# Deniable Ring Authentication*

Moni Naor

Weizmann Institute of Science
Rehovot 76100, Israel
naor@wisdom.weizmann.ac.il

**Abstract.** Digital Signatures enable authenticating messages in a way that disallows repudiation. While non-repudiation is essential in some applications, it might be undesirable in others. Two related notions of authentication are: Deniable Authentication (see Dwork, Naor and Sahai [?]) and Ring Signatures (see Rivest, Shamir and Tauman [?]). In this paper we show how to combine these notions and achieve *Deniable Ring Authentication*: it is possible to convince a verifier that a member of an *ad hoc* subset of participants (a ring) is authenticating a message $m$ without revealing which one (source hiding), and the verifier $V$ cannot convince a third party that message $m$ was indeed authenticated – there is no 'paper trail' of the conversation, other than what could be produced by $V$ alone, as in zero-knowledge.
We provide an efficient protocol for deniable ring authentication based on any strong encryption scheme. That is once an entity has published a public-key of such an encryption system, it can be drafted to any such ring. There is no need for any other cryptographic primitive. The scheme can be extended to yield threshold authentication (e.g. at least $k$ members of the ring are approving the message) as well.

## 1 Introduction

An authentication protocol allows a receiver of a message, Bob, to verify that the message received is indeed the one sent by the sender, Alice. It is one of the basic issues which cryptography deals with. One of the key insights in the seminal paper of Diffie and Hellman [?] was the idea that it is possible to make authentication transferable, i.e. that Bob can convince a third party that Alice had indeed sent him the message. This involves Alice having a public-key as well as a secret-key that allows her to produce a digital signature of the message, verifiable by anyone knowing her public-key, but one that cannot be generated by anyone not holding her secret-key. This non-repudiation property is essential to contract signing, e-commerce and a host of other applications. In the last 25 years a lot of effort has been devoted to digital signatures in the research community, as well as the legal and business one.

However, one question to consider is whether non-repudiation of messages is *always* desirable. One obvious reason is privacy - one need not be a card

* Research supported in part by the RAND/APX grant from the EU Program IST

carrying EFF[1] member to appreciate that not everything we ever say should be transferable to anyone else - but this is precisely the case as more and more of our interactions move on-line. Another motivation arises where Bob is *paying* for the authentication (e.g. for checking a piece of software); should he be free to turn and give it away to Charlie? To address these concerns several notions of deniable authentication were developed (see more in Section **??** below.) In general an authentication provides (plausible) deniability if the recipient could have generated the authentication all by itself.

A different form of protection to the sender of the messages is hiding its identity or source. This is needed for leaking information - something that can be viewed as an important part of the checks and balances that monitor an open society. Keeping the sender's identity secret while being sure that it is a valid confirmation of the message may sound paradoxical, since the receiver verifies the authenticity of the message with respect to some public information related to the party doing the authentication (e.g. a public key). However a method for doing just that was recently suggested by Rivest, Shamir and Tauman [**?**]. They proposed the notion of *Ring Signatures* (a generalization of group signatures of Chaum and van Heyst [**?**]) that allows a member of an *ad hoc* collection of users $S$ (e.g. Crypto'2002 Program Committee members), to prove that a message is authenticated by a member of $S$. The assumption is that each member of $S$ has published a public signature key of a scheme with certain properties (where RSA and Rabin are examples). The construction given in [**?**] is very efficient, but its analysis is based on the ideal cipher model (a strengthening of the random oracle one.)

In this work we propose a notion that merges Ring Signatures and Deniable Authentication to form *Deniable Ring Authentication*. Roughly speaking, for a scheme to be Deniable Ring Authentication it should: (i) Enable the sender for any message he wishes and for any ad hoc collection $S$ of users containing the sender to prove (interactively) that a member of $S$ is the one confirming the message. (ii) Be a good authentication scheme, i.e. not allowing forgeries, where the notions of forgeability of Goldwasser, Micali and Rivest [**?**] are relevant. Ideally an adversary should not be able to make a receiver accept *any* message not sent by a member of $S$. (iii) The authentication is deniable in the zero-knowledge sense, i.e. the recipient could have simulated the conversation alone and the result would have been indistinguishable. (iv) The authentication should be *source hiding* or preserve the "anonymity in a crowd" of the sender: for any arbitrary subset $S$ of users, any two members of $S$ generate indistinguishable conversations to the recipient. (v) The scheme should not assume that the verifier of the authentication is part of the system and has established a public key. This is needed for two reasons: The PKI may be of a special nature (e.g. high-ranking government officials) and thus there is no reason for the recipient to be part of it. The second reason is that it is difficult to assure the independence of keys in the PKI and there is no reason to assume that the receiver has chosen its key properly (see Footnote **??** for an example.)

[1] Electronic Frontier Foundation.

We provide a construction of a Deniable Ring Authentication protocol based sole on the assumption that users have public-keys of some[2] good encryption scheme. The scheme is quite efficient: it requires $|S|$ encryptions by the sender and receiver and a single decryption by the sender. One can view the scheme as evolving from the deniable authentication scheme of Dwork, Naor and Sahai [?] (described in Section **??**.) The analysis of the scheme is based on the security of the encryption scheme, without resorting to additional random oracles (or any additional cryptographic primitive.) Note that users have "no choice" about being recruited to the subset $S$. Once a user has established an encryption key he might be drafted to such a crowd $S$.

### 1.1 Related Work

Issues related to deniability and anonymity have been investigated quite extensively from the early days of open scientific investigations of Cryptography[3]. Hence there are quite a few variants of deniability and anonymity protection and we will try to briefly describe them and their relationship to our work.
*Group and Ring Signatures:* a group signature scheme allows members of a *fixed* group to sign messages on the group's behalf while preserving their anonymity. This anonymity is conditional and a group manager can revoke it. Note that groups here are not ad hoc and the group manager sets up a special type of key assignment protocol. There are quite a few papers on the subject [?,?,?], yielding reasonably efficient protocols. A related notion is that of *identity Escrow* allowing proofs of membership in a subset, with the group manager being able to identify and revoke membership [?,?]. Some of the protocols do support subsets authentication [?] as well as general key choices by the participants, but these all assume special set-up and managers. Ring Signatures, as introduced in [?], support *ad hoc* subset formation and by definition do not require special setup. They rely on a Public-Key Infrastructure (for signatures of certain type in the construction of [?].) Note that some of the protocols for group signatures can actually be used as ring signatures, e.g. [?].
*Designated Verifier Proofs* were proposed by [?] to enable signatures that convince only the intended recipient, who is assumed to have a public-key. See Footnote **??** for the problems this approach might encounter in our setting.
*Deniable Authentication:* the work of Dwork, Naor and Sahai [?] on deniable authentication provides a system that addresses the deniability aspects, i.e. that following the protocol there is no paper trail for the authentication of the message. This is the same property we are trying to achieve, and the protocols presented there are our starting point (see Section **??**.)
*Undeniable signatures* are digital signatures in which the recipient cannot transfer the signature without the help of the signer. If forced to either acknowledge or deny a signature, however, the signer cannot deny it if it is authentic (thus the term "invisible" is probably better). They were introduced in 1989 by Chaum

