Deniable Encryption

(Prelimiary version)

Ran Canetti Cynthia Dwork Moni Naor Rafi Ostrovsky

May 10, 1996

Abstract

Consider a situation in which the transmission of an encrypted message can be intercepted by an authority, and subsequently (say, in response to court order) the sender can be coerced to reveal the keys and random choices used in generating the ciphertext, thereby revealing the message sent. An encryption scheme is *deniable* if the sender can generate "plausible" keys and random choices that will satisfy the authority and at the same time keep the past communication private. Analogous requirements can be formulated with respect to coercion of the receiver and with respect to coercion of both parties.

Deniable encryption is a strong primitive. In particular, it yields the first solution to the problem of incoercible ("receipt-free") voting *requiring no physical security assumptions.* Deniable encryption also provides a simple and elegant implementation of adaptively secure multiparty computation.

In this paper we define and construct various types of deniable encryption schemes.

1 Introduction

Imagine a country in which the government practices the following policy, intended to learn the contents of encrypted data transmitted between citizens under suspicion. A Ciphertext Investigation Authority records the (encrypted) data transmitted (to or from) a suspected citizen. When enough suspicion is accumulated, and a suitable court-order is obtained, the suspect is required to reveal the cleartexts that correspond to the ciphertexts that the Authority has recorded, along with any private keys and random choices used for encryption and decryption. The Authority then verifies that the cleartexts, combined with the private keys and the random choices, indeed "look convincing". In particular, they should match the recorded ciphertexts. (If the suspect fails to come up with the matching cleartexts then this fact is held against him in court.)

This policy is based on the fact that, when standard semantically secure encryption is used it is infeasible - or even impossible - for a party to come up with keys, randomness, and cleartexts that match a given ciphertext *unless these were the real ones used.* (This holds both when the coerced party is the sender of the encrypted data and when the party is the receiver.) Is it possible to devise encryption schemes that allow people to maintain the privacy of their communication, even in the presence of such government policy? That is, are there deniable encryption schemes that allow the "coerced" citizens (senders, receivers, or perhaps both simultaneously) to come up with "fake" keys and random choices that will convince the Authority that the exchanged ciphertexts correspond to different *cleartexts,* other than the real ones?

Deniable encryption seems impossible at first glance: consider a ciphertext c sent from Sender to Receiver. If, using two different random choices, Sender could have generated c both as an encryption of a message m_1 and as an encryption of a different message, m_2, then how can Receiver, upon receipt, correctly decide whether Sender meant to send m_1 or m_2? A more careful inspection shows that deniable encryption schemes can be constructed, in a variety of ways. In this paper we demonstrate how.

We remark that the coercive policy described above is useless against communicating parties that are equipped with a physically secure channel, since on such a channel the Authority gets no ciphertext to match against the cleartexts, keys, and random choices. We obtain a similar situation in the absence of physically secure channels. Indeed, the bottom line of our work is that, even without physically secure channels, coercion is impossible: nothing useful can be learned by the coercion, so there is no point in even trying.

This coercive scenario is not far from real life, all the more so as "authority" can be legitimate or illegitimate, and "pressure" to release information can be legal or felonious. Indeed, a classic motivation for deniable encryption is the prevention of coercion in electronic secret voting schemes [6]: a coercer may demand to know a person's vote after hearing the corresponding ciphertext. In contrast with our work, all previous work on uncoercible (also known as *receipt-free*) voting requires that, for a crucial part of the conversation, the communicating parties share a physically secure channel; thus, the coercer hears no ciphertext and the problem disappears [3, 10, 11][1]. Having made this physical assumption, these papers are concerned with achieving accurate and private voting in the presence of untrustworthy *voting centers*, while maintaining incoercibility. Deniable encryption schemes provide the first solution to the receipt-free voting problem requiring no physical security assumptions. Indeed, combining our results with earlier work of Benaloh [2] yields an incoercible, accurate, and private, voting scheme requiring no physical security assumptions.

A particularly interesting use of deniable encryption is for storing private data, encrypting it in such a way as to protect it from coercion. Consider a record, such as a private diary, that may be publicly accessible but is kept private using a deniable encryption scheme. (Alternative examples include a psychiatrist's or lawyer's notes.) When coerced (say, by a court order) to reveal the cleartext and the secret key used for encryption and decryption, the owner of the record can instead "reveal" (1) a variety of fake cleartexts of her choice, and (2) for each such cleartext, a fake key that, together with the fake cleartext, will yield the public ciphertext. One of our schemes, described in Section 4, is

[1] In [11] a slightly different physical security assumption is made, namely that the random choices used for encryption are physically unavailable. The result is the same: the 'deniability problem' disappears.

particularly suitable for this application.

In this work we construct several types of deniable encryption schemes. We distinguish deniable encryption schemes according to which parties may be coerced: a sender-deniable scheme is resilient against coercion (*i.e.*, demanding to see the secret data of) the sender of the ciphertext. Receiver-deniable and sender-and-receiver-deniable schemes are defined analogously. We also distinguish between shared-key and public-key deniable encryption schemes. In a shared-key scheme the sender and receiver are assumed to share a secret key, about which the adversary has no *a priori* information. A public-key scheme "starts from scratch" by assuming that the sender and receiver have no shared secret information.

Yet another issue is how dynamic the *senders* actions can be: it seems better to design a scheme in which the sender can decide how to open the message as late as *at time of coercion*, than design a scheme where the choices of the fake messages have to be made *at time of encryption.* Precise definitions appear in Section 2.

We show, via simple constructions, how to transform any deniable encryption scheme resilient against coercing the sender into a scheme resilient against coercing the receiver, and vice-versa (Section 5). We also show how a scheme resilient against corrupting both the sender and the receiver can be constructed based on a scheme resilient against corrupting the sender. This last construction requires the help of other parties in a network, and works as long as at least one other party remains uncoerced.

We describe two *sender-resilient, public-key* deniable encryption schemes, based on any trapdoor one-way permutation, *e.g.*, RSA (Section 3). The schemes differ in the type and quality of deniability achieved (described rigorously below). Next we present two shared-key schemes. One is based on any pseudorandom generator (Section 4) and the other (in Appendix C) on the Quadratic Residuocity Assumption.

