Karlsruhe Institute of Technology

KASTEL

A Comparison of Deniable Encryption Notions

Bachelor's Thesis of

Haotian Wu

At the KIT Department of Informatics
KASTEL – Institute of Information Security and Dependability

First examiner:	Prof. Dr. Jörn Müller-Quade
Second examiner:	Prof. Dr. Thorsten Strufe
First advisor:	M.Sc. Saskia Bayreuther
Second advisor:	M.Sc. Astrid Ottenhues

01. June 2025 – 30. November 2025

Karlsruher Institut für Technologie
Fakultät für Informatik
Postfach 6980
76128 Karlsruhe

A Comparison of Deniable Encryption Notions (Bachelor's Thesis)

I declare that I have developed and written the enclosed thesis completely by myself. I have not used any other than the aids that I have mentioned. I have marked all parts of the thesis that I have included from referenced literature, either in their original wording or paraphrasing their contents. I have followed the by-laws to implement scientific integrity at KIT.

Karlsruhe, 30. November 2025

. . . . . .

(Haotian Wu)

Abstract

Deniability is an important feature of modern cryptography. This property allows a participant to plausibly deny an action they have performed. Deniable encryption enables participants to achieve deniability in addition to the two fundamental properties of encryption: security and correctness. In a state with a public and impartial electronic voting system using encrypted ballots, the election should, in theory, be resilient to coercion. However, if the transmission of encrypted ballots takes place over a non-secure channel and a powerful but malicious party can coerce voters into revealing their votes, the election can no longer be considered impartial. By employing sender-deniable encryption to encrypt the ballots, the system could once again resist coercion.

The deniability notion in deniable encryption can be classified into three categories: sender-deniability, receiver-deniability, and sender-and-receiver deniability. With sender-deniability, the sender can plausibly deny what they have sent. With receiver-deniability, the receiver can plausibly deny what they have received. With sender-and-receiver deniability, both the sender and the receiver can plausibly deny what they have sent or received, respectively.

In this thesis, we examine four deniable encryption schemes, two of which are sender-deniable schemes, one is a receiver-deniable scheme, and the other is a sender-and-receiver-deniable scheme. We first examine how they work and how they achieve the desired deniability property. In addition, we evaluate their performance with respect to the cost of time and space. We also observe how a general transformation between a sender-deniable and a receiver-deniable scheme is constructed. Furthermore, we compare the two sender-deniable schemes proposed by Howlader and Basu [10] and Barakat [2]. Finally, we prove that the scheme proposed by Howlader and Basu is not receiver-deniable, while the scheme proposed by Ibrahim is not sender-deniable.

Zusammenfassung

Abstreitbarkeit ist ein wichtiges Merkmal der modernen Kryptographie. Diese Eigenschaft erlaubt es einem Teilnehmer, eine von ihm ausgeführte Handlung plausibel abzustreiten. Abstreitbare Verschlüsselung ermöglicht es den Beteiligten, neben den zwei grundlegenden Eigenschaften der Verschlüsselung – Sicherheit und Korrektheit – Abstreitbarkeit zu erreichen. In einem Staat mit einem öffentlichen und unparteiischen elektronischen Wahlsystem, das verschlüsselte Stimmzettel verwendet, sollte die Wahl theoretisch resistent gegen Nötigung sein. Findet die Übertragung der verschlüsselten Stimmzettel jedoch über einen unsicheren Kanal statt und kann eine mächtige, böswillige Partei Wähler dazu zwingen, ihre Stimmen offenzulegen, kann die Wahl nicht mehr als unparteiisch gelten. Durch den Einsatz sender-abstreitbarer Verschlüsselung zur Verschlüsselung der Stimmzettel könnte das System erneut gegenüber Nötigung widerstandsfähig sein.

Der Abstreitbarkeitsbegriff in abstreitbarer Verschlüsselung lässt sich in drei Kategorien einteilen: Sender-Abstreitbarkeit, Empfänger-Abstreitbarkeit und Sender-und-Empfänger-Abstreitbarkeit. Bei Sender-Abstreitbarkeit kann der Absender plausibel abstreiten, was er gesendet hat. Bei Empfänger-Abstreitbarkeit kann der Empfänger plausibel abstreiten, was er empfangen hat. Bei Sender-und-Empfänger-Abstreitbarkeit können sowohl Absender als auch Empfänger plausibel abstreiten, was sie gesendet beziehungsweise empfangen haben.

In dieser Arbeit untersuchen wir vier abstreitbare Verschlüsselungsverfahren: Zwei davon sind sender-abstreitbar, eines ist empfänger-abstreitbar und eines ist sender-und-empfänger-abstreitbar. Zunächst analysieren wir, wie sie funktionieren und wie sie die gewünschte Abstreitbarkeits-Eigenschaft erreichen. Darüber hinaus bewerten wir ihre Leistung hinsichtlich Zeit- und Speicheraufwand. Ferner betrachten wir, wie eine allgemeine Transformation zwischen einem sender-abstreitbaren und einem empfänger-abstreitbaren Verfahren konstruiert wird. Außerdem vergleichen wir die beiden sender-abstreitbaren Verfahren von Howlader und Basu [10] und Barakat [2]. Abschließend zeigen wir, dass das von Howlader und Basu vorgeschlagene Verfahren nicht empfänger-abstreitbar ist, während das von Ibrahim vorgeschlagene Verfahren nicht sender-abstreitbar ist.

Contents

1 Introduction

Consider a direct-election state with an e-voting system in which an authoritarian regime seeks to stay in power indefinitely by abolishing elections through a mandated referendum. Voters are forced to reveal their votes, and those who vote against the regime's will face strict punishment. As an e-voting system encrypts ballots using the voter's public key and transmits the encrypted ballots over a public, non-secure network, the regime—acting as an eavesdropper or spy—may intercept and store the encrypted ballots. Later, the regime could demand that voters reveal their public keys and corresponding votes. With access to the public key and the intercepted ciphertexts, the regime would then be able to verify whether the voters complied with its instructions. However, if deniable encryption were used instead of traditional public-key encryption, a voter could generate a "fake" public key and a corresponding "fake" vote that together produce a "fake" ciphertext indistinguishable from the intercepted one. With such encryption, a voter may confidently vote according to their own will, yet reveal a different result if placed under coercion after the election. The coercer, however, is unable to prove or disprove whether the voter actually voted as claimed.

In 1982, Yao proposed Yao's Millionaires' Problem [18]. He introduced a game in which Alice and Bob each have assets valued at i and j million USD, respectively. They want to determine who is wealthier without revealing their actual wealth to each other. Their communication occurs over a non-secure channel, which may have zero, one, or multiple eavesdroppers or saboteurs. Additionally, both Alice and Bob wish to learn only the comparative result—who is wealthier—without disclosing their exact wealth or gaining any additional information about the other's assets. Yao aimed to develop an algorithm that would allow Alice and Bob to determine who is wealthier without revealing their actual wealth. He then extended this concept to multi-party scenarios, creating a more general secure computation protocol. His work became the foundation of multi-party secure computation and inspired deniable encryption: what if the participants in a multi-party computation are placed under the coercion of a third party?

Canetti et al. introduced the concept of deniable encryption in their 1997 paper. Suppose an encrypted message is intercepted by an adversary and the adversary later demands that the sender reveal the private key and the corresponding randomness, which were presumably used to produce the ciphertext. An encryption scheme is deniable if the sender can generate a fake private key and a fake randomness under which the intercepted ciphertext decrypts to a different plaintext, making it indistinguishable from an encryption of the original plaintext under the real private key. Canetti et al. classified deniable encryption schemes according to which parties may be coerced: sender-deniable, receiver-deniable, and sender-and-receiver-deniable schemes. A sender-deniable encryption scheme allows the sender to plausibly

deny what the sender has sent when a third party demands disclosure of the message or plaintext. Similarly, a receiver-deniable encryption scheme allows the receiver to deny what the receiver has received when a third party demands disclosure of the message or plaintext. If a scheme allows both the sender and receiver to plausibly deny what they have sent or received when a third party demands disclosure of the message or plaintext, it is called a sender-and-receiver-deniable encryption scheme, or bi-deniable encryption scheme for short.

In this thesis, we introduce the deniability notions with respect to deniable encryption. A formal definition is provided to define the three key properties: correctness, security, and deniability. We then present two protocols for sender-deniable encryption, one protocol for receiver-deniable encryption, and one protocol for sender-and-receiver-deniable encryption. Furthermore, we examine the differences between these protocols, discussing their advantages and disadvantages. Finally, we explore the transformation between sender-deniable and receiver-deniable encryption schemes. Still, there are a few more notions and applications that are not introduced in this thesis, such as off-the-record deniability, receipt-freeness, steganography, etc.

2 Related Work

As this thesis mainly focuses on deniable encryption, which represents only a small area within the broader topic of deniability, it is worth noting that many other studies explore deniability in combination with different fields.