[2] Actually each user can use their favorite encryption scheme.
[3] An early proponent was David Chaum, e.g. [?].

and Van Antwerpen [?] and further developed in [?]. A specific and appealing version of them are the *Chameleon Signatures* of Krawczyk and Rabin [?]. The difference in the deniability requirement between this line of work and that of deniable authentication [?] as well as the current paper, is that in our case the authentication is *not* intended for ultimate adjudication by a third party, but rather to assure $V$ – and only $V$ – of the validity of the message.

*Contributions of this work:* We present a simple and efficient scheme that allows leaking an authenticated secret, without the danger of being traced (Protocol **??** below.) The scheme does not assume any special infrastructure, beyond the one given by standard PKI for encryption. The analysis of the scheme is straightforward and does not resort to random oracles.

We also extend the scheme to be able to authenticate more complex statements than "a member of $S$ is confirming the message", to statements such as "at least $k$ members of $S$ confirm the message" and other access structures (Protocol **??** in Section **??**.)

We also deal in Section **??** with the case where the adversary $\mathcal{A}$ may have all the secret keys of the authenticator, which is the appropriate model for Identity Based Encryption [?,?] and the Subset Cover Framework of [?]. Protocol **??** handles this case at the cost of two additional rounds.

There are a number of differences between the properties of our setting and scheme and those of Rivest, Shamir and Tauman [?]: on the negative side (from this paper's point of view), our scheme requires interaction, since the verifier is not assumed to have established a public-key. This requires some mechanism of anonymous routing (e.g. MIX-nets.) Also our scheme involves sending longer messages (proportional to the size of $S$). On the neutral side, the time complexity of our scheme and that of [?] are roughly comparable (to within multiplicative constants), if one uses an encryption scheme where the encryption process is very efficient, such as RSA with low exponent. On the positive side: (i) Our analysis does not rely on any additional assumptions except the underlying encryption scheme is good (immune to chosen ciphertext attacks.) (ii) Since we only need that the encryption scheme is good, there is no way for an organization that wishes that its members have public-keys to try and fight our system by establishing ones with some weird formats (that deter the [?] scheme, e.g. tree based ones.) (iii) Our deniability guarantees are stronger than in [?]: their deniability is achieved by assuming that the verifier is a member of the system and has established a public key. He is then added to the Ring (and hence could have generated the conversation himself). However this assumes not only that the verifier has a public key, but that this key was properly chosen[4]. (iv) it is not clear how to extend the [?] protocol to handle threshold and other access

[4] To see why this issue may be problematic, consider a large corporation $A$ dealing with a small user $B$. The user $B$ chooses its public-key $K_B$ to be the *same* as $K_A$, the public key of corporation $A$. Now suppose that $A$ sends to $B$ a message signed in a ring scheme were the ring consists of $A$ and $B$. Given that the public keys used are $\{K_A, K_A\}$ this is hardly deniable for $A$.

structures over the ring[5], whereas we do that with no computational penalty in Protocol ??.

## 2 Definition of Deniable Ring Authentication

We now summarize the setup and requirements of a deniable ring authentication scheme.

*Setup:* We assume that participants have published public-keys. The public keys are generated via some key generation process that gives corresponding secret keys. We do not make any particular assumption about this process, except that good participants choose their keys properly, i.e. following the key generation protocol. However bad participants, that are under the control of the adversary, may have chosen them arbitrarily and in particular as a function of the good public keys. A ring $S$ is a any subset of participants. A (good) authenticator $P$ is a member of $S$. The verifier of a message is an arbitrary party and has not necessarily published a public-key. The only assumption is that both the verifier and the authenticator know the public-keys of *all* members of $S$. The authenticator $P$ engages with the verifier in an interactive protocol to authenticate a message $m$. At the end of the interaction the verifier accepts or rejects the authentication.

Given that the protocol is interactive (it must be so, since the verifier has not established any credentials) we must assume that it is possible to route messages anonymously, i.e. that the verifier and prover can exchange message without the adversary being able to trace who is the recipient. How this is achieved is beyond the scope of this paper.

We assume that the adversary $\mathcal{A}$ controls some of the participants of the system. For those participants it chooses (and knows) all the secret bits (we do not deal here with dynamic corruption of good users, though the methods presented seem to be resilient to such attacks as well). The authentication protocol should satisfy:

**Completeness:** For any subset of participants $S$ and for any good authenticator $P \in S$, for any message $m$, if the prover and verifier follow the protocol for authenticating the message $m$ (with $P$ using his secret key), then the verifier accepts; this can be relaxed to "accepts with high probability."

**Soundness - Existential Unforgeability:** Consider an adversary $\mathcal{A}$ trying to forge a message. It may know and choose the secret keys of all bad participants, but the good members choose their public-keys properly. The adversary runs an attack on the protocol as follows: it *adaptively* chooses a sequence of arbitrary messages $m_1, m_2, \ldots$, arbitrary rings $S_1, S_2, \ldots$ and good participants $P_1, P_2, \ldots$ where $P_i \in S_i$, and asks that $P_i$ will authenticate message $m_i$ as part of ring $S_i$ (using the deniable ring authentication protocol) where the verifier is controlled by $\mathcal{A}$. We say that $\mathcal{A}$ successfully attacks the scheme if it can find a ring $S$ of good participants so that a forger

[5] But see the recent work [?].

$C$, under control of $\mathcal{A}$ and pretending to be a member of $S$, succeeds in authenticating to a third party $D$ (running properly the verifier's $V$ protocol) a message $m \notin \{m_i\}_{i=1,2,\ldots}$. The soundness requirement is for all probabilistic polynomial time adversaries $\mathcal{A}$ the probability of success is negligible.

**Source Hiding:** For any two *good* participants $A_1$ and $A_2$, for any subset $S$ containing $A_1$ and $A_2$, it is computationally infeasible for any $V^*$ acting as the verifier to distinguish between protocols where $A_1$ is doing the authentication and $A_2$ is the one running it (that is the probability it guesses correctly which case it is should be negligibly close to 1/2.) Note that not all the members of $S$ are necessarily good, but we only protect the anonymity of the good ones.

