A Method for Obtaining Deniable Public-Key Encryption

Maged Hamada Ibrahim[*]
Department of Electronics, Communications and Computers, Faculty of Engineering, Helwan University
1, Sherif St., Helwan, Cairo, Egypt (Email: mhii72@hotmail.com)

(Received June 6, 2007; revised and accepted Apr. 2, 2008)

Abstract

Deniable encryption is an important notion that allows a user (a sender and/or a receiver) to escape a coercion attempted by a coercive adversary. Such an adversary approaches the coerced user after transmission forcing him to reveal all his random inputs used during encryption or decryption. Since traditional encryption schemes commits the user to his random inputs, the user is forced to reveal the true values of all his random inputs (including the encrypted/decrypted messages and the encryption/decryption keys) which are verifiable by this coercer using the intercepted ciphertext. In this scenario, a coercer may force the user to perform actions against his wish. In this paper we present a scheme for sender-deniable public-key encryption, by which, the sender is able to lie about the encrypted message to a coercer and hence, escape a coercion. While the receiver is able to decrypt for the true message, the sender has the ability to open a fake message of his choice to the coercer which when verified gives the same ciphertext as the true message. Our schemes rely on quadratic residuosity of a two-prime modulus. Deniability improvements to these schemes considering the sender's local randomness are also presented. We also show how to build an efficient deniable public-key encryption from any trapdoor permutation. Compared to the schemes proposed in [5], our schemes require much less bandwidth, provide stronger deniability and no decryption errors.

Keywords: Blum primes, coercive adversary, deniable encryption, factoring attacks, public key, quadratic residuosity, trapdoor permutation

1 Introduction

While standard encryption schemes protect the privacy of the sender and the receiver against passive eavesdroppers (semantic security), they fail to provide protection against coercers. A coercive adversary has the power (weaker than corruptive adversary) to approach the sender (or the receiver or both) after the ciphertext has been transmitted and of course recorded by this adversary. Assume for an instant that this adversary approaches the sender. She commands the sender to reveal all his random inputs used to produce this particular ciphertext. Since the ciphertext produced using standard encryption schemes (specially, public-key encryption) commits the sender to his random inputs, he cannot lie about his true plaintext. Such commitments allow the coercive adversary to ensure that the sender will do as she wishes.

In situations where the coercer accepts the sender's claim that he erased his plaintext and all the local randomness involved during encryption, then deniable encryption is not needed. However, there are cases where the coercer knows that the sender will be prosecuted if he erased his data and that his data is still there. Consequently, the sender's claim will not be accepted to a coercer. In this later case, deniable encryption comes to play.

Deniable encryption allows a party to escape a coercion. Namely, it allows the sender to produce a ciphertext C that looks like an encryption of a true message M_t and as an encryption of a fake message M_f . Both messages are chosen by the sender. While the receiver is able to decrypt C for M_t, the sender is able to open either M_f or M_t to a coercer which when verified, produces the same ciphertext C.

Deniable encryption maybe classified according to which party is coerced: A sender-deniable encryption schemes are resilient against coercing the sender. The definitions for receiver-deniable and sender-receiver-deniable follow analogously. When the sender and the receiver initially share a common secret key, this is spoken off as shared-key deniable encryption. In deniable public-key encryption, no pre-shared information and no communications are assumed prior to the encryption process. This follows from the assumptions of standard public-key encryption schemes. Yet, deniable public-key encryption is more challenging than deniable shared-key encryption since the public key of the receiver is already known to everyone including the coercer, consequently, the sender cannot lie about the receivers public key.

[*]Part of the work in this paper was accomplished while the author was visiting Dept. of Computer Science, ITE-UCONN, University of Connecticut, USA.

The work in [5] showed that it is possible to transform any sender-deniable encryption scheme to a receiver-deniable encryption scheme and vice-versa. Also, they showed that, with the help of other parties with at least one of them remains un-attacked, it is possible to transform a sender-deniable encryption scheme to a sender-receiver-deniable encryption scheme. Hence, in this paper, we focus on building an efficient and secure sender-deniable public-key encryption scheme.

Deniable encryption is very useful in the protocols where coercive adversaries come to play as a potential threat. For example, deniable encryption protects voters from being coerced during electronic elections [7, 8]. It is also very useful to protect bidders in electronic auctions. Generally, deniable encryption is very important when a party is forced to act with a gun pointing at his/her head.

We distinguish two types of deniability according to the time of coercion: plan-ahead-deniability and unplanned-deniability. In plan-ahead deniability, the sender chooses his fake message during encryption based on what the coercive adversary previously commanded him to do. In unplanned-deniability, the sender must be able to generate the fake message after transmission whenever a coercive adversary approaches him. Our proposed method is of the later type i.e. we assume that the coercer approaches the sender after transmission and the sender must be able to open any message satisfactory to the coercer.

We reduce the required bandwidth (ciphertext bit-length) to be in the order of $\lg N + 2\ell$ bits in case of single bit encryption and in the order of $m \lg \lg N + \lg N + 2^m \ell$ bits in case of multi-bit message encryption, where $m > 1$ is the plaintext bit-length, N is a two-prime modulus and ℓ is the output bit-length of a strong hash function (e.g. SHA-256). At the same time, our method provides strong deniability (i.e. undetectable cheating) equivalent to the infeasibility to factor a sufficiently large two-prime modulus. Unlike the schemes of [5], our methods produce no decryption errors and hence, more reliable.