In a functional encryption system, a decryption key allows a user to learn a function of the encrypted data [4]. De Caro, Iovino, and O'Neill proposed the essential properties of deniable functional encryption and additionally summarized two models: full deniability from trapdoor circuits and multi-distributional deniability from delayed trapdoor circuits [8].

Canetti, Park, and Poburinnaya introduced fully deniable interactive encryption. The goal was to deal with situations where both parties are placed under the coercion of a single third party. In this case, the third party may check whether the plaintexts revealed by both parties correspond to each other. Therefore, a new deniability notion was proposed: off-the-record deniability, which guarantees protection for each party independently of the other party's actions without prior coordination [5].

In electronic voting, several schemes have been proposed to solve the coercion scenario described in the introduction by achieving deniability. Alwen et al. defined a security notion for incoercible multiparty computation and proposed a protocol that achieves such deniability [1]. The new deniability notions: receipt-freeness and coercion resistance are introduced alongside.

Steganography is another field where deniability is applied. Steganography enables a sender to conceal a secret message inside a piece of media that appears completely normal. For example, a message can be embedded in an image, allowing the sender to transmit the image instead of the message or ciphertext itself. Xu et al. introduced deniable steganography and presented a receiver-deniable image steganography protocol [16].

Apart from deniable encryption, another major research focus is deniable message authentication. Deniable message authentication protects the authentication process from spies or malicious verifiers. It occurs between a sender (prover) and a receiver (verifier) and allows the prover, in the strongest sense, to dispute that such a verification process has taken place. Fischlin and Mazaheri further classified its deniability properties into five types: content deniability, context deniability, time deniability, source deniability, and destination

deniability [9].

Additionally, Yao and Zhao [17] proposed deniable Internet Key Exchange, which integrates deniability into key exchange protocols and makes it compatible with widely deployed key exchange standards. In the same work, Yao and Zhao introduced two additional deniability notions: concurrent deniability and forward deniability.

3 Preliminaries

We introduce in this section some important definitions and security assumptions. In addition, a few algorithms that are used in Section 4 are also included.

Definition 3.0.1 (Negligible Function[1])**.** *A function negl is negligible if for every polynomial $p(\cdot)$ there exists an N such that for all integers $n > N$ it holds that $negl(n) < \frac{1}{p(n)}$.*

Definition 3.0.2 (Computational Indistinguishability[2])**.** *Two probability ensembles $X = \{X_n\}_{n\in\mathbb{N}}$ and $Y = \{Y_n\}_{n\in\mathbb{N}}$ are computationally indistinguishable, denoted $X \overset{c}{\equiv} Y$, if for every probabilistic polynomial-time distinguisher D there exists a negligible function* negl *such that:*

$$|\Pr[D(X_n) = 1] - \Pr[D(Y_n) = 1]| \leq \mathsf{negl}(n)$$

The distinguisher D is given the unary input 1^n so that it can run in time that is polynomial in n in its attempt to distinguish.

Definition 3.0.3 (Transmission (txm)[3])**.** *We denote a transmission of a message m between a sender $\mathcal{S}$ and a receiver $\mathcal{R}$ as $txm(m, r_S, r_R)$ with r_S, r_R being two randomly chosen inputs from the sender $\mathcal{S}$ and the receiver $\mathcal{R}$ respectively.*

Definition 3.0.4 (Quadratic Residue (QR)[4])**.** *Let n be an odd, positive integer, and let x be an integer that is relatively prime to n. The integer x is a quadratic residue modulo n if the equation*

$$x \equiv y^2 \bmod n$$

has an integer solution y. We denote in this case $x \in Q_n$. In other words, the integer x is a square modulo n. The integer x is a quadratic non-residue (QNR) otherwise, and we denote in this case $x \in \bar{Q}_n$

[1] [13, Def. 3.5]
[2] [13, Def. 6.31]
[3] Adapted from [6]
[4] [12]

Definition 3.0.5 (Invertible Sampling[5]). *Let $A : X \times \{0,1\}^* \to Y$ be a probabilistic polynomial-time algorithm. We say A has invertible sampling and that A is a probabilistic polynomial-time algorithm with invertible sampling, if there exists a probabilistic polynomial-time random-bits-faking-algorithm $A^{-1} : Y \times X \to \{0,1\}^*$ such that for all inputs $x \in X$. uniformly random bits $r \leftarrow \mathcal{R}_A$, output value $y \leftarrow A(x,r)$, and fake random bits $r' \leftarrow A^{-1}(y,x)$ the random variables (x,y,r') and (x,y,r) are computationally indistinguishable (see 3.0.2).*

3.1 Security Assumptions

In this section, several important security assumptions are introduced. These assumptions provide the theoretical foundation for the schemes discussed in the later sections.

Definition 3.1.1 (Strong RSA Assumption[6]). *Let $1 < \tau \in \mathbb{Z}$ be a security parameter. Let $n = pq$ be a product of two random $\tau-$bit primes and let x be an element of group $\mathbb{Z}_N^*$. The strong RSA-Problem is defined as follows:*

$$\begin{gathered} \textit{given (n,x) as an input, find a pair } a, b \in \mathbb{Z} \\ \textit{such that } a^b = x \mod n \textit{ and } b \neq \pm 1 \end{gathered}$$

The strong RSA assumption states that for a sufficiently large τ the strong RSA problem cannot be solved in probabilistic polynomial-time.

Definition 3.1.2 (Quadratic Residuosity Assumption [7]). *The Quadratic Residuosity Problem (QRP) is to determine whether an integer x with Jacobi Symbol 1 modulo a composite number n ($n = pq$ and p, q are prime) is a QR (see 3.0.4), where the Jacobi symbol of x modulo n is defined as the product of the Legendre symbols of x modulo each prime factor of n. In other words, QRP is to determine whether an integer x exists so that $(\frac{x}{n}) = 1$.*
The Quadratic Residuosity Assumption (QR-Assumption) states that the QRP cannot be efficiently solved if only given x and n, but could be solved efficiently if p, q, the factorization of n, are given.

Definition 3.1.3 (Computational Diffie–Hellman Assumption[8]). *Fix a cyclic group $\mathbb{G}$ and generator $g \in \mathbb{G}$. Let a, b be two randomly chosen group elements. The Computational Diffie–Hellman (CDH) problem is to compute g^{ab} given g^a, g^b . The CDH assumption states that there is no known feasible algorithms capable of solving the CDH problem in probabilistic polynomial-time.*

[5] [7][Def.1]
[6] [3]
[7] [12]
[8] [13, pp.264-265]

3.2 Symmetric and Asymmetric Encryption

Definition 3.2.1 (Symmetric Encryption [9])**.** *A symmetric encryption scheme is a tuple of probabilistic polynomial-time algorithms* (GEN, ENC, DEC) *such that:*

1. *The key-generation algorithm* GEN *takes as input the security parameter* 1^n *(i.e. length of the key) and outputs a key* k*. We write this as* $k \leftarrow GEN(1^n)$ *. We will assume without loss of generality that any key* k *output by* $GEN(1^n)$ *satisfies* $|k| \geq n$*.*
2. *The encryption algorithm* ENC *takes as input a key* k *and a plaintext message* $m \in \{0,1\}^*$*, and outputs a ciphertext* c*. Since* ENC *may be randomized, we write this as* $c \leftarrow ENC_k(m)$*.*
3. *The decryption algorithm* DEC *takes as input a key* k *and a ciphertext* c*, and outputs a result* m'*. We assume without loss of generality that* DEC *is deterministic, and so write this as* $m' := DEC_k(c)$*.*

It is required for correctness that for every n*, every key* k *output by* $GEN(1^n)$*, and every* $m \in \{0,1\}^*$*, it holds that* $DEC_k(ENC_k(m)) = m'$ *and* $m = m'$*. Symmetric encryption is also known as shared-key encryption or private key encryption.*

Definition 3.2.2 (Asymmetric Key Encryption[10])**.** *An asymmetric encryption scheme is a tuple of probabilistic, polynomial-time algorithms* (GEN, ENC, DEC) *that satisfies the following:*

1. *Algorithm* GEN *takes as input a security parameter* 1^n *(i.e. length of the key) and outputs a pair of keys* (pk, sk)*. We refer to the first of these as the public key and the second as the private key or secret key. We assume for convenience that* pk *and* sk *each have length at least* n*, and that* n *can be determined from* pk, sk*.*
2. *Algorithm* ENC *takes as input a public key* pk *and a message* m *from some underlying plaintext space (that may depend on* pk*). It outputs a ciphertext* c*, and we write this as* $c \leftarrow ENC_{pk}(m)$*.*
3. *Algorithm* DEC *takes as input a private key* sk *and a ciphertext* c*, and outputs a message* m *or a special symbol* $\perp$ *denoting failure. We assume without loss of generality that* DEC *is deterministic and write this as* $m' := DEC_{sk}(c)$*.*