Finally, deniable encryptions provide a conceptually simple and elegant alternative solution to the recently solved problem of general secure multiparty computation in the presence of an *adaptive* adversary. We elaborate briefly. Protocols for securely computing any function in a multiparty scenario in the presence of a non-adaptive adversary – one that chooses whom to corrupt *before* execution of the protocol begins – were shown in [8]. Recently protocols secure against adaptive adversaries – ones that choose whom to corrupt *during* the course of the computation, based on the information seen so far – were presented [4].[2] These protocols are based on another new type of encryption protocol, called non-committing encryption. Non-committing encryptions have the same flavour as deniable encryptions, in that there exist ciphertexts that can be "opened" as encryptions of, say, both '1' and '0'. However, non-committing encryptions are strictly weaker than deniable ones. One main difference, among others, is that in non-committing encryptions the parties *using* the scheme are, in general, unable to generate ciphertexts that can be opened both ways. Such ciphertexts can only be generated by a *simulator* (which is an artifact of the model in [4]). In deniable encryption each ciphertext generated by parties using the scheme has unique decryption, and at the same time can be "opened" in several ways for a coercer. In particular, the non-committing encryption scheme in [4] is *not* deniable. However, any deniable encryption scheme, resilient against coercing both the sender and the receiver, is non-committing. Indeed, one of the schemes shown in the sequel constitutes a conceptually simple and elegant alternative to the [4] scheme. See Appendix D for more details.

2 Definitions

Let us recall the definition of computational distance of distributions. Here and in the sequel a real-valued function is negligible if it approaches zero faster than any polynomial.

[2] [5, 1] obtained solutions for the case of adaptive adversaries; however, the security of their solutions depended in a crucial way on trusting each party to continuously *erase* used random bits and private keys. Such solutions are unsatisfactory because such trust makes no sense in a scenario in which no party is thoroughly trusted by others.

Definition 1 *Let $\mathcal{A} = \{A_n\}_{n\in\mathrm{N}}$ and $\mathcal{B} = \{B_n\}_{n\in\mathrm{N}}$ be two ensembles of probability distributions, and let $\delta : \mathrm{N} \to [0,1]$. We say that $\mathcal{A}$ and $\mathcal{B}$ are $\delta(n)$-*close *if for every polytime distinguisher D and for all large enough n, $|\text{Prob}(D(A_n) = 1) - \text{Prob}(D(B_n) = 1)| < \delta(n)$.*

If $\delta(n)$ is negligible then we say that $\{A_n\}$ and $\{B_n\}$ are computationally indistinguishable *and write $A_n \overset{c}{\approx} B_n$.*

PUBLIC-KEY ENCRYPTION. Consider a sender S and a receiver R that, *a priori*, have no shared secret information. They engage in some protocol in order to transmit a message from S to R. (If a standard public key encryption scheme is used then this protocol may consist of the receiver sending his encryption key to the sender, who responds with the encrypted message.) We first define intuitively what we desire from a deniable encryption protocol. First, the receiver should be able to decrypt the correct value (except for negligible probability of error). Next the protocol should be semantically secure against eavesdroppers, as in [9]. Finally, the sender should have a 'faking' algorithm ϕ with a property described roughly as follows. Given an encryption c of a message m_1 and the random input of the sender, together with another message m_2, ϕ generates a fake random input for the sender, that makes c look like an honest encryption of m_2. We now present a more precise definition.

Let M be the set of all possible messages to be sent from S to R. (M can be $\{0,1\}^s$ for some s.) Let π be a protocol for transmitting a message $m \in M$ from S to R. Let $\text{COM}_\pi(m, r_S, r_R)$ denote the communication between S and R for transmitting m, when S has random input r_S and R has random input r_R. Let $\text{COM}_\pi(m)$ denote the random variable describing $\text{COM}_\pi(m, r_S, r_R)$ when r_S and r_R are uniformly and independently chosen.

Definition 2 *A protocol π with sender S and receiver R, and with security parameter n, is a $\delta(n)$-*sender-deniable encryption *protocol if:*

Correctness: *The probability that R's output is different than S's input is negligible (as a function of n).*

Security: *For any $m_1, m_2 \in M$ we have $\text{COM}_\pi(m_1) \overset{c}{\approx} \text{COM}_\pi(m_2)$.*

Deniability: *There exists an efficient 'faking' algorithm ϕ having the following property with respect to any $m_1, m_2 \in M$. Let r_S, r_R be uniformly chosen random inputs of S and R, respectively, let $c = \text{COM}_\pi(m_1, r_S, r_R)$, and let $\tilde{r}_S = \phi(m_1, r_S, c, m_2)$. Then, the random variables*

$$(m_2, \tilde{r}_S, c) \quad \text{and} \quad (m_2, r'_S, \text{COM}_\pi(m_2, r'_S, r'_R)) \tag{1}$$

are $\delta(n)$-close, where r'_S, r'_R are independent, uniformly chosen random inputs of S and R, respectively.

The right hand side of (1) describes the adversary's view of an honest encryption of m_2 according to protocol π. The left hand side of (1) describes the adversary's view when a party falsely claims that c is an encryption of m_2. The definition requires that the adversary cannot distinguish between the two cases with probability more than $\delta(n)$.

Schemes resilient against coercion of the receiver, or simultaneous coercion of both the sender and the receiver, are defined in a similar way. They appear as Definitions 5 and 6 in Appendix A.

SHARED-KEY ENCRYPTION. In a shared-key scenario, the sender and receiver share a random, secret key about which the adversary is assumed to have no *a priori* information. Consequently, here the parties can also present the coercer with a "fake shared key", on top of presenting her with fake random inputs. This is captured as follows. The communication between the parties depends also on a shared key k, and is denoted $\text{COM}_{\pi,k}(m, r_S, r_R)$ (where m, r_S, r_R are the same as before). Below we define sender-deniability. Receiver-deniability and sender-and-receiver-deniability are defined analogously.

Definition 3 *A protocol π with sender S and receiver R, and with security parameter n, is a* shared-key $\delta(n)$-sender-deniable encryption *protocol if:*

Correctness: *The probability that R's output is different than S's input is negligible (as a function of n).*

Security: *For any $m_1, m_2 \in M$ and for a shared-key k chosen at random, we have* $\text{COM}_{\pi,k}(m_1) \overset{c}{\approx} \text{COM}_{\pi,k}(m_2)$.

Deniability: *There exists an efficient 'faking' algorithm ϕ having the following property with respect to any $m_1, m_2 \in M$. Let k, r_S, r_R be uniformly chosen shared-key and random inputs of S and R, respectively, let $c = \text{COM}_{\pi,k}(m_1, r_S, r_R)$, and let $(\tilde{k}, \tilde{r}_S) = \phi(m_1, k, r_S, c, m_2)$. Then, the random variables*

$$(m_2, \tilde{k}, \tilde{r}_S, c) \quad \text{and} \quad (m_2, k', r'_S, \text{COM}_{\pi,k'}(m_2, r'_S, r'_R))$$

are $\delta(n)$-close, where k', r'_S, r'_R are independent, uniformly chosen shared key and random inputs of S and R, respectively.

