Deniable Encryption*

Ran Canetti † Cynthia Dwork ‡ Moni Naor § Rafail Ostrovsky ¶

June 2, 1997

Abstract

Consider a situation in which the transmission of encrypted messages is intercepted by an adversary who can later ask the sender to reveal the random choices (and also the secret key, if one exists) used in generating the ciphertext, thereby exposing the cleartext. An encryption scheme is deniable if the sender can generate 'fake random choices' that will make the ciphertext 'look like' an encryption of a different cleartext, thus keeping the real cleartext private. Analogous requirements can be formulated with respect to attacking the receiver and with respect to attacking both parties.

In this paper we introduce deniable encryption and propose constructions of schemes with polynomial deniability. In addition to being interesting by itself, and having several applications, deniable encryption provides a simplified and elegant construction of *adaptively secure* multiparty computation.

1 Introduction

The traditional goal of encryption is to maintain the privacy of communicated data against passive eavesdroppers. That is, assume that Alice wants to communicate private information to Bob over a channel where Eve can eavesdrop. Alice obtains Bob's (public) encryption key of an asymmetric encryption scheme and uses it, together with local randomness, to encrypt her messages. Now only Bob, who possesses the decryption key, should be able to decrypt. Semantic security [15] captures the security requirements that this setting imposes on the encryption function. Basically, semantic security means that Eve learns nothing from the ciphertexts she hears: whatever she can compute having heard the ciphertexts she can also compute from scratch. It follows that Alice must use local randomness in order to achieve semantic security.

While (passive) semantic security appropriately captures the security needed against passive eavesdroppers, there are settings in which it falls short of providing the desired degree of protection. Such settings include protection against chosen ciphertext attacks (e.g., [17, 18]), non-malleable encryption [8], and protection against adaptive adversaries [7].

We investigate the additional properties required to protect the privacy of transmitted data in yet another hostile setting. Assume that the adversary Eve now has the power to approach Alice (or Bob, or both) *after* the ciphertext was transmitted, and demand to see all the private information: the cleartext, the random bits used for encryption and any private keys Alice (or Bob) have. Once Alice hands over this information, Eve can verify that the cleartext and randomness provided by Alice indeed match the transmitted ciphertext. Can the privacy of the communicated data be still somehow maintained, in face of such an attack?

*Research on this paper was supported by BSF Grant 32-00032. A slightly shorter version of this work appears in the proceedings of Crypto'97.

†IBM T.J. Watson Research Center. *email:* canetti@watson.ibm.com

‡IBM Almaden Research Center. *email:* dwork@almaden.ibm.com

§Dept. of Computer Science, the Weizmann Institute. *email:* naor@wisdom.weizmann.ac.il.

¶Bellcore. *email:* rafail@bellcore.com

We first concentrate on the case where Eve attacks only Alice in the above way. Certainly, if Alice must hand Eve the *real* cleartext and random bits then no protection is possible. Also if Eve approaches Alice *before* the transmission and requires Alice to send specific messages there is no way to hide information. However, in case Eve has no direct physical access to Alice's memory, and Alice is allowed to hand Eve *fake* cleartext and random bits, is it possible for Alice to maintain the privacy of the transmitted data? That is, we ask the following question. Assume Alice sent a ciphertext $c = E(m_1, r)$, where m_1 is some message, E is the public encryption algorithm and r is Alice's local random input. Can Alice now come up with a fake random input r' that will make c 'look like' an encryption of a different message m_2? We call encryption schemes that have this property deniable.

The following valid question may arise at this point: if Eve has no physical access to Alice's memory, then why should Alice present Eve with any data at all? That is, why not have Alice tell Eve: 'Sorry, I erased the cleartext and the random bits used'. Indeed, if Eve will be willing to accept such an answer, then deniable encryption is not needed. But there may well exist cases where being able to provide Eve with convincing fake randomness will be valuable to Alice. (Presenting convincing data is almost always more credible than saying 'I erased', or 'I forgot'.) In fact, there may be cases where Alice is required to record all her history including the randomness used, and can be punished/prosecuted if she claims to have destroyed the "evidence", *i.e.* any part of her history. Furthermore, the mere fact that Alice is able to 'open' any ciphertext in many ways makes it impossible for Alice to convince Eve in the authenticity of *any* opening. This holds even if Alice *wishes* to present Eve with the real data. In this sense, the privacy of Alice's data is protected even from the *future behavior of Alice herself.*

Standard encryption schemes do not guarantee deniability. Indeed, typically there do not *exist* two different messages that may result in the same ciphertext (with *any* random input). In fact, encryption is often conceived of as a *committing* process, in the sense that the ciphertext may serve as a commitment to the cleartext. (This is a common use for encryption schemes, e.g. in [13, 14].) Deniable encryption radically diverges from this concept.