We require that for every n*, every* (pk, sk) *pair output by* $GEN(1^n)$*, and every message* m *in the appropriate underlying plaintext space, it holds that* $DEC_{sk}(ENC_{pk}(m)) = m'$ *and* $m = m'$*. Asymmetric encryption is also known as public-key encryption.*

[9] [13, Def.3.8]
[10] [13, Def.10.1]

Definition 3.2.3 (RSA encryption)**.** *In RSA encryption, we need to select two prime numbers (sufficiently big[11]) p and q and consequently compute $n = pq$. Then we compute $r = (p-1)(q-1)$. Next we select an e, with $1 \leq e \leq r-1$ such that $gcd(r,e) = 1$. We take (n,e) as the public key and (n,d) as the private key. In the following parts of this thesis, we take the public key as e instead of (n,e) and the private key as d instead of (n,d) since n is available to everyone (see figure 3.1). To encrypt a message m, the sender needs to use the receiver's public key to calculate the ciphertext with $c = m^e \mod n$. The receiver may decrypt the ciphertext by computing $m = c^d \mod n$ (see figure 3.2).*

The strong RSA assumption 3.1.1 states that given any $n = pq$, where p, q are prime and unknown, for any given ciphertext c, there is no probabilistic polynomial-time algorithm to find any pair (m,e) such that $c \equiv m^e \mod n$. Therefore, RSA encryption is considered secure.

In the variant mediated RSA (mRSA), a neutral third-party, also called the SEcurity Mediator (SEM), is introduced, and the secret key is split into two parts: the user-owned private key $d^{Receiver}$ and the SEM-owned private key d^{SEM}. The encryption proceeds exactly as RSA encryption. What distinguishes it from a naive implementation is the decryption. The detailed encryption and decryption procedures of mRSA will be presented with the protocol in 4.2.2 [11].

Generator GEN

Take in security parameter k
Choose two large prime numbers p, q of length k
Compute $n = pq, r = (p-1)(q-1)$
Compute $\lambda = lcm(p-1, q-1)$
Select e, where $1 \leq e \leq r$ and $gcd(r,e) = 1$
Compute d where $ed \equiv 1 \mod \lambda$
Save $pk = (n,e), sk = (n,d)$
Discard everything else

Figure 3.1: Key Generation of RSA Encryption

[11] as of 2025, typically 2048-4096 bits long

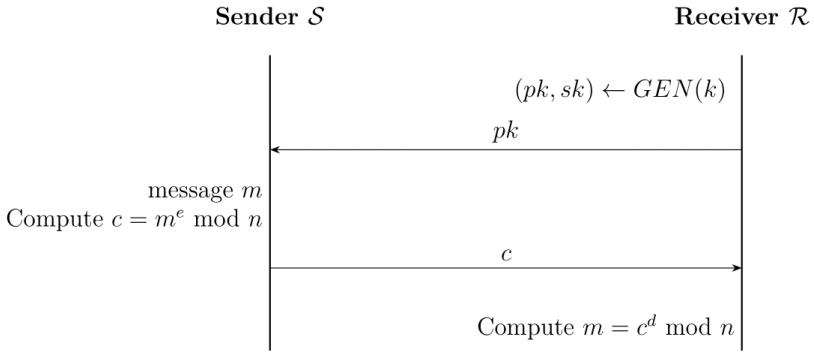

Figure 3.2: RSA Encryption and Decryption

Definition 3.2.4 (Simulatable Public Key Encryption[12])**.** *Let Gen, Enc, Dec be the algorithms for key generation, encryption, and decryption as defined in 3.2.2, respectively. Let M be the message space.*
Let $OGen(1^n, r_{OGen})$ denote an oblivious key generation algorithm which produces a public key Opk and has invertible sampling via algorithm I_{OGen} .
Let $OEnc(pk)$ denote an oblivious encryption scheme which produces a ciphertext Oc and has invertible sampling via I_{OEnc}.
For a simulatable public key encryption scheme $(Gen, Enc, Dec, OGen, OEnc)$, the distribution of pk should be computationally indistinguishable from Opk. Further, the outputs of the following games, the generator game (left) and the simulatable game (right) shall be computationally indistinguishable:

$pk \leftarrow \text{Gen}(1^n, r_R)$	$pk \leftarrow \text{Gen}(1^n, r_R)$
$m \leftarrow \mathcal{M}$	$Oc \leftarrow \text{OEnc}(pk)$
$c \leftarrow \text{Enc}(pk, m)$	Return (pk, r_R, Oc)
Return (pk, r_R, c)	

3.3 Oblivious Transfer

Oblivious transfer (OT) was first proposed by Rabin [15]. It was designed to transmit secrets between a sender and a receiver: the sender sends multiple secrets at once without knowing which one the receiver receives, and the receiver obtains exactly one secret with no acknowledgment of the other secrets.
Let G_q be a subgroup of order q of Z_p^* where p is prime and $p - 1$ is divided by q with no remainder (i.e. $q|p - 1$). Let g be a generator of the group, and assume that the CDH assumption holds. Let H be a hash function. We call an oblivious transfer consisting of n messages OT_n^1 : The sender owns n strings, $r_0, ..., r_{n-1}$ and picks $n - 1$ random values $U_1, ..., U_{n-1}$ and publishes them. The sender also picks a random R and sends g^R to the receiver. The receiver selects a random k and sets $pk_\sigma = g^k$ with $\sigma \in \{0, ..., n - 1\}$ at the receiver's own choice. The receiver sends pk_0 to the sender. The sender computes

[12] [7]

pk_0^R and $pk_i^R = U_i^R / pk_0^R$, $\forall i \in \{0, ..., n-1\}$. The sender sends the encryption of every $r_i, H(pk_i^R, w, i) \oplus r_i$, where w is a random string known to both parties. Finally, the receiver is able to decrypt the receiver's choice using pk_σ (see figure 3.3)[11].

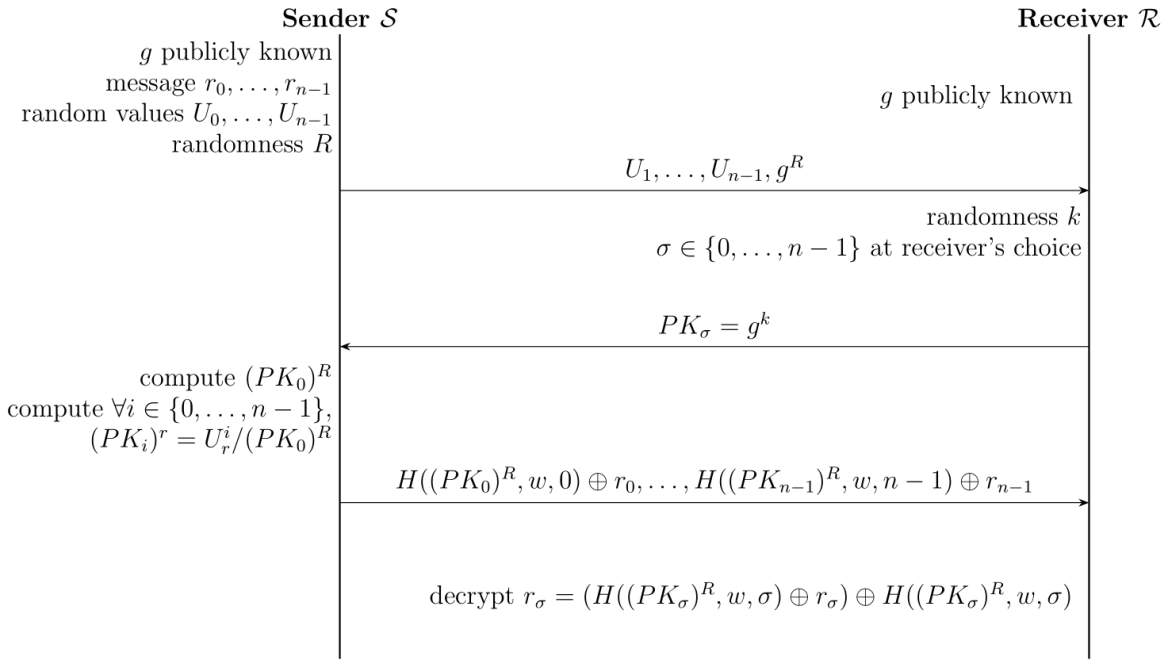

Figure 3.3: Oblivious Transfer in Game Display

4 Fundamental Deniability Notions

In this section, we first define the concept of deniable encryption. Then, we introduce the three primary notions of deniability. Subsequently, several classic encryption schemes are discussed with respect to these notions. Finally, we analyze the schemes and provide the amortized cost of time and space.