Here it may be the case that the same key k was used to encrypt several messages. Thus the "fake key" $\tilde{k}$ must account for all these messages together. We elaborate on this point in Section 4.

PLAN-AHEAD DENIABILITY. Definitions 2 and 3 require that the encryption of a message m_1 be generated independently of the "fake" message m_2. This is a strong property, and in particular it allows a coerced party to choose m_2 as late as *at time of coercion,* and then to invoke algorithm ϕ to generate the appropriate fake random input. A somewhat weaker deniability property allows the encryption algorithm to have the fake message(s) as part of its input. This forces the sender to choose the fake message(s) *at time of encryption.* This restriction may be considerable. In particular, since all fake messages have to be specified in advance, there may be only a polynomial number of them. Still, this type of deniability, called plan-ahead deniability, is very useful (e.g. for maintaining 'deniable records' of data, as proposed in the Introduction).

Here we define plan-ahead sender-deniability with respect to shared-key schemes. We restrict the encryption protocol to send only one message from S to R. Furthermore, we require that both encryption and decryption be deterministic. (We impose these restrictions since they simplify the definition considerably and the scheme described in the sequel complies with them. General definitions of plan-ahead deniable schemes, public-key and shared-key, are formulated analogously.)

We motivate our definition. A standard (non deniable) shared-key encryption scheme consists of a encryption algorithm E and a decryption algorithm D, as follows. On input k, m, where k is the shared key and m is the message to be encrypted, E outputs a ciphertext c such that $D(k, c) = m$. In our scenario the sender has, in addition to k, m, also $l-1$ pairs $(k_2, m_2), ..., (k_l, m_l)$, where $k_2...k_l$ are fake shared keys (chosen beforehand uniformly at random), and $m_2, ...m_l$ are arbitrary messages. Upon coercion, the sender wants to be able to choose an arbitrary $i \in 2...l$ and claim that the ciphertext c is an encryption of m_i under shared key k_i. Below we let $k = k_1, m = m_1$, and write $E(k_1, m_1; k_2, m_2, ...k_l, m_l) = c$ where c is the generated ciphertext. When we don't specify the values of $k_2, m_2, ...k_l, m_l$, we write $c = E(k_1, m_1; ...)$.

Definition 4 *A* shared-key plan-ahead sender-deniable encryption *scheme consists of an encryption algorithm E and a decryption algorithm D, such that:*

Correctness: *For any $m \in M$,* Prob$(D(k, E(k, m, ...)) \neq m)$ *is negligible, where the probability is taken over the choices of the key k.*

Security: *For any $m, m' \in M$ we have $E(k, m; ...) \overset{c}{\approx} E(k', m'; ...)$, where the probabilities are taken over the choices of k, k'.*

Deniability: *For all $m_1, m_2, ..., m_l \in M$ and for all $i \in 2...l$ we have*

$$E(k_1, m_1; k_2, m_2, ..., k_l, m_l) = c = E(k_i, m_i; k_1, m_1, ..., k_{i-1}, m_{i-1}, k_{i+1}, m_{i+1}, ..., k_l, m_l).$$

Remarks:

1. The number, l, of fake messages that correspond to a single ciphertext is a parameter of the scheme. One of the schemes described here is limited to $l = 2$; another can handle any value of l. (Of course, the work done is proportional to l.)

2. Plan-ahead sender-deniable schemes also differ with respect to whether the *receiver* learns the fake messages $m_1 ... m_l$. (This is determined by whether the distribution of c depends on $m_1 ... m_l$.) We prefer schemes where the receiver does not learn anything about the fake messages, since the sender may not want the receiver to learn this information.

3 Public-key Deniable Encryption

OVERVIEW. We describe three public-key deniable encryption schemes. The first, called the basic scheme, is only a partial solution to the problem. We use it as a building-block to construct complete schemes. (It can also be used to construct a non-committing encryption scheme, as described in Appendix D.) The Parity Scheme is a $\frac{1}{n}$-sender-deniable scheme according to Definition 2. Roughly speaking, this means that in this scheme the probability of successful coercion vanishes *linearly* in the amount of work invested in encryption and decryption. This may be sufficient in most situations.[3]

We also describe another scheme, called the Undetectable Parity Scheme, where the probability of successful coercion is negligible. But this comes at a price: this scheme offers a qualitatively different notion of deniability than the one formalized in Definitions 2 through 6. We present this notion of deniability below. An interesting open problem is whether there exist schemes that comply with, say, Definition 2 with negligible $\delta(n)$.

All three schemes are sender-deniable. Receiver-deniable and Sender-and-receiver-deniable schemes can be constructed from these using the techniques of Section 5. Our schemes encrypt one bit at a time. Longer messages can be encrypted bit by bit.

SPARSE SETS. Our schemes are based on a combinatorial primitive that can be informally described as follows. We assume that there exists a set $\mathcal{S} \subset \{0,1\}^t$, together with trapdoor information d, such that:
1. $\mathcal{S}$ is small: $|\mathcal{S}| \leq 2^{t-k}$ for some sufficiently large k.
2. It is easy to generate random elements $x \in \mathcal{S}$, even without the trapdoor d.
3. Given $x \in \{0,1\}^t$ and d it is easy to decide whether $x \in \mathcal{S}$.
4. Without d it is infeasible to decide whether a given value $x \in \{0,1\}^t$ was chosen from $\mathcal{S}$ or uniformly from $\{0,1\}^t$.
We call such sets sparse. (Formal definitions can be extracted from this description.)

We present two simple constructions of sparse sets. Both use a trapdoor permutation $f : \{0,1\}^s \rightarrow \{0,1\}^s$, and its hard-core bit function $B : \{0,1\}^s \rightarrow \{0,1\}$ (say, use the Goldreich-Levin bit [7]).
Construction I: Let $t = sk$. Represent each $x \in \{0,1\}^t$ as a vector $x = x_1 ... x_k$ where each $x_i \in \{0,1\}^s$. Then let $\mathcal{S} = \{x \in \{0,1\}^{sk} \mid \forall i = 1..k,\ B(f^{-1}(x_i)) = 0\}$. Here $|\mathcal{S}| = 2^{(s-1)k} = 2^{t-k}$.
Construction II: Let $t = s + k$. Represent each $x \in \{0,1\}^t$ as $x = x_0, b_1 ... b_k$ where $x_0 \in \{0,1\}^s$ and for $i > 0$ each $b_i \in \{0,1\}$. Then let $\mathcal{S} = \{x \in \{0,1\}^{s+k} \mid \forall i = 1..k,\ B(f^{-i}(x_0)) = b_i\}$. Here $|\mathcal{S}| = 2^s = 2^{t-k}$.
It is easy to verify that both constructions satisfy requirements 1-4. Construction II is more efficient in that it uses only $t = s + k$ instead of $t = sk$.

