our proof reveals some information about $(m, r)$. To prevent this, we add a sufficiently large random prefix $s \gg P$ to the message $m$ and use $\mathcal{H}(s, m)$ instead of $\mathcal{H}(m)$ in $M$. Then

$(s, m, r)$ mod $P$ hides $m, r$ statistically. Note that the receiver cannot check immediately that the answer to the challenge is right. But a simulator, repeating the proof of knowledge with different primes, is able to verify that it has reconstructed the correct value $(s, m, r)$ by comparing $g_0^{\mathcal{H}(s',m')} h_0^{r'}$ for the extracted values $s', m', r'$ to $M$.

We introduce some notations. Let $\boldsymbol{x} \cdot \boldsymbol{y}$ denote the concatenation of two binary strings $\boldsymbol{x}, \boldsymbol{y} \in \{0,1\}^*$ and $\mathrm{val}(\boldsymbol{x}) := \sum_{i \geq 1} x_i 2^{i-1}$ the value of string $\boldsymbol{x} = x_1 x_2 \ldots x_n$. Conversely, denote by $\mathrm{bin}(r)$ the standard binary representation of number $r$. Suppose that we are given an efficiently computable function $\mathbb{P}_{K,k/2}$ mapping $K$-bit strings to $k/2$-bit primes such that for $\boldsymbol{x} \in_{\mathrm{R}} \{0,1\}^K$ the prime $\mathbb{P}_{K,k/2}(\boldsymbol{x})$ is uniformly distributed in a superpolynomial subset of all $k/2$-bit primes. See [CS99,M95] for fast algorithms to generate primes from random strings. We will later present an alternative proof technique based on polynomials instead of primes.

**Fig. 4.** Non-Malleable Commitment Scheme with fast a-posteriori verifiable POK

| **Sender $\mathcal{S}$** | $p, q, g_0, g_1, h_0, h_1, \mathcal{H}$ | **Receiver $\mathcal{R}$** |
|---|---|---|
| message $\boldsymbol{m} \in \{0,1\}^\ell$ | | |
| *a) commitment phase*: | | |
| choose $\boldsymbol{a} \in_{\mathrm{R}} \{0,1\}^K$ | | |
| choose $\boldsymbol{s} \in_{\mathrm{R}} \{0,1\}^k$ | | |
| choose $r, u \in_{\mathrm{R}} \mathbb{Z}_q$ | | |
| set $M := g_0^{\mathcal{H}(\boldsymbol{s} \cdot \boldsymbol{m})} h_0^r$ | | |
| set $A := (g_1 M)^{\mathcal{H}(\boldsymbol{a})} h_1^u$ | $\xrightarrow{M, A}$ | choose $\boldsymbol{b} \in_{\mathrm{R}} \{0,1\}^K$ |
| | $\xleftarrow{\boldsymbol{b}}$ | |
| set $P := \mathbb{P}_{K,k/2}(\boldsymbol{a} \oplus \boldsymbol{b})$ | | |
| set $y := \mathrm{val}(\boldsymbol{s} \cdot \boldsymbol{m} \cdot \mathrm{bin}(r)) \bmod P$ | | |
| | $\xrightarrow{\boldsymbol{a}, u, y}$ | check $A \overset{!}{=} (g_1 M)^{\mathcal{H}(\boldsymbol{a})} h_1^u$ |
| *b) decommitment phase*: | | |
| | $\xrightarrow{\boldsymbol{s}, \boldsymbol{m}, r}$ | check $M \overset{!}{=} g_1^{\mathcal{H}(\boldsymbol{s} \cdot \boldsymbol{m})} h_0^r$ |
| | | set $P := \mathbb{P}_{K,k/2}(\boldsymbol{a} \oplus \boldsymbol{b})$ |
| | | check $y \overset{!}{=} \mathrm{val}(\boldsymbol{s} \cdot \boldsymbol{m} \cdot \mathrm{bin}(r)) \bmod P$ |