In the scheme of [5], a one bit plaintext requires tn bits of ciphertext where t is the bit-length of elements in a translucent set $\mathcal{S}_t$ and $t = s + k$ for security parameters n, s and k. The scheme provides deniability of $4/n$ and decryption error of $n2^{-k}$. Hence, to achieve a high level of deniability and a sufficiently low decryption error, the ciphertext is super-polynomial and almost impractical.

The rest of this paper is organized as follows: Section 2 presents the related work in the field. Our motivations and contributions are given in Section 3. Section 4 describes the notion of deniability, the quadratic residuosity of a composite as an underlying primitive and the main idea. Section 5 describes our schemes for single bit encryption. The schemes for multiple bits encryption are presented in Section 6. Section 7 discusses the problem of the sender's local randomness and gives a solution for the problem. Deniability transformation techniques are discussed in Section 8. Finally, the conclusions are given in Section 9.

2 Related Work

The work in [5] constructed a sender-deniable public-key encryption scheme based on trapdoor permutations. However, the scheme (as stated in [5]) falls short of achieving an appropriate level of deniability, that is, to achieve a high deniability, the size of the ciphertext corresponding to a one bit encryption is super-polynomial and hence inefficient. They constructed two deniable public-key encryption schemes based on translucent sets, the first represents the building block for the second which they have called, the "Parity Scheme". The work in [5] also notified that in order to build one-round schemes, different approaches are required. Also, [5] introduced techniques for the less challenging, deniable shared-key encryption and showed that the one-time-pad is a perfect deniable shared-key encryption. Based on the sender-deniable public-key encryption, the work in [4] described a general multiparty computations allowing a set of players to compute a common function of their inputs with the ability to escape a coercion.

In our recent work [9], we devised a scheme for a one-move receiver-deniable public-key encryption which is built over any mediated PKI. Yet, when the scheme is transformed to a sender-deniable scheme, it is no more a one-move scheme.

3 Motivations and Contributions

3.1 Motivations

Deniable public-key encryption is a strong primitive, essential in all cryptographic protocols where a coercive adversary comes to play with high potential. Deniable public-key encryption realizes the Receipt-freeness attribute which is a very important attribute in electronic voting, electronic bidding and auctions. The schemes proposed in [5] fall short of achieving the desired level of deniability and correctness unless the size of the ciphertext corresponding to a one bit encryption is super-polynomial.

Deniable encryption has an impact on the design of adaptively secure multiparty computations [3] since, the notion of deniability is stronger than the notion of non-committing encryption.

3.2 Contributions

The contributions of this paper is to introduce an efficient sender-deniable public-key encryption scheme. We introduce three versions of our scheme, all rely on the quadratic residuosity of a two-prime modulus. We introduce three versions of our scheme. Scheme I and II are for single bit encryption while the third scheme is for multi-bit message encryption. Scheme I is efficient in the case of single bit encryption and loses efficiency in the case of multi-bit message encryption. Scheme II is an introductory scheme to our third scheme and is less efficient than

Scheme I in the case of single bit encryption. We also introduce Scheme I* to improve the deniability of Scheme I considering the sender's local randomness. We show how to build an efficient deniable public-key encryption from any trapdoor permutation. Our proposed schemes enjoy the following properties:

- They are one-move schemes without any pre-encryption information required to be sent from the receiver to the sender prior to encryption.
- No pre-shared secret information is required between the sender and the receiver.
- Achieve a high level of deniability equivalent to the factorization of a large two-prime modulus.
- No deciphering errors.
- The bandwidth (ciphertext bit-length) is significantly improved compared to previous constructions.

We introduce our schemes in the weakest notion of semantic security, namely, probabilistic encryption or equivalently, indistinguishable chosen plaintext attack (IND-CPA) model. Since we focus on the deniability notion, we do not consider CCA security models or non-malleability in this paper although one may realize security against such attacks by applying the generic constructions from [1].

4 Preliminaries

In this section we first describe the notion of deniability and then we introduce the quadratic residuosity of a composite in some details as it represents the basic primitive of the schemes presented in this paper.

4.1 Notion of Deniability

While a standard encryption scheme for encrypting a single bit b can be generally viewed as a scheme having two distributions on ciphertexts: A distribution T_0 for $b = 0$ and a distribution T_1 for $b = 1$; deniable encryption has four distributions [5], T_0, T_1, F_0 and F_1. The distribution T_b is used by the sender as long as he is not willing to open the encryption dishonestly. The distribution F_b allows the sender to open either honestly or dishonestly when coerced. An important property in the deniable encryption scheme is that whenever the sender opens an encryption dishonestly, it must appear as belonging to T_b.

A protocol π is a sender-deniable public-key encryption if:

- *Correctness:* The probability that the receiver output is different from the sender input is negligible.
- *Security:* For any two different messages M_t and M_f, the communications for transmitting M_t are computationally indistinguishable from the communications for transmitting M_f.
- *Deniability:* The adversary's view of an honest encryption of M_t according to protocol π is indistinguishable from the adversary's view when the ciphertext was generated while transmitting M_t and the sender falsely claims that it is an encryption of M_f.