[3] Linearly vanishing probability of successful coercion may be too high if a large number (or very long) messages are being examined. In this case, coercion can be avoided by using the public-key scheme only to exchange a short shared key, and then carrying out the rest of the communication using a deniable shared-key scheme, such as those of Section 4.

THE BASIC SCHEME. The public encryption key is a sparse set $\mathcal{S} \subset \{0,1\}^t$. The private decryption key is the corresponding trapdoor d.

Encryption: *To encrypt 1, send a random element of $\mathcal{S}$. To encrypt 0, send a random element in $\{0,1\}^t$.*

Decryption: *If the ciphertext x is in $\mathcal{S}$ then output 1. Else output 0.*

Opening an encryption honestly: *reveal the true random choices used.*

Opening an encryption dishonestly: *If the encrypted bit is 1, i.e., the ciphertext x is a random element in $\mathcal{S}$, then claim that x was chosen at random from $\{0,1\}^t$ and thus x is an encryption of 0. If the encrypted bit is 0 then lying is infeasible since the ciphertext x is in $\mathcal{S}$ only with negligible probability 2^{-k}.*

Analysis: An encryption of 1 is always decrypted correctly. An encryption of 0 may be decrypted as 1 with probability 2^{-k}. Since lying is possible only in one direction, this is only a partial solution to the problem. Next we describe a scheme where lying is possible in both directions.

THE PARITY SCHEME. We use $V \in \{\mathcal{S}, \mathcal{R}\}^n$ to denote a length n vector formed as follows. First choose $\hat{V} \in \{0,1\}^n$. Next, if the ith element of $\hat{V}$ is 1 (resp., 0) then the ith element of V chosen uniformly at random from $\mathcal{S}$ (resp., $\{0,1\}^t$). We call elements drawn uniformly from $\mathcal{S}$ (resp., $\{0,1\}^t$) $\mathcal{S}$-elements (resp., $\mathcal{R}$-elements).

Encryption: *To encrypt 0 (1), choose a random even (odd) $i \in 0..n$, and send a random $V \in_R \{\mathcal{S}, \mathcal{R}\}^n$ containing i $\mathcal{S}$-elements.*

Decryption: *Output the parity of the number of elements in V that belong to $\mathcal{S}$.*

Opening an encryption honestly: *Reveal the real random choices used in generating V.*

Opening an encryption dishonestly: *Let i be the number chosen by the sender in the encryption process. The sender claims that she has chosen $i-1$ rather than i. (Consequently, the parity of i flips.) For this, she chooses a randomly selected $\mathcal{S}$-element in the ciphertext, and claims that it is an $\mathcal{R}$-element (i.e., a random string in $\{0,1\}^t$).*

Analysis: Security of the Parity Scheme against eavesdroppers that see only the ciphertext is straightforward. The probability of erroneous decryption is at most $n2^{-k}$. We show deniability. Assume that n is odd, and let x be an encryption of 1 to generate which the sender chose the value i. Then, i was chosen at random from $1, 3, ...n$. Consequently, the value $i-1$ is uniformly distributed over $0, 2, ..., n-1$. Thus, when the sender claims that she has chosen $i-1$, she demonstrates the correct distribution of i for encrypting 0. Thus, cheating in this direction is undetectable (as long as $\mathcal{S}$-elements cannot be distinguished from $\mathcal{R}$-elements). Assume now that x is an encryption of 1. Thus i is chosen uniformly from $0, 2, ..., n-1$. Now, deniability is possible (and undetectable) as long as $i \neq 0$. However, $i = 0$ only with probability $\frac{1}{n}$.

The Parity Scheme can be modified slightly so as to permit deniability with all but negligible probability. In the modified scheme, to encrypt a bit 0 (1) the vector V is chosen *uniformly* from all vectors in $\{\mathcal{S}, \mathcal{R}\}^n$ with even (odd) number of $\mathcal{S}$-elements. In this variant, however, the statistical distance between the distribution of the number of $\mathcal{S}$-elements in an honestly opened encryption and in a dishonestly opened encryption is as large as $\frac{1}{\sqrt{n}}$. Finally, a second variant, omitted here, combines the virtues of both schemes: negligible probability of being unable to lie combined with statistical distance $O(\frac{1}{n})$.

In this modified Parity Scheme, the sender can send an arbitrarily long message of her choice and later, upon coercion, decide what text she would like to claim to have sent. The problem, addressed in the next construction, is that if she wishes to open too many encryptions dishonestly, her behavior becomes statistically suspect.

Using our second construction of sparse sets, the Parity Scheme uses $ns = nt + nk$ ciphertext bits to encrypt each bit of the plaintext. We also construct a scheme with similar performance, where only $t + nk$ ciphertext bits are used for each plaintext bit. The construction is omitted from this extended abstract.

THE UNDETECTABLE PARITY SCHEME. In this scheme, the sender chooses the ciphertext according to one of four distributions, T_0, T_1, C_0, C_1. Distribution T_b is used by a sender who wishes to send the binary value b and does not wish to have the ability to open dishonestly upon coercion. Distribution C_b is also used to send the bit value b, but by a sender who wishes to preserve both the ability to open "honestly" and the ability to open dishonestly upon coercion. *This choice can be made at time of coercion.* In particular, if the sender encrypts according to distribution C_b then, upon coercion, the sender can appear to have chosen either from T_0 or T_1. This is why we put the word "honestly" in quotes: if the sender has chosen from C_b and if she does not wish to lie about the value sent, then she acts so as to appear to have chosen from T_b, which may require lying about her random choices. The philosophical question arises: "Why would anyone ever choose to send a message according to the T_b's?" We do not have a satisfactory reply to this question. However, we cannot eliminate the option to do so from the protocol because then the coercer would "know" that the sender is lying if the sender were to appear to have chosen according to one of the T_b's. Thus we get a slightly different type of deniability, called multi-distribution deniability, for this protocol. The full requirements for the protocol are formally specified in Definition 7 in Appendix B. We focus here on multi-distribution deniability.