The modified commitment scheme is given in Figure 4. We replace the commitment $M = g_0^m h_0^r$ by $M = g_0^{\mathcal{H}(\boldsymbol{s} \cdot \boldsymbol{m})} h_0^r$. Additionally, the proof of the DLog-based scheme shows that we can also hash the value $\boldsymbol{a}$ for the coin flips before computing the commitment $A$. This enables us to generate longer random strings from a single commitment. Moreover, we now use the Chinese-Remainder-based proof of knowledge instead of the one by Okamoto. To commit to $\boldsymbol{m} \in \{0,1\}^\ell$ the sender $\mathcal{S}$ prepends $k$ random bits $\boldsymbol{s} \in_{\mathrm{R}} \{0,1\}^k$ to the message. To prove knowledge of $\mathrm{val}(\boldsymbol{s} \cdot \boldsymbol{m} \cdot \mathrm{bin}(r))$ the sender reveals the residue

$$\mathrm{val}(\boldsymbol{s} \cdot \boldsymbol{m} \cdot \mathrm{bin}(r)) \equiv 2^k \, \mathrm{val}(\boldsymbol{m} \cdot \mathrm{bin}(r)) + \mathrm{val}(\boldsymbol{s}) \pmod P$$

modulo a $k/2$-bit prime $P$. The value $\mathrm{val}(\boldsymbol{s}) \in [0, 2^k)$ acts as a one-time pad since its distribution modulo $P$ is statistically close to the uniform distribution on $[0, P)$.

We omit the formal proof that the scheme is liberal non-malleable with respect to opening under the discrete-logarithm assumption and given that $\mathcal{H}$ is collision-intractable (or, more precisely, given that $\mathcal{H}$ is a family of collision-intractable hash functions). As for the proof of knowledge, the extractor rewinds the adversary $\mathcal{A}$ which mounts a PIM attack, and forces $\mathcal{A}$ to answer different primes while the original sender always has to answer to the same prime. Suppose that $k = \ell = \log q$. Thus, if $\mathcal{A}$ correctly reveals the residues of $\mathrm{val}(\boldsymbol{s} \cdot \boldsymbol{m} \cdot \mathrm{bin}(r))$ modulo seven different primes $P_1, \ldots, P_7$, then using the Chinese Remainder Theorem we retrieve $\mathrm{val}(\boldsymbol{s} \cdot \boldsymbol{m} \cdot \mathrm{bin}(r))$, because $\mathrm{val}(\boldsymbol{s} \cdot \boldsymbol{m} \cdot \mathrm{bin}(r)) \leq 2^{3k} < P_1 \cdots P_7$. If $\mathcal{A}$ has only a noticeable success probability, then the extractor creates a list of residues modulo a polynomial number of primes and applies the Chinese Remainder Theorem to all subsets of seven residues finding the right values with overwhelming probability. Using the more sophisticated algorithm by Goldreich, Ron and Sudan [GRS99] or the improved one by Boneh [B00], we derive:

**Theorem 2.** *Under the discrete-logarithm assumption and if $\mathcal{H}$ is a (family of) collision-intractable hash function(s), then for all polynomials $K, k, \ell$ with $K, \ell = poly(k)$ and $k \geq (\log q)^{1-\epsilon}$ for a constant $\epsilon > 0$ the scheme in Figure 4 is a statistically-secret commitment scheme which is liberal non-malleable with respect to opening.*