4.1 Deniable Encryptions

In this section, we introduce the fundamentals of deniable encryption and define three notions of deniability: sender deniability, receiver deniability, and sender-and-receiver deniability. We consider in this thesis only asymmetric deniable encryption.

Definition 4.1.1 (Deniable Encryption [1])**.** *Canetti et al. [6] formalized deniable encryption as follows. Let* (pk, sk) *be a key pair, where* $pk \neq sk$*. We denote a transmission (3.0.3) between sender and receiver as* $txm(m, r_S, r_R)$ *with* r_S, r_R *being two randomly chosen inputs from the sender and the receiver respectively.*
A deniable encryption scheme must satisfy the regular encryption notions, which are:

1. ***Correctness****: For every message m, if*

$$c \leftarrow ENC_{\mathsf{pk}}(m), \quad m' \leftarrow DEC_{\mathsf{sk}}(c)\,,$$

   then the probability that the outcome of decryption m' *differs from the original plaintext* m *is negligible, i.e.* $\Pr[m' \neq m]$ *is smaller than a negligible function (3.0.1).*

2. ***Security***[2]*: For any two messages* m_1, m_2 *of equal length and any public key pk, the distributions*

$$txm(m_1, r_S, r_R) \quad \textit{and} \quad txm(m_2, r_S, r_R)$$

   are computationally indistinguishable.

   Additionally, it must satisfy a third notion.

3. ***Deniability****: There exists an efficient* faking algorithm f *that allows a party to plausibly produce alternative randomness consistent with some other plaintext. The formal definition will be addressed in 4.1.1.*

[1] Adapted from [6]
[2] Equivalent to the IND-CPA security

4.1.1 Three Fundamental Types of Deniability

We detail the three types of deniability:

Definition 4.1.2 (Sender Deniability[3]). *A sender-deniable encryption shall satisfy the above mentioned correctness and security. Besides, it shall satisfy the deniability property defined as follows:*

There is an efficient algorithm f_S such that, for any two messages $m_1 \neq m_2$ and any sender randomness r_S, (m_2, r'_S, c) is computationally indistinguishable (3.0.2) from $(m_2, r_S, txm(m_2, r_S, r_R))$, where

$$c \leftarrow txm(m_1, r_S, r_R), \quad r'_S \leftarrow f_S(m_1, r_S, c, m_2),$$

Intuitively, the sender can "fake" having encrypted m_2 instead of m_1. We use a simple game to illustrate the process (see 4.1). The sender wins the game if the chance that the adversary makes the right guess is less than $1/2 + negl(n)$. The game is rather simple, we would only visualize the game here. The game for receiver-deniability or sender-and-receiver deniability can be constructed analogously.

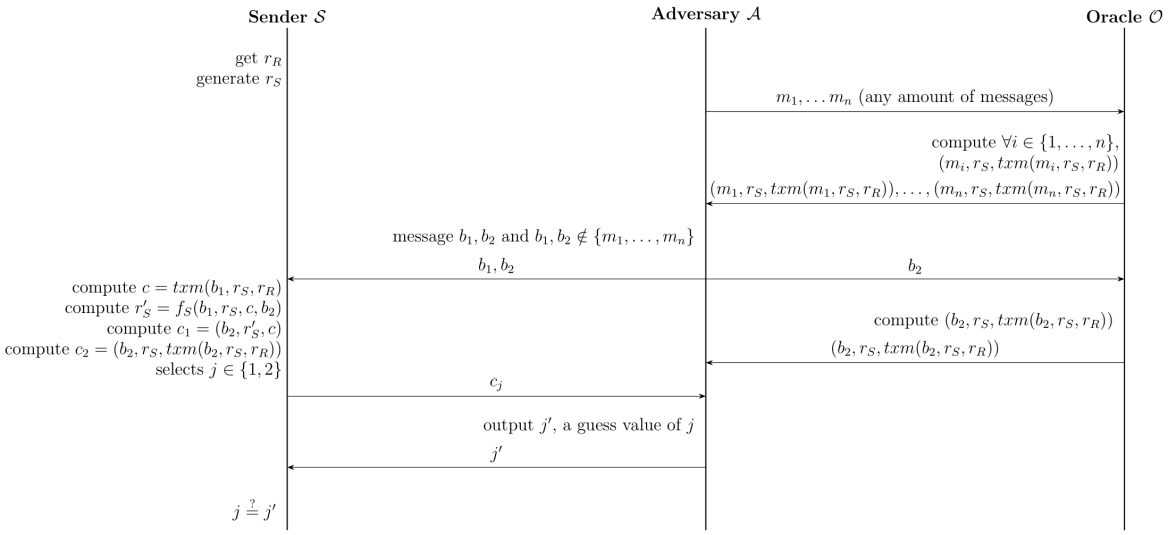

Figure 4.1: Sender Deniability in Game Display

Definition 4.1.3 (Receiver Deniability[4]). *A receiver deniable encryption shall satisfy the above mentioned correctness and security. Besides, it shall satisfy the deniability property defined as follows:*

[3] [6, Def.2]
[4] [6, Def.9]

There is an efficient algorithm f_R such that, for any $m_1 \neq m_2$ and any receiver randomness r_R, (m_2, r'_R, c), is computationally indistinguishable (3.0.2) from $(m_2, r_R, txm(m_2, r_S, r_R))$, where

$$c \leftarrow txm(m_1, r_S, r_R), \quad r'_R \leftarrow f_R(m_1, r_R, c, m_2),$$

Intuitively, the receiver can "fake" a claim to have decrypted c as m_2.

Definition 4.1.4 (Sender-and-Receiver Deniability[5])**.** *A sender-and-receiver deniable encryption shall satisfy the above mentioned correctness and security. Besides, it shall satisfy the deniability property defined as follows:*

There is an efficient algorithm f_{SR} such that, for any $m_1 \neq m_2$, sender randomness r_S and receiver randomness r_R, if

$$c \leftarrow txm(m_1, r_S, r_R), \quad r'_S \leftarrow f_{SR}(m_1, r_S, c, m_2), \quad r'_R \leftarrow f_{SR}(m_1, r_R, c, m_2),$$

then both

$$(m_2, r'_S, c) \approx (m_2, r_S, txm(m_2, r_S, r_R)) \quad and \quad (m_2, r'_R, c) \approx (m_2, r_R, txm(m_2, r_S, r_R))$$

are computationally indistinguishable (3.0.2). In other words, both sender and receiver can independently claim the plaintext was m_2. Sender-and-receiver deniable encryption is also known as bi-deniable encryption.

However, if both parties do not coordinate with each other on how to generate fake randomness before the coercion, the bi-deniability does not provide the desired deniability when both parties are placed under coercion by the same coercer.

4.2 Example Deniable Encryption Schemes

We introduce two sender-deniable public-key encryption schemes, one receiver-deniable public-key encryption scheme, and one sender-and-receiver deniable encryption scheme. We assume that all key-exchange procedures are secure.

4.2.1 Sender Deniable Public Key Scheme

In this subsection, we first introduce a sender deniable public key encryption scheme proposed by Howlader and Basu, which achieves sender-deniability.

[5] [6, Def.10]

Let $n \geq 3$ be an odd composite number.J_n^+ is the set of all pseudosquares, and J_n^+, J_n^- are defined as follows.

$$J_n^+ = \{\, a \in \mathbb{Z}_n^* \mid \left(\tfrac{a}{n}\right) = 1\}, \quad J_n^- = \{\, a \in \mathbb{Z}_n^* \mid \left(\tfrac{a}{n}\right) = -1\}.$$

Let n be a product of two distinct primes. Then half of the elements in J_n^+ are quadratic residues and the other half are quadratic nonresidues. That is, if $a \in J_n^+$, then

$$\Pr[a \in Q_n] \;=\; \tfrac{1}{2}.$$

We then define how to communicate a binary stream y of k bits.

For each bit $b_i^y, 0 \leq i \leq k-1$: If the i^{th} bit is 1, then the sender selects t elements $x_j \in \mathbb{Z}_n^*$, for $0 \leq j \leq t-1$ and computes $a_j = x_j^2 \; mod \; n$. Otherwise, the sender selects t numbers of elements such that $a_j \in J_n^+$ for $0 \leq j \leq t-1$.
The binary streams y can be represented as:

$$A_{(i,j)} \;=\; \begin{vmatrix} a_{(0,0)} & a_{(0,1)} & \cdots & a_{(0,t-1)} \\ a_{(1,0)} & a_{(1,1)} & \cdots & a_{(1,t-1)} \\ \vdots & \vdots & \ddots & \vdots \\ a_{(k-1,0)} & a_{(k-1,1)} & \cdots & a_{(k-1,t-1)} \end{vmatrix}$$

Let y be a binary stream of k bits. For every single bit in the binary stream, the sender does one of the following, depending on the bit b_i^y.