Deniable encryption may seem impossible at first glance: consider a ciphertext c sent from Alice to Bob. If, using two different random choices, Alice 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 Bob correctly decide, from c alone, whether Alice meant to send m_1 or m_2? A more careful inspection shows that such schemes can indeed be constructed, based on trapdoor information unavailable to Eve.

Deniable encryption has applications to the prevention of vote-buying in electronic voting schemes [4, 10, 11, 19], storing encrypted data in a deniable way, and uncoercible multiparty computation [5]; it also yields an alternative solution to the adaptive security problem [7]. We elaborate on these applications in Section 1.1.

We classify deniable encryption schemes according to which parties may be coerced: a sender-deniable scheme is resilient against coercing (*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 schemes, in which the sender and receiver initially share some information, and public-key deniable encryption schemes, in which no prior communication is assumed. Another issue is the time at which the coerced party must decide on the fake message: at time of attack (preferable) or at time of encryption.

Let us informally sketch the requirements for a one-round public-key, sender-deniable, bit-by-bit encryption scheme (Section 2 contains a more general definition). Let E_k be the sender's encryption algorithm with public key k. First, a deniable encryption scheme should be semantically secure in the sense of [15]. In addition we require that the sender have a (publicly known) faking algorithm. Given a bit b, a random input r, and the resulting ciphertext $c = E_k(b, r)$, the faking algorithm generates a fake random input $\rho = \phi(b, r, c)$ that 'makes c look like an encryption of $\bar{b}$'. That is, given b, ρ, c, the adversary should be

unable to distinguish between the following cases:
(a) ρ is uniformly chosen and $c = E_k(b, \rho)$
(b) c was generated as $c = E_k(\bar{b}, r)$ where r is independently and uniformly chosen, and $\rho = \phi(\bar{b}, r, c)$.
We say that a scheme is δ-deniable if the adversary can distinguish between cases (a) and (b) with probability at most δ.

We construct a sender-deniable public-key encryption scheme based on any trapdoor permutation (Section 3). However, our scheme falls short of achieving the desired level of deniability. That is, while we can construct a δ-deniable scheme for arbitrarily small δ, the length of the ciphertext is *linear* in $1/\delta$. Consequently, if we want δ to be negligible, we end up with ciphertexts of super-polynomial length. (The semantic security of our scheme against passive eavesdroppers holds in the usual sense.) We present evidence that constructing substantially better one-round schemes requires a different approach (Section 4).

We also consider a more flexible notion of deniability than the one sketched above. An encryption scheme for encrypting a single bit can be generally viewed as defining two distributions on ciphertexts: a distribution T_0 of encryptions of 0, and a distribution T_1 of encryptions of 1. Here, in contrast, 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 when attacked. 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 when attacked. (This choice can be made at time of attack.) In particular, if the sender encrypts according to distribution C_b then, when attacked, the sender can appear to have chosen either from T_0 or T_1. This alternative notion allows us to construct efficient deniable schemes with negligible δ.

Section 6 shows, via simple constructions, how to transform any sender-deniable encryption scheme into a receiver-deniable scheme, and vice-versa. 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 unattacked. In Section 5 we review some shared-key deniable schemes.

1.1 Applications and related work

A natural application of deniable encryption is to prevent coercion in electronic secret voting schemes [10]: a coercer may offer bribe in exchange for proof of a person's vote, after hearing the corresponding ciphertext. The coercion problem in the context of voting has been studied in the past [4, 19, 11]. However, these previous works assume that, for a crucial part of the conversation, the communicating parties share a physically secure channel; thus, the coercer hears no ciphertext and the 'deniability problem' disappears.[1] Deniable encryptions may be incorporated in these works to replace these physical security assumptions. (One still has to make sure, as before, that the voters are not coerced *prior* to the elections.)

Based on the public-key, sender-deniable construction presented here, [5] describe a general multiparty protocol permitting a set of parties to compute a common function of their inputs while keeping their internal data private even in the presence of a coercer.

