A Framework for Efficient and Composable Oblivious Transfer

Chris Peikert, SRI International
Vinod Vaikuntanathan, MIT*
Brent Waters†, SRI International

December 21, 2007

Abstract

We propose a simple, general, and unified framework for constructing oblivious transfer (OT) protocols that are *efficient*, *universally composable*, and *generally realizable* from a variety of standard number-theoretic assumptions, such as the decisional Diffie-Hellman assumption and the Quadratic Residuosity assumption. More interestingly, we can also instantiate our framework with *worst-case* complexity assumptions relating to *lattices*.

Our OT protocols are round-optimal (one message each way), efficient in the parties' communication and local computation, and use only one reference string for an unbounded number of executions. Furthermore, the protocols can provide *unconditional* security to either the sender or receiver, simply by changing the distribution of the reference string. (For several versions of the protocol, even a common *random* string suffices.)

One of our key technical contributions is a simple and novel abstraction that we call a *dual-mode* cryptosystem. We implement dual-mode cryptosystems by taking a unified view of several cryptosystems in the literature that have what we call "message-lossy" public keys, whose defining property is that a ciphertext produced under such a "key" carries *no information* (even statistically) about the encrypted message.

As a contribution of independent interest, we also provide a multi-bit version of Regev's lattice-based cryptosystem (STOC 2005) whose time and space efficiency are improved by a linear factor. In particular, the amortized runtime per message bit is only $\tilde{O}(n)$ bit operations, and the ciphertext expansion can be made as small as a constant.

---

*Work performed while at SRI International.
†Supported by NSF CNS-0524252, CNS-0716199, and the US Army Research Office under the CyberTA Grant No. W911NF-06-1-0316.

1 Introduction

Secure multi-party computation (MPC) [GMW87, BGW88] is one of the oldest and most important concepts in cryptography. A secure MPC protocol allows a set of untrusted parties to compute any function over their respective inputs, both privately and robustly. In principle, this can be used to construct almost any type of on-line cryptographic application between mutually distrustful parties.

Recently, there has been a renewed interest in bringing secure two-party computation into practice. Malhki *et al.* created Fairplay [MNPS04], a software tool that compiles a program into a protocol that two parties can run over the Internet, using Yao's garbled circuit technique [Yao86]. However, Fairplay offers only very weak soundness guarantees in the presence of a malicious party. More recently, Lindell and Pinkas [LP07] constructed an efficient two-party protocol that is secure against malicious adversaries by replacing inefficient zero knowledge proofs with cut-and-choose techniques and combinatorial analysis. These works demonstrate that the primary bottleneck for constructing efficient two-party computation is in the underlying *oblivious transfer* (OT) protocol.

Oblivious transfer was first introduced by Rabin [Rab81] and has received considerable attention from the research community; see, e.g., [Kil88, Cré87, NP01]. Oblivious transfer allows one party, called the receiver, to get exactly one of two (or more) values from another other party, called the sender. The receiver is oblivious to the other value(s), and the sender is oblivious to which value was received. OT protocols that are secure against semi-honest adversaries can be constructed from (enhanced) trapdoor permutations and made robust to malicious adversaries using zero knowledge proofs [GMW87]; however, this general approach is very inefficient.

Naor and Pinkas [NP01] first constructed efficient OT protocols based on the decisional Diffie-Hellman (DDH) assumption. Generalizing their approach, Tauman Kalai [Kal05] presented efficient OT protocols based on a variety of concrete assumptions, building upon the projective hash framework of Cramer and Shoup [CS02]. The primary drawback of these constructions is that their security is only proven according to a "half-simulation" definition of security, where an ideal world simulator is demonstrated only in the case of a cheating receiver. Protocols that are proven secure in the half-simulation model will not necessarily be secure when integrated into a larger protocol, such as a multi-party computation. Indeed, Naor and Pinkas [NP01] point out that a half-simulation secure protocol may fall to selective-failure attacks, where the sender causes a failure that depends upon the receiver's selection.

Recently, Camenisch, Neven, and shelat [CNS07] proposed practical OT protocols that are secure according to a full-simulation definition. They provided constructions that are based on unique blind signatures in the random oracle model, and give standard model constructions using bilinear groups. Subsequently, Green and Hohenberger [GH07] realized simulation-secure OT under a weaker set of static assumptions on bilinear groups, and developed a framework for constructing OT from blind IBE with efficient proofs of knowledge. One may securely plug these schemes into an MPC protocol by using *sequential* OT invocations, but this results in a protocol with a large number of rounds. Unfortunately, these schemes are not known to be secure when composed (say) in parallel, due to their use of rewinding in the simulation.

In summary, the literature still lacks an oblivious transfer protocol enjoying all three of the following desirable properties:

1. **Secure and composable:** Oblivious transfer protocols should be proven secure according to a full simulation-based definition, and should securely compose (e.g., in parallel) with each

other and with other protocols such as MPC.

2. **Efficient:** OT protocols should be efficient in the local computations of the parties, the communication complexity and number of rounds, and their usage of any external resources.

3. **Generally realizable:** It is important to have a general framework for oblivious transfer that is realizeable under a variety of cryptographic assumptions. This justifies the generality of the approach, and can also protect against future advances in cryptanalysis, such as improved algorithms for specific problems or the development of a large-scale quantum computer.

1.1 Our Approach

We present a new framework for constructing efficient, secure, and generally realizable oblivious transfer protocols. We work in Canetti's universal composability (UC) model [Can01] with static corruptions.

Our protocols are based on a new abstraction that we call a *dual-mode* cryptosystem. Such a system starts with a setup phase that produces a common reference string (CRS), which is made available to all parties (we discuss this assumption in more detail below). The CRS is created acccording to one of two possible *modes*, called the *extraction* mode and the *decryption* mode.