1. If $b_i^y = 1$, select t elements

   $$x_j \in \mathbb{Z}_n^*, \quad 0 \leq j \leq t-1,$$

   and compute

   $$a_j \;\equiv\; x_j^2 \pmod{n}, \quad 0 \leq j \leq t-1.$$

2. If $b_i^y = 0$, select t elements

   $$a_j \in J_n^+, \quad 0 \leq j \leq t-1.$$

The sender then encrypts the message m to c as

$$c = \begin{bmatrix} b_{k-1}^c \\ \vdots \\ b_1^c \\ b_0^c \end{bmatrix} = \begin{bmatrix} b_{k-1}^m \\ \vdots \\ b_1^m \\ b_0^m \end{bmatrix} \oplus \begin{bmatrix} b_{k-1}^y \\ \vdots \\ b_1^y \\ b_0^y \end{bmatrix}$$

the ciphertext c, together with $y = A_{(i,j)}$ will be sent. Receiver decrypts the message c to m as

$$m = \begin{bmatrix} b^m_{k-1} \\ \vdots \\ b^m_1 \\ b^m_0 \end{bmatrix} = \begin{bmatrix} b^c_{k-1} \\ \vdots \\ b^c_1 \\ b^c_0 \end{bmatrix} \oplus \begin{bmatrix} b^y_{k-1} \\ \vdots \\ b^y_1 \\ b^y_0 \end{bmatrix}$$

with b^y the binary stream reconstructed from matrix $A_{(i,j)}$. The reconstruction is executed as follows:

$$b^y_i = \begin{cases} 0, & \text{if } \exists\} a_{(i,j)} \in \bar{Q}_n,\ 0 \le j \le t-1, \\ 1, & \text{if } \forall\, a_{(i,j)} \in Q_n,\ 0 \le j \le t-1. \end{cases}$$

In case the sender is under coercion, the sender may modify $A_{(i,j)}$ and flip some bit in y from 1 to 0. The fake binary stream $\bar{y}$ can be constructed without being noticed by the coercer. The sender may then construct a fake message m_f based on the fake binary stream $\bar{y}$ and ciphertext c with $m_f = \bar{y} \bigoplus c$.

However, this scheme suffers from the Quadratic Residue Problem (QRP) (3.1.2). The scheme is considered secure as there are no feasible algorithms known to date, that resolve QRP efficiently. However, as of now, it is still not known if QRP can be reduced to the problem of integer factorization.

Performance Observation:
We recall that t is the number of randomly chosen elements. To encrypt a message of k bits, $tO(k)$ modular exponentiation computations and $O(k)$ XOR operations are required.
To decrypt a message of k bits, $tO(k)$ modular exponentiation computations are required.
The ciphertext of a k-bit plaintext takes up $tO(k \log n)$ bits of space.

Lemma 1. *The scheme is not receiver deniable.*

Proof. The trick of the sender-deniable scheme is that the sender can manipulate the matrix $A_{(i,j)}$ efficiently without being detected by the coercer. However, the receiver has no information about the secret key and therefore cannot interpret $A_{(i,j)}$ dishonestly. Thus, the scheme is not receiver-deniable.

□

We use a simple example to disprove the receiver deniability.

For a better understanding, we present an example deniable encryption as follows:

Let $p = 11$, $q = 13$; hence $n = pq = 143$ (in practice, p and q are much bigger and might be 512 bits long in binary format. We use simple numbers for a simplified explanation).

Let the message to be encrypted be $m = 1011_2$. Let y be a binary string: $y = 1100_2$.

We can produce

$$A_{(i,j)} = \begin{bmatrix} 25 & 64 \\ 100 & 49 \\ 9 & 4 \\ 81 & 16 \end{bmatrix}.$$

The sender sends $A_{(i,j)}$ and ciphertext c together to the receiver. In the honest case, the receiver can decrypt the ciphertext and obtain $m = 1011_2$.

If the receiver is under coercion and tries to convince the coercer that m is 0000_2 instead of 1011_2, the receiver would need to find a y' such that $y' = 0111_2$. But 143 factors only as 11×13. So we cannot find alternative p', q' such that $p' \neq 11$, $q' \neq 13$, and $p'q' = 143$.

Therefore, y is fixed at 1100_2, and the scheme is not receiver-deniable.

Sender Deniable Public Key Scheme by Barakat[2] In this paragraph, we focus on an improved version of sender-deniable encryption scheme built on the scheme by Howlader and Basu. The scheme by Howlader and Basu is not secure against the QRP (3.1.2) and executes slowly as multiple square root computations are required.
Same as the previous scheme, y is communicated separately and displayed as matrix A, where a_j represents the $j - th$ column of matrix A. If the sender would encrypt the true message (honest encryption), the sender may proceed as follows:

- Selects two primes p, q with $p \neq q$.
- Let PK be $n = pq$ with p and q secret.
- Selects a pseudosquare $y \in \mathbb{Z}_n$ (i.e., y is QNR).
- Let message m be a binary string $m = m_1, m_2, \ldots, m_l$.
- For $i = 1, \ldots, l$ do:
  - Select $x \in \mathbb{Z}_n^*$ at random.
  - If $m_i = 0$, sender computes $a_j = X_j^2 \mod n$, where $X_j \in \mathbb{Z}_n^*$, for $0 \leq j \leq m - 1$.
  - Otherwise, sender computes $a_j = y \cdot X_j^2 \mod n$.
- The sender scans the binary representation of y for an index i_j such that $b_{i_j}^{(y)} = b_j^{(M_t)}$.
- To ensure that the receiver is able to distinguish whether $X \in Q_N$ or $X \in \overline{Q_N}$ as well as to allow the receiver to stop at the correct QNR which is y in our scheme, we use a strong hash function $\mathcal{H}$ with an output bit-length L as follows:
  - Let $\varepsilon = 2^m - 1$. Defines strings $R_0, \ldots, R_\varepsilon$, selects a random $i \leq \varepsilon$, and sets $R_i = \mathcal{H}(y)$. Then, sets each other $R_j \neq i \in \{0,1\}^\ell$.

- Randomly selects $0 < r < n$, and then the sender computes $C = g^{y+nr} \mod n$, where g is some element of $\mathbb{Z}_n^*$.
- Sends $(i_{m-1}, \ldots, i_0, C, R_0, \ldots, R_\varepsilon)$ to the receiver.

If the sender would encrypt the false message (dishonest encryption), the sender may proceed as follows.

- Selects two primes p, q and n where $n = pq$.
- Selects a bit stream y of k bits, where y is a QNR.
- Picks two small integers $0 < (r_1, r_2) < n$ and lets g be some element of $\mathbb{Z}_n^*$.
- Computes $y_1 = g^{y+nr_1} \mod n$.
- Scans the binary representation of both y and y_1 such that

$$b_{i_{m-1}}^{(y)} = b_{m-1}^{(M_t)} \ldots b_{i_0}^{(M_t)} \quad \text{and} \quad b_{i_{m-1}}^{(y_1)} = b_{m-1}^{(M_f)} \ldots b_{i_0}^{(M_f)}.$$

- Let $\varepsilon = 2^m - 1$ be the number of strings y_j (i.e., each y_j corresponds to one fake M_f).
- Defines strings $R_0, \ldots, R_\varepsilon$, selects a random $i \leq \varepsilon$, and sets $R_i = \mathcal{H}(y)$, and sets each other $R_j \neq i \in \{0,1\}^\ell$ as a value of $\mathcal{H}(y_1)$.
- Computes $C = g^{y_1+nr_2} \mod n$.
- Sends $(i_{m-1}, \ldots, i_0, C, R_0, \ldots, R_\varepsilon)$ to the receiver.

The decryption process works as follows:
The receiver decrypts the received message $(i_{m-1}, \ldots, i_0, C, R_0, \ldots, R_\varepsilon)$ starting with C. Then, the receiver keeps on computing $y \mod n$ until the receiver reaches

$$y = \frac{L(C^\alpha \bmod n^2)}{L(g^\alpha \bmod n^2)} \mod n$$

as a QNR in $\mathbb{J}_n^+$ satisfying that $R_i = \mathcal{H}(y)$ for any $i = 0, \ldots, \varepsilon$. Hence, the receiver decrypts $b_{i_{m-1}}^{(y)}, \ldots, b_{i_0}^{(y)}$ as the cleartext bits, where $L(x) = \frac{x-1}{n}$.

Performance Observation:
To encrypt a message of k-bits, $O(k)$ modular exponentiation computations, 1 hash function computation are required.
To decrypt a message of k-bits, in best practice $O(|n|^{2+\alpha})$ if g is chosen, so that $|\alpha| = |n|^\epsilon$, where $\epsilon = 2^m - 1$.
Space Cost: $O(2^k)$

4.2.2 Receiver Deniable Public Key Scheme

In this subsection, we introduce a receiver-deniable public-key encryption scheme proposed by Ibrahim [11].