**Zero-Knowledge - Deniability:** Consider an adversary $\mathcal{A}$ as above and suppose that a member of $S$ is willing to authenticate any polynomial number of messages. Then for each $\mathcal{A}$ there exists a polynomial-time simulator $\mathcal{Z}$ that outputs an indistinguishable transcript (to everyone but the sender). A possible relaxation is to allow the simulator to depend on $\varepsilon$, the distinguishing advantage (this is known as $\varepsilon$ -knowledge.)

Note that Source Hiding and Deniability seem to be related but they are incomparable. In particular, the requirement for Source Hiding should hold even for an online verifier, whereas the requirement for Deniability is only after the fact.

*Concurrency:* One issue that we have not specified is whether the many various protocols that the adversary may be running are executed concurrently, where timing is under the control of the adversary, or sequentially. This is largely orthogonal to those definitions and we will specify for our main scheme (Protocol **??**) for each property whether it withstands concurrent attacks or not.

*Big brother:* A stronger model for deniability and source hiding is when the adversary $\mathcal{A}$ knows the secret keys of the good players as well as those of the bad ones. This case and its motivating examples is discussed in Section **??**.

## 3 Tools

### 3.1 Encryption Schemes

Our main tool is encryption schemes. We assume some *good* public-key encryption scheme $E$. To specify what we mean by good, we have to provide the type of attack that the encryption scheme is assumed to withstand , e.g. known[6] or chosen plaintext, or chosen ciphertext. And we have to specify what breaking the encryption scheme means, where the two leading notions are semantic security and non-malleability. The latter is the relevant notion we will require from $E$. Roughly speaking, a public key cryptosystem is non-malleable if, seeing an encryption $E(\alpha)$ "does not help" an attacker to generate an encryption $E(\beta)$ such

[6] Not really relevant in public-key encryption.

that $\alpha$ and $\beta$ are related (with certain trivial exceptions). This is formalized and treated at length in [**?**]. See also [**?**] and [**?**].

As for the type of attack, this varies depending on the precise properties we want from our deniable ring authentication process (in particular whether we want to withstand concurrent attacks). We can already gain some properties simply assuming that $E$ is immune to chosen plaintext attack. However for the full strength we require that the scheme be immune against chosen ciphertexts in the post-processing mode, also known as CCA2. This means that the attacker has access to a decryption device and can feed it with ciphertexts of its choice. At some point it gets a challenge ciphertext (so that it should perform some operation on the corresponding plaintext) and still has access to the decryption device, except that now it cannot feed it with the challenge ciphertext. Under such an attack semantic security and non-malleability coincide. (See [**?**,**?**] for background one the subject.)

For a public key $K$ the encryption scheme $E_K$ maps a plaintext into a ciphertext. This mapping must be probabilistic, otherwise the scheme cannot even be semantically secure. Therefore $E_K$ induces for each message $m$ a distribution of ciphertexts. To encrypt $m$ one has to choose a random string $\rho$ and then $C = E_K(m, \rho)$ is a ciphertext of $m$. Given $C$ and the corresponding private decryption key $K^{-1}$ the decryption process retrieves $m$, but we do not assume that it retrieves $\rho$ as well (in some schemes the process does retrieve while in others it does not). When we write "generate $C = E_{K_i}(m)$" we mean choose random $\rho$ and let $C = E_{K_i}(m, \rho)$.

A procedure we use quite extensively in our protocols is for the creator of a ciphertext $C$ to prove that $C$ is an encryption of a message $m$. In order to perform this it is sufficient to produce $\rho$, the random bits used to generate $C$ and then anyone can verify that $C = E_{K_i}(m, \rho)$). The property we require from $E$ is that it be binding or unique opening. If $K$ was generated properly, then for any ciphertext $C$ there should be a unique message $m$ for which there exists a $\rho$ such that $C = E_K(m, \rho)$ (there could be more than one $\rho$ but no more than one plaintext corresponding to $C$)[7]. We do not assume any binding in case the key are badly formed (except for Section **??** where this issue arises.) Thus when we write "open ciphertext $C$" we mean give the plaintext and the random bits $\rho$ used to generate $C$.

*Implementations of the Encryptions Schemes:* There are a number of possibilities for encryption scheme meeting the standards outlines above. If one wants to avoid employing random oracles, (which is one of the goals of this paper) then the famed Cramer-Shoup [**?**] one is the most efficient. It is based on the Decisional Diffie-Hellman problem. One drawback of it is that encryption is as expensive as decryption, i.e. requires (a few) modular exponentiations. Otherwise the system

[7] This property more or less follows from the non-malleability requirement (without it one has to specify what is the meaning of such a ciphertext) but we added it explicitly to prevent confusion. Note that the complement of this property was used in [**?**] to obtain deniable *encryption*.

known as OAEP [?] over low exponent RSA or Rabin offers the most efficient implementation (See [?,?,?] for the state-of-the-art on the subject.) Using such an encryption in Protocol **??** yields a scheme of complexity comparable (up to multiplicative constants to that of [?]. Another possibility for avoiding random oracles while maintaining efficiency is is to use interactive encryption, as proposed by Katz [?]. We do not explore it further in this paper, but see [?] for its application for deniable encryption.

### 3.2 Commitment Schemes

A commitment scheme allows the sender to deposit a hidden value with the receiver, so that the latter has no idea what it is, but at a later point the sender can reveal the hidden value and the receiver can be sure that this is the original one. There are a number of variants on the precise security offered to the two sides. We will be interested in commitment schemes where the sender is offered computational secrecy and the receiver is assured that there is a unique value. More precisely, following the commitment phase the receiver cannot decide (with non-negligible advantage) whether the hidden value is $r_1$ or $r_2$. There are quite simply and efficient protocols with these properties (e.g. [?]).

For most of the protocols of the paper we will actually use encryption for the purpose of commitment (this means that it is not secret to the owner of the secret key.) The reason is that we need to obtain *non-malleability* with respect to another encryption, and this is achieved in the easiest way using an encryption scheme which is non-malleable. However, for the big brother setting, where $\mathcal{A}$ is assumed to know all the secret keys in the system, this is not good enough and we will need a more involved solution in Protocol **??** in Section **??**.

### 3.3 Zero-Knowledge

We do not apply zero-knowledge protocols as tools, but the deniability requirement means that our protocol should be zero-knowledge[8]. We use the standard tricks of the trade to come up with a simulators $\mathcal{Z}$.

The subject of preserving zero-knowledge for concurrently executed protocols has received much attention recently and in general it is a quite difficult problem. One way to bypass it was proposed in [?] by adding relatively benign timing assumptions. It is possible to use the same techniques to achieve deniability in the presence of concurrent attacks for Protocol **??**.