Finally, our work on deniable encryption provides a conceptually simple and elegant alternative solution to the problem of general secure multiparty computation in the presence of an *adaptive* adversary – one that chooses whom to corrupt *during* the course of the computation, based on the information seen as the execution unfolds. Protocols for securely computing any function in a multiparty scenario in the presence of a non-adaptive adversary were shown in [14]. Almost a decade passed before the restriction to non-adaptive

[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.

adversaries was lifted [7].[2] These protocols are based on another type of encryption protocol, called non-committing encryption. Non-committing encryptions have the same flavor 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. For example, in non-committing encryptions the parties *using* the scheme are, in general, *not* able to generate ciphertexts that can be opened both ways; such ciphertexts can only be generated by a *simulator* (which is an artifact of the [7] model). In contrast, 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 an adversary (thus, the non-committing encryption scheme in [7] is not deniable). The key insight is that any deniable encryption scheme resilient against attacking both the sender and the receiver is non-committing. Indeed, applying the transformation of Section 6 to the basic scheme described in Section 3 yields a complete solution to the adaptive security problem.

2 Definitions

Let us first recall the definition of computational distance of distributions. Here and in the sequel a function $\delta : \mathbf{N} \to [0,1]$ is negligible if it approaches zero faster than any polynomial (when its argument approaches infinity).

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

If $\delta(n)$ is negligible then we say that $\mathcal{A}$ and $\mathcal{B}$ are computationally indistinguishable *and write $\mathcal{A} \stackrel{c}{\approx} \mathcal{B}$.*

2.1 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 public encryption key to the sender, who responds with the encrypted message.) Intuitively, we desire: (1) the receiver should be able to decrypt the correct value (except, perhaps, with negligible probability of error); (2) the protocol should be semantically secure against eavesdroppers; and (3) the sender should have a faking algorithm ϕ such that, given (m_1, r_S, c, m_2) (where m_1 is the transmitted message, r_S is the sender's random input, c is a transcript of the conversation between S and R for transmitting m_1, and m_2 is the required fake message), ϕ generates a fake random input for the sender, that makes c look like a conversation for transmitting m_2.

More precisely, 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 $\mathrm{COM}_\pi(m, r_S, r_R)$ 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 $\mathrm{COM}_\pi(m)$ denote the random variable describing $\mathrm{COM}_\pi(m, r_S, r_R)$ 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 $\mathrm{COM}_\pi(m_1) \stackrel{c}{\approx} \mathrm{COM}_\pi(m_2)$.*

[2] [9, 3] obtain solutions for this problem under the assumption that the parties are trusted to keep *erasing* past information. Such solutions are unsatisfactory in a setting where parties aren't trusted since erasing cannot be externally verified. Furthermore, the physical design of computer systems makes erasing information difficult and unreliable [16].

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 and independently 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*

$$(m_2, \tilde{r}_S, \text{COM}_\pi(m_1, r_S, r_R)) \quad \text{and} \quad (m_2, r_S, \text{COM}_\pi(m_2, r_S, r_R)) \tag{1}$$

are $\delta(n)$*-close.*

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 c was generated while transmitting m_1, and the sender 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 $\delta(n)$.

REMARKS: 1. When the domain of messages is $M = \{0, 1\}$ the definition may be simplified. In the sequel we concentrate on such schemes, encrypting one bit at a time.

2. Definition 2 requires the parties to choose new public keys for each message transmitted. The definition can be modified in a natural way to capture schemes where a 'long-lived' public key is used to encrypt several messages, requiring the sender to be able to 'fake' each message independently of the other messages encrypted with the same public key. The scheme described in the sequel indeed enjoys this additional property.

3. Schemes in which the coerced party chooses the fake message m_2 at time of encryption are called *plan-ahead* deniable encryption schemes. Some modifications of the constructions described below yield plan-ahead deniable encryption schemes with negligible $\delta(n)$.

Next we define a somewhat weaker notion of deniability, called flexible deniability.

Definition 3 *A protocol* π *with sender* S *and receiver* R, *binary* Preserve *parameter* P, *and security parameter* n, *is a* $\delta(n)$-flexible-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 any* $P \in \{T, C\}$ *we have* $\text{COM}_\pi(P, m_1) \stackrel{c}{\approx} \text{COM}_\pi(P, m_2)$ *(encryptions of* m_1 *and* m_2 *are indistinguishable independent of* P*).*

Weak 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* $c = \text{COM}_\pi(C, m_1, r_S, r_R)$, *and let* $\tilde{r}_S = \phi(m_1, r_S, c, m_2)$. *Then, the random variables*

$$(m_2, \tilde{r}_S, c) \quad \text{and} \quad (m_2, r'_S, \text{COM}_\pi(T, m_2, r'_S, r'_R)) \tag{2}$$

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_2 (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_2.

Schemes resilient against attacking the receiver, or simultaneous attack of both the sender and the receiver, are defined analogously. They appear as Definitions 9 and 10 in Appendix B.