The OT protocol is very simple, and works roughly as follows (in both modes): the receiver uses its selection bit (and the CRS) to generate a "base" public key and secret key, and delivers the public key to the sender. The sender uses this key, along with the CRS, to compute two "derived" public keys. It encrypts each of its values under the respective key, and delivers the ciphertexts to the receiver. Finally, the receiver uses its secret key to decrypt the appropriate value. Note that the protocol consists of only two rounds (one message in each direction).

A dual-mode system has three main security properties. When the system is set up in *extraction* mode, at least one of the sender's values is *statistically* hidden by the encryption. Moreover, a simulator that produces the CRS obtain a trapdoor that allows it to extract, from the receiver's message, which of the values will be hidden. This allows us to prove *unconditional* security against even an unbounded cheating receiver.

When the system is in *decryption* mode, the honest receiver's selection bit is instead statistically hidden by the base key. On the other hand, both ciphertexts will contain full information about their respective encrypted values; furthermore, the creator of the CRS can use a trapdoor to decrypt both ciphertexts. This allows us to prove unconditional security against even an unbounded cheating sender.

Finally, a dual-mode system has the property that no bounded adversary can distinguish between extraction mode and decryption mode (note that this is the only computational property required of the system). The protocol can therefore provide unconditional security for *either* the sender or the receiver, depending on the chosen mode. Computational security for the other party follows directly from statistical security in the other mode, and the indistinguishability of modes. (Groth *et al.* [GOS06] used a similar "parameter-switching" argument in the context of non-interactive zero knowledge.)

Our dual-mode abstraction has a number of nice properties and consequences. First, the definition is quite simple: for any candidate construction, we need only show three simple security properties, only one of which depends on a computational assumption. Second, the symmetric nature of our definition yields protocols that can provide unconditional security for either the sender or receiver, simply by choosing the appropriate distribution of the CRS. Third, by working in the UC framework, we automatically get security under arbitrary composition, e.g., in parallel and/or with an MPC protocol. Fourth, we are able to efficiently realize our abstraction under any one of several standard cryptographic assumptions, including the DDH assumption in cyclic groups, the quadratic residuosity (QR) assumption, and even *worst-case* assumptions on lattices.

The advantage of secure composition deserves some additional discussion. In terms of efficiency, the main nice consequence is that we can securely compose several invocations *in parallel* with each other. Because our protocols are only two rounds, we immediately obtain (using standard tools and techniques) *round-optimal* secure two-party computation in the UC model (cf. [HK07]). Another benefit of composition in the UC model is that we actually prove our protocol secure with *joint state* [CR03], i.e., every invocation of the protocol between the same two parties can use the same CRS.

Of course, our entire system depends upon a trusted setup of the CRS. We believe that in context, this assumption is reasonable (or even quite mild). First, it is known that secure oblivious transfer in the plain UC model *requires* some type of setup assumption [CF01]. Second, our protocols can use the same CRS for an *unbounded* number of OT invocations between two specific parties, meaning that the CRS can be produced once and for all time. Third, several of our instantiations require only a common *random* string, which may be obtainable without a trusted party via, e.g., a natural process.

1.2 Techniques

In constructing dual-mode systems from various assumptions, we build upon several existing public key cryptosystems that all have what we call *message-lossy* public keys. Their defining property is that a ciphertext produced under such a key carries *no information* (even statistically) about the encrypted message. More precisely, the distribution of an encryption of m_0 under pk is *statistically* close to that of an encryption of m_1. Several cryptosystems, ranging from a non-IBE variant of Cocks' scheme [Coc01] to those based on lattices [AD97, Reg04, Reg05], use message-lossy keys as a key component of their security proofs.

In our dual-mode constructions, message-lossy keys actually play a crucial role in the *protocol itself.* In extraction mode, the CRS will be constructed to guarantee that at least one of the derived keys is message-lossy; this ensures statistical security for the sender. In decryption mode, neither of the public keys will be truly message-lossy, but the indistinguishability of modes will guarantee that at least one value is *computationally* hidden from a bounded cheating receiver.

For our constructions based on DDH and QR, we obtain dual-mode systems via relatively straightforward abstraction and modification of the Naor-Pinkas OT protocol [NP01] and Cocks' identity-based cryptosystem [Coc01], respectively. For these systems, we have a precise characterization of message-lossy keys. For example, a public key that is a DDH tuple produces decryptable ciphertexts, whereas a public key that is not a DDH tuple is message-lossy.

Our lattice-based constructions build off of a cryptosystem of Regev [Reg05], and are more subtle. In particular, we do not have an *explicit* characterization of message-lossy keys for this cryptosystem. However, we are able to choose the parameters so that all but an *extremely small* fraction of keys are message-lossy. By a counting argument, we can then guarantee that for almost all choices of the CRS (in extraction mode), the at least one of the derived keys is message-lossy. See Section 7 for details.

As an additional contribution of independent interest, we give a multi-bit version of Regev's cryptosystem whose time and space efficiency is improved by a linear factor in the security parameter n. This yields a very efficient cryptosystem, in which the amortized runtime per message bit is only $\tilde{O}(n)$ bit operations, and the size of the ciphertext is only a constant factor larger than the message. The public key size and underlying lattice assumptions remain essentially unchanged.

1.3 Organization

In Section 2 we give background that is relevant to the entire paper. In Section 3 we define our new abstraction called dual-mode encryption. In Section 4 we present our OT protocol based on dual-mode encryption and prove it secure in the UC framework. In Sections 5 and 6 we construct dual-mode encryption based on the DDH and QR assumptions, respectively. In Section 7 we describe our efficient lattice-based cryptosystem and construct a dual-mode cryptosystem from it.

2 Preliminaries

2.1 Notation

We let $\mathbb{N}$ denote the natural numbers. For $n \in \mathbb{N}$, $[n]$ denotes the set $\{1, \ldots, n\}$. We use standard O, Ω, o, and ω notation to classify the growth of functions. We say that $f = \tilde{O}(g)$ if $f = O(g \log^c n)$ for some fixed constant c. We let $\text{poly}(n)$ denote an unspecified function $f(n) = O(n^c)$ for some constant c.