## 4 Some Background Protocols

In this section we describe two protocols that can be viewed as the precursors of our main protocol. We recommend reading them before Protocol **??**. We

[8] This is a relatively rare case where zero-knowledge is needed as an end result and not as a tool in a subprotocol.

use the term "prover" for the party doing the authentication or proving the statement "message $m$ is authentic" and "verifier" or $V$ to the receiver or the party doing the verification of the claim. The first protocol simply provides an interactive authenticated protocol. It is based on adding a random secret value to $m$ encrypted under $P$'s public key as a challenge. Note that in all our protocols we assume that the sender and receiver already know what is the candidate message (otherwise an additional preliminary round is needed.)

**Protocol 1 Interactive Authentication**
*The prover has a public key $K$ of an encryption scheme $E$. The prover wishes to authenticate the message $m$. The parameter $\ell$ sufficiently large that $2^{-\ell}$ is negligible. The concatenation of $x$ and $y$ is denoted $x \circ y$.*

*1. $V \to P$ : Choose random $r \in_R \{0,1\}^\ell$.*
*Generate and send the encryption $C = E_K(m \circ r)$ to the prover.*
*2. $P \to V$ : Decrypt $C$ to obtain $r$.*
*Verify that the prefix of the plaintext equals $m$. Send $r$.*
*The verifier $V$* accepts *if the value $P$ sends in Step 2 equals $r$.*

This protocol was proposed in [**?**] (see Section 3.5 there) and proved to be existentially unforgeable assuming that the encryption scheme is secure against chosen ciphertext attacks (post-processing, or CCA2). Note that if $E$ is malleable in certain ways then the scheme is not secure, since it is possible to switch the prefix of the message.

Is this scheme deniable? It is deniable against an honest verifier that chooses $r$ at random. However one cannot hope to argue that it remains so against a malicious verifier, since zero-knowledge is impossible to obtain in two rounds (see [**?**], at least with auxiliary input).

Consider now the following extension that was proposed by Dwork, Naor and Sahai, [**?**], where the idea is that the verifier should prove knowledge of $r$ before the prover reveals it. For this we use the "opening" of ciphertexts as defined in Section **??**, by giving away the plaintext and the random bits used to generate it.

**Protocol 2 Deniable Authentication**
*The prover $P$ has a public key $K$ of an encryption scheme $E$. The message to be authenticated in $m$, known to both parties.)*

*1. $V \to P$ : Choose $r \in_R \{0,1\}^\ell$. Generate and send $C = E_K(m \circ r)$*
*2. $P \to V$ : Decrypt $C$ to obtain $r$ (the suffix of the message).*
*Generate and send $D = E_K(r)$.*
*3. $V \to P$ : Open $C$ by sending $r$ and $\rho$,*
*the random bits used in the encryption in Step 1.*
*4. $P \to V$ : Verify that the prefix of the opened $C$ equals $m$.*
*Open $D$ by sending $r$ and $\sigma$,*
*the random bits used in the encryption in Step 2.*
*$V$* accepts *if the value sent in Step 4 equals $r$ and $D$ was opened correctly.*

Note that the verification $m$ is only done at Step 4, that is if a bad $C$ was sent, then the prover does not reveal the fact that it detected it at Step 2. The deniability of the scheme is obtained by the possibility that the simulator extract the value of $r$ from any verifier $V^*$, at least in expected polynomial time, as is common in proofs of zero-knowledge. After $r$ has been extracted it is possible to finish the execution of the protocol. Soundness follows from the fact that the ciphertext $D = E_K(r)$ serves as a non-malleable commitment to $r$.

## 5 The Main Scheme

The idea for obtaining a ring authentication protocol from Protocol **??** is to run in parallel a copy of the protocol for each member of $S$, but using the same $r$, but otherwise with independent random bits. However there are a few delicate points. In particular if we want to assure source hiding, then it is unsafe for the prover to encrypt the decrypted $r$ using all the $K_i$'s before it verifies the consistency of the Step 1 encryption in all the protocols. Otherwise by using a different $r$ for each encryption key the adversary who may be controlling one member of $S$ may figure out the identity of $P$. To handle this we let $P$ split $r$ into $r_1, r_2, \ldots r_n$ and encrypt each one separately in Step 2.

*Setup:* Participants in the system have public keys of an encryption scheme $E$, as described in Section **??**. Each good member knows the corresponding secret key. Let the ring be denoted by $S$ and by slight abuse of notation we will also identify $S$ with the set of public keys of its member $\{K_1, K_2, \ldots, K_n\}$. Both $P$ (where we assume $P \in S$) and the verifier $V$ know all the public keys in $S$.

**Protocol 3 Deniable Ring Authentication**
*for $S = \{K_1, K_2, \ldots, K_n\}$ where $P$ knows the $j$th decryption key. The message to be authenticated is $m$.*

*1.* $V \to P$ : *Choose random $r \in \{0,1\}^\ell$. Generate and Send*
$\langle C_1 = E_{K_1}(m \circ r), C_2 = E_{K_2}(m \circ r), \ldots C_n = E_{K_n}(m \circ r)\rangle$.
*2.* $P \to V$ : *Decrypt $C_j$ to obtain $r$.*
*Choose random $r_1, r_2, \ldots r_n$ so that $r = r_1 + r_2 + \cdots + r_n$.*
*Generate and send $\langle D_1 = E_{K_1}(r_1), \ldots, D_n = E_{K_n}(r_n)\rangle$.*
*3.* $V \to P$ : *Open $C_1, C_2 \ldots C_n$ by sending $r$ and $\rho_1, \rho_2, \ldots \rho_n$,*
*the random bits used in the encryption process in Step 1.*
*4.* $P \to V$ : *Verify that $C_1, C_2 \ldots C_n$ were properly formed (same $m$ and $r$).*
*Send $r_1, r_2, \ldots r_n$ and $\sigma_1, \sigma_2, \ldots \sigma_n$,*
*the random bits used to generate $D_1, D_2, \ldots D_n$.*

$V$ accepts *if $r = r_1 + r_2 + \cdots + r_n$ and $D_1, D_2, \ldots D_n$ were properly formed.*

*Complexity:* Running the protocol involves on the verifier's side $n$ encryptions and $n$ verifications of encryptions. On the prover's side it involves one decryption, $n$ encryptions and $n$ verifications of encryption. If the underlying encryption scheme is based on low exponents (Rabin or low exponent RSA with OAEP), then this consists of $O(n)$ multiplication and $O(1)$ exponentiations. If the encryption is Diffie-Hellman based (for instance Cramer-Shoup [**?**]) then $O(n)$ exponentiations are involved. In term of communication, the major burden is sending (both ways) $n$ ciphertexts.