2.2 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 adversary with

a fake shared key, on top of presenting fake random inputs. This is captured as follows. The communication between the parties now depends also on a shared key k, and is denoted $\text{COM}_\pi(m, k, r_S, r_R)$ (where m, r_S, r_R are the same as before). Below we define sender-deniability.

Definition 4 *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 $\text{COM}_\pi(m_1, k) \stackrel{c}{\approx} \text{COM}_\pi(m_2, k)$.*

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 $c = \text{COM}_\pi(m_1, k, r_S, r_R)$, and let $(\tilde{k}, \tilde{r}_S) = \phi(m_1, k, r_S, c, m_2)$. Then, the random variables*

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

are $\delta(n)$-close.

Note that Definition 4 also covers the case where the same key is used to encrypt several messages: let m_1 (resp., m_2) in the definition denote the concatenation of all real (resp., fake) messages. In Section 5 we mention some shared-key schemes.

3 Public-key Deniable Encryption

OVERVIEW. We describe two 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 our main scheme. (It can also be used to construct a non-committing encryption scheme, as described in Appendix A.) Our main scheme, called the Parity Scheme, is $\frac{4}{n}$-sender-deniable according to Definition 2. Roughly speaking, this means that the probability of successful attack vanishes *linearly* in the security parameter. (By a simple renaming of parameters this scheme can be regarded as $\frac{1}{n^c}$-sender-deniable for any $c > 0$. Yet, the probability of successful attack vanishes only linearly in the amount of work invested in encryption and decryption.)

The schemes are sender-deniable. Receiver-deniable and Sender-and-receiver-deniable schemes can be constructed from these using the techniques of Section 6. Our schemes encrypt one bit at a time. Here they are described in the standard terms of encryption and decryption algorithms. In terms of Definition 2, the interaction consists of the receiver sending the public encryption key to the sender, who responds with the encrypted message.

THE BASIC APPROACH. Our schemes are based on the following simple idea. Assume that the sender can pick an element in some domain either *randomly,* or according to some *pseudorandom* distribution. Assume further that the receiver, having some secret information, can tell whether the element was chosen randomly or pseudorandomly; other parties cannot tell the difference. Then, the sender can proceed as follows: to encrypt a 1 (resp., 0) send a pseudorandom (resp., random) element. The receiver will be able to decrypt correctly; but if a *pseudorandom* element e was transmitted, then when attacked the sender can claim that e was *randomly* chosen — and the adversary will not be able to tell the difference.

Here the sender could fake its message only in one direction (from 1 to 0). Using simple tricks one can come up with schemes that allow faking in both directions. We now describe the schemes in detail.

TRANSLUCENT SETS. Our schemes are based on a construct that can be informally described as follows. (Formal definitions can be extracted from this description.) We assume that there exists a family $\{\mathcal{S}_t\}_{t \in \mathbf{N}}$ of sets, where $\mathcal{S}_t \subset \{0,1\}^t$, together with secret 'trapdoor information' d_t, such that:

1. $\mathcal{S}_t$ is small: $|\mathcal{S}_t| \leq 2^{t-k}$ for some sufficiently large $k(t)$.
2. It is easy to generate random elements $x \in \mathcal{S}_t$, even without the secret d_t.
3. Given $x \in \{0,1\}^t$ and d_t it is easy to decide whether $x \in \mathcal{S}_t$.
4. Without d_t, values chosen uniformly from $\mathcal{S}_t$ are indistinguishable from values chosen uniformly from $\{0,1\}^t$.

We first present two simple constructions of translucent sets. Both use a trapdoor permutation $f : \{0,1\}^s \to \{0,1\}^s$, and its hard-core predicate $B : \{0,1\}^s \to \{0,1\}$ (say, use the Goldreich-Levin predicate [12]).

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 $\mathcal{S}_t = \{x_1...x_k \in \{0,1\}^{sk} \mid \forall i = 1..k,\ B(f^{-1}(x_i)) = 0\}$. Here $|\mathcal{S}_t| \approx 2^{(s-1)k} = 2^{t-k}$.

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 $i \geq 1$ each $b_i \in \{0,1\}$. Then let $\mathcal{S}_t = \{x_0, b_1...b_k \in \{0,1\}^{s+k} \mid \forall i = 1..k,\ B(f^{-i}(x_0)) = b_i\}$. Here $|\mathcal{S}_t| = 2^s = 2^{t-k}$.

It is easy to verify that both constructions satisfy requirements 1-4. Construction II is more efficient in that, given a trapdoor permutation on $\{0,1\}^s$, the length of x is only $t = s + k$ instead of $t = sk$.