The above concepts must be kept in mind while designing a sender-deniable public-key encryption scheme.

4.2 Quadratic Residuosity

The deniability of our proposed scheme relies mainly on the quadratic residuosity problem [2, 6, 10, 11], specially for a two-prime modulus. In this subsection, we show the mathematical idea we rely on in building our scheme.

Basic definitions:

- For any integer $N > 1$, an integer $a \in Z_N^*$, is a quadratic residue (QR) modulo N if there exists some $x \in Z_N^*$ such that $a = x^2 \bmod N$, and a is a quadratic nonresidue (QNR) modulo N otherwise.
- The set of all quadratic residues in Z_N^* is denoted by Q_N, while the set of all quadratic non-residues in Z_N^* is denoted by $\overline{Q}_N$. We have $\phi(N) = |Q_N| + |\overline{Q}_N|$ where $\phi(N) = |Z_N^*|$ is the Euler totient.
- *Quadratic Residue Problem (QR)* - given integers (a, N), where $N > 1$ and $a \in Z_N^*$, determine if $a \in Q_N$, i.e., if there exists $x \in Z_N^*$, such that $a = x^2$ mod$N$.
- *Quadratic Nonresidue Problem (QNR)* - the complementary problem of determining if $a \in \overline{Q}_N$.
- Define $J_N^+ \subset Z_N^*$ to be the subset of all integers such that for any $a \in J_N^+$, the jacobi symbol $(\frac{a}{N}) = +1$ and define $J_N^- \subset Z_N^*$ to be the subset of all integers such that for any $a \in J_N^-$, the jacobi symbol $(\frac{a}{N}) = -1$. We have $Q_N \subset J_N^+$.

Lemma 1. *Let $a \in_R Z_N^*$, for a sufficiently large two-prime modulus N with unknown odd prime factors. If $a \in J_N^+$ (i.e. $(\frac{a}{N}) = +1$) then deciding whether $a \in Q_N$ is hard. Whereas, if $a \in J_N^-$(i.e. $(\frac{a}{N}) = -1$) then $a \in \overline{Q}_N$.*

Although Lemma 1 has no proof, it is widely believed that determining a quadratic residue modulo N is equivalent to factoring N. Also, note that, while the Jacobi symbol can be efficiently calculated without knowing the factors of N, it is infeasible to compute a square root modulo N without knowing the factors of N [6, 10, 11].

Blum primes. Now, let $N = pq$ where p and q are sufficiently large primes subject to the condition that p and q are Blum primes (i.e. both are congruent to 3 mod 4). In our method, N represents the public key of the receiver while p and q represent his secret parameters. Since $(\frac{a}{N}) = (\frac{a}{p})(\frac{a}{q})$; for any integer $a \in J_N^+$ (i.e. $(\frac{a}{N}) = +1$), we have, $(\frac{a}{p}) = (\frac{a}{q})$ and either (only one) of the following two cases must be true [6]:

1) a is a QR modulo both p and q and hence is a QR modulo N.

2) $-a$ is a QR modulo p and q and hence is a QR modulo N.

The second case arises because by construction p and q are both congruent to 3 mod 4, thus, $(\frac{-1}{p}) = (\frac{-1}{q}) = -1$. So, either a or $-a$ is a QR modulo p and q. It is also important to notice that both a and $-a$ are in J_N^+.

Lemma 2. *Let $N = pq$ be the product of two Blum primes. Let $a \in Q_N$, then a has exactly four square roots, exactly one of which is in Q_N itself.*

Proof. The reader refers to [10] for a complete proof of Lemma 2.

Since N is a composite of two Blum primes p and q, it follows that $(\frac{-1}{p}) = (\frac{-1}{q}) = -1$. For any square root s of a, the possible combinations of $((\frac{s}{p}), (\frac{s}{q}))$ are $(+, +)$, $(+, -)$, $(-, +)$, $(-, -)$. For s to be a QR modulo N then $(+, +)$ must be the case for s. It is important to notice that among the three other QNR roots, only one root is in J_N^+ and it has the combination $(-, -)$ which is $-s$. Since we are concerned only with the integers in J_N^+, we have two roots in J_N^+, one is a QR and the other is a QNR modulo N.

4.3 Our Triggering Idea

In our proposed schemes in Sections 5 and 6, the sender is required to choose blindly (i.e. without the help of the receiver and without knowledge of the factors of N) some random integer $y \in J_N^+$, hopping that y is a quadratic non-residue modulo N. His way to do so is to pick $s \in_R Z_N^*$ and computes $a = s^2 \bmod N$, then computes $y = -a \bmod N$. Since a is a QR modulo N then it must be that y is a QNR modulo N given that N is a composite of two Blum primes. We refer this idea (although used for different purpose) to the pioneer work of Clifford Cocks [6]. Actually, using this idea, Cocks was able to establish the first practical identity-based encryption scheme. *We must emphasize that s is generated and used on the fly to reach a QNR value in J_N^+. The program does not store s anywhere on the system since it is not part of the encryption pattern (neither the plaintext nor the ciphertext). Therefore, a coercer will not ask for the disclosure of s.*

The reader may argue that denying the knowledge of s as the sender's local randomness maybe unacceptable to a coercer, in this scenario, the schemes described in Sections 5 and 6 are not deniable, therefore, in Section 7, we introduce Scheme I* as an improvement to Scheme I to satisfy this argument by allowing the sender to safely lie about his local randomness. Until Section 7, we assume that the coercer accepts the sender's claim of not knowing s.