The security parameter will be denoted by n throughout the paper. We let $\mathsf{negl}(n)$ denote some unspecified function $f(n)$ such that $f = o(n^{-c})$ for *every* fixed constant c, saying that such a function is *negligible* (in n). We say that a probability is *overwhelming* if it is $1 - \mathsf{negl}(n)$.

2.2 The Universal Composability Framework (UC)

We work in the standard universal composability framework of Canetti [Can01] with static corruptions of parties. For consistency, we use the definition of computational indistinguishability, denoted by $\stackrel{c}{\approx}$, from that work. The UC framework defines a probabilistic poly-time (PPT) *environment* machine $\mathcal{Z}$ that oversees the execution of a protocol in one of two worlds. The "ideal world" execution involves "dummy parties" (some of whom may be corrupted by an *ideal adversary* $\mathcal{S}$) interacting with a *functionality* $\mathcal{F}$. The "real world" execution involves PPT parties (some of whom may be corrupted by a PPT *real world adversary* $\mathcal{A}$) interacting only with each other in some protocol π. We refer to [Can01] for a detailed description of the executions, and a definition of the real world ensemble $\mathsf{EXEC}_{\pi,\mathcal{A},\mathcal{Z}}$ and the ideal world ensemble $\mathsf{IDEAL}_{\mathcal{F},\mathcal{S},\mathcal{Z}}$. The notion of a protocol π *securely emulating* a functionality $\mathcal{F}$ is as follows:

Definition 2.1. Let $\mathcal{F}$ be a functionality. A protocol π is said to UC-realize $\mathcal{F}$ if for any adversary $\mathcal{A}$, there exists a simulator $\mathcal{S}$ such that for all environments $\mathcal{Z}$,

$$\mathsf{IDEAL}_{\mathcal{F},\mathcal{S},\mathcal{Z}} \stackrel{c}{\approx} \mathsf{EXEC}_{\pi,\mathcal{A},\mathcal{Z}}.$$

The common reference string functionality $\mathcal{F}_{\text{CRS}}^{\mathcal{D}}$ produces a string with a fixed distribution that can be sampled by a PPT algorithm $\mathcal{D}$. Its definition is given in Figure 1.

Oblivious Transfer (OT) is a two-party functionality, involving a sender $\mathbf{S}$ with input x_0, x_1 and a receiver $\mathbf{R}$ with an input $\sigma \in \{0, 1\}$. The receiver $\mathbf{R}$ learns x_σ (and nothing else), and the

Functionality $\mathcal{F}_{\mathbf{CRS}}^{\mathcal{D}}$

$\mathcal{F}_{\mathrm{CRS}}^{\mathcal{D}}$ runs with parties $P_1, \ldots, P_n$ and is parametrized by an algorithm $\mathcal{D}$.

- When receiving a message (sid, P_i, P_j) from P_i, let $\mathsf{crs} \leftarrow \mathcal{D}(1^n)$, send $(\mathsf{sid}, \mathsf{crs})$ to P_i and send (crs, P_i, P_j) to the adversary. Next, when receiving (sid, P_i, P_j) from P_j (and only P_j), send $(\mathsf{sid}, \mathsf{crs})$ to P_j and to the adversary, and halt.

Figure 1: The common reference string functionality $\mathcal{F}_{\mathrm{CRS}}^{\mathcal{D}}$ [CR03].

sender **S** learns nothing at all. These requirements are captured by the specification of the OT functionality $\mathcal{F}_{\mathrm{OT}}$ from [CLOS02], given in Figure 2

Functionality $\mathcal{F}_{\mathbf{OT}}$

$\mathcal{F}_{\mathrm{OT}}$ interacts with a sender **S** and a receiver **R**.

- Upon receiving a message $(\mathsf{sid}, \mathsf{sender}, x_0, x_1)$ from **S**, where each $x_i \in \{0,1\}^\ell$, store (x_0, x_1) (The lengths of the strings ℓ is fixed and known to all parties).
- Upon receiving a message $(\mathsf{sid}, \mathsf{receiver}, \sigma)$ from **R**, check if a $(\mathsf{sid}, \mathsf{sender}, \ldots)$ message was previously sent. If yes, send (sid, x_σ) to **R** and (sid) to the adversary $\mathcal{S}$ and halt. If not, send nothing to **R** (but continue running).

Figure 2: The oblivious transfer functionality $\mathcal{F}_{\mathrm{OT}}$ [CLOS02].

Our OT protocols operate in the common reference string model, or, in the terminology of [Can01], the $\mathcal{F}_{\mathrm{CRS}}$-hybrid model. For efficiency, we would like to reuse the same common reference string for distinct invocations of oblivious transfer whenever possible. As described in [CR03], this can be achieved by designing a protocol for the *multi-session extension* $\hat{\mathcal{F}}_{\mathrm{OT}}$ of the OT functionality $\mathcal{F}_{\mathrm{OT}}$. Intuitively, $\hat{\mathcal{F}}_{\mathrm{OT}}$ acts as a "wrapper" around any number of independent executions of $\mathcal{F}_{\mathrm{OT}}$, and coordinates their interactions with the parties via subsessions (specified by a parameter ssid) of a single session (specified by sid).

The *UC theorem with joint state* (JUC theorem) [CR03] says that any protocol π operating in the $\mathcal{F}_{\mathrm{OT}}$-hybrid model can be securely emulated in the real world by appropriately composing π with a *single* execution of a protocol ρ implementing $\hat{\mathcal{F}}_{\mathrm{OT}}$. This single instance of ρ might use fewer resources (such as common reference strings) than several independent invocations of some other protocol that only realizes $\mathcal{F}_{\mathrm{OT}}$; in fact, the protocols ρ that we specify will do exactly this.