The bottleneck of this scheme is the generation of primes. An even faster approach is based on polynomials over the finite Field $\mathbb{F}$ with $2^{\sqrt{k}}$ elements. Let $K = k$ and $\Pi : \{0,1\}^* \to \mathbb{F}[\tau]$ denote the mapping which maps a bit string of length $d\sqrt{k}$ to a monic polynomial of degree $d$ over $\mathbb{F}$ by taking every block of $\sqrt{k}$ bits as a coefficient of the polynomial. In Figure 4, we replace the prime $P$ by the polynomial $P(\tau) := \Pi(\boldsymbol{a} \oplus \boldsymbol{b})$ of degree $\sqrt{k}$ and set $y(\tau) := \Pi(\boldsymbol{s} \cdot \boldsymbol{m} \cdot \mathrm{bin}(r)) \bmod P(\tau)$. To retrieve $\boldsymbol{s}, \boldsymbol{m}, r$ from the proof of knowledge, we apply the Chinese Remainder Theorem for polynomials [K98, 4.6.2, Ex. 3]. Two randomly chosen monic polynomials of same degree over $\mathbb{F}$ are co-prime with probability $1 - 2^{-\sqrt{k}}$ [K98, 4.6.5, Ex. 5]. For instance, if $\log q = \ell = k = K$, then given $\Pi(\boldsymbol{s} \cdot \boldsymbol{m} \cdot \mathrm{bin}(r))$ modulo four randomly chosen monic polynomials $P_1, P_2, P_3, P_4 \in \mathbb{F}[\tau]$ of degree $\sqrt{k}$, we retrieve with overwhelming probability the polynomial $\Pi(\boldsymbol{s} \cdot \boldsymbol{m} \cdot \mathrm{bin}(r))$ modulo $(P_1 P_2 P_3 P_4)$. This yields $\boldsymbol{s}, \boldsymbol{m}$ and $r$, because $\deg \Pi(\boldsymbol{s} \cdot \boldsymbol{m} \cdot \mathrm{bin}(r)) \leq 3\sqrt{k} < \deg(P_1 P_2 P_3 P_4)$.

## Acknowledgments

We are indebted to Cynthia Dwork for discussions about non-malleability. We also thank the participants of the Luminy 1999 crypto workshop for stimulating discussions, as well as the Crypto 2000 reviewers and program committee, especially Shai Halevi. We are also grateful to Yehuda Lindell.

## References

[BG92] M. Bellare and O. Goldreich: *On Defining Proofs of Knowledge*, Advances in Cryptology — Proceedings Crypto '92, Lecture Notes in Computer Science, vol. 740, pp. 390–420, Springer Verlag, 1993.

[BR94] M. Bellare and P. Rogaway: *Optimal Asymmetric Encryption*, Advances in Cryptology — Proceedings Eurocrypt '94, Lecture Notes in Computer Science, vol. 950, pp. 92–111, Springer Verlag, 1993.

[BR93] M. Bellare and P. Rogaway: *Random Oracles are Practical: a Paradigm for Designing Efficient Protocols*, First ACM Conference on Computer and Communication Security, ACM Press, pp. 62–73, 1993.

[B00] D. Boneh: *Finding Smooth Integers Using CRT Decoding*, to appear in Proceedings of the $32^{\text{nd}}$ Annual ACM Symposium on Theory of Computing (STOC), ACM Press, 2000.

[BCC88] G. Brassard, D. Chaum and C. Crépeau: *Minimum Disclosure Proofs of Knowledge*, Journal of Computer and Systems Science, vol. 37(2), pp. 156–189, 1988.

[CS98] R. Cramer and V. Shoup: *A Practical Public Key Cryptosystem Provable Secure Against Adaptive Chosen Ciphertext Attack*, Advances in Cryptology — Proceedings Crypto '98, Lecture Notes in Computer Science, vol. 1492, pp. 13–25, Springer Verlag, 1998.

[CS99] R. Cramer and V. Shoup: *Signature Schemes Based on the Strong RSA Assumption*, ACM Conference on Computer and Communication Security, ACM Press, 1999.

[DIO98] G. Di Crescenzo, Y. Ishai and R. Ostrovsky: *Non-interactive and Non-Malleable Commitment*, Proceedings of the $30^{\text{th}}$ Annual ACM Symposium on Theory of Computing (STOC), pp. 141–150, ACM Press, 1998.

[DDN00] D. Dolev, C. Dwork and M. Naor: *Non-Malleable Cryptography*, manuscript, to appear in SIAM Jornal on Computing, January 2000. Preliminary version in Proceedings of the $21^{\text{st}}$ Annual ACM Symposium on Theory of Computing (STOC), pp. 542–552, ACM Press, 1991.

[FFS88] U. Feige, A. Fiat and A. Shamir: *Zero-Knowledge Proofs of Identity*, Journal of Cryptology, vol. 1(2), pp. 77–94, Springer-Verlag, 1988.