5 Single Bit Encryption

In this section we introduce our sender-deniable public-key encryption schemes allowing one bit encryption at a time. We introduce two schemes. The first scheme is efficient (from the point of view of bandwidth) as a single bit encryption but loses efficiency for multi-bit message encryption. The second scheme represents an introductory to our multi-bit message encryption, yet, as a single bit encryption, it requires a larger bandwidth than the first scheme. Both schemes have almost the same computation complexity.

Let b_t be the true bit to be encrypted while b_f be the fake bit. By $(b_{k-1}^{(x)} ... b_0^{(x)})$ we denote the binary representation of $x \in \{0, 1\}^k$.

The receiver picks two large Blum primes p and q. He publishes $N = pq$ as his public key while keeping p and q secret.

5.1 Scheme I

This scheme is suitable for one bit encryption.

Encryption: The sender proceeds as follows:

- *Honest encryption* ($b_t = b_f$).
  - Picks $s \in_R Z_N^*$ and computes $a = s^2 \bmod N$.
  - Computes $y_0 = -a \bmod N$. It is clear that y_0 is a QNR and $y_0 \in J_N^+$. Assume for an instant that the receiver knows how to identify y_0. Soon we will show how this is done.
  - Computes the bit $b = \bigoplus_{i=0}^{k-1} b_i^{(y_0)} \oplus b_t$.
  - Picks a small integer $r > 0$. He computes $X = 2^r$.
  - Computes $C = y_0^X \bmod N$ and sends (b, C) to the receiver.
- *Dishonest encryption* ($b_t = \overline{b}_f$).
  - Picks $s \in_R Z_N^*$ and computes $a = s^2 \bmod N$.
  - Computes $y_0 = -a \bmod N$. We still assume that the receiver somehow knows how to identify y_0.
  - Computes the bit $b = \bigoplus_{i=0}^{k-1} b_i^{(y_0)} \oplus b_t$.
  - Keeps on squaring, that is, to compute $y_j = y_{j-1}^{2^j}$, $j = 1, 2, ...$ until there exists some y_j with its binary representation satisfying: $\bigoplus_{i=0}^{k-1} b_i^{(y_j)} = b_f \oplus b$.
  - Performs at least one more squaring to compute $C = y_0^{2^r} \bmod N$. He sends (b, C) to the receiver.

Decryption: On the reception of b and C, the receiver keeps on computing square roots modulo N until he reaches y_0 as a QNR. He simply computes $b_t = \bigoplus_{i=0}^{k-1} b_i^{(y_0)} \oplus b$ as the encrypted bit.

How to distinguish y_0 at the receiver. The reader now argue how the receiver will know when to stop computing square roots. We did not mention this during the description of the scheme for not disturbing the flow of the scheme. For any $x \in J_N^+$, we benefit from the fact that only the receiver is able to distinguish whether $x \in Q_N$ or $x \in \overline{Q}_N$. As the receiver receives C he will start by computing $\sqrt{C}$ which results in four roots. Since p and q are Blum primes, Only one root is a QR which is of course in J_N^+ and only one root is a QNR in J_N^+. The other two roots are out of the receiver's concern and are discarded. For the two roots in J_N^+ we are done with the fact that the receiver is able to distinguish which of them is a QR and which is a QNR. To allow the receiver to stop at the correct QNR which is y_0 in our scheme, all we need is a strong hash function H with an output bit-length say ℓ and a simple trick described next. The sender defines two ℓ-bit values R_0 and R_1 and a random bit $e \in_R \{0,1\}$. In case of honest encryption, the sender sets $R_e = H(y_0)$ and $R_{1-e} \in_R \{0,1\}^\ell$. In case of dishonest encryption, the sender sets $R_e = H(y_0)$ and $R_{1-e} = H(y_j)$. He sends R_0 and R_1 in some order to the receiver. *We emphasize that y_0 is a QNR while y_j is a QR.*

Opening an encryption: To open an encryption honestly, the sender reveals y_0. To open dishonestly, the sender reveals y_j, claims that y_j is a QNR and R_e is random.

Correctness. In the decryption process, on the reception of (b, C, R_0, R_1), the receiver (starting from C) keeps on computing square roots. After each computation, he, (i) discards the two roots in J_N^-, (ii) hashes the QNR root in J_N^+ to see whether it matches either R_0 or R_1. If a match is found he stops and takes this QNR as y_0, else, he continue computing the square roots of the QR in J_N^+ and repeats (i) and (ii). Hence, correctness follows immediately.

Security. For any b_t, $b_f \in \{0,1\}$, the communications between the sender and the receiver for transmitting b_t is indistinguishable from that for transmitting b_f. Also, semantic security is immediate.