### 5.1 Functionality and Security of the Scheme

To prove that Protocol **??** is indeed a deniable Ring Authentication Protocol we have to argue that the four requirements, completeness, soundness, source-hiding and deniability are satisfied, as we now sketch. As for completeness it is easy to verify that if both sides follows the protocol than they accept. The only requirement we need from a bad public key $K_i$ is that it will be easy to verify even for bad keys that $C = E_{K_i}(m, \rho)$ which we can assume without loss of generality that holds.

*Soundness/Unforgeability:* Recall that we may assume that all keys in $S$ are properly formed for this property. The key point of to understanding why the protocol is that $\langle D_1 = E_{K_1}(r_1), D_2 = E_{K_2}(r_2), \ldots D_n = E_{K_n}(r_n)\rangle$ is a non-malleable commitment to $r = r_1 + r_2 + \cdots r_n$, where the non-malleability is with respect to $\langle C_1 = E_{K_1}(m \circ r), C_2 = E_{K_2}(m \circ r), \ldots C_n = E_{K_n}(m \circ r)\rangle$.

For this to hold it is sufficient that $E$ be non-malleable against chosen plaintext attacks (no need for protection against chosen ciphertext attacks, unless we are interested in concurrent attacks .) The fact that $P$ is committed to the value follows from the binding property of $E$ (See Section **??**.) Once we have established this then soundness follows, as it does for Protocol **??**. To handle a concurrent attacks we assume that $E$ is secure against chosen ciphertext secure attacks (post-processing, or CCA2). (We do not know whether this is essential, see Section **??**.)

*Source Hiding:* we claim that the of the key which was used in Step 2 (among well chosen keys in $S$) is *computationally* hidden during the protocol and *statistically* hidden after protocol, if things went well, i.e. the protocol terminated successfully. This follows from the fact that if at Step 1 all the $\langle C_1 = E_{K_1}(m \circ r), C_2 = E_{K_2}(m \circ r), \ldots C_n = E_{K_n}(m \circ r)\rangle$ are consistent (with the same $m$ and $r$), at least among the good keys, then the hiding of the source is perfect. Suppose that they are not consistent. Then at step 3 they will be caught (from the binding property of $E$, and hence Step 4 will not take place.

This property is maintained even when the adversary can schedule concurrent executions. The reason is that witness indistinguishable protocols can be composed concurrently.

*Deniability:* we can run a simulator 'as usual' and extract $r$: run the protocol with $P$ using first a random $r$. If at Step 3 the verifier opens then rewind to just after Step 1 and run again with the correct $r$. A few things worth noting: the complications to address, as is usual in proofs of a zero-knowledge property are a $V^*$ that refuses to open. One key point to notice is that the semantic security of $E$ means that it is enough that one key $K_i$ be good and unknown to $V^*$ for $\langle D_1 = E_{K_1}(r_1), D_2 = E_{K_2}(r_2), \ldots D_n = E_{K_n}(r_n)\rangle$ to be a semantically secure commitment scheme to $r$.

This is the only property that is problematic under concurrent executions. We can appeal to the timing model of [**?**] and get a variant of this protocol that would work there. However this is beyond the scope of this paper.

## 6 Extension: Threshold and other Access Structures

One can view Ring Authentication (both ours and the Rivest Shamir and Tauman one [**?**]) as a proof system that 1 out of the ring $S$ is confirming the message. In this section we discuss an extension of Rings into proving more general statements, e.g. that (a least) $k$ members out of the ring $S$ are confirming the message, without revealing any information about the subset $T$ of confirmers. In general, we can deal with *any* monotone access structure, provided that it has a good secret sharing scheme (see [**?**] for bibliography on the subject.)

In this setting we assume that there is a subset $T \subset S$ of members that collude and want to convince the verifier that $T$ satisfies some monotone access structure $\mathcal{M}$. As in the rest of this paper, all this can be ad hoc, i.e. there is no need to fix neither $S$ nor $M$ in advance (or $T$ of course). We do assume that there is one representative $P$ of $T$ that communicates with the verifier. Note that the members of $T$ need to trust $P$ to the extent that a bad $P$ can make the protocol loose its deniability and source hiding, but the unforgeability.

We adapt an idea suggested by Cramer, Damgård and Shoenmaker [**?**] and DeSantis et al. [**?**] for combining zero-knowledge statements via secret sharing. In our context we use this idea by letting the verifier split $r$ according to the secret sharing scheme for $\mathcal{M}$. Only if enough shares are known, then $r$ can be reconstructed, otherwise it remains completely unknown.

We do not have to assume any additional properties from the access structure for $\mathcal{M}$, i.e. the protection could be information theoretic or computational. We assume of course that secret generation and reconstruction are efficient. We also assume that given shares $s_1, s_2, \ldots s_n$ it is possible to verify that they were properly formed, i.e. that for each subset $T$ that satisfies $\mathcal{M}$ the reconstruction algorithm will output the same secret. This is very simple in most, if not all, schemes we are aware of, e.g. Shamir's polynomial based one [**?**].

**Protocol 4 Ring Authentication for Monotone Access Structure $\mathcal{M}$**
*Ring $S = \{K_1, K_2, \ldots, K_n\}$ where $P$ represents a subset $T \subseteq S$.*

*1. $V \rightarrow P$ : Choose random $r \in \{0,1\}^\ell$.*
*Generate shares $s_1, s_2, \ldots s_n$ of $r$ according to the scheme for $\mathcal{M}$.*
*Send $\langle C_1 = E_{K_1}(m \circ s_1), C_2 = E_{K_2}(m \circ s_2), \ldots C_n = E_{K_n}(m \circ s_n)\rangle$.*
*2. $P \rightarrow V$ : $P$ gets from each $j \in T$ the decryption of $C_j$.*
*$P$ reconstructs $r$ from the shares $s_j$ for $j \in T$.*
*Choose random $r_1, r_2, \ldots r_n$ such that $r = r_1 + r_2 + \cdots + r_n$.*

*Generate and send* $\langle D_1 = E_{K_1}(r_1), \ldots, D_n = E_{K_n}(r_n)\rangle$.
*3.* $V \rightarrow P$ : *Open* $C_1, C_2 \ldots C_n$ *by sending* $s_1, s_2, \ldots s_n$ *and* $\rho_1, \rho_2, \ldots \rho_n$, *the random bits used in the encryption in Step 1.*
*4.* $P \rightarrow V$ : *Verify that* $s_1, s_2, \ldots s_n$ *yield the same secret for all subsets.*
*Verify that* $C_1, C_2 \ldots C_n$ *were properly formed*
*( same* $m$ *and corresponding* $s_i$*.)*
*Send* $r_1, \ldots r_n$ *and* $\sigma_1, \ldots \sigma_n$ *(bits used to generate* $D_1, \ldots D_n$*.)*

$V$ accepts *if* $r = r_1 + r_2 + \cdots + r_n$ *and* $D_1, \ldots D_n$ *were properly formed.*

This extended protocol is not much more complex (computationally as well as to implement) than the original Protocol **??**. The additional computation consists simply of the secret sharing generation, reconstruction and verification.