A third construction relies on the latticed-based public-key cryptosystem described in [2]. Roughly speaking, the secret information is an n-dimensional vector u of length at most 1. Let $\mathcal{K}$ denote the cube $2^{n \log n} U^{(n)}$, where $U^{(n)}$ is the n-dimensional unit cube. The vector u induces a collection of $(n-1)$-dimensional hyperplanes as follows: for integer i the *ith* hyperplane is the set of all vectors v whose inner product with u is equal to i. Let X be the intersection of the hyperplanes with $\mathcal{K}$. The public key consists of a collection of $m = n^c$ points $v_1, \ldots, v_m$, each of which is a small perturbation of a randomly chosen point in X. The encryption procedure makes use of a certain parallelepiped $\mathcal{P}$, computable from the public key. An encryption of zero is a point chosen uniformly at random from $\mathcal{K} \cap 2^{-n}\mathbb{Z}^n$. An encryption of one is $\sum_{i=1}^{m} \delta_i v_i \mod \mathcal{P}$, where each $\delta_i \in_R \{0,1\}$. Thus, encryptions of one are close to hyperplanes in X, while encryptions of zero, typically, are not. Decryption of the ciphertext is performed by computing the distance of the ciphertext from the nearest hyperplane in X: if the distance is sufficiently small the ciphertext is decrypted as one (there is a polynomial probability of error). This construction yields a translucent set in which t is the length of a ciphertext ($t \approx n^2$), and, once the public key has been chosen, $\mathcal{S}_t$ is the set of encryptions of one, and $\mathcal{R}_t$ is the set of encryptions of zero.

THE BASIC SCHEME. The public encryption key is a method for generating uniformly at random a member of a translucent set $\mathcal{S}_t \subset \{0,1\}^t$. The private decryption key is the corresponding secret d.

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

Decryption: *If the ciphertext x is in $\mathcal{S}_t$ 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 $\mathcal{S}_t$, 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 will be infeasible since the ciphertext x is in $\mathcal{S}_t$ only with negligible probability 2^{-k}.* Analysis: Correctness: An encryption of 1 is always decrypted correctly. An encryption of 0 may be decrypted as 1 with probability 2^{-k}. Standard security against eavesdroppers is straightforward. Deniability: the faking algorithm ϕ and its validity are described above. 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. Let $\mathcal{S}_t \subset \{0,1\}^t$ be a translucent set. We call elements drawn uniformly from $\mathcal{S}$ (resp., from $\{0,1\}^t$) $\mathcal{S}$-elements (resp., $\mathcal{R}$-elements).

Encryption: *To encrypt 0 (resp., 1), choose a random even (resp., odd) number $i \in 0, \ldots, n$. Construct a ciphertext consisting of i $\mathcal{S}$-elements followed by $n - i$ $\mathcal{R}$-elements.*

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

Opening an encryption honestly: *Reveal the real random choices used in generating the ciphertext.*
Opening an encryption dishonestly: *Let i be the number chosen by the sender. The sender claims that she has chosen $i-1$ rather than i. (Consequently, the parity of i flips.) For this, she claims that the ith element in the ciphertext is an $\mathcal{R}$-element (whereas it was chosen as an $\mathcal{S}$-element). If there are no $\mathcal{S}$-elements (i.e., $i=0$) then cheating fails.*

Theorem 5 *Assume trapdoor permutations exist. Then the Parity Scheme is a $4/n$-sender-deniable encryption scheme.*

Proof (Sketch): The probability of erroneous decryption is at most $n2^{-k}$. Security of the Parity Scheme against eavesdroppers that see only the ciphertext is straightforward. We show deniability. Assume that n is odd, and let c be an encryption of 1. Let i be the number chosen for generating c. 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 c is an encryption of 0. Thus i is chosen uniformly from $0, 2, ..., n-1$. Now, $i-1$ is distributed uniformly in $-1, 1, 3, ..., n-2$ (where -1 is interpreted as "cheating impossible"). It is easy to verify that the statistical distance between the distribution of i in the case of an honest opening (i.e., uniform on $1, 3, ..., n$) and the distribution of i in the case of fake opening (i.e., uniform on $-1, 1, 3, ..., n-2$) is $4/n$. It follows that, as long as $\mathcal{S}$-elements cannot be distinguished from $\mathcal{R}$-elements, cheating is detectable with probability at most $4/n$. □

The Parity Scheme can be modified to let the sender first choose a vector v uniformly out of all vectors in $\{0,1\}^n$ with the parity of the bit to be encrypted. Next the ciphertext is constructed by replacing each 1 entry in v with an $\mathcal{S}$-element, and replacing each 0 with an $\mathcal{R}$-element. Here the probability of $i=0$ (i.e., the probability of the case where cheating is impossible) is negligible. Now, however, the statistical distance between i's distribution in honest and fake openings grows to $\Omega(\sqrt{\frac{1}{n}})$. A 'hybrid' scheme, omitted from this abstract, achieves both negligible probability of impossible cheating and probability $O(1/n)$ of detection.