The scheme requires an extra binary parameter, Preserve, to control whether the sender chooses from the T or the C distributions. The definitions of COM_π and ϕ are modified accordingly to take an additional parameter.

Multi-Distribution Deniability: There exists an efficient "faking" algorithm ϕ having the following property with respect to any $m_1, m_2 \in M$. Let r_S, r_R be uniformly chosen random inputs of S and R, respectively, let $c = \text{COM}_\pi(1, m_1, r_S, r_R)$, and let $\tilde{r}_S^{(i)} = \phi(i, m_1, r_S, c, m_2)$, for $i \in \{1, 2\}$. The new parameter i tells the faking algorithm which message the sender wants to claim to have sent. Then, the random variables

$$(m_i, \tilde{r}_S^{(i)}, c) \quad \text{and} \quad (m_i, r'_S, \text{COM}_\pi(0, m_i, r'_S, r'_R)) \tag{2}$$

are $\delta(n)$-close, where r'_S, r'_R are independent, uniformly chosen random inputs of S and R, respectively.

The left-hand side of Equation 2 describes the view of the adversary when the sender, having preserved the ability to open dishonestly when sending m_1, opens with value m_i (which might or might not equal m_1). The right-hand side of Equation 2 describes the adversary's view when the sender, not having preserved the ability to open dishonestly, opens an encryption of m_i. The definition requires that the adversary cannot distinguish between the two cases with probability exceeding $\delta(n)$.

For the scheme, let $T_0 = \{\mathcal{R}, \mathcal{R}\}, T_1 = C_1 = \{\mathcal{S}, \mathcal{R}\}$, and $C_0 = \{\mathcal{S}, \mathcal{S}\}$.

Encryption: *To encrypt b without preserving the ability to open dishonestly (that is, if* Preserve $= 0$*), send $V \in_R T_b$. To encrypt b preserving the ability to open dishonestly (*Preserve $= 1$*), send $V \in_R C_b$.*

Decryption: *Output the parity of the number of elements in V that belong to $\mathcal{S}$.*

Opening an encryption drawn from T_b: *Reveal the true random choices used.*

Opening an encryption drawn from C_0 as value v: *Let b be the number of elements in the ciphertext drawn from $\mathcal{S}$.If $v = 0$ then claim that V was chosen as $\{\mathcal{R}, \mathcal{R}\}$. If $v = 1$ then claim that V was chosen as $\{\mathcal{S}, \mathcal{R}\}$.*

Opening an encryption drawn from C_1 as value v: *If $v = 0$ then claim that V was chosen as $\{\mathcal{R}, \mathcal{R}\}$. If $v = 1$ then reveal the real random choices used in generating V.*

Analysis: Security of the Undetectable Parity Scheme against eavesdroppers seeing only the ciphertext is straightforward. The probability of erroneous decryption is at most 2^{-k}. Multi-Distribution deniability is immediate by inspection, assuming polynomial time indistinguishability of $\mathcal{S}$ and $\mathcal{R}$.

4 Shared-Key Deniable Encryption

Here we assume that the sender and receiver share a random, secret key on which the adversary has no *a priori* information. Consequently, the parties can also present the coercer with a "fake shared key", on top of presenting her with fake random inputs. As we will see, this added capability allows schemes of a completely different nature.

One-time-pad is a perfectly good shared-key deniable encryption scheme: Assume that the sender and the receiver share a sufficiently long random string, and each message m is encrypted by bitwise xoring it with the next unused $|m|$ bits of the key. Let k denote the part of the random key used to encrypt m, and let $c = m \oplus k$ denote the corresponding ciphertext. Then, in order to claim that c is an encryption of a message $m' \neq m$, the parties claim that the shared key is $k' = c \oplus m'$. It is easy to verify that this trivial scheme satisfies Definition 3. Here the message m' can be chosen as late as *at time of coercion.* However, using a one-time pad is generally impractical, since the key has to be as long as all the communication between the parties. For example, a one-time pad would be inappropriate for encrypting a private diary. We want schemes with a relatively short key, and in particular we want the length of the key to be independent (or almost independent) of the amount of communication between the parties.

We present a scheme where the length of the key depends only on a security parameter. (In fact, it can be as short as $\log l$, where l is the number of fake messages to which a generated ciphertext can correspond.) The scheme is a plan-ahead shared-key scheme, that complies with Definition 4. That is, the fake messages have to be chosen at time of encryption. However, l is unlimited. Furthermore, the scheme is resilient against coercing simultaneously both the sender and the receiver. The construction can be based on any pseudorandom number generator.

An especially interesting application for this scheme is a method for keeping diaries or other records of private data encrypted in a deniable fashion. The user chooses an encryption key k_1 (as well as $l-1$ fake keys $k_2 ... k_l$, as described below) and runs this scheme to generate the encrypted records. Next, upon coercion (say a request to reveal the encryption key), the user can demonstrate that the records were any one of l different records, chosen by the user at time of encryption.

The Appendix contains a second plan-ahead scheme, which is resilient only against coercing the sender. We include it because it employs a different technical idea which may be useful in developing better schemes. This scheme is based on the Quadratic Residuocity Assumption.

It is an interesting open problem whether there exist shared-key deniable schemes that comply with Definition 3 (i.e., schemes that are not plan-ahead), and have a short shared key.

4.1 A scheme based on pseudorandom generators

Consider a sender S that wishes to send encrypted information, of unknown length, to a receiver R. Furthermore, the sender wants to be able to later 'open' the ciphertext in any one of l different ways. We partition the information into blocks, or messages of n bits each, where n is a security parameter. Each message will be encrypted separately. We let

$$\vec{m}_1 = m_1^{(1)}, m_1^{(2)}, ...$$

denote the information to be encrypted, where each each $m^{(j)}$ is an n-bit message. Also, let

$$\begin{aligned}
\vec{m}_2 &= m_2^{(1)}, m_2^{(2)}, ... \\
\vec{m}_3 &= m_3^{(1)}, m_3^{(2)}, ... \\
&\cdots\cdots \\
\vec{m}_l &= m_l^{(1)}, m_l^{(2)}, ...
\end{aligned}$$

denote the $l-1$ different sequences of fake messages. The generated ciphertext is

$$\vec{c} = c^{(1)}, c^{(2)}, ...$$

where each $c^{(j)}$ is an encryption of the jth message. (It will be seen that each $c^{(j)}$ is generated as a function of $m_1^{(j)} ... m_l^{(j)}$.)