The Completeness and Deniability of Protocol **??** follow from the same principals as Protocol **??**. As for unforgeability, we should argue that if the subset $S^* \subset S$ of participants the adversary controls does not satisfy $\mathcal{M}$, then it cannot make the adversary accept a message $m$ (not authenticated by the good participants) with non-negligible probability. One point is the binding property of $E$. In Protocol **??** the assumption was that all of $S$ consists of good players (otherwise unforgeability is not meaningful), but here some members of $S$ might be under the control of $\mathcal{A}$ (but not enough to satisfy $\mathcal{M}$) and might have chosen their public-keys improperly. For this we either have to assume that $E_K$ is binding even if the key $K$ was not properly chosen, or modify the protocol to and add for each $D_i$ a commitment to $r_i$ as well. The protection against reconstruction by a non qualified subset that the secret sharing scheme $\mathcal{M}$ offers then assures that $\mathcal{A}$ cannot retrieve $r$ following Step 1.

As for source hiding, we should argue that for two sets $T_1$ and $T_2$ deciding which one is doing the confirmation is difficult. This follows form the fact that at Step 4 $P$ checks the consistency of $s_1, s_2, \ldots s_n$ and hence revealing $r_1, r_2, \ldots r_n$ will not yield information about $T$.

## 7 Deniable Ring Authentication in the Presence of Big Brother

In this Section we deal with the case of where the adversary $\mathcal{A}$ actually knows the secret key of the authenticator. Why is this an interesting case, after all we usually think of the public-key setting as providing users with the freedom of choosing their own keys? There are several possible answers: first, there are settings where users do not choose their own private keys. These include *Identity Based Encryption* [**?**,**?**,**?**] where a center provides a key to each users as a function of their identity, and a broadcast encryption type of setting where users receive secret keys of various subsets to which they belong (see in particular [**?**] and more below). Another answer is that it is desirable to avoid a situation where the distinguisher has an incentive to extract the secret key for $j$ by, say, legal means.

Protocol **??** does not offer anonymity and deniability in case the adversary $\mathcal{A}$ knows the *secret* key of $P$ - it is possible for $\mathcal{A}$ to figure out whether it is $j$ who is authenticating the message by the following active attack: In Step 1 in protocol send $\langle C_1 = E_{K_1}(m \circ r), C_2 = E_{K_2}(m \circ r), \ldots C_j = E_{K_j}(m \circ r'), \ldots C_n = E_{K_n}(m \circ r)\rangle$, that is $C_j$ is the only one with suffix $r'$. When receiving $D_1, D_2, \ldots D_n$ the adversary can check whether the suffix of decryption of $D_j$ equals $r$ or $r'$.

There are two possible approaches for correcting this problem, one is for $P$ to make sure that all the $C_i$'s are proper before decrypting any of them. This requires some form of proof of consistency. The other is for $P$ not to commit to $r$ using the $E$'s but rather using a scheme that is secure against everyone. This requires coming up with non-malleable commitment with respect to the encryptions of Step 1. Both approaches seem viable, but we found a reasonably efficient implementation of only the second one. The major obstacle is to preserve soundness. The idea is simple and is an adaptation of old tricks (e.g. the commitment scheme in [**?**]): the prover splits $r$ into two parts and will reveal one of them to prove knowledge.

Let $W$ be a commitment scheme with perfect binding and computational protection to the sender, as in, e.g. [**?**]. We assume that the commitment phase is unidirectional (this even fits the scheme of [**?**], since it can be sent together with the message.) We assume that commitment to value $r$ involves choosing a random string $\sigma$ and sending $D = W(r, \sigma)$. The computational protection it offers the sender means that given $D$ which is a commitment to $r_1$ or $r_2$ it is hard to distinguish between the two cases. We will have as our security parameter $\ell$, we assume is sufficiently large so that $2^{-\ell}$ is negligible.

**Protocol 5 Ring Authentication in the presence of big brother**
*for Ring $S = \{K_1, K_2, \ldots, K_n\}$ where $P$ knows the $j$th decryption key.*
*The message to be authenticated is $m$.*

*1. $V \rightarrow P$ : Choose random $r \in \{0,1\}^\ell$. Generate and Send*
*$\langle C_1 = E_{K_1}(m \circ r), C_2 = E_{K_2}(m \circ r), \ldots C_n = E_{K_n}(m \circ r)\rangle$.*
*2. $P \rightarrow V$ : Decrypt $C_j$ to obtain $r$.*
*Choose $\ell$ pairs $(r_1^0, r_1^1), (r_2^0, r_2^1), \ldots (r_\ell^0, r_\ell^1)$ such that $r_i^0 + r_i^1 = r$*
*Generate and send $\ell$ pairs $\langle (D_1^0, D_1^1), (D_2^0, D_2^1), \ldots, (D_\ell^0, D_\ell^1)\rangle$*
*where for $1 \le i \le \ell$ and $b \in \{0,1\}$ generate $D_i^b = W(r_i^b, \sigma_i^b)$.*
*3. $V \rightarrow P$ : Choose and send $\ell$ random bits $b_1, b_2, \ldots b_\ell$*
*4. $P \rightarrow V$ : For $1 \le i \le \ell$ open $D_i^{b_i}$ by sending $r_i^{b_i}$ and $\sigma_i^{b_i}$ .*
*5. $V \rightarrow P$ : Verify that the opening are consistent: $\forall 1 \le i \le \ell$ $D_i^{b_i} = W(r_i^{b_i}, \sigma_i^{b_i})$*
*Open $C_1, \ldots, C_n$ by sending $r$ and $\rho_1, \rho_2, \ldots \rho_n$,*
*the random bits used in the encryption process in Step 1.*
*6. $P \rightarrow V$ : Verify that $C_1, \ldots, C_n$ were properly formed (same $m$ and $r$).*
*Open the remaining members of $\langle (D_1^0, D_1^1), (D_2^0, D_2^1), \ldots (D_\ell^0, D_\ell^1)\rangle$*
*by sending $r_i^{1-b_i}$ and $\sigma_i^{1-b_i}$.*
$V$ accepts *if the revealed values $(r_1^0, r_1^1), (r_2^0, r_2^1), \ldots (r_\ell^0, r_\ell^1)$ were properly formed and $\forall 1 \le i \le \ell$ we have $r_i^0 + r_i^1 = r$.*

In order to prove that soundness/unforgeability still holds, consider Protocol **??**. For this protocol we are assured of its soundness in case $E$ is secure against chosen ciphertext attacks in the postprocessing mode (CCA2) [**?**]. We will need the same assumption here. The key point is that it is possible to extract the value $r_i^0 + r_i^1$ (which should be $r$) by rewinding the forger to just before Step 3. If the forger has probability $\delta$ of succeeding, then with probability at least $\delta^2$ such a value can be extracted and hence a guess for $r$ can be mounted. This (plus the original proof of Protocol **??**) are sufficient for proving the soundness of Protocol **??**.