Deniability. When the sender reveals y_j and falsely claims that y_j is a QNR and b_f is the encrypted bit, the coercer (without knowing the factors of N) cannot prove the contradiction of this claim, i.e, he cannot prove that y_j is a QR and that R_e is not random. The coercer will not ask the sender for $\sqrt{-y_j}$ since as we mentioned before, s is picked on the fly. The deniability of our scheme is equivalent to the inability of a coercer to factorize N.

Bandwidth. It is required, one bit to encode b, $\lg N$ bits to encode C and 2ℓ bits for the pair (R_0, R_1) totalling $1 + \lg N + 2\ell$ bits of ciphertext.

Remark. We remark that it is possible to remove b from the ciphertext and pick y_0 such that $b_t = \bigoplus_{i=0}^{k-1} b_i^{(y_0)}$ and then search for y_j such that $b_f = \bigoplus_{i=0}^{k-1} b_i^{(y_j)}$, however, including b allows picking y_0 without any trials.

5.2 Scheme II

We present this scheme as an introductory to our multi-bit message sender-deniable public-key encryption scheme.

Encryption: The sender proceeds as follows:

- *Honest encryption* ($b_t = b_f$).
  - Picks $s \in_R Z_N^*$ and computes $a = s^2 \bmod N$.
  - Computes $y_0 = -a \bmod N$.
  - Scans the binary representation of y_0 for an index i such that $b_i^{(y_0)} = b_t = b_f$.
  - Picks a small integer $r > 0$. He computes $X = 2^r$ and $C = y_0^X \bmod N$.
  - Picks $e \in_R \{0,1\}$, sets $R_e = H(y_0)$ and $R_{1-e} \in_R \{0,1\}^\ell$.
  - Sends (i, C, R_0, R_1) to the receiver.
- *Dishonest encryption* ($b_t = \overline{b}_f$).
  - Picks $s \in_R Z_N^*$ and computes $a = s^2 \bmod N$.
  - Computes $y_0 = -a \bmod N$.
  - Picks two random small integers $(r_1, r_2) > 0$. He computes $X_1 = 2^{r_1}$ and $X_2 = 2^{r_2}$.
  - Computes $y_1 = y_0^{X_1} \bmod N$.
  - Scans the binary representations of both y_0 and y_1 for an index i such that $b_i^{(y_0)} = b_t$ and $b_i^{(y_1)} = b_f$.
  - Computes $C = y_1^{X_2} \bmod N$.
  - Picks $e \in_R \{0,1\}$, sets $R_e = H(y_0)$ and $R_{1-e} = H(y_1)$.
  - Sends (i, C, R_0, R_1) to the receiver.

Decryption: On the reception of (i, C, R_0, R_1), starting with C, the receiver keeps on computing square roots modulo N until he reaches y_0 as a QNR in J_N^+ satisfying either $R_0 = H(y_0)$ or $R_1 = H(y_0)$. He takes $b_i^{(y_0)}$ as the encrypted bit.

Opening an encryption: To open an encryption honestly, the sender reveals y_0. To open dishonestly, the sender reveals y_1 and claims that R_e is a random string.

Correctness. Immediate.

Security. For any b_t, $b_f \in \{0,1\}$, the communications between the sender and the receiver for transmitting b_t

is indistinguishable from that for transmitting b_f . Also, semantic security is immediate.

Deniability. When the sender reveals y_1 and falsely claims that y_1 is a QNR and $b_i^{(y_1)}$ is the encrypted bit, the coercer (without knowing the factors of N) has nothing to do but accepting this claim since he cannot prove the contradiction, i.e, he cannot prove that y_1 is a QR and that R_e is not random. The deniability of our scheme is equivalent to the infeasibility to factorize N.

Bandwidth. For one bit encryption it is required, $\lg\lg N$ bits to encode the index i, $\lg N$ bits to encode C and 2ℓ bits to encode R_0 and R_1 totalling, $\lg\lg N + \lg N + 2\ell$ bits of ciphertext.

6 Multiple Bits Encryption

It is possible by trivial extension to perform multi-bit message encryption by repeating Scheme I or Scheme II for each encrypted bit. However in this case the size of the ciphertext will blowup inefficiently. With a slight increase in computation complexity, Scheme II may allow m-bit message encryption while reducing bandwidth to $m(\lg\lg N) + \lg N + 2^m\ell$ bits. Scheme I fails to achieve such reduction, although it is more suitable in case of single bit encryption. Let M_t be the true message to be encrypted. Let $\mathcal{M}_f$ be the set of all possible fake binary messages of m bits (excluding M_t). Obviously, $|\mathcal{M}_f| = 2^m - 1$. It is required that the sender is able to open any message $M_f \in\mathcal{M}_f$ which looks like an honest encryption to a coercer. We assume that m is no more than several bits. The receiver picks two large Blum primes p and q. He publishes $N = pq$ as his public key while keeping p and q secret. The scheme is described next.