The parties share an n-bit random key, denoted k_1. In addition, the sender chooses $l-1$ independent n-bit random numbers $k_2, ..., k_l$, called the fake keys. Later, in order to demonstrate that the transmitted ciphertext $\vec{c}$ is an encryption of $\vec{m}_i$, the sender will claim that the value of the shared key is k_i)

ENCRYPTION AND DECRYPTION. The scheme makes use of a pseudorandom number generator G that expands n-bit inputs to $3n$-bit outputs. We use G iteratively as follows. Initially we invoke G on the shared key. In iteration j, the last $2n$ bits of G's output are used to generate $c^{(j)}$. The first n bits are fed to G to generate output for the next iteration. Let $k_i^{(0)} = k_i$. For $j > 0$ let $k_i^{(j)}$ (resp., $a_i^{(j)}$, $b_i^{(j)}$) denote the first (resp., second, third) n-bit chunk of $G(k_i^{(j-1)})$.

The jth ciphertext, $c^{(j)}$, is generated as follows. Each one of $a_i^{(j)}$, $b_i^{(j)}$ and $m_i^{(j)}$ is considered as an element of Z_p for some appropriate prime $p \approx 2^n$. The sender finds the polynomial $Q^{(j)}$ of degree $l-1$ in Z_p, such that $Q^{(j)}(a_i^{(j)}) = m_i^{(j)} + b_i^{(j)}$ for each $i = 1..l$, and lets $c^{(j)} = \langle j, Q^{(j)} \rangle$. (That is, $c^{(j)}$ lists the l coefficients of $Q^{(j)}$.)

Having the shared key k_1 and $c^{(j)} = \langle j, Q^{(j)} \rangle$, the receiver computes $a_1^{(j)}$, $b_1^{(j)}$, and $m_1^{(j)} = Q^{(j)}(a_1^{(j)}) - b_1^{(j)}$.

ANALYSIS. We show that the scheme complies with Definition 4. First note that, unless $a_{i_1}^{(j)} = a_{i_2}^{(j)}$ for some $i_1 \neq i_2$, the polynomial Q is uniquely defined. Furthermore, as long as such collisions do not happen decryption is errorless. It follows from the pseudo-randomness of G that such collisions happen only with negligible probability.

Security of our scheme against eavesdroppers follows from the fact that each polynomial $Q^{(j)}$ is indistinguishable from a random polynomial in Z_p, regardless of $m_1^{(j)}...m_l^{(j)}$. To see why this holds, first observe that if $a_1^{(j)}...a_l^{(j)}$ and $b_1^{(j)}...b_l^{(j)}$ were truly random values then $Q^{(j)}$ would be a truly random polynomial of degree $l-1$ over Z_p. Thus, any test that distinguishes between $Q^{(j)}$ and a random polynomial can be turned into a test that breaks the pseudo-randomness of G.

Deniability is straightforward: For every $i = 1...l$, if the value of the shared key is k_i and the information to be transmitted is $\vec{m}_i$ then the *same ciphertext* $\vec{c}$ is generated, and the receiver decrypts $\vec{c}$ to $\vec{m}_i$.

REMARKS:

1. The length of each ciphertext $c^{(j)}$ is ln bits. This is optimal: no shorter ciphertext can hold the information contained in $m_1^{(j)}...m_l^{(j)}$.

2. Note that the coercer gains no extra information by coercing also the receiver. Thus the scheme is resilient against simultaneous coercion of both parties. (This of course holds as long as the sender and the receiver "coordinate their stories" and claim the same value for the key. This can be aided by having the sender and receiver share also the fake keys $k_2...k_l$ on top of the real key $k = k_1$.)

3. The size of the shared key can be further reduced to $\log l$ bits (instead of n bits), as follows. The shared key will represent a random number $v \in 1...l$. Before transmitting any messages, the sender chooses l random values $k_1...k_l$ (of n bits each), and will let the receiver learn *one* of those values using a one-out-of-l Oblivious Transfer for strings.[4] The receiver will choose to learn k_v. Next the parties will run the above protocol with k_v as the shared key and the other $l-1$ values as fake keys. The properties of Oblivious Transfer make sure that, upon coercion, the parties will be able to demonstrate to the coercer that the value learned by receiver in the Oblivious Transfer was any one out of $k_1...k_l$.

4. The number l of possible cleartexts that correspond to the ciphertext $\vec{c}$ is public. Thus, one may envision a coercer that demands to see all l possible cleartexts $\vec{m}_1...\vec{m}_l$, along with all keys $k_1...k_l$. Our scheme is resilient against

[4] OT schemes that work only for honest receivers are sufficient here.

such a coercer: even after seeing all of $(\vec{m}_1, k_1)...(\vec{m}_l, k_l)$, the coercer is unable to tell which of the l possibilities was actually used.

5 Coercing the Sender vs. Coercing the Receiver

We describe simple constructions that transform sender-deniable schemes into receiver-deniable schemes and vice-versa. If there are other parties that can help in transmitting the data, we also construct a sender-and-receiver-deniable scheme from any sender-deniable scheme. We describe the constructions with respect to schemes that encrypt only one bit at a time. Generalizing these constructions to schemes that encrypt arbitrarily long messages is straightforward. These constructions apply to both shared-key and public-key settings.

RECEIVER-DENIABILITY FROM SENDER-DENIABILITY. Assume a sender-deniable encryption scheme $\mathcal{A}$, and construct the following scheme $\mathcal{B}$. Let b denote the bit to be transmitted from S to R. First R chooses a random bit r, and invokes the scheme $\mathcal{A}$ to send r to S. (That is, R is the sender and S is the receiver with respect to scheme $\mathcal{A}$.) Next, S sends $b \oplus r$ to R, in the clear.

If scheme $\mathcal{A}$ is sender-deniable then, upon coercion, R can convincingly claim that the value of r was either 0 or 1, as desired. Consequently R can claim that the bit b was either 0 or 1, at wish, and scheme $\mathcal{B}$ is receiver-deniable. (We note that a similar idea is implicit in [3, 10].)

SENDER-DENIABILITY FROM RECEIVER-DENIABILITY. We use the exact same construction. It is easy to verify that if $\mathcal{A}$ is receiver-resilient then $\mathcal{B}$ is sender-resilient.

SENDER-AND-RECEIVER-DENIABILITY. Assume that S and R can use other parties $I_1, ..., I_n$ as intermediaries in their communication. The following scheme is resilient against coercing the sender, the receiver and some intermediaries, as long as at least *one* intermediary remains uncoerced.