3 Dual-Mode Encryption

Here we describe our new abstraction, called a *dual-mode* cryptosystem. Such a cryptosystem is initialized in a trusted setup phase, which produces a common string crs known to all parties along with some auxiliary "trapdoor" information t (which is only used in the security proof). The string crs may be either uniformly random or from a prescribed distribution, depending on the concrete scheme. The cryptosystem can be set up in one of two possible *modes*: *extraction* mode or *decryption* mode. The first crucial security property of a dual-mode cryptosystem is that no (efficient) adversary can distinguish, given the crs, between the modes.

Once the system has been set up and the crs made available, the system operates much like a standard public-key cryptosystem, but with an added notion that we call *encryption branches.* When a party generates a public/secret key pair, it specifies to the key generation algorithm a *message-preserving* branch $\sigma \in \{0,1\}$; the resulting secret key sk will correspond to that branch of the public key pk. When encrypting a message under pk, the encrypter similarly specifies a branch $b \in \{0,1\}$ on which to encrypt the message. Essentially, ciphertexts produced on branch $b = \sigma$ can be decrypted using sk, while those on the other branch cannot. Precisely what this means depends on the mode of the system.

In extraction mode, branch $b \neq \sigma$ is what we call *message-lossy.* That is, encrypting on branch b *loses all information* about the encrypted message — not only in the sense of semantic security, but *statistically.* Moreover, the extraction mode trapdoor allows one to find a message-lossy branch of *any* given public key — even a *malformed* one that could never be produced by the key generator.

In decryption mode, however, the trapdoor allows one to circumvent lossiness. Specifically, given the trapdoor, one can generate a public key pk and corresponding secret keys sk_0, sk_1 that enable decryption on both branches 0 and 1 (respectively). Moreover, the distribution of the key pair (pk, sk_σ) is *statistically close* to that of an "honestly-generated" key pair with decryption branch σ, for both values of $\sigma \in \{0,1\}$.

We now proceed more formally. A dual-mode cryptosystem consists of a tuple of probabilistic algorithms $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{FindLossy}, \mathsf{TrapKeyGen})$ having the following interfaces:

- $\mathsf{Setup}(1^n, \mu)$, given security parameter n and mode $\mu \in \{0,1\}$, outputs (crs, t). The crs is a common string for the remaining algorithms, and t is a trapdoor value that enables either the $\mathsf{FindLossy}$ or $\mathsf{TrapKeyGen}$ algorithm, depending on the selected mode.

  For notational convenience, we define a separate extraction mode setup algorithm $\mathsf{SetupExt}(\cdot) := \mathsf{Setup}(\cdot, 0)$ and a decryption mode setup algorithm $\mathsf{SetupDec}(\cdot) := \mathsf{Setup}(\cdot, 1)$.

  All the remaining algorithms take crs as their first input, but for notational clarity, we often omit it in their lists of arguments.

- $\mathsf{KeyGen}(\sigma)$, given a desired message-preserving branch value $\sigma \in \{0,1\}$, outputs (pk, sk) where pk is a public encryption key and sk is a corresponding secret decryption key for messages encrypted on branch σ.

- $\mathsf{Enc}(pk, b, m)$, given a public key pk, a branch value $b \in \{0,1\}$, and a message $m \in \{0,1\}^\ell$, outputs a ciphertext c encrypted on branch b.

- $\mathsf{Dec}(sk, c)$, given a secret key sk and a ciphertext c, outputs a message $m \in \{0,1\}^\ell$.

- $\mathsf{FindLossy}(t, pk)$, given a trapdoor t and some (possibly even malformed) public key pk, outputs a branch value $b \in \{0,1\}$ corresponding to a message-lossy branch of pk.

- $\mathsf{TrapKeyGen}(t)$, given a trapdoor t, outputs (pk, sk_0, sk_1), where pk is a public encryption key and sk_0, sk_1 are corresponding secret decryption keys for branches 0 and 1, respectively.

We now describe the required security properties.

Definition 3.1 (Dual-Mode Encryption)**.** A *dual-mode cryptosystem* is a tuple of algorithms described above that satisfy the following properties:

1. **Completeness for message-preserving branch:** For every $\mu \in \{0,1\}$, every $(\mathsf{crs}, t) \leftarrow \mathsf{Setup}(1^n, \mu)$, every $\sigma \in \{0,1\}$, every $(pk, sk) \leftarrow \mathsf{KeyGen}(\sigma)$, and every $m \in \{0,1\}^\ell$, decryption is correct on the message-preserving branch σ, i.e., $\mathsf{Dec}(sk, \mathsf{Enc}(pk, \sigma, m)) = m$.

   It is also suffices for decryption to be correct with overwhelming probability over the randomness of the entire experiment.

2. **Indistinguishability of modes:** the first outputs of $\mathsf{SetupExt}$ and $\mathsf{SetupDec}$ are computationally indistinguishable, i.e., $\mathsf{SetupExt}_1(1^n) \overset{c}{\approx} \mathsf{SetupDec}_1(1^n)$.

3. **(Extraction mode) Trapdoor extraction of message-lossy branch:** For every $(\mathsf{crs}, t) \leftarrow \mathsf{SetupExt}(1^n)$ and *every* (possibly malformed) pk, $\mathsf{FindLossy}(t, pk)$ outputs a branch value $b \in \{0,1\}$ such that $\mathsf{Enc}(pk, b, \cdot)$ is message-lossy. Namely, for every $m_0, m_1 \in \{0,1\}^\ell$, $\mathsf{Enc}(pk, b, m_0) \overset{s}{\approx} \mathsf{Enc}(pk, b, m_1)$.

4. **(Decryption mode) Trapdoor generation of keys decryptable on both branches:** For every $(\mathsf{crs}, t) \leftarrow \mathsf{SetupDec}(1^n)$, $\mathsf{TrapKeyGen}(t)$ outputs (pk, sk_0, sk_1) such that for every $\sigma \in \{0,1\}$, $(pk, sk_\sigma) \overset{s}{\approx} \mathsf{KeyGen}(\sigma)$.