Encryption: To encrypt a message, the sender proceeds as follows:

- *Honest Encryption.*
  - Picks $s \in_R Z_N^*$ and computes $a = s^2 \bmod N$.
  - Computes $y_0 = -a \bmod N$.
  - Picks a small integer $r > 0$ and computes $X = 2^r$.
  - $\forall j = 0, ..., m-1$, the sender scans the binary representation of y_0 for an index i_j such that $b_{i_j}^{(y_0)} = b_j^{(M_t)}$.
  - Computes $C = y_0^X \bmod N$.
  - Let $n = 2^m - 1$. Defines the $n+1$ strings $R_0, ..., R_n$, selects a random $i \leq n$ and sets $R_i = H(y_0)$. Sets each other $R_{j\neq i} \in_R \{0,1\}^\ell$.
  - He sends $(i_{m-1}, ..., i_0, C, R_0, ..., R_n)$ to the receiver.
- *Dishonest Encryption.*
  - Picks $s \in_R Z_N^*$ and computes $a = s^2 \bmod N$.
  - Computes $y_0 = -a \bmod N$.
  - The sender keeps on squaring, that is, to compute $y_j = y_{j-1}^2 = y_0^{2^j} \bmod N$, $j = 1, 2, ...$, until there exist distinct and fixed indices $i_{m-1}, ..., i_0$ satisfying the following conditions:
    1) $b_{i_{m-1}}^{(y_0)} = b_{m-1}^{(M_t)}, ..., b_{i_0}^{(y_0)} = b_0^{(M_t)}$.
    2) For each $M_f \in\mathcal{M}_f$ there exists a square y_j such that $b_{i_{m-1}}^{(y_j)} = b_{m-1}^{(M_f)}, ..., b_{i_0}^{(y_j)} = b_0^{(M_f)}$.
  - Performs at least one more squaring to compute $C = y_0^{2^r}$.Obviously, $r \geq 2^m$.
  - Let $n = 2^m - 1$ be the number of strings y_j's (each y_j corresponds to a one fake M_f). Defines the $n+1$ strings $R_0, ..., R_n$, selects a random $i \leq n$ and sets $R_i = H(y_0)$. Assigns each other $R_{k\neq i}$ a value $H(y_j)$.
  - Sends $(i_{m-1}, ..., i_0, C, R_0, ..., R_n)$ to the receiver.

Decryption: On the reception of the ciphertext $(i_{m-1}, ..., i_0, C, R_0, ..., R_n)$, the receiver keeps on computing square roots modulo N until he finds y_0 as a QNR with $H(y_0)$ matches R_i for any $i = 0, ..., n$. He takes the bits $b_{i_{m-1}}^{(y_0)}, ..., b_{i_0}^{(y_0)}$ as the cleartext bits.

Opening an encryption: To open the encryption honestly the sender reveals y_0. To open dishonestly, whenever the coercer approaches the sender, the sender chooses a fake message M_f satisfactory to the coercer and opens the corresponding y_j, he claims that y_j is a QNR and that all values $H(y_0), ..., H(y_{j-1})$ are random. Notice that the coercer is able to find matches $H(y_{j+1}), ..., H(y_n)$ by squaring y_j, we emphasize that this attempt does not threaten deniability since the sender's claim is that y_j is QNR and what is important is that the coercer cannot detect matched values with index less than j.

Bandwidth. The scheme provides a bandwidth of $m(\lg\lg N) + \lg N + 2^m\ell$. For small values of $m > 1$, the scheme is more efficient than Scheme I and II. Larger messages could be partitioned into blocks of m bits each, then performing the encryption for each block independently. Numerically, for $\ell = 256$, $\lg N = 1024$ and $m = 4$, Scheme I provides a ciphertext of 6148 bits, Scheme II provides a ciphertext of 6184 bits while the multiple bits scheme provides a ciphertext of 5160 bits.

The correctness, security and deniability of the scheme follow from Scheme II.

7 The Sender's Local Randomness

As we mentioned earlier, our schemes are unplanned-deniable, that is, we assume that the coercer approaches the sender after transmission. If the coercer approaches the sender before encryption, our schemes fall short of achieving the desired deniability since in this case, the coercer is able to plan with the sender for the sender's local randomness, more precisely, in our method, if the coercer forces the sender to use some y_0 the sender will not be able to escape the coercion. However, if the sender claims that he has no control of his local randomness and this claim is convincing to the coercer, then our method is still deniable.

7.1 Scheme I*: Full Deniability

Now we turn our attention to the problem where the sender – when opening some y_j and claims that y_j is a QNR – is asked to reveal one of the roots of $(-y_j)$, which is supposed to be known to the sender. It is obvious that in this scenario, since $(-y_j)$ is a QNR, the sender is trapped and fails to satisfy the coercer. We modify Scheme I to overcome this problem. In our modification, the sender is able to lie about his local randomness, s. The modifications to Scheme II follow in a similar way.