The *unique shortest vector* problem for lattices is: "Find the shortest nonzero vector in an n dimensional lattice L where the shortest vector v is unique in the sense that any other vector whose length is at most $n^c\|v\|$ is parallel to v." The unique shortest vector problem is one of the three famous problems listed in [1]. There, a random method is given to generate hard instances of a particular lattice problem so that if it has a polynomial time solution then all of the three worst-case problems (including the unique-shortest vector problem) has a solution. The cryptosystem in [2] outlined above is secure provided the unique shortest vector problem is hard in the worst case. From this and the proof of Theorem 5 we have:

Theorem 6 *Assume that the unique shortest vector problem is hard in the worst case. Then the Parity Scheme is a $4/n$-sender-deniable encryption scheme.*

A FLEXIBLY DENIABLE 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 against eavesdroppers seeing only the ciphertext is straightforward. The probability of

erroneous decryption is at most 2^{-k}. Weak deniability with negligible $\delta(n)$ is immediate by inspection, assuming polynomial time indistinguishability of $\mathcal{S}$ and $\mathcal{R}$.

4 Efficiency Vs. Deniability

In this section we describe an attack suggesting that no one-round scheme of the type presented above can enjoy negligible $\delta(n)$. The attack works against all schemes that we describe as separable (the reason for the name will become clear shortly). Roughly, in a separable scheme the decryption key is the trapdoor of some translucent set $\mathcal{S} \subset \{0,1\}^t$; a ciphertext consists of a sequence of elements $y_1....y_m$ in $\{0,1\}^t$. The sender chooses some of the y_i's at random, and the rest at random from $\mathcal{S}$. The encrypted bit is encoded in the number and placement of the y_i's that are in the translucent set $\mathcal{S}$. To fake the value of the cleartext the sender claims that one (or more) of the y_i's was randomly chosen, whereas this y_i was chosen from $\mathcal{S}$.

For any separable scheme, and for each value $b \in \{0,1\}$, one can compute the expected number of y_i's in $\mathcal{S}$ in an encryption of b. Denote this number by E_b. Now, since the faking algorithm always *decreases* the number of y_i's for which the sender claims to know the preimage, the adversary decides that the sender is lying if the sender claims to have sent b but the number of y_i's which the sender claims to have chosen from $\mathcal{S}$ is less than E_b. It is shown below that this strategy succeeds with probability at least $\Omega(\frac{1}{m})$.

A more precise (and somewhat more general) description follows.

Definition 7 *A $\frac{1}{k}$-sender-deniable public key encryption scheme π is m-*separable *if there exists an efficient, deterministic* classification algorithm *C that, on any input ρ (interpreted as a claimed random input of the sender), outputs a number $C(\rho) \in 1,\ldots,m$. Furthermore:*

1. *For a value ρ (interpreted as a random input for the sender), let $\rho^{(b)}$ be the random variable describing $\phi(b,\rho,c)$, where ϕ is the sender's faking algorithm, $b \in \{0,1\}$, r_R is the receiver's random input, and and $c = \text{COM}_\pi(b,\rho,r_R)$ is the resulting communication. Let $EC^{(b)}(\rho)$ denote the expected value (over the choices of r_R) of $C(\rho^{(b)})$.*

   Then for any value ρ such that $C(\rho) > 1$, either $EC^{(0)}(\rho) \leq C(\rho) - 1$ or $EC^{(1)}(\rho) \leq C(\rho) - 1$.

2. *If the sender's random input ρ satisfies $C(\rho) = 1$ then the faking algorithm fails, i.e. it outputs a special symbol denoting that no suitable fake random input was found.*

Claim 8 *For any m-*separable, *$\frac{1}{k}$-*sender-deniable public key encryption scheme we have $2m \geq k$.

REMARKS:

- Using the terminology of the above informal description of separable schemes, the coercer will use the classification algorithm that outputs the number of y_i's which the sender claims to have chosen as $\mathcal{S}$-elements. It follows that any such scheme with only m y_i's is m-separable.
- In all the m-separable schemes that we know of, the length of the ciphertext grows linearly with m. This seems to be inherent in our approach for constructing deniable schemes.