It is straightforward to generalize these definitions to larger sets $\{0,1\}^k$ of branches, for $k > 1$ (in this generalization, $\mathsf{FindLossy}$ would return $2^k - 1$ different branches that are all message-lossy). Such a dual-mode cryptosystem would yield a 1-out-of-2^k oblivious transfer in an analogous way. All of our constructions can be suitably modified to satisfy the generalized definition; for simplicity, we will concentrate on the branch set $\{0,1\}$ throughout the paper, briefly noting inline how to generalize each construction.

4 Oblivious Transfer Protocol

Here we construct a protocol dm that emulates the multi-session functionality $\hat{\mathcal{F}}_{\mathrm{OT}}$ functionality in the $\mathcal{F}_{\mathrm{CRS}}$-hybrid model. Let $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{FindLossy}, \mathsf{TrapKeyGen})$ be a dual-mode cryptosystem. The dm protocol is given in Figure 3.

The protocol actually can operate in one of two *modes*, depending only on the distribution of the CRS that is used. In the extraction mode, the receiver's security is computational and the sender's security is *statistical*, i.e., security is guaranteed even against an *unbounded* cheating receiver. In decryption mode, the security properties are reversed: the sender enjoys computational security, while the receiver's security is statistical (i.e., holds against an unbounded cheating sender).

To implement the two modes, we define two different instantiations of $\mathcal{F}_{\mathrm{CRS}}^{\mathcal{D}}$ that produce common strings according to the appropriate setup algorithm: $\mathcal{F}_{\mathrm{CRS}}^{\mathsf{ext}}$ uses $\mathcal{D} = \mathsf{SetupExt}_1$, and $\mathcal{F}_{\mathrm{CRS}}^{\mathsf{dec}}$ uses $\mathcal{D} = \mathsf{SetupDec}_1$.

Theorem 4.1. *Let* $\mathsf{mode} \in \{\mathsf{ext}, \mathsf{dec}\}$. *Protocol* $\mathsf{dm}^{\mathsf{mode}}$ *securely realizes the functionality* $\hat{\mathcal{F}}_{OT}$ *in the* $\mathcal{F}_{CRS}^{\mathsf{mode}}$*-hybrid model.*

For $\mathsf{mode} = \mathsf{ext}$*, the sender's security is unconditional and the receiver's security is computational; for* $\mathsf{mode} = \mathsf{dec}$*, the security properties are reversed.*

Proof. Given all the properties of a dual-mode cryptosystem, the proof is conceptually quite straightforward. There is a direct correspondence between completeness and the case that neither party is corrupted, between trapdoor extraction and statistical security for the sender, and

Protocol $\mathsf{dm}^{\mathsf{mode}}$ for Oblivious Transfer

The $\mathsf{dm}^{\mathsf{mode}}$ protocol is parameterized by $\mathsf{mode} \in \{\mathsf{ext}, \mathsf{dec}\}$ indicating the type of crs to be used.

SENDER INPUT: $(\mathsf{sid}, \mathsf{ssid}, x_0, x_1)$, where $x_0, x_1 \in \{0,1\}^\ell$.

RECEIVER INPUT: $(\mathsf{sid}, \mathsf{ssid}, \sigma)$, where $\sigma \in \{0,1\}$.

When activated with their inputs, the sender $\mathbf{S}$ queries $\mathcal{F}_{\mathrm{CRS}}^{\mathsf{mode}}$ with $(\mathsf{sid}, \mathbf{S}, \mathbf{R})$ and gets back $(\mathsf{sid}, \mathsf{crs})$. The receiver $\mathbf{R}$ then queries $\mathcal{F}_{\mathrm{CRS}}^{\mathsf{mode}}$ with $(\mathsf{sid}, \mathbf{S}, \mathbf{R})$ and gets $(\mathsf{sid}, \mathsf{crs})$.

$\mathbf{R}$ computes $(pk, sk) \leftarrow \mathsf{KeyGen}(\mathsf{crs}, \sigma)$, sends $(\mathsf{sid}, \mathsf{ssid}, pk)$ to $\mathbf{S}$, and stores $(\mathsf{sid}, \mathsf{ssid}, sk)$.

$\mathbf{S}$ gets $(\mathsf{sid}, \mathsf{ssid}, pk)$ from $\mathbf{R}$, computes $y_b \leftarrow \mathsf{Enc}(pk, b, x_b)$ for each $b \in \{0,1\}$, and sends $(\mathsf{sid}, \mathsf{ssid}, y_0, y_1)$ to $\mathbf{R}$.

$\mathbf{R}$ gets $(\mathsf{sid}, \mathsf{ssid}, y_0, y_1)$ from $\mathbf{S}$ and outputs $(\mathsf{sid}, \mathsf{ssid}, \mathsf{Dec}(sk, y_\sigma))$, where $(\mathsf{sid}, \mathsf{ssid}, sk)$ was stored above.

Figure 3: The protocol for realizing $\hat{\mathcal{F}}_{\mathrm{OT}}$.

between trapdoor decryption and statistical security for the receiver. The indinstinguishability of modes will establish computational security for the appropriate party in the protocol. We now proceed more formally.

Let $\mathcal{A}$ be a static adversary that interacts with the parties $\mathbf{S}$ and $\mathbf{R}$ running the $\mathsf{dm}^{\mathsf{mode}}$ protocol. We will construct an ideal world adversary (simulator) $\mathcal{S}$ interacting with the ideal functionality $\hat{\mathcal{F}}_{\mathrm{OT}}$, such that no environment $\mathcal{Z}$ can distinguish an interaction with $\mathcal{A}$ in the above protocol from an interaction with $\mathcal{S}$ in the ideal world. Recall that $\mathcal{S}$ interacts with both the ideal functionality $\hat{\mathcal{F}}_{\mathrm{OT}}$ and the environment $\mathcal{Z}$.