In our solution, the receiver is required to publish a small integer (say d), where d is the maximum number of square roots computations that he will perform during decryption (e.g. $d = 10$). Notice that d is required to be a little larger in case of multiple bits message encryption depending on the bit length of the message. The receiver picks two large Blum primes p and q, he publishes the pair $(N = pq, d)$ as his public key while keeping p and q secret. Let b_t be the true bit to be encrypted while b_f is the fake bit. All the computations are performed in Q_N. Scheme I* is a modification of Scheme I and is described next:

Encryption: The sender proceeds as follows:

- *Honest encryption* $(b_t = b_f\ )$.
  - Picks $s \in_R Z_N$ and computes $y_0 = s^2 \bmod N$.
  - Computes the bit $b = \bigoplus_{i=0}^{k-1} b_i^{(y_0)} \oplus b_t$.
  - Picks a small integer $0 < r < d$. He computes $X = 2^r$.
  - Computes $C = y_0^X \bmod N$.
  - Picks $e \in_R \{0, 1\}$, sets $R_e = H(y_0)$ and $R_{1-e} \in_R \{0, 1\}^\ell$.
  - Sends (b, C, R_0, R_1) to the receiver.
- *Dishonest encryption* $(b_t = \overline{b}_f)$.
  - Picks $s \in_R Z_N$ and computes $y_0 = s^2 \bmod N$.
  - Computes the bit $b = \bigoplus_{i=0}^{k-1} b_i^{(y_0)} \oplus b_t$.
  - Keeps on squaring, that is, to compute $y_j = y_{j-1}^{2^j}$, $(j = 1, 2, ...) < d$ until there exists some y_j with its binary representation satisfying: $\bigoplus_{i=0}^{k-1} b_i^{(y_j)} = b_f \oplus b$.
  - Performs at least one more squaring to compute $C = y_0^{2^r} \bmod N$, $r \leq d$.
  - Picks $e \in_R \{0, 1\}$, sets $R_e = H(y_0)$ and $R_{1-e} = H(y_j)$.
  - Sends (b, C, R_0, R_1) to the receiver.

Decryption: On the reception of b, C, R_0 and R_1, the receiver starts computing square roots of C, after each computation he:

- Discards the three QNR roots.
- Hashes the QR root to see whether it matches either R_0 or R_1. If a match is found (say R_w), he records this root (say W, where $R_w = H(W)$) and continue computing square roots until one of the following two situations occurs:
  - The receiver reaches d square roots computations. In this case, he takes W as y_0.
  - The receiver finds another QR root that matches R_{1-w}. He takes this new root as y_0 and discards W.
- Finally, he computes $b_t = \bigoplus_{i=0}^{k-1} b_i^{(y_0)} \oplus b$.

Opening the encryption: To open the encryption honestly, the sender reveals y_0 and s. To open dishonestly, the sender reveals y_j, claims that y_{j-1} is s and that R_e is random.

Deniability. To a coercer, y_j is totally indistinguishable from y_0 and y_{j-1} is totally indistinguishable from s.

Correctness is immediate, security and bandwidth follow from Scheme I.

7.2 Generic Constructions

By investigating Scheme I*, we notice that the scheme could be built given any trapdoor permutation (f, f^{-1}), where, f is the receiver's public function and f^{-1} is its trapdoor inverse known only to the receiver. Let $f^{(j)}(y_0) = f(f(...f(f(y_0))...))$ be the process of encrypting y_0 (i.e. applying f to y_0) j times. Again, let d be the maximum number of decryptions that will be performed by the receiver (i.e. the receiver will apply f^{-1} no more than d times). The pair (f, d) is the receiver's public key. The scheme is described next.

Encryption: The sender proceeds as follows:

- *Honest encryption* $(b_t = b_f\ )$.
  - Picks y_0 at random from the domain of f.
  - Computes the bit $b = \bigoplus_{i=0}^{k-1} b_i^{(y_0)} \oplus b_t$.
  - Picks a small integer $0 < r < d$.
  - Computes $C = f^{(r)}(y_0)$.
  - Picks $e \in_R \{0, 1\}$, sets $R_e = H(y_0)$ and $R_{1-e} \in_R \{0, 1\}^\ell$.

  - Sends (b, C, R_0, R_1) to the receiver.

- *Dishonest encryption* $(b_t = \overline{b}_f)$.
  - Picks y_0 at random from the domain of f.
  - Computes the bit $b = \bigoplus_{i=0}^{k-1} b_i^{(y_0)} \oplus b_t$.
  - Keeps on applying f to y_0, that is, to compute $y_j = f^{(j)}(y_0)$, $(j = 1, 2, ...) < d$ until there exists some y_j with its binary representation satisfying: $\bigoplus_{i=0}^{k-1} b_i^{(y_j)} = b_f \oplus b$.
  - Applies f at least one more time to compute $C = f^{(r)}(y_0)$, $j < r \leq d$.
  - Picks $e \in_R \{0, 1\}$, sets $R_e = H(y_0)$ and $R_{1-e} = H(y_j)$.
  - Sends (b, C, R_0, R_1) to the receiver.

Decryption: On the reception of b, C, R_0 and R_1, the receiver starts decrypting by applying f^{-1} to C a maximum of d times, after each computation of f^{-1}, he hashes the output to see whether it matches either R_0 or R_1. If a match is found (say R_w), he records this output (say W, where $R_w = H(W)$) and continue applying f^{-1} until one of the following two situations occurs:

- The receiver reaches d decryptions. In this case, he takes W as y_0.
- The receiver finds another output that matches R_{1-w}. He takes this new output as y_0 and discards W.