[FS90] A. Fiat and A. Shamir: *Witness Indistinguishable and Witness Hiding Protocols* Proceedings of the $22^{\text{nd}}$ Annual ACM Symposium on the Theory of Computing (STOC), pp. 416–426, ACM Press, 1990.

[G98] O. Goldreich: **Foundations of Cryptography**, Fragments of a Book, Version 2.03, 1998.

[GRS99] O. Goldreich, D. Ron and M. Sudan: *Chinese Remainder With Errors*, Proceedings of the $31^{\text{st}}$ Annual ACM Symposium on Theory of Computing (STOC), pp. 225–234, ACM Press, 1999.

[K98] D.E. Knuth: **Seminumerical Algorithms**, The Art of Computer Programming, vol. 2, 3rd edition, Addison Wesley, 1998.

[L00] Y. Lindell: Personal communication, based on work on authenticated key-exchange with Oded Goldreich. May 2000.

[M95] U. Maurer: *Fast Generation of Prime Numbers and Secure Public-Key Cryptographic Parameters*, Journal of Cryptology, vol. 8, pp. 123–155, Springer-Verlag, 1995.

[O92] T. Okamoto: *Provable Secure and Practical Identification Schemes and Corresponding Signature Schemes*, Advances in Cryptology — Proceedings Crypto '92, Lecture Notes in Computer Science, vol. 740, pp. 31–53, Springer Verlag, 1993.

[P91] T.P. Pedersen: *Non-Interactive and Information-Theoretical Secure Verifiable Secret Sharing*, Crypto '91, Lecture Notes in Computer Science, Vol. 576, Springer-Verlag, pp. 129–140, 1991.

[RSA78] R. Rivest, A. Shamir and L. Adleman: *A Method for Obtaining Digital Signatures and Public-Key Cryptosystems*, Communication of the ACM, vol. 21(2), pp. 120–126, 1978.

## A Sketch of Proof of Theorem 1

We address a more formal proof that our protocol constitutes a non-malleable commitment scheme. Our aim is to extract the adversary's message within a negligibly close bound to the adversary's success probability $\pi_{\mathsf{open}}(\mathcal{A}, R)$ to derive some information about the unknown message for a given commitment. We omit a proof that this implies non-malleability of the commitment scheme because this proof already appears in [DDN00]. Instead, we show how to extract the adversary's message in our scheme. To this end, we repeat some basic facts about proofs of knowledge and knowledge extractors [FFS88,BG92]; we discuss them for the example of Okamoto's discrete-log-based proof of knowledge [O92] for a given $M = g_0^m h_0^r$. The knowledge extractor interacting with the prover works in two phases. Namely, it first generates a random conversation $S, c, y, z$ by running the prover to obtain $S$, by selecting $c$ and by letting the prover answer with $y, z$ to $S, c$. If this communication is invalid, then the extractor aborts. Else the extractor also stops with probability $1/q$. Otherwise it extracts at all costs. That is, the extractor fixes this communication up to the challenge, and then loops (till success) to seek another accepting conversation with the same communication prefix $S$ and different $c$. This is done by rewinding the execution to the choice of the challenge and re-selecting other random challenges. The extractor runs in expected polynomial time and outputs a representation of $M$ with respect to $g_0, h_0$ with probability $\pi - 1/q$. Here, $\pi$ denotes the probability that the prover makes the verifier accept, and $1/q$ is called the error of the protocol.

Next, we transfer the proof-of-knowledge technique to our setting. As in Section 3.2 we, too, adopt the convention that the adversary $\mathcal{A}$ does not "mix" the order of messages but rather catches up. We show how to remove this restriction in the final version of the paper.

Assume that we communicate with some party $\mathcal{C}$ which is going to commit to an unknown message $m \in_{\mathrm{R}} \mathbb{M}$. We choose random primes $p, q$ and two generators

$g_0, h_0$ and send them to $\mathcal{C}$. Party $\mathcal{C}$ selects $r, s, t \in_{\mathrm{R}} \mathbb{Z}_q$ and sends $M := g_0^m h_0^r$, $S := g_0^s h_0^t$. We answer with a random challenge $c \in_{\mathrm{R}} \mathbb{Z}_q$ and $\mathcal{C}$ returns $y := s + cm, z := t + cr \bmod q$. Finally, we check the correctness. Put differently, we perform all the steps of the sender in our protocol except for the coin flipping.