$\mathcal{S}$ starts by invoking a copy of $\mathcal{A}$ and running a simulated interaction of $\mathcal{A}$ with $\mathcal{Z}$ and the players $\mathbf{S}$ and $\mathbf{R}$. More specifically, $\mathcal{S}$ works as follows:

Simulating the communication with $\mathcal{Z}$: Every input value that $\mathcal{S}$ receives from $\mathcal{Z}$ is written into the adversary $\mathcal{A}$'s input tape (as if coming from $\mathcal{A}$'s environment). Every output value written by $\mathcal{A}$ on its output tape is copied to $\mathcal{S}$'s own output tape (to be read by the environment $\mathcal{Z}$).

Simulating the case when only the receiver $\mathbf{R}$ is corrupted: Regardless of the mode of the protocol, $\mathcal{S}$ does the following. Run the extraction mode setup algorithm, letting $(\mathsf{crs}, t) \leftarrow \mathsf{SetupExt}(1^n)$. When the parties query the ideal functionality $\mathcal{F}_{\mathrm{CRS}}^{\mathsf{mode}}$, return $(\mathsf{sid}, \mathsf{crs})$ to them. (Note that when $\mathsf{mode} = \mathsf{ext}$, the crs thus returned is identically distributed to the one returned by $\mathcal{F}_{\mathrm{CRS}}^{\mathsf{mode}}$, whereas when $\mathsf{mode} = \mathsf{dec}$, the simulated crs has a different distribution from the one returned by $\mathcal{F}_{\mathrm{CRS}}^{\mathsf{mode}}$ in the protocol).

When $\mathcal{A}$ produces a protocol message $(\mathsf{sid}, \mathsf{ssid}, pk)$, $\mathcal{S}$ lets $b \leftarrow \mathsf{FindLossy}(\mathsf{crs}, t, pk)$. $\mathcal{S}$ then sends $(\mathsf{sid}, \mathsf{ssid}, \mathsf{receiver}, 1-b)$ to the ideal functionality $\hat{\mathcal{F}}_{\mathrm{OT}}$, receives the output $(\mathsf{sid}, \mathsf{ssid}, x_{1-b})$, and stores it along with the value b.

When the dummy $\mathbf{S}$ is activated for subsession $(\mathsf{sid}, \mathsf{ssid})$, $\mathcal{S}$ looks up the corresponding b and x_{1-b}, computes $y_{1-b} \leftarrow \mathsf{Enc}(pk, 1-b, x_{1-b})$ and $y_b \leftarrow \mathsf{Enc}(pk, b, 0^\ell)$ and sends the adversary $\mathcal{A}$ the message $(\mathsf{sid}, \mathsf{ssid}, y_0, y_1)$ as if it were from $\mathbf{S}$.

Simulating the case when only the sender S is corrupted: Regardless of the mode of the protocol, $\mathcal{S}$ does the following. Run the decryption mode setup algorithm, letting $(\mathsf{crs}, t) \leftarrow \mathsf{SetupDec}(1^n)$. When the parties query the ideal functionality $\mathcal{F}_{\mathrm{CRS}}^{\mathsf{mode}}$, return $(\mathsf{sid}, \mathsf{crs})$ to them.

When the dummy $\mathbf{R}$ is activated on $(\mathsf{sid}, \mathsf{ssid})$, $\mathcal{S}$ computes $(pk, sk_0, sk_1) \leftarrow \mathsf{TrapKeyGen}(\mathsf{crs}, t)$, sends $(\mathsf{sid}, \mathsf{ssid}, pk)$ to $\mathcal{A}$ as if from $\mathbf{R}$, and stores $(\mathsf{sid}, \mathsf{ssid}, pk, sk_0, sk_1)$. When $\mathcal{A}$ replies with a message $(\mathsf{sid}, \mathsf{ssid}, y_0, y_1)$, $\mathcal{S}$ looks up the corresponding (pk, sk_0, sk_1), computes $x_b \leftarrow \mathsf{Dec}(sk_b, y_b)$ for each $b \in \{0, 1\}$ and sends to $\hat{\mathcal{F}}_{\mathrm{OT}}$ the message $(\mathsf{sid}, \mathsf{ssid}, \mathsf{sender}, x_0, x_1)$.

Simulating the remaining cases: When both parties are corrupted, the simulator $\mathcal{S}$ just runs $\mathcal{A}$ internally (who itself generates the messages from both $\mathbf{S}$ and $\mathbf{R}$).

When neither party is corrupted, $\mathcal{S}$ internally runs the honest $\mathbf{R}$ on input $(\mathsf{sid}, \mathsf{ssid}, \sigma = 0)$ and honest $\mathbf{S}$ on input $(\mathsf{sid}, \mathsf{ssid}, x_0 = 0^\ell, x_1 = 0^\ell)$, activating the appropriate algorithm when the corresponding dummy party is activated in the ideal execution, and delivering all messages between its internal $\mathbf{R}$ and $\mathbf{S}$ to $\mathcal{A}$.

The proof will be completed via the following claims, shown below:

1. *(Claim 4.2, statistical security for* $\mathbf{S}$ *in extraction mode.)* When $\mathcal{A}$ corrupts the receiver $\mathbf{R}$,

$$\mathsf{IDEAL}_{\hat{\mathcal{F}}_{\mathrm{OT}}, \mathcal{S}, \mathcal{Z}} \stackrel{s}{\approx} \mathsf{EXEC}_{\mathsf{dm}^{\mathsf{ext}}, \mathcal{A}, \mathcal{Z}}.$$

2. *(Claim 4.3, statistical security for* $\mathbf{R}$ *in decryption mode.)* When $\mathcal{A}$ corrupts the receiver $\mathbf{S}$,

$$\mathsf{IDEAL}_{\hat{\mathcal{F}}_{\mathrm{OT}}, \mathcal{S}, \mathcal{Z}} \stackrel{s}{\approx} \mathsf{EXEC}_{\mathsf{dm}^{\mathsf{dec}}, \mathcal{A}, \mathcal{Z}}.$$