Finally, the receiver computes $b_t = \bigoplus_{i=0}^{k-1} b_i^{(y_0)} \oplus b$.

Opening the encryption: To open the encryption honestly, the sender reveals y_0. To open dishonestly, the sender reveals y_j, claims that y_j is picked at random from the domain of f and that R_e is random.

8 Deniability Transformation

Of course, our proposed scheme cannot withstand coercion of the receiver, since a coerced receiver is forced to reveal the prime factors of his public key N. Following the results of [5], a sender-deniable encryption is easily transformed to a receiver-deniable encryption as follows: Let $\mathcal{A}$ be our sender-deniable public-key scheme. Let b be the bit to be encrypted and transmitted from the sender to the receiver. The receiver chooses a random bit r and invokes scheme $\mathcal{A}$ to encrypt and send r to the sender (as if the sender and the receiver have exchanged places). The sender replies by $r \oplus b$ in the clear. A sender-receiver deniable scheme requires n intermediaries, $I_1, ..., I_n$, with at least one of them remains honest (un-attacked). The sender chooses n bits $b_1, ..., b_n$ such that $\bigoplus_i b_i = b$ and sends b_i to each I_i using the sender-deniable public-key encryption. Each I_i transmits b_i to the receiver using the receiver-deniable public-key encryption. Finally, the receiver computes $b = \bigoplus_i b_i$.

9 Conclusions

We proposed schemes for sender-deniable public-key encryption. Our schemes prove efficiency over that proposed in [1] in the sense of bandwidth, deniability and decipherability. Scheme I and II for single bit encryption while the third scheme is for multi-bit message encryption. Scheme I is efficient in the case of single bit encryption and loses efficiency in the case of multi-bit message encryption. Scheme II is an introductory scheme to our third scheme and is less efficient than Scheme I in the case of single bit encryption. Our schemes are one-move and do not require any pre-shared secret information between the sender and the receiver. The schemes are unplanned-deniable and are not secure as plan-ahead-deniable unless the coercer has no control on the sender's local randomness.

We introduced Scheme I* to improve the deniability of our schemes considering the sender's local randomness. We showed how to efficiently build a deniable public-key encryption from any trapdoor permutation.

One final thing worth noting is that, when our schemes are transformed to receiver-deniable schemes, they are no more one-move schemes. For one-move receiver-deniable public-key encryption, the reader may refer to [9].

Acknowledgments

The author is greatly indebted to the anonymous reviewers of the IJNS for their valuable comments. The work in this paper is supported in part by a grant from the Egyptian Culture and Education Bureau (ECEB), Washington-DC, USA.

References

[1] M. Bellare, and P. Rogaway, "Random oracles are practical: a paradigm for designing efficient protocols," *1st Conference on Computer and Communications Security*, pp. 62-73, ACM, 1993.

[2] J. Cai, and R. A. Threlfall, "A note on quadratic residuosity and UP," *Information Processing Letters*, vol. 92, no. 3, pp. 127-131, 2004.

[3] R. Canetti, U. Feige, O. Goldreich, and M. Naor, "Adaptively secure multi-party computation," *28th Annual ACM Symposium on Theory of Computing (STOC)*, pp. 639-648, 1996.

[4] R. Canetti, and R. Gennaro, "Incoercible multiparty computation," *Proceedings of the 37th Annual Symposium on Foundations of Computer Science*, pp. 504-513, 1996.

[5] R. Canetti, C. Dwork, M. Naor, and R. Ostrovsky, "Deniable encryption," *Proceedings of the 17th Annual international Cryptology Conference on Advances in Cryptology*, pp. 90-104, Springer-Verlag, London, 1997.

[6] C. Cocks, "An identity based encryption scheme based on quadratic residues," *Proceedings of IMA 2001*, LNCS 2260, pp. 360-363, Springer-Verlag, 2001.

[7] R. Cramer, R. Gennaro, and B. Schoenmakers, "A secure and optimally efficient multi-authority election scheme," *Eurocrypt '97*, pp. 103-118, 1997.

[8] M. Hirt, and K. Sako, "Efficient receipt-free voting based on homomorphic encryption," *Eurocrypt '00*, pp. 539-556, 2000.

[9] M. H. Ibrahim, "Receiver-deniable public-key encryption," *International Journal of Network Security (IJNS)*, to appear.

[10] P. Junod, *Cryptographic Secure Pseudo-Random Bits Generation: the Blum-Blum-Shub Generator*, manuescript found on the net at http://crypto.junod.info/publications.html, 1999.

[11] A. J. Menezes, P. C. V. Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography*, CRC Press, ISBN: 0-8493-8523-7 October 1996.

Maged Hamada Ibrahim received his BSc in communications and computers engineering from Helwan University, Cairo; Egypt. Received his MSc and PhD in Cryptography and Network security systems from Helwan University in 2001 and 2005 respectively. Currently, working as a lecturer, post doctor researcher and also joining several network security projects in Egypt. His main interest is Cryptography and network security. More specifically, working on the design of efficient and secure cryptographic algorithms, in particular, secure distributed multiparty computations. Other things that interest him are number theory and the investigation of mathematics for designing secure and efficient cryptographic schemes.