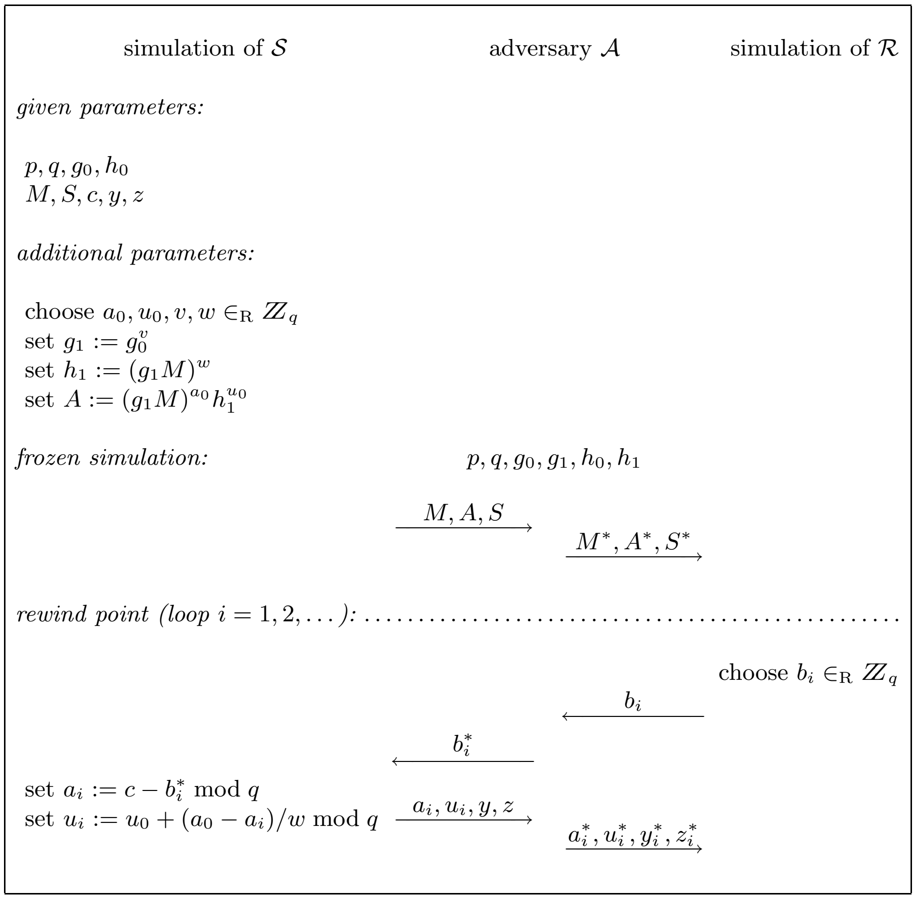
We want to use the PIM adversary to learn some information about $\mathcal{C}$'s message $m$. To this end, we incorporate the values of $\mathcal{C}$'s commitment in a knowledge extraction procedure for $M$. The extractor chooses additional generators $g_1, h_1$ by setting $g_1 := g_0^v$ and $h_1 := (g_1 M)^w$ for random $v, w \in_{\mathrm{R}} \mathbb{Z}_q^*$, and computes $A := (g_1 M)^{a_0} h_1^{u_0}$ according to the protocol description for random $a_0, u_0 \in_{\mathrm{R}} \mathbb{Z}_q$. Then it starts to emulate the PIM attack by pretending to be $\mathcal{S}$ and $\mathcal{R}$ and with values $p, q, g_0, g_1, h_0, h_1, M, S, A$. Because of the assumption about the order of messages, the adversary commits then to $M^*, S^*, A^*$. Next, we use the same stop-or-extract technique as in [O92]. In our case, the rewind point (if we do rewind) is the step where the receiver sends $b$. In each repetition, we send a random value $b_i \in_{\mathrm{R}} \mathbb{Z}_q$ —the subscript denotes the number $i = 1, 2, \ldots$ of the loop— on behalf of the receiver and the adversary hands some value $b_i^*$ to the simulated sender. Knowing the trapdoor $w = \log_{(g_1 M)} h_1$ we open $A$ with $a_i, u_i = u_0 + (a_0 - a_i)/w \bmod q$ such that $a_i + b_i^*$ equals the given value $c$, and send the valid answer $y, z$ to the challenge $c$ in the proof of knowledge for $M$. The adversary replies with $a_i^*, u_i^*, y_i^*, z_i^*$ to the receiver. A description is shown in Figure 5.