Proof: Consider an m-separable deniable scheme π with faking algorithm ϕ. We show an algorithm A that for some $b \in \{0,1\}$ distinguishes between

$$(\bar{b}, r_S^{(b)}, \text{COM}_\pi(b, r_S, r_R)) \quad \text{and} \quad (\bar{b}, r_S, \text{COM}_\pi(\bar{b}, r_S, r_R)) \tag{3}$$

with probability $\frac{1}{2m}$, where r_S, r_R are random inputs for the sender and the receiver respectively, and $r_S^{(b)} = \phi(b, r_S, \text{COM}_\pi(b, r_S, r_R))$.

Let C be the classification algorithm. For $b \in \{0,1\}$, let DC denote the distribution of $C(r_S)$ where r_S is chosen at random from the domain of random inputs of the sender, and let $DC^{(b)}$ denote the distribution of $C(r_S^{(b)})$ when r_R is chosen at random. Let $EC, EC^{(b)}$ denote the expected values of $DC, DC^{(b)}$, respectively. It follows from Definition 7 that either $EC - EC^{(0)} \geq \frac{1}{2}$ or $EC - EC^{(1)} \geq \frac{1}{2}$.

Let $\mathrm{SD}(D_1, D_2)$ denote the statistical distance between two distributions D_1, D_2 over $1, \ldots, m$,[3] and let E_1, E_2 denote the corresponding expected values. It can be verified that $|E_1 - E_2| \leq m \cdot \mathrm{SD}(D_1, D_2)$. In our case this implies that either $\mathrm{SD}(DC, DC^{(0)}) > \frac{1}{2m}$ or $\mathrm{SD}(DC, DC^{(1)}) > \frac{1}{2m}$.

The distinguisher A is now straightforward. Assume that $\mathrm{SD}(DC, DC^{(0)}) > \frac{1}{2m}$. Then A distinguishes between $(0, r_S^{(1)}, \mathrm{COM}_\pi(1, r_S, r_R))$ and
$(0, r_S, \mathrm{COM}_\pi(0, r_S, r_R))$ as follows. Let $Z \subset 1...m$ be the set of numbers that have higher probability under $DC^{(0)}$ than under DC. Then, given a triplet $(0, \rho, c)$, first check that the ciphertext c is consistent with 0 and ρ. Next, if $C(\rho) = 1$ then by Definition 7 above A can distinguish between the two distributions of (3). Otherwise, say that the triplet describes an honest encryption of 0 iff $C(\rho) \in Z$. By definition of statistical distance, A distinguishes correctly with probability at least $\frac{1}{2m}$. (Since Z is a subset of $1...m$, it can be found by sampling.) □

5 Shared-key deniable encryption

In this section we briefly remark on some shared-key deniable schemes. Clearly, a public-key deniable scheme is also deniable in the shared-key setting. Thus the public key constructions described in previous sections apply here as well. Yet better shared-key deniable schemes may be easier to find than public-key ones.

A one-time-pad is a 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 4. Here the message m' can be chosen as late as *at time of attack.* However, using a one-time pad is generally impractical, since the key has to be as long as all the communication between the parties.

Recall that in plan-ahead sender-deniability the sender chooses the fake message(s) *at time of encryption.* Although restrictive, this notion can be useful, *e.g.* for maintaining 'deniable records' of data, 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.) The records are deniable if, when coerced to reveal the cleartext and the secret key used for encryption and decryption, the owner of the record can instead "reveal" a variety of fake cleartexts of her choice.

Plan-ahead shared-key deniability is trivially solved: given l alternative messages to encrypt, use l different keys, and construct the ciphertext as the concatenation of the encryptions of all messages, where the ith message is encrypted using the ith key. When coerced, the party simply claims that the key he used is the one that corresponds to the message he wishes to open.

One problem with this simple scheme is that the size of the ciphertext grows linearly in the number of different messages to be encrypted. It is possible (details omitted for lack of space) to transform any given shared key encryption to a deniable one, without any increase in the message length, and with a key of length $1 - \frac{1}{l}$ times the length of the message. The shared-key deniable schemes can also be used to make public-key deniable schemes more efficient by way of first sending (using public-key deniable scheme) a deniable shared key and then switching to a private-key (deniable) scheme. We omit the details from this

[3] That is, $\mathrm{SD}(D_1, D_2) = \sum_{i \in 1,\ldots,m} |\mathrm{Prob}_{D_1}(i) = \mathrm{Prob}_{D_2}(i)|$.

abstract.