3. *(Claim 4.4, parameter switching.)* For any protocol π^{mode} in the $\mathcal{F}_{\mathrm{CRS}}^{\mathsf{mode}}$-hybrid model, any adversary $\mathcal{A}$ and any environment $\mathcal{Z}$,

$$\mathsf{EXEC}_{\pi^{\mathsf{ext}}, \mathcal{A}, \mathcal{Z}} \stackrel{c}{\approx} \mathsf{EXEC}_{\pi^{\mathsf{dec}}, \mathcal{A}, \mathcal{Z}}.$$

We now complete the proof as follows. Consider the protocol $\mathsf{dm}^{\mathsf{ext}}$. When $\mathcal{A}$ corrupts $\mathbf{R}$, by item 1 above we have statistical security for $\mathbf{S}$ (whether or not $\mathbf{S}$ is corrupted). When $\mathcal{A}$ corrupts $\mathbf{S}$, by items 2 and 3 above we have

$$\mathsf{IDEAL}_{\hat{\mathcal{F}}_{\mathrm{OT}}, \mathcal{S}, \mathcal{Z}} \stackrel{s}{\approx} \mathsf{EXEC}_{\mathsf{dm}^{\mathsf{dec}}, \mathcal{A}, \mathcal{Z}} \stackrel{c}{\approx} \mathsf{EXEC}_{\mathsf{dm}^{\mathsf{ext}}, \mathcal{A}, \mathcal{Z}},$$

which implies computational security for $\mathbf{R}$.

It remains to show computational security when neither the sender nor the receiver is corrupted. Let $\mathsf{EXEC}_{\mathsf{dm}^{\mathsf{ext}}, \mathcal{A}, \mathcal{Z}}(x_0, x_1, b)$ (resp, $\mathsf{EXEC}_{\mathsf{dm}^{\mathsf{dec}}, \mathcal{A}, \mathcal{Z}}(x_0, x_1, b)$) denote the output of an environment in the protocol $\mathsf{dm}^{\mathsf{ext}}$ (resp, $\mathsf{dm}^{\mathsf{dec}}$) that sets the inputs of the sender $\mathbf{S}$ to be (x_0, x_1) and the input of the receiver $\mathbf{R}$ to be the bit b. The following sequence of hybrids establishes what we want.

$$\begin{aligned} &\mathsf{EXEC}_{\mathsf{dm}^{\mathsf{ext}}, \mathcal{A}, \mathcal{Z}}(x_0, x_1, 1) \stackrel{s}{\approx} \mathsf{EXEC}_{\mathsf{dm}^{\mathsf{ext}}, \mathcal{A}, \mathcal{Z}}(0^\ell, x_1, 1) \stackrel{c}{\approx} \\ &\quad \mathsf{EXEC}_{\mathsf{dm}^{\mathsf{ext}}, \mathcal{A}, \mathcal{Z}}(0^\ell, x_1, 0) \stackrel{s}{\approx} \mathsf{EXEC}_{\mathsf{dm}^{\mathsf{ext}}, \mathcal{A}, \mathcal{Z}}(0^\ell, 0^\ell, 0) \end{aligned}$$

The first two and the last two experiments are statistically indistinguishable because of the message-lossy property in the extraction mode, and the second and third experiments are computationally indistinguishable because of the computational hiding of the receiver's selection bit. The first experiment corresponds to the real world execution, whereas the last experiment is what the simulator runs. Furthermore, by the completeness of the dual-mode cryptosystem, the first experiment is statistically indistinguishable from the ideal world exection with inputs (x_0, x_1, b).

The proof of security for protocol $\mathsf{dm}^{\mathsf{dec}}$ follows symmetrically, and we are done. $\square$

Claim 4.2. *If the adversary $\mathcal{A}$ corrupts the receiver $\mathbf{R}$ in an execution of $\mathsf{dm}^{\mathsf{ext}}$, then we have*

$$\mathit{IDEAL}_{\hat{\mathcal{F}}_{OT},\mathcal{S},\mathcal{Z}} \stackrel{s}{\approx} \mathit{EXEC}_{\mathsf{dm}^{\mathsf{ext}},\mathcal{A},\mathcal{Z}}.$$

Proof. The real world execution can be seen as a game that proceeds as follows, interacting with the environment $\mathcal{Z}(z)$: first, $\mathsf{crs} \leftarrow \mathsf{SetupExt}_1(1^n)$. Then the environment arbitrarily schedules some number of subsessions, where in each subsession, $\mathcal{Z}$ chooses an arbitrary pk and arbitrary inputs (x_0, x_1) for the honest sender $\mathbf{S}$, who sends $y_b \leftarrow \mathsf{Enc}(\mathsf{crs}, pk, b, x_b)$ for each $b \in \{0,1\}$ to $\mathcal{Z}$.

The ideal world execution proceeds similarly; however, first $(\mathsf{crs}, t) \leftarrow \mathsf{SetupExt}(1^n)$ (but only crs is visible to $\mathcal{Z}$). Then the environment arbitrarily schedules subsessions, where in each subsession $\mathcal{Z}$ produces an arbitrary pk and arbitrary inputs (x_0, x_1) for the dummy $\mathbf{S}$. The simulator $\mathcal{S}$ runs $b \leftarrow \mathsf{FindLossy}(t, pk)$ and learns x_{1-b} from the functionality $\hat{\mathcal{F}}_{\mathrm{OT}}$. It then sends $y_b \leftarrow \mathsf{Enc}(\mathsf{crs}, pk, b, 0^\ell)$ and $y_{1-b} = \mathsf{Enc}(\mathsf{crs}, pk, 1-b, x_{1-b})$ to $\mathcal{Z}$.