In order to transmit a bit b to R, S first chooses n bits $b_1...b_n$ such that $\oplus_i b_i = b$. Next, S transmits b_i to each intermediary I_i, using a sender-deniable scheme. Next, each I_i transmits b_i to R using a receiver-deniable scheme. Finally R computes $\oplus_i b_i = b$.

When an intermediary I_i is coerced, it reveals the true value of b_i. However, as long as one intermediary I_j remains uncoerced, both S and R can convincingly claim, upon coercion, that the value of b_j (and consequently the value of b) is either 0 or 1.

Note that this scheme holds only if the parties can 'coordinate their stories', in the sense of Remark 2 to Definition 6. In particular, the sender and receiver must know, upon coercion, which intermediaries are being coerced.

6 Discussion

This work raises a number of areas for further thought and open problems. Among the most interesting to us is the difference between the notions of deniability and multi-distribution deniability and in general the value and meaning of a guarantee of multi-distribution deniability.

Natural open questions include searching for a public-key sender-and-receiver-deniable encryption scheme with no intermediaries, or proving that none exists; searching for a public-key sender-deniable encryption scheme enjoying the property of deniability (rather than multi-distribution deniability, as in the Multi-Distribution Parity Scheme), or proving that none exists; and finding a shared-key sender-deniable scheme that does not require the sender to plan ahead.

References

[1] D. Beaver and S. Haber, "Cryptographic Protocols Provably secure Against Dynamic Adversaries", *Eurocrypt,* 1992.

[2] Josh Benaloh, "Verifiable Secret Ballot Elections," Ph.D. thesis, Yale University, Technical Report 561, 1987.

[3] Josh Benaloh and Dwight Tunistra, "Receipt-Free Secret-Ballot Elections", *26th STOC,* 1994, pp. 544-552.

[4] Ran Canetti, Uri Feige, Oded Goldreich and Moni Naor, "Adaptively secure computation", *28th STOC,* 1996, to appear.

[5] Feldman, P., *Private Communication*, 1986.

[6] Amir Herzberg, Rump-Session presentation at CRYPTO 1991.

[7] O. Goldreich and L. Levin, "A Hard-Core Predicate to any One-Way Function", *21st STOC,* 1989, pp. 25-32.

[8] O. Goldreich, S. Micali and A. Wigderson, "How to Play any Mental Game", *19th STOC,* pp. 218-229, 1987.

[9] Shafi Goldwasser and Silvio Micali, "Probabilistic encryption", *JCSS,* Vol. 28, No 2, April 1984, pp. 270-299.

[10] Kazue Sako and Joe Kilian, "Receipt-Free Mix-Type Voting Scheme", *Eurocrypt* 1995, pp. 393-403.

[11] "A multi-authority election scheme which is voter-efficient and receipt-free against a dynamic coercer", submitted to these proceedings.

A Definitions 5 and 6

Definition 5 *A protocol π with sender S and receiver R, and with security parameter n is a $\delta(n)$-*receiver-deniable encryption *protocol if it satisfies Definition 2, with the exception that the Deniability property is replaced as follows.*

Deniability: *There exists an efficient 'faking' algorithm ϕ having the following property with respect to any $m_1, m_2 \in M$. Let r_S, r_R be uniformly chosen random inputs of S and R, respectively, let $c = \text{COM}_\pi(m_1, r_S, r_R)$, and let $\tilde{r}_R = \phi(m_1, r_R, c, m_2)$. Then, the random variables*

$$(m_2, \tilde{r}_R, c) \quad \text{and} \quad (m_2, r'_R, \text{COM}_\pi(m_2, r'_S, r'_R))$$

are $\delta(n)$-close, where r'_S, r'_R are independent, uniformly chosen random inputs of S and R, respectively.

Definition 6 *A protocol π with sender S and receiver R, and with security parameter n is a $\delta(n)$-*sender-and-receiver-deniable encryption *protocol if it satisfies Definition 2, with the exception that the Deniability property is replaced as follows.*

Deniability: *There exist two efficient 'faking' algorithms ϕ_S and ϕ_R having the following property with respect to any $m_1, m_2 \in M$. Let r_S, r_R be uniformly chosen random inputs of S and R, respectively, let $c = \text{COM}_\pi(m_1, r_S, r_R)$, let $\tilde{r}_S = \phi_S(m_1, r_S, c, m_2)$, and let $\tilde{r}_R = \phi_R(m_1, r_R, c, m_2)$. Then, the random variables*

$$(m_2, \tilde{r}_S, \tilde{r}_R, c) \quad \text{and} \quad (m_2, r'_S, r'_R, \text{COM}_\pi(m_2, r'_S, r'_R))$$

are $\delta(n)$-close, where r'_S, r'_R are independent, uniformly chosen random inputs of S and R, respectively.

Remarks:

1. Note that the requirement for a sender-and-receiver-deniable encryption protocol is stronger than requiring that the protocol be both sender-deniable and receiver-deniable.
2. Definition 6 does not address the following issue. In order for the sender and the receiver to convince an adversary that coerces both, the parties need to have 'consistent stories'. That is, they both should claim the same (fake or true) value for the cleartext. We believe that this 'coordination' issue should be treated separately. (We stress that this issue does *not* appear when only the sender or only the receiver are coerced.)

B Definition 7

For this definition we let an extra binary parameter, Preserve, control whether the sender chooses from the T or the C distributions. (See discussion in section 3.) The definition of COM_π is modified accordingly to take an additional parameter. We use the term "multi-distribution deniable," defined below, to describe that we are purchasing deniability at the cost of adding alternative distributions from which the sender can choose.

Definition 7 *A protocol π with sender S and receiver R, binary* Preserve *parameter P, and security parameter n, is a* $\delta(n)$-multi-distribution-sender-deniable encryption *protocol if:*

Correctness: *The probability that R's output is different than S's message input is negligible (as a function of n), independent of P.*

Security: *For any $m_1, m_2 \in M$ we have $\text{COM}_\pi(P, m_1) \overset{c}{\approx} \text{COM}_\pi(P, m_2)$ (encryptions of m_1 and m_2 are indistinguishable independent of P).*

Multi-Distribution Deniability: *There exists an efficient 'faking' algorithm ϕ having the following property with respect to any $m_1, m_2 \in M$. Let r_S, r_R be uniformly chosen random inputs of S and R, respectively, let $c = \text{COM}_\pi(1, m_1, r_S, r_R)$, and let $\tilde{r}_S^{(i)} = \phi(i, m_1, r_S, c, m_2)$, for $i \in \{1, 2\}$. The new parameter i tells the faking algorithm which message the sender wants to claim to have sent. Then, the random variables*

$$(m_i, \tilde{r}_S^{(i)}, c) \quad \text{and} \quad (m_i, r'_S, \text{COM}_\pi(0, m_i, r'_S, r'_R)) \tag{3}$$

are $\delta(n)$-close, where r'_S, r'_R are independent, uniformly chosen random inputs of S and R, respectively.