This scheme is realized based on Mediated RSA (mRSA) (3.2.3) and Oblivious Transfer (3.3).

Key Generation A public key and private key pair is generated as described in 3.2.3. Let (e, N) represent the receiver's public key.

Denote by d_R the piece of receiver's private key held by the receiver and d_{SEM} the piece of SEM's private key held by the SEM.

Encryption Let m be the message to be encrypted by the sender for the receiver, and let m_t be the t-th bit in the message. The sender encrypts each message bit m_t in m as follows

- Picks a $\log N$-bit string $R \in \mathbb{Z}_N$. Let $r_0, r_1, \ldots, r_{n-1}$ be the binary representation of R.
- Scans the binary representation of R for an index i such that $r_i = m_t$.
- Computes and sends C_i and C_R, where

$$C_i = i^e \mod N, \quad C_R = R^e \mod N$$

  to the receiver.

Decryption The decryption is executed jointly by the SEM and the receiver, and proceeds as follows:

Step 1: Receiver partial decryption $(PD)^{(Receiver)}$ The receiver computes

$$PD_R^{(\text{Receiver})} = C_R^{d_R} \mod N,$$

$$PD_i^{(\text{Receiver})} = C_i^{d_R} \mod N.$$

The receiver then sends $PD_R^{(\text{Receiver})}$ and C_i to the SEM.

Step 2: SEM partial decryption $(PD)^{(SEM)}$ The SEM computes

$$PD_i^{(\text{SEM})} = C_i^{d_{SEM}} \mod N.$$

$$R = PD_R^{(\text{Receiver})} C_R^{d_{SEM}} \mod N.$$

The SEM then sends $PD_i^{(\text{SEM})}$ back to the receiver.

Step 3: Reconstruction. The receiver computes

$$i = PD_i^{(\text{SEM})} \cdot PD_i^{(\text{Receiver})} \mod N,$$

Step 4: Oblivious Transfer. Using the reconstructed values, the receiver and the SEM perform an Oblivious Transfer protocol. As a result, the receiver obtains the bit m_t. The scheme is receiver-deniable under the assumption that the communication between the receiver and the SEM is beyond the eavesdropping capabilities of the coercer.

Performance Observation:
The encryption and decryption of a single-bit message have no significant difference compared with mRSA encryption schemes together with Oblivious Transfer. Only negligible overhead is required. However, the encryption and decryption of a k-bit message would cost k times mRSA encryptions and k times Oblivious Transfers.
The ciphertext of a k-bit plaintext takes up $NO(k)$ bits.

Lemma 2. *The scheme is not secure against sender coercion.*

Proof. The scheme is not secure against sender coercion as a coerced sender is forced to reveal R and the index i, which are verifiable by the coercer using the receiver's public key. □

4.2.3 Sender-and-receiver Deniable (Bi-deniable) Encryption Scheme

Bi-deniable encryption proposed by O'Neill, Peikert, and Waters utilized simulatable public key encryption as well as Oblivious Transfer (3.3). The scheme with message space $\{0, 1\}$ works as follows:

Honest Key Generation:

$$
\begin{array}{l}
\textbf{BI-DEN.Gen}(1^n): \\
\quad \mathcal{R} \leftarrow \mathcal{P}_n([5n]) \\
\quad \text{for } i = 1 \text{ to } 5n \text{ do:} \\
\quad\quad \text{if } i \in \mathcal{R} \text{ then} \\
\quad\quad\quad pk_i \leftarrow \text{Gen}(1^n; r_{\mathcal{R},i}) \\
\quad\quad \text{else} \\
\quad\quad\quad pk_i \leftarrow \text{OGen}(1^n; r_{\mathcal{R},i}) \\
\quad pk \leftarrow pk_1 \parallel \ldots \parallel pk_{5n} \\
\quad \textbf{return } pk
\end{array}
$$

Let $\mathcal{R}$ be a subset of size n sampled uniformly at random from a natural number set of size $5n$. Any sub-public-key pk_i is generated by $\text{Gen}(1^n; r_{\mathcal{R},i})$ if $i \in \mathcal{R}$, otherwise generated by $\text{OGen}(1^n; r_{\mathcal{R},i})$. The public key will be the concatenation of all sub-public-keys in order, i.e., $pk \leftarrow pk_1||...||pk_{5n}$.

Honest Encryption:

$$
\begin{aligned}
&\textbf{BI-DEN.Enc}(pk, m):\\
&\quad \mathcal{S} \leftarrow \mathcal{P}_n([5n])\\
&\quad \text{for } i = 1 \text{ to } 5n \text{ do:}\\
&\qquad \text{if } i \in \mathcal{S} \text{ then}\\
&\qquad\quad c_i \leftarrow \mathrm{Enc}(pk_i, b; r_{\mathcal{S},i})\\
&\qquad \text{else}\\
&\qquad\quad c_i \leftarrow \mathrm{OEnc}(pk_i; r_{\mathcal{S},i})\\
&\quad c \leftarrow c_1 \parallel \ldots \parallel c_{5n}\\
&\quad \textbf{return } c
\end{aligned}
$$

Let $\mathcal{S}$ be a subset of size n sampled uniformly at random from a natural number set of size $5n$. Any sub-ciphertext c_i is the outcome of $\mathrm{Enc}(pk_i, m; r_{\mathcal{S},i})$ if $i \in \mathcal{S}$, otherwise the outcome of $\mathrm{OEnc}(pk_i; r_{\mathcal{S},i})$. The ciphertext will be the concatenation of all sub-ciphertexts in order, i.e. $c \leftarrow c_1||...||c_{5n}$.

Decryption:

$$
\begin{aligned}
&\textbf{BI-DEN.Dec}((\mathcal{R}, r_R), c):\\
&\quad \text{for all } i \in \mathcal{R} \text{ do:}\\
&\qquad d_i \leftarrow \mathrm{Dec}(r_{R,i}, c_i)\\
&\quad \text{if most of the } d_i\text{'s are 1 then}\\
&\qquad \textbf{return } 1\\
&\quad \textbf{else return } 0
\end{aligned}
$$

$i \in R, d_i \leftarrow \mathrm{Dec}(r_{R,i}, c_i)$. If more d_i are 1, then return $true$, otherwise $false$.

Deniable Key Generation:

$$
\begin{aligned}
&\textbf{BI-DEN.DenGen}(1^n):\\
&\quad \mathcal{R} \leftarrow \mathcal{P}_n([5n])\\
&\quad \text{for } i = 1 \text{ to } 5n \text{ do:}\\
&\qquad pk_i \leftarrow \mathrm{Gen}(1^n; r_{\mathcal{R},i})\\
&\quad pk \leftarrow pk_1 \parallel \ldots \parallel pk_{5n}\\
&\quad r \leftarrow r_{\mathcal{R},1} \parallel \ldots \parallel r_{\mathcal{R},5n}\\
&\quad \textbf{return } (pk, \mathcal{R}, r)
\end{aligned}
$$

Let $\mathcal{R}$ be a subset of size n sampled uniformly at random from a natural number set of size $5n$. Any sub-public-key pk_i is generated by $\mathrm{Gen}(1^n; r_{\mathcal{R},i})$. The public key will be the concatenation of all sub-public-keys in order, i.e. $pk \leftarrow pk_1||...||pk_{5n}$. The randomness r will be the concatenation of all sub-randomness, i.e. $r_{\mathcal{R},1}||...||r_{\mathcal{R},5n}$. Instead of returning pk, $(pk, (\mathcal{R}, r))$ will be returned.

Deniable Encryption:

BI-DEN.DenEnc(pk, m') :
$\mathcal{S}_0 \leftarrow \mathcal{P}_n([5n])$
$\mathcal{S}_1 \leftarrow \mathcal{P}_n([5n] \setminus \mathcal{S}_0)$
$\mathcal{Y} \leftarrow \mathcal{P}_n([5n] \setminus (\mathcal{S}_0 \cup \mathcal{S}_1))$
for $i = 1$ to $5n$ do:
  if $i \in \mathcal{S}_0$ then $c_i \leftarrow \mathrm{Enc}(pk_i, 0; r_{\mathcal{S},i})$
  if $i \in \mathcal{S}_1$ then $c_i \leftarrow \mathrm{Enc}(pk_i, 1; r_{\mathcal{S},i})$
  if $i \in \mathcal{Y}$ then $c_i \leftarrow \mathrm{Enc}(pk_i, m'; r_{\mathcal{S},i})$
  else $c_i \leftarrow \mathrm{OEnc}(pk_i; r_{\mathcal{S},i})$
$c \leftarrow c_1 \| \ldots \| c_{5n}$
return c

Let $\mathcal{S}_0$ be a subset of size n sampled uniformly at random from a natural number set of size $5n$. Let $\mathcal{S}_1$ be a subset of size n sampled uniformly at random from a natural number set of size $5n$ excluding $\mathcal{S}_0$. Let $\mathcal{Y}$ be a subset of size n sampled uniformly at random from a natural number set of size $5n$ excluding $\mathcal{S}_0$ and $\mathcal{S}_1$. Any sub-ciphertext c_i is the outcome of $\mathrm{Enc}(pk_i, 0; r_{\mathcal{S},i})$ if $i \in \mathcal{S}_0$, the outcome of $\mathrm{Enc}(pk_i, m'; r_{\mathcal{S},i})$, otherwise the outcome of $\mathrm{OEnc}(pk_i; r_{\mathcal{S},i})$. The ciphertext will be the concatenation of all sub-ciphertexts in order, i.e. $c \leftarrow c_1||...||c_{5n}$.