As for source hiding, it follows from the semantic security of $W$. The only problem is when $V^*$ is not following the protocol. But in this case in Step 6 the prover does not open the remaining commitments. Similarly, for deniability, the simulator should extract $r$ from $V^*$, which can be done by sending commitments to random values in Step 2.

*The Subset Cover Environment:* An interesting case of using Protocol **??** is in the Subset Cover Environment described in [**?**]. In this setting there is a collection of subsets of users $U_1, U_2, \ldots$. For each subset $U_i$ in the collection there is an associated public key $L_i$. Each user $u$ is given secret information enabling it to compute the corresponding secret key of $L_i$ for all subsets $U_i$ such that $u \in U_i$. There are many types of rings $S$ where it is easy express $S$ as the union of not too many subsets from the collection. One such example is where the users correspond to points on a line and the collection $U_1, U_2, \ldots$ to segments[9]. If the ring $S$ consists of a small number of segments, then the number of subsets in the union is small. In such cases using Protocol **??** is very attractive: first even though $S$ may be large, there is no need to perform the encryption in Step 1 for each member of $S$ but rather for each segment. The deniable and source hiding properties assure us that even though the prover did not choose his keys by himself, and they might be known to the adversary, he can still enjoy anonymity.

## 8 Open Problems and Discussion

There are several specific questions that arise from this work, as well as more general ones:

- In the access scheme of Section **??** is it possible for the members to be mutually untrusting with respect to deniability. The protocol is presented so that $P$ is the one responsible for checking that the shares are proper. Is this necessary (without resorting to a complex multi-part computation).
- What is the communication complexity of ring authentication, in particular is it possible to perform such authentication by sending $o((|S|)$ bits, assuming the identity of the members of $S$ is known, or a the very least without sending $|S|$ encryptions (or signatures). Note that [**?**] manage to achieve that using

[9] The Subtree Difference example of [**?**] can be adapted to work in this case.

random oracles. Also Boaz Barak (personal communication) has pointed out that using Kilian's arguments [?] it is possible in principle to obtain inefficient but succinct protocols, so the remaining question is whether it is possible to do so while maintaing the efficiency of the protocol.
– What is the weakest form of security required from an encryption scheme that is sufficient to be used in our ring deniable authentication scheme. In particular what are the minimum requirements for Protocol **??**?
– Is obtaining *source hiding* only an easier task than achieving deniability? Our approach was to take a deniable scheme (Protocol **??** and turn it into a ring scheme, but perhaps aiming directly for source hiding will yield other schemes and in particular 2-round ones (note that Witness Indistinguishability is possible in 2-rounds [?].)
– Is it possible to obtain *source hiding* in the case of shared keys, say in the Subset Cover Framework? Note that shared key authentication implies *deniability*, but running a protocol like **??** is problematic, since proving in zero-knowledge the consistency of shared-key encryptions is difficult.
– Is it possible to use the Fiat-Shamir heuristic[10] and remove the interaction from authentication protocols such as Protocols **??**, **??** and **??** and thus get new types of signature schemes from encryptions schemes?

An important social concern that both this work and [?] raise is the implication to PKI. The fact that a user that has established a public key can be 'drafted' to a ring $S$ without his consent might be disturbing to many users. On the other hand we believe in a more positive interpretation of the results. Allowing some degree of anonymity as well as leaking secret has always been important at least in modern societies and this form of protocols allows the re-introduction of it.

In general we find the issue of anonymity and deniability to be at the heart of the open scientific investigation of cryptography. A very natural research program is to find the precise mapping between possible and impossible in this area. It seems that behind every impossibility result lies a small twist (in the model perhaps) that allows the tasks to be performed.

## Acknowledgments

I wish to thank Boaz Barak, Cynthia Dwork and Adi Shamir for helpful discussions and the anonymous referees for useful comments.

## References

1. M. Bellare, A. Desai, D. Pointcheval and P. Rogaway. *Relations among notions of security for public-key encryption schemes,* Advances in Cryptology – CRYPTO'98, LNCS 1462, Springer, pp. 26–45.

[10] Fiat and Shamir [?] proposed a general method for converting *public* coins zero-knowledge proof systems into signatures. The analysis of the method is based on the random oracle model.