The left-hand side of Equation 3 describes the view of the adversary when the sender, having preserved the ability to open dishonestly when sending m_1, opens with value m_i (which might or might not equal m_1). The right-hand side of Equation 3 describes the adversary's view when the sender, not having preserved the ability to open dishonestly, opens an encryption of m_i. The definition requires that the adversary cannot distinguish between the two cases with probability exceeding $\delta(n)$.

C A scheme based on the hardness of Factoring

We describe a shared-key scheme where, on top of the secret key shared by the sender and the receiver, the receiver has trapdoor information which remains unknown to the coercer. Thus this scheme is not resilient to coercing both parties simultaneously.

The shared key is chosen as follows: The receiver R picks at random a pair of large and distinct primes p and q and computes $N = pq$. We will make the *quadratic Residuocity assumption* which states that for a random element in Z_N with Jacobi symbol $+1$ it is hard to determine whether it is a square or a non-square if factorization of N is not known.

R picks two numbers y_b and y_{1-b}, such that y_b is a random quadratic residue (QR) mod N with Jacobi symbol $+1$, and y_{1-b} is a random quadratic non-residue (QNR) mod N with Jacobi symbol $+1$. R sends $\{y_0, y_1\}$ to the sender S in the clear. The shared key is one bit b, telling S which out of $\{y_0, y_1\}$ is a QR.

We let

$$\vec{m}_1 = m_1^{(1)}, m_1^{(2)}, ...$$

denote the information to be encrypted, where each each $m^{(j)}$ is a one bit message. Also, let

$$\vec{m}_2 = m_2^{(1)}, m_2^{(2)}, ...$$

denote the the sequence of fake messages. The protocol for the sender A, to send the ith bit as as follows:

- If $m_1^{(j)} = 1$ and $m_2^{(j)} = 0$, then
  S chooses a random r in Z_N and sends $r^2 * y_b$ to R.
- If $m_1^{(j)} = m_2^{(j)} = 1$, then
  S chooses a random r in Z_N and sends r^2 to R.
- If $m_1^{(j)} = 0$ and $m_2^{(j)} = 1$, then
  S chooses a random r in Z_N and sends $r^2 * y_{1-b}$ to R
- If $m_1^{(j)} = m_2^{(j)} = 0$, then
  S chooses a random r in Z_N and sends $r^2 * y_0 * y_1$ to R.

If S wants to prove that the transmitted string was $\vec{m}_1$ then he will (correctly) claim that y_b is QR and y_{1-b} is QNR. If S wants to claim that the transmitted string was $\vec{m}_2$ then he will claim that y_{1-b} is QR and y_b is QNR. Notice that S can not compute the value of b by itself, since S does not know factorization of N – the scheme is based on the fact that R correctly tells S what b is, but without a proof.

It is easy to verify that the receiver always deciphers the right bit (i.e., $m_1^{(j)}$). Also, if y_b is a QR and y_{1-b} is a QNR then the transmitted string is $\vec{m}_1$. If y_b is a QNR and y_{1-b} is a QR then the transmitted string is $\vec{m}_2$.

Notice that the above scheme is 2-plan-ahead (i.e. only two possible plan-ahead strings). Next, we show how to convert *any* 2-plan-ahead scheme into k-plan-ahead scheme. The idea is to build a binary tree of 2-plan-ahead schemes. For example, if we want a 4-plan-ahead scheme, we can use 3 2-plan-ahead schemes. The root scheme will tell whether to go to the left branch or the right branch 2-plan-ahead sub-schemes, etc.

D An Alternative Non-Committing Encryption Scheme

Recently, Canetti, Feige, Goldreich, and Naor constructed general multiparty protocols secure against an adaptive adversary that corrupts parties based on past information [4]. At the heart of this construction is another new type of encryption protocol, called non-committing encryptions.

As in the case of deniable encryption, non-committing encryption is concerned with foiling an adversary that registers the ciphertext and later asks the sender and receiver to reveal their secret keys and random choices. Non-committing encryption offers a considerably weaker solution to this problem than deniable encryption, (although non-committing encryption suffice for proving adaptive security.) Very roughly, a non-committing encryption scheme consists of the usual key-generation, encryption, and decryption algorithms that ensure correctness and semantic security, with the following extra property: There exists a simulator that simulates the process of encryption and decryption in a deniable. That is, the simulator first generates dummy ciphertexts that are indistinguishable, in the eyes of the adversary, from authentic ciphertexts. Next, when the adversary asks to see the internal data of the sender and the receiver, the simulator is able to choose an arbitrary cleartext and then present the adversary with data that is consistent the chosen cleartext.

Note that in a non-committing scheme only the simulator can generate 'dummy ciphertexts' that can be 'opened' as encryptions of more than one value. The sender and receiver are unable to do so. Furthermore, the dummy ciphertexts of a non-committing scheme need not have meaningful decryption, whereas in deniable encryption each ciphertext has unique, meaningful decryption and at the same time can be opened in several ways for a coercer. In particular the non-committing scheme described in [4] is not deniable.

In principle, any public-key $\delta(n)$-sender-and-receiver-deniable encryption scheme according to Definition 6 is non-committing if $\delta(n)$ is negligible. However, we do not know if such a deniable scheme exists. (In this paper we only describe $\frac{1}{n}$-sender-and-receiver-deniable schemes.)

The basic scheme described in Section 3, after being transformed into a sender-and-receiver-deniable scheme using the technique of Section 5, is a valid non-committing encryption scheme. In particular, this scheme is much simpler to present and prove than the [4] scheme. Call this scheme the transformed basic scheme.

To show that the transformed basic scheme is non-committing, we have to describe the simulator. We use the fact that, in the basic scheme, cheating is undetectable in one direction (except for negligible probability). That is, the sender can encrypt 1 and later claim that the encrypted bit was 0. The dummy ciphertexts generated by the simulator will all be legal encryptions of 1. When the sender and receiver are asked to demonstrate their internal data they can show that the encrypted bit was either 0 or 1, as desired.