Generation of fake randomness:

BI-DEN.FakeCoins(pk, f_k, r_S, m', m) :
$c \leftarrow \mathrm{BI\text{-}DEN.Enc}(pk, .'; r_S)$
$z \leftarrow \mathrm{HGD}(5n, n, n)$
$\mathcal{Z} \leftarrow \mathcal{P}_z(S_b)$
$\mathcal{Z}' \leftarrow \mathcal{P}_{n-z}([5n] \setminus (\mathcal{S}_0 \cup \mathcal{S}_1 \cup \mathcal{Y}))$
$\mathcal{R}^* \leftarrow \mathcal{Z} \cup \mathcal{Z}'$
$\mathcal{S}^* \leftarrow S_b$
for $i = 1$ to $5n$ do:
  if $i \in \mathcal{S}^*$ then $r^*_{S,i} \leftarrow r_{S,i}$
  else $r^*_{S,i} \leftarrow \mathrm{IOEnc}(pk_i, c_i)$
  if $i \in \mathcal{R}^*$ then $r^*_{R,i} \leftarrow r_{R,i}$
  else $r^*_{R,i} \leftarrow \mathrm{IOGen}(pk_i)$
$r^*_S \leftarrow r^*_{S,1} \| \ldots \| r^*_{S,5n}$
$r^*_R \leftarrow r^*_{R,1} \| \ldots \| r^*_{R,5n}$
return (r^*_S, r^*_R)

Let c be the honest encryption of fake message m'. Let z be the random integer sampled from a hypergeometric distribution $\mathrm{HGD}(5n, n, n)$[6]. Let $\mathcal{Z}$ be a subset of size z sampled from subset $\mathcal{S}_m$. Let $\mathcal{Z}'$ be a subset of size $n - z$ sampled from integer set of size $5n$ excluding subset $\mathcal{S}_0, \mathcal{S}_1, \mathcal{Y}$. Let R^* be the union of $\mathcal{Z}$ and $\mathcal{Z}'$. Let S^* be S_m. For any index i,

[6]Hypergeometric distribution is to describe the distribution of sampling without replacement, which in our case means that $Pr[z = k] = \frac{\binom{n}{k}\binom{4n}{n-k}}{\binom{5n}{n}}$ and $k \in \{1, \ldots, n\}$

if $i \in \mathcal{S}^*$ and $i \in \mathcal{S}^*$ then $r^*_{S,i} \leftarrow r_{S,i}, r^*_{R,i} \leftarrow r_{R,i}$.
If $i \in \mathcal{S}^*$ and $i \notin \mathcal{S}^*$ then $r^*_{S,i} \leftarrow r_{S,i}, r^*_{R,i} \leftarrow I_{O\mathrm{Gen}}(pk_i)$.
If $i \notin \mathcal{S}^*$ and $i \in \mathcal{S}^*$ then $r^*_{S,i} \leftarrow I_{O\mathrm{Enc}}(pk, c_i), r^*_{R,i} \leftarrow r_{R,i}$.
If $i \notin \mathcal{S}^*$ and $i \notin \mathcal{S}^*$ then $r^*_{S,i} \leftarrow I_{O\mathrm{Enc}}(pk, c_i), r^*_{R,i} \leftarrow I_{O\mathrm{Gen}}(pk_i)$.
The fake sender randomness is the concatenation of $r^*_{S,i}$ i.e. $r^*_S \leftarrow r^*_{S,1} \parallel ... \parallel r^*_{S,5n}$. The fake receiver randomness is the concatenation of $r^*_{R,i}$ i.e.$r^*_R \leftarrow r^*_{R,1} \parallel ... \parallel r^*_{R,5n}$.

Performance Observation:
The time cost depends on the encryption schemes. Compared with the normal encryption scheme, the encryption and the decryption of bi-deniable schemes, require at least five times cost of time. The fake coin generation requires twice the cost of time as is required in decryption or encryption. To encrypt a message of k-bits, the cost of space is $5k$ bits.

4.3 Transformation between Sender-deniable and Receiver-deniable Encryptions

In this section, we observe a generally applicable transformation between sender-deniable and receiver-deniable encryption.

4.3.1 From Sender-deniable Encryptions to Receiver-deniable Encryptions

Let A be a sender-deniable encryption scheme, then a receiver-deniable scheme B may be constructed as follows:

Let m denote the message of length k to be transmitted from sender $\mathcal{S}$ to receiver $\mathcal{R}$. $\mathcal{R}$ selects a randomness r of length k and sends the r to $\mathcal{S}$ utilizing the scheme A. Afterwards, $\mathcal{S}$ sends $m \oplus r$ to $\mathcal{R}$. If A is sender deniable, then, $\mathcal{R}$ may plausibly claim that r is r' as desired, where r' is a fake randomness of length k. As a result, $\mathcal{R}$ may plausibly claim that the message the receiver received is $m' = m \oplus r \oplus r'$ accordingly. The scheme B therefore, allows the receivers to plausibly deny what they have received, i.e., receiver-deniable. The figure 4.2 visualizes the construction of the receiver-deniable encryption scheme B with ENC and DEC representing the encryption and decryption algorithms of the sender-deniable encryption scheme A.

4.3.2 From Receiver-deniable to Sender-deniable Encryptions

Let A be a receiver-deniable encryption scheme, then a sender-deniable scheme B may be constructed as follows:

Let m denote the message of length k to be transmitted from sender $\mathcal{S}$ to receiver $\mathcal{R}$. The receiver $\mathcal{R}$ selects a randomness r and sends the r to $\mathcal{S}$ utilizing the scheme A. Afterwards,

$\mathcal{S}$ sends $m \oplus r$ to $\mathcal{R}$. If A is receiver deniable, then, $\mathcal{S}$ may plausibly claim that the value of r is r' as desired, where r' is a fake randomness of length k. As a result, $\mathcal{S}$ may plausibly claim that the message the sender sent is $m' \oplus r' = m \oplus r$ accordingly. The scheme B, therefore, allows the senders to plausibly deny what they have sent, i.e. sender-deniable. The figure 4.3 visualizes the construction of the sender-deniable scheme B with ENC and DEC representing the encryption and decryption algorithms of the receiver-deniable encryption scheme A.

4.3.3 Remarks

Although a generally applicable transformation between a sender-deniable and a receiver-deniable scheme has been proposed, in practice it is not common to achieve a sender-deniable scheme from a receiver-deniable scheme with such a transformation or vice versa. This transformation makes no modification to the original schemes but treats them as black boxes. This provides a simple and straightforward way for the user to achieve a sender-deniable/ receiver-deniable scheme when one only has a feasible receiver-deniable/ sender-deniable scheme at hand. Since additional communications and computations are required, the performance with respect to space and time is both negatively impacted. To realize sender-deniability or receiver-deniability, it is often more practical to propose a new scheme.