6 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, with respect to scheme $\mathcal{A}$, R is the sender and S is the receiver.) Next, S sends $b \oplus r$ to R, in the clear.

If scheme $\mathcal{A}$ is sender-deniable then, when attacked, 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.

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

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 attacking the sender, the receiver and some intermediaries, as long as at least *one* intermediary remains unattacked.

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 attacked, it reveals the true value of b_i. However, as long as one intermediary I_j remains unattacked, both S and R can convincingly claim, when attacked, that the value of b_j (and consequently the value of b) is either 0 or 1.

Note that this scheme works only if the parties can 'coordinate their stories', in the sense of Remark 2 to Definition 10. In particular, the sender and receiver must know, when attacked, which intermediaries are being attacked.[4]

References

[1] M. Ajtai, Generating Hard Instances of Lattice Problems, STOC'96

[2] M. Ajtai, C. Dwork, A Public-Key Cryptosystem with Average-Case/Worst-Case Equivalence, *STOC'97*; see also *Electronic Colloquium on Computational Complexity TR96-065*, http://www.eccc.uni-trier.de/eccc-local/Lists/TR-1996.html

[3] D. Beaver and S. Haber, Cryptographic Protocols Provably Secure Against Dynamic Adversaries, *Eurocrypt,* 1992.

[4] J. Benaloh and D. Tunistra, Receipt-Free Secret-Ballot Elections, *26th STOC,* 1994, pp. 544-552.

[4] In general, when both the sender and the receiver are attacked they are faced with a 'coordination problem': to be consistent, they both should claim the same (fake or true) value for the cleartext. We believe that this 'coordination' issue should be treated separately. This problem is extensively treated in [5].)

[5] R. Canetti and R. Gennaro, Incoercible multiparty computation, *FOCS'96*

[6] R. Canetti, C. Dwork, M. Naor and R. Ostrovsky, Deniable Encryption, *Theory of Cryptology Library,* `http://theory.lcs.mit.edu/ tcryptol`, 1996.

[7] R. Canetti, U. Feige, O. Goldreich and M. Naor, Adaptively secure computation, *28th STOC,* 1996.

[8] D. Dolev, C. Dwork and M. Naor, Non-malleable cryptography, *STOC'91*

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

[10] A. Herzberg, Rump-Session presentation at CRYPTO 1991.

[11] R. Gennaro, unpublished manuscript.

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

[13] O. Goldreich, S. Micali and A. Wigderson, Proofs that Yield Nothing but the Validity of the Assertion, and a Methodology of Cryptographic Protocol Design, *27th FOCS,* 174-187, 1986.

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

[15] S. Goldwasser and S. Micali, Probabilistic encryption, *JCSS,* Vol. 28, No 2, April 1984, pp. 270-299.

[16] P. Gutman, Secure Deletion of Data from Magnetic and Solid-State Memory, Sixth USENIX Security Symposium Proceedings, San Jose, California, July 22-25, 1996, pp. 77-89.

[17] M. Naor and M. Yung " Public key cryptosystems provably secure against chosen ciphertext attacks", Proc. 22nd ACM Annual Symposium on the Theory of Computing, 1990, pp. 427–437.

[18] C. Rackoff and D. Simon, Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack, *CRYPTO'91, (LNCS 576),* 1991.

[19] K. Sako and J. Kilian, Receipt-Free Mix-Type Voting Scheme, *Eurocrypt* 1995, pp. 393-403.

A An Alternative Non-Committing Encryption Scheme

Recently, general multiparty protocols secure against an adaptive adversary that corrupts parties based on past information were constructed [7]. At the heart of this construction is another 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 an adversary. In particular the non-committing scheme described in [7] is not deniable.

Any public-key $\delta(n)$-sender-and-receiver-deniable encryption scheme according to Definition 10 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 $\frac{1}{n^c}$-sender-and-receiver-deniable schemes, we're c is a constant.)

It turns out, however, that the basic scheme described in Section 3, after being transformed into a sender-and-receiver-deniable scheme using the technique of Section 6, is a valid non-committing encryption scheme. In particular, this scheme is much simpler to present and prove than the [7] 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, in a simulated execution, 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.

B Definitions 9 and 10

Definition 9 *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.

Definition 10 *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.

Remarks:

1. Note that the requirement from a sender-and-receiver-deniable encryption protocol is stronger than requiring that the protocol be both sender-deniable and receiver-deniable.

2. Definition 10 does not address the following issue. In order for the sender and the receiver to convince an adversary that attacks 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. (Indeed, it is extensively treated in [5].) Note that this issue does *not* appear when only the sender or only the receiver are attacked.