2. M. Bellare and P. Rogaway, *Optimal Asymmetric Encryption,* Advances in Cryptology-Eurocrypt '94, LNCS 950, Springer, 1995, pp. 92–111.
3. Dan Boneh, *Simplified OAEP for the RSA and Rabin Functions*, Advances in Cryptology - CRYPTO 2001, LNCS2139, Springer 2001, pp. 275–291.
4. D. Boneh and M. Franklin, *Anonymous Authentication with Subset Queries*, ACM Conference on Computer and Communications Security 1999, pp. 113–119.
5. D. Boneh and M. Franklin, *Identity-Based Encryption from the Weil Pairing,* Advances in Cryptology - CRYPTO 2001, LNCS 2139, Springer, 2001, pp. 213–229.
6. J. Boyar, D. Chaum, I. Damgård and T. P. Pedersen: *Convertible Undeniable Signatures,* Advances in Cryptology – CRYPTO'90, Springer, 1991, pp. 189–205.
7. E. Bresson, J. Stern and M. Szydlo, *Threshold Ring Signatures for Ad-hoc Groups,* Advances in Cryptology – CRYPTO'2002, (these proceedings).
8. J. Camenisch, *Efficient and Generalized Group Signatures*, Advances in Cryptology - EUROCRYPT'97, LNCS 1233, Springer, 1997, pp. 465–479.
9. J. Camenisch and I. Damgård, *Verifiable Encryption, Group Encryption, and Their Applications to Group Signatures and Signature Sharing Schemes,* Advances in Cryptology - Asiacrypt 2000, LNCS 1976, Springer, 2000, pp. 331–345.
10. J. Camenisch and A. Lysyanskaya, *An Identity Escrow Scheme with Appointed Verifiers,* Advances in Cryptology - Crypto 2001, LNCS 2139, Springer, 2001, pp. 388–407.
11. J. Camenisch, M. Michels, *Separability and Efficiency for Generic Group Signature Schemes*, Advances in Cryptology - CRYPTO'99, LNCS 1666, Springer, 1999, pp. 106–121.
12. J. Camenisch and M. Stadler, *Efficient Group Signature Schemes for Large Groups*, Advances in Cryptology - CRYPTO'97, LNCS 1294, Springer, 1997, pp. 410–424.
13. R. Canetti, C. Dwork, M. Naor and R. Ostrovsky, *Deniable Encryption,* Advances in Cryptology - CRYPTO'97, LNCS 1294, Springer, 1997, pp. 90–104.
14. D. Chaum, *Untraceable electronic mail, return addresses, and digital pseudonyms*, Comm. of ACM, vol. 24(2), 1981, pp. 84–88.
15. D. Chaum and H. van Antwerpen, *Undeniable Signatures*, Advances in Cryptology – CRYPTO'89, LNCS 435, Springer, 1990, pp. 212–216.
16. D. Chaum and E. van Heyst, *Group Signatures*, Advances in Cryptology - EUROCRYPT'91, LNCS 541, Springer, 1991, pp. 257–265.
17. D. Chaum and E. van Heyst and B. Pfitzmann, *Cryptographically Strong Undeniable Signatures, Unconditionally Secure for the Signer*, Advances in Cryptology - CRYPTO'91, LNCS 576, Springer, 1992, pp. 470–484.
18. C. Cocks. *An identity based encryption scheme based on quadratic residues*, Cryptography and Coding, LNCS 2260, Springer, 2001, pp. 360–363.
19. R. Cramer, I. Damgård, B. Schoenmakers, *Proofs of Partial Knowledge and Simplified Design of Witness Hiding Protocols,* Advances in Cryptology - CRYPTO'94, LNCS, Springer, 1994, pp. 174–187.
20. R. Cramer and V. Shoup, *A Practical Public Key Cryptosystem Provably Secure against Adaptive Chosen Ciphertext Attack*, Advances in Cryptology - CRYPTO'98, LNCS 1462, Springer, 1998, pp. 13–25.
21. A. De Santis, G. Di Crescenzo, G. Persiano, M. Yung, *On Monotone Formula Closure of SZK*, Proc. 35th IEEE FOCS, 1994, pp. 454–465.
22. W. Diffie, and M.E. Hellman. New Directions in Cryptography. *IEEE Trans. on Info. Theory*, IT-22 (Nov. 1976), pages 644–654.
23. D. Dolev, C. Dwork and M. Naor, *Non-malleable Cryptography*, Siam J. on Computing, vol 30, 2000, pp. 391–437.

24. C. Dwork and M. Naor, *Zaps and Their Applications*, Proc. 41st IEEE Symposium on Foundations of Computer Science, pp. 283–293. Full version: ECCC, Report TR02-001, `www.eccc.uni-trier.de/eccc/`.
25. C. Dwork, M. Naor and A. Sahai, *Concurrent Zero-Knowledge*, Proc. 30th ACM Symposium on the Theory of Computing, Dallas, 1998, pp. 409–418.
26. A. Fiat and A. Shamir, *How to Prove Yourself: Practical Solutions to Identification and Signature Problems,* Advances in Cryptology - CRYPTO'86, LNCS 263, Springer, 1987, pp. 186–194.
27. E. Fujisaki, T. Okamoto, D. Pointcheval, J. Stern, *RSA-OAEP Is Secure under the RSA Assumption*, Advances in Cryptology - CRYPTO 2001, pp. 260–274.
28. R. Gennaro, H. Krawczyk and T. Rabin, *RSA-Based Undeniable Signatures*, Advances in Cryptology – CRYPTO'97, LNCS 1294, Springer, 1997, pp. 132–149.
29. O. Goldreich and Y. Oren, *Definitions and properties of Zero-Knowledge proof systems,* J. of Cryptology, Vol 7, 1994, pp.1–32.
30. S. Goldwasser, S. Micali and R. Rivest, *A secure digital signature scheme*, SIAM J. on Computing 17, 1988, pp. 281–308.
31. M. Jakobsson, K. Sako and R. Impagliazzo, *Designated Verifier Proofs and Their Applications,* Advances in Cryptology - EUROCRYPT '96, pp. 143–154.
32. J. Katz, *Efficient and Non-Malleable Proofs of Plaintext Knowledge and Applications Cryptology*, ePrint Archive, Report 2002//027, `http://eprint.iacr.org/`
33. J. Kilian, *A Note on Efficient Zero-Knowledge Proofs and Arguments*, Proc. 24th ACM Symposium on the Theory of Computing, 1992, pp. 723–732.
34. J. Kilian and E. Petrank, *Identity Escrow*, Advances in Cryptology - CRYPTO '98 LNCS 1462, 1998, pp. 169–185.
35. H. Krawczyk and T. Rabin, *Chameleon Hashing Signatures*, Proceedings of Network and Distributed Systems Security Symposium (NDSS) 2000, Internet Society, pp. 143–154.
36. M. Naor. *Bit Commitment Using Pseudo-Randomness*, Journal of Cryptology, vol. 4, 1991, pp. 151–158.
37. D. Naor, M. Naor and J. B. Lotspiech, *Revocation and Tracing Schemes for Stateless Receivers*, Advances in Cryptology – CRYPTO 2001, pp. 41–62. LNCS 2139, Springer, 2001, pp. 205–219. Full version: Cryptology ePrint Archive, Report 2001/059, `http://eprint.iacr.org/`
38. R. L. Rivest, A. Shamir, and Y. Tauman, *How to Leak A Secret*, Advances in Cryptology - ASIACRYPT 2001, Lecture Notes in Computer Science, Vol. 2248, Springer, pp. 552–565.
39. A. Sahai, *Non-Malleable Non-Interactive Zero Knowledge and Achieving Chosen-Ciphertext Security*, Proc. 40th IEEE Symposium on Foundations of Computer Science, 1999, pp. 543–553.
40. A. Shamir, *How to Share a Secret*, Communications of the ACM 22, 1979, pp. 612–613.
41. A. Shamir, *Identity-Based Cryptosystems and Signature Schemes*, Advances in Cryptology - CRYPTO'84, LNCS 196, Springer, 1985, pp. 47–53.
42. V. Shoup, *OAEP Reconsidered,* Advances in Cryptology - CRYPTO 2001, LNCS, Springer, 2001, pp. 239–259.
43. Bibliography on Secret Sharing Schemes, maintained by D. Stinson and R. Wei. `www.cacr.math.uwaterloo.ca/~dstinson/ssbib.html`