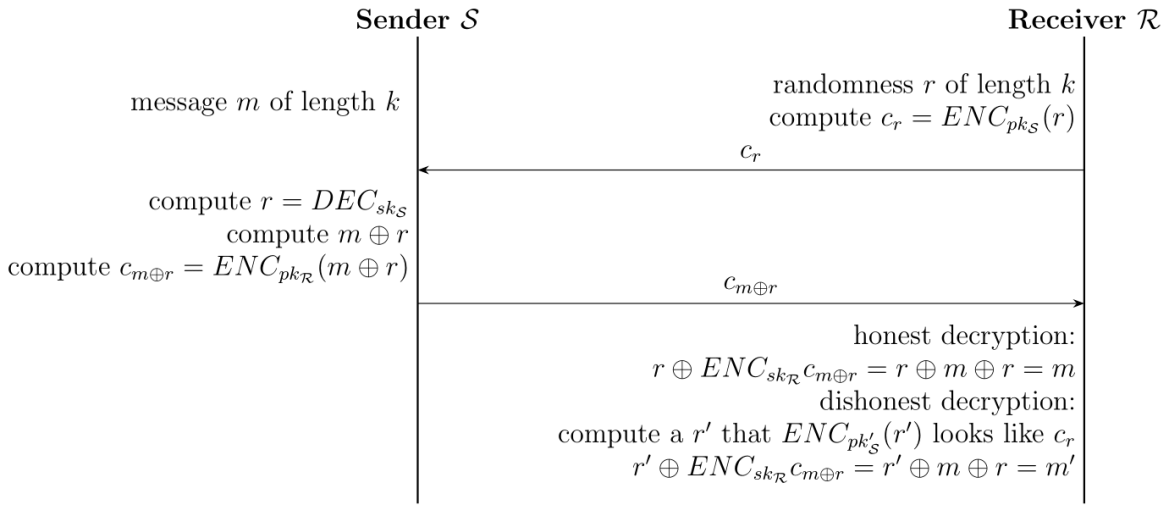

Figure 4.2: The construction of scheme B from scheme A

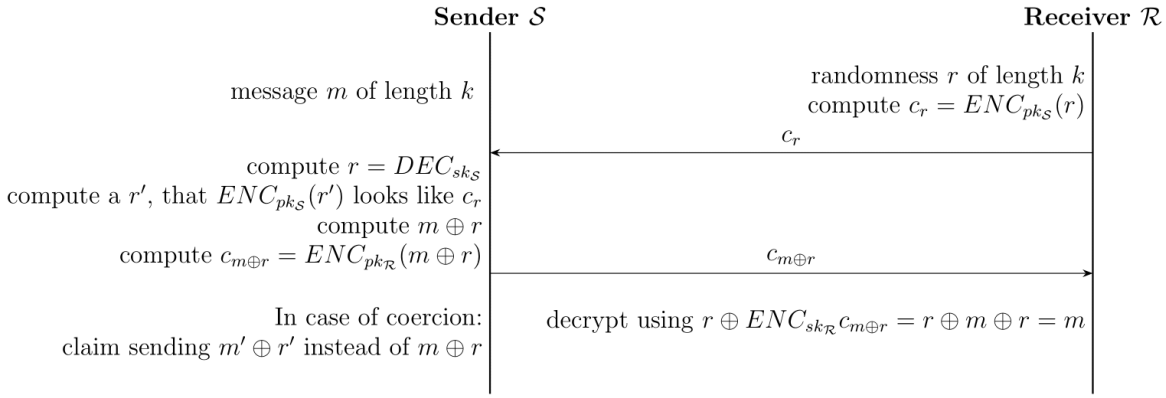

Figure 4.3: Caption

5 Conclusion

The goal of this paper was to briefly introduce the notions of deniable encryption and observe their performance. We presented the three key deniability notions: sender-deniability, receiver-deniability, and sender-and-receiver-deniability. For each notion, we introduced at least one feasible encryption scheme. We also described a generally applicable transformation between a sender-deniable encryption scheme and a receiver-deniable encryption scheme. Next, we compared the schemes proposed by Howlader and Basu [10] and Barakat [2]. Additionally, we examined the scheme proposed by Ibrahim [11] to determine whether it also achieves receiver deniability. Finally, we provided general performance observations for all three schemes.

Due to the limited time available for this bachelor's thesis, we discussed only the three fundamental deniability notions. As deniability in modern cryptography is highly interdisciplinary, many related topics lie outside the scope of this work. However, future research could explore these interdisciplinary fields as introduced in Section 2.

While the schemes mentioned above achieve the desired properties, they nonetheless leave several questions unanswered.

First, all of these schemes rely on specific assumptions. For example, the receiver-deniable encryption scheme proposed by Ibrahim is secure only under the strong RSA assumption. The RSA assumption is generally believed to be secure, as no feasible algorithm is currently known to solve the factoring problem. However, Shor's algorithm can efficiently solve the factoring problem on a quantum computer. Therefore, once large-scale quantum computers become available, all encryption schemes based on the RSA cryptosystem will no longer be secure. Research into quantum-resistant deniable encryption is still in a preliminary stage.

Second, all existing schemes require additional computational power and storage compared to standard encryption schemes without deniability. It remains an open question whether a deniable encryption scheme can be designed that does not require extra computational or storage overhead compared to standard encryption schemes.

6 Disclaimer

AI generative tools were utilized in the preparation of this thesis in accordance with the "Guidelines for the Use of Generative AI in Teaching at the KIT Faculty of Computer Science (AI Guidelines for Computer Science)" [[https://www.informatik.kit.edu/downloads/studium/Guidelines_Generative_AI_Informatics.pdf]
The following outlines the AI tools used and their specific contributions:

Search Engine In the initial phase of this thesis, ChatGPT was used to gather background information on the topic and to generate summaries of potential research directions.

Code Generation ChatGPT was used to convert mathematical equations from screenshots or handwritten notes into LaTeX code. Figure 3.1 is also generated by ChatGPT based on my handwritten draft.

Grammar Correction The thesis was originally written in English. ChatGPT was used solely for grammar checking and correction to improve readability. Writefull assistant in Overleaf was also used.

Translation The "Zusammenfassung" section was translated entirely using ChatGPT, based on the English abstract.

Bibliography

[1] Joël Alwen et al. "Incoercible multi-party computation and universally composable receipt-free voting". In: *Advances in Cryptology–CRYPTO 2015: 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part II 35*. Springer. 2015, pp. 763–780.

[2] Tamer Mohamed Barakat. "A new sender-side public-key deniable encryption scheme with fast decryption". In: *KSII Transactions on Internet and Information Systems (TIIS)* 8.9 (2014), pp. 3231–3249.

[3] Dan Boneh. "Strong RSA Assumption". In: *Encyclopedia of Cryptography, Security and Privacy*. Springer, 2025, pp. 2545–2545.

[4] Dan Boneh, Amit Sahai, and Brent Waters. "Functional encryption: Definitions and challenges". In: *Theory of Cryptography Conference*. Springer. 2011, pp. 253–273.

[5] Ran Canetti, Sunoo Park, and Oxana Poburinnaya. "Fully deniable interactive encryption". In: *Advances in Cryptology–CRYPTO 2020: 40th Annual International Cryptology Conference, CRYPTO 2020, Santa Barbara, CA, USA, August 17–21, 2020, Proceedings, Part I 40*. Springer. 2020, pp. 807–835.

[6] Ran Canetti et al. "Deniable encryption". In: *Advances in Cryptology—CRYPTO'97: 17th Annual International Cryptology Conference Santa Barbara, California, USA August 17–21, 1997 Proceedings 17*. Springer. 1997, pp. 90–104.

[7] Ivan Damgård and Jesper Buus Nielsen. "Improved non-committing encryption schemes based on a general complexity assumption". In: *Annual International Cryptology Conference*. Springer. 2000, pp. 432–450.

[8] Angelo De Caro, Vincenzo Iovino, and Adam O'Neill. "Receiver-and sender-deniable functional encryption". In: *IET Information Security* 12.3 (2018), pp. 207–216.

[9] Marc Fischlin and Sogol Mazaheri. "Notions of deniable message authentication". In: *Proceedings of the 14th ACM Workshop on Privacy in the Electronic Society*. 2015, pp. 55–64.

[10] Jaydeep Howlader and Saikat Basu. "Sender-side public key deniable encryption scheme". In: *2009 International Conference on Advances in Recent Technologies in Communication and Computing*. IEEE. 2009, pp. 9–13.

[11] Maged Hamada Ibrahim. "Receiver-deniable Public-Key Encryption." In: *Int. J. Netw. Secur.* 8.2 (2009), pp. 159–165.

[12] Burt Kaliski. "Quadratic Residuosity Problem". In: *Encyclopedia of Cryptography and Security*. Ed. by Henk C. A. van Tilborg and Sushil Jajodia. Boston, MA: Springer US, 2011, pp. 1003–1003. ISBN: 978-1-4419-5906-5. DOI: 10.1007/978-1-4419-5906-5_429. URL: https://doi.org/10.1007/978-1-4419-5906-5_429.

[13] Jonathan Katz and Yehuda Lindell. *Introduction to modern cryptography: principles and protocols*. Chapman and hall/CRC, 2007.

[14] Adam O'Neill, Chris Peikert, and Brent Waters. "Bi-deniable public-key encryption". In: *Annual Cryptology Conference*. Springer. 2011, pp. 525–542.

[15] Michael O Rabin. "How to exchange secrets with oblivious transfer". In: *Cryptology ePrint Archive* (2005).

[16] Yong Xu et al. "Deniable steganography". In: *arXiv preprint arXiv:2205.12587* (2022).

[17] Andrew C Yao and Yunlei Zhao. "Deniable internet key exchange". In: *International Conference on Applied Cryptography and Network Security*. Springer. 2010, pp. 329–348.

[18] Andrew C. Yao. "Protocols for secure computations". In: (1982), pp. 160–164. DOI: 10.1109/SFCS.1982.38.