The only difference between the two games is therefore in y_b in each subsession. But by trapdoor extraction of a message-lossy branch, we have in the ideal game that $\mathsf{Enc}(\mathsf{crs}, pk, b, 0^\ell) \stackrel{s}{\approx} \mathsf{Enc}(\mathsf{crs}, pk, b, x_b)$. Therefore the two games are statistically indistinguishable. $\square$

Claim 4.3. *If the adversary $\mathcal{A}$ corrupts the sender $\mathbf{S}$ in an execution of $\mathsf{dm}^{\mathsf{dec}}$, then we have that*

$$\mathit{IDEAL}_{\hat{\mathcal{F}}_{OT},\mathcal{S},\mathcal{Z}} \stackrel{s}{\approx} \mathit{EXEC}_{\mathsf{dm}^{\mathsf{dec}},\mathcal{A},\mathcal{Z}}.$$

Proof. The real world execution can be seen as a game that proceeds as follows, interacting with the environment $\mathcal{Z}(z)$: first, $\mathsf{crs} \leftarrow \mathsf{SetupDec}_1(1^n)$. Then the environment arbitrarily schedules some number of subsessions. In each subsession, $\mathcal{Z}$ chooses an input σ for the honest $\mathbf{R}$, who generates $(pk, sk) \leftarrow \mathsf{KeyGen}(\mathsf{crs}, \sigma)$ and sends pk to $\mathcal{Z}$, then $\mathcal{Z}$ produces arbitrary (y_0, y_1) and the honest $\mathbf{R}$ outputs $\mathsf{Dec}(\mathsf{crs}, sk, y_\sigma)$.

The ideal world execution proceeds similarly; however, first $(\mathsf{crs}, t) \leftarrow \mathsf{SetupDec}(1^n)$ (but only crs is visible to $\mathcal{Z}$). Then the environment arbitrarily schedules subsessions, where in each subsession $\mathcal{Z}$ produces arbitrary σ (not known to $\mathcal{S}$), then $\mathcal{S}$ runs $(pk, sk_0, sk_1) \leftarrow \mathsf{TrapKeyGen}(t)$ and gives pk to $\mathcal{Z}$, then $\mathcal{S}$ receives arbitrary (y_0, y_1) from $\mathcal{Z}$. The dummy $\mathbf{R}$ outputs the value $\mathsf{Dec}(\mathsf{crs}, sk_\sigma, y_\sigma)$, acquiring it from the functionality, which was provided the messages $x_b = \mathsf{Dec}(\mathsf{crs}, sk_b, y_b)$ by $\mathcal{S}$.

The only difference between the two games is therefore in the creation of the public and secret keys. However, by trapdoor key generation, $(pk, sk_\sigma) \stackrel{s}{\approx} \mathsf{KeyGen}(\mathsf{crs}, \sigma)$ for any value of crs generated by $\mathsf{SetupDec}$. Therefore the two games are statistically indistinguishable. $\square$

Claim 4.4. *For any protocol π^{mode} in the $\mathcal{F}_{CRS}^{\mathsf{mode}}$-hybrid model, adversary $\mathcal{A}$ and environment $\mathcal{Z}$,*

$$\mathit{EXEC}_{\pi^{\mathsf{ext}},\mathcal{A},\mathcal{Z}} \stackrel{c}{\approx} \mathit{EXEC}_{\pi^{\mathsf{dec}},\mathcal{A},\mathcal{Z}}.$$

Proof. By the indistinguishability of modes in the dual-mode cryptosystem, the output of $\mathcal{F}_{\text{CRS}}^{\mathsf{ext}}$ and $\mathcal{F}_{\text{CRS}}^{\mathsf{dec}}$ are computationally indistinguishable. Environment $\mathcal{Z}$ running protocol π^{mode} can be seen as an efficient algorithm that receives a polynomial number of samples from either $\mathcal{F}_{\text{CRS}}^{\mathsf{ext}}$ or $\mathcal{F}_{\text{CRS}}^{\mathsf{dec}}$. By a standard hybrid argument, the two executions are indistinguishable. $\square$

4.1 Application: Round-Optimal Two-Party Secure Computation

Using standard tools like (non-interactive) zero-knowledge and Yao's garbled circuit method, we can use our OT protocol to obtain round-optimal UC-secure two party computation for non-reactive functionalities. The result gives two-round protocols when only one party receives output, or three-round protocols when both parties receive output. We omit the details.

Theorem 4.5. *There is a* 2*-round (respectively,* 3*-round) protocol that securely realizes any non-reactive functionality* $\mathcal{F}_{2PC}$ *for which only one party (resp., both parties) receives output, in the* $\mathcal{F}_{CRS}$*-hybrid model.*

5 Realization from DDH

5.1 Background

Let $\mathcal{G}$ be a an algorithm that takes as input a security parameter 1^n and outputs a group description $\mathbb{G} = (G, p, g)$, where G is a cyclic group of prime order p and g is a generator of G.

Our construction will make use of groups for which the DDH problem is believed to be hard. The version of the DDH assumption we use is the following: for random *generators* $g, h \in G$ and for *distinct* but otherwise random $a, b \in \mathbb{Z}_p$, the tuples (g, h, g^a, h^a) and (g, h, g^a, h^b) are computationally indistinguishable.[1] This version of the DDH assumption is equivalent to another common form, namely, that $(g, g^a, g^b, g^{ab}) \stackrel{c}{\approx} (g, g^a, g^b, g^c)$ for independent $a, b, c \leftarrow Z_p$, because g^a is a generator and $c \neq ab$ with overwhelming probability.

5.2 Cryptosystem Based on DDH

We start by presenting a cryptosystem based on the hardness of the Decisional Diffie-Hellman problem, which slightly differs from the usual ElGamal cryptosystem in a few ways. The cryptosystem depends on a randomization procedure that we describe below. We note that the algorithm $\mathsf{Randomize}$ we describe below is implicit in the OT protocol of Naor and Pinkas [NP01].

Lemma 5.1 (Randomization)**.** *Let* G *be an arbitrary multiplicative group of prime order* p*. For each* $x \in \mathbb{Z}_p$*, define* $\text{DLOG}_G(x) = \{(g, g^x) : g \in G\}$*. There is a probabilistic algorithm* $\mathsf{Randomize}$ *that takes generators* $g, h \in G$ *and elements* $g', h' \in G$*, and outputs a pair* $(u, v) \in G^2$ *such that:*

- *If* $(g, g'