An important modification of the knowledge extractor in comparison to the one in [FFS88,O92] is that, once having entered the loop phase, not only does our extractor stop in case of success; it also aborts with no output if in some repetitions $i, j$ the adversary both times successfully finishes the commitment phase —which includes a correct decommitment of $A^*$— but opens $A^*$ with distinct values $a_i^* \neq a_j^*$. We say that $\mathcal{A}$ *wins* if this happens. In this case, the extractor fails to extract a message.

Our first observation is that our knowledge extractor stops (either with success or aborting prematurely) in expected polynomial-time. This follows as in [FFS88,O92]. Let us analyze the success probability of our extractor. We assert that the extractor succeeds in outputting a message with probability at least $\pi_{\mathsf{open}}(\mathcal{A}, R) - 1/q - \delta(n)$, where $\delta(n)$ denotes the probability that $\mathcal{A}$ wins (for security parameter $n$). The reason for this is that, given $\mathcal{A}$ does not win, the adversary's openings $a_{i_1}^* = a_{i_2}^* = \ldots$ in the valid commitment conversations are all equal. But then the values $b_{i_j} + a_{i_j}^* \bmod q$ for $j = 1, 2, \ldots$ of challenges in the proof of knowledge between $\mathcal{A}$ and $\mathcal{R}$ are uniformly and independently distributed. Analogously to [FFS88,O92] it follows that the extractor finds a message with probability $\pi_{\mathsf{open}}(\mathcal{A}, R) - 1/q$ in this case.

It remains to bound the probability $\delta(n)$ that $\mathcal{A}$ wins. We will prove that $\delta(n)$ is negligible under the discrete-log assumption. For this, we first remark that we are only interested in the case that $\mathcal{A}$ sends distinct openings of $A^*$ in *accepting* executions, because the extractor only relies on these executions. In order to derive a contradiction to the intractability of the discrete-log problem we observe that the notion of non-malleability with respect to opening requires

**Fig. 5.** Knowledge Extraction


simulation of $\mathcal{S}$
adversary $\mathcal{A}$
simulation of $\mathcal{R}$
*given parameters:*
$p, q, g_0, h_0$
$M, S, c, y, z$
*additional parameters:*
choose $a_0, u_0, v, w \in_{\mathrm{R}} \mathbb{Z}_q$
set $g_1 := g_0^v$
set $h_1 := (g_1 M)^w$
set $A := (g_1 M)^{a_0} h_1^{u_0}$
*frozen simulation:*
$p, q, g_0, g_1, h_0, h_1$
$\xrightarrow{M, A, S}$
$\xrightarrow{M^*, A^*, S^*}$
*rewind point (loop $i = 1, 2, \dots$):*
choose $b_i \in_{\mathrm{R}} \mathbb{Z}_q$
$\xleftarrow{b_i}$
$\xleftarrow{b_i^*}$
set $a_i := c - b_i^* \bmod q$
set $u_i := u_0 + (a_0 - a_i)/w \bmod q$
$\xrightarrow{a_i, u_i, y, z}$
$\xrightarrow{a_i^*, u_i^*, y_i^*, z_i^*}$

that $\mathcal{A}$ also reveals a valid decommitment. Hence, we view the decommitment phase as an additional step of the proof of knowledge. In other words, a correct decommitment is part of an accepting conversation of the proof of knowledge. Yet, this step has an extra property: the adversary must finish his commitment before he is allowed to ask $\mathcal{S}$ to open the original commitment. This corresponds to the fact that the decommitment phase of both parties $\mathcal{S}$ and $\mathcal{A}$ is delayed until both commitment phases are complete.

**Lemma 1.** *The probability that $\mathcal{A}$ wins is negligible.*

*Proof.* We show that if the claim of Lemma 1 does not hold this contradicts the intractability of the discrete-log problem. We are given randomly generated primes $p, q$, a generator $g$, and a value $X \in G_q$ for which we are supposed to compute $\log_g X$. We show how to use $\mathcal{A}$ to do so.

The first observation is that if $\mathcal{A}$ wins with noticeable probability within the (expected) polynomial number of loops, then by standard techniques it follows that $\mathcal{A}$ does so with noticeable probability in the first $p(n)$ repetitions for an appropriate polynomial $p(n)$. Thus, we simply truncate any loop beyond this, and the running time of our derived discrete-log algorithm is strictly polynomial. Instead of using the commitment $M$ of the third party $\mathcal{C}$, this time we run the knowledge extraction procedure incorporating the given values $p, q, g, X$, but generating the same distribution as the extractor. That is, select a message $m \in_{\mathrm{R}} \mathbb{M}$, as well $v, w \in_{\mathrm{R}} \mathbb{Z}_q^*$, set

$$g_0 := g^{-1/m}X, \qquad g_1 := g, \qquad h_0 := X^v, \qquad h_1 := X^w,$$

and compute $M, A, S, c, y, z$ according to the protocol description. Wlog. assume that $X \neq 1$ and $X^m \neq g$, else we already know the discrete log of $X$. Then $g_0$, $g_1$, $h_0$ and $h_1$ are random generators of the subgroup $G_q$. Furthermore, $g_1 M = g g_0^m h_0^r = X^{m+rv}$ and thus $\log_{(g_1 M)} h_1 = (m + rv)/w \bmod q$. Next we emulate $\mathcal{A}$ on values $p, q, g_0, g_1, h_0, h_1$ and $M, A, S$ by running the extraction procedure above. Note that this, too, means that we may abort before even starting to loop. Once we have entered the rewind phase, whenever the extractor is supposed to open $A$ to determine the challenge $c$ in the loop, we also open the commitment such that the coin flipping protocol always yields the same value $c$. This is possible as we know $\log_{(g_1 M)} h_1$ and are therefore able to open $A$ ambiguously. Observe that the communication here is identically distributed to the one in the extraction procedure. Hence, given that $\mathcal{A}$ wins with noticeable probability in the extraction procedure, $\mathcal{A}$ finds some $a_i^* \neq a_j^*$ for two accepting executions $i, j$ with the same probability in this experiment here. Let $u_i^*, u_j^*$ denote the corresponding portions of the decommitment for $A^*$ in these loops. Recall that we take the decommitment stage as an additional step of the proof of knowledge. Therefore, after having revealed $m, r$ in place of the sender in loop no. $j$, we also obtain some $m^*, r^*$ satisfying the verification equation $M^* = g_0^{m^*} h_0^{r^*}$ from the adversary. Particularly, we have:

$$h_1^{(u_i^* - u_j^*)/(a_j^* - a_i^*)} = g_1 M^* = g_1 g_0^{m^*} h_0^{r^*} = g^{1 - m^*/m} X^{m^* + r^* v}$$

Since $h_1 = X^w$ we can transform this into

$$g^{1 - m^*/m} = X^x \qquad \text{for } x = w(u_i^* - u_j^*)/(a_j^* - a_i^*) - (m^* + r^* v) \bmod q$$

Observe that $x$ is computable from the data that we have gathered so far. From $m^* \neq m$ we conclude that $1 - m^*/m \neq 0 \bmod q$ and therefore $x \neq 0 \bmod q$ has an inverse modulo $q$. Thus the discrete logarithm of $X$ to base $g$ equals $(1 - m^*/m)/x \bmod q$. $\square$

Summerizing, with probability $\pi_{\mathsf{open}}(\mathcal{A}, R) - 1/q - \delta(n)$ (which is negligibly close to the adversary's success probability) we extract some message $m'$. The final step in the proof is to show that indeed $m'$ equals the adversary's decommitment $m^*$ except with negligible probability; this follows by standard techniques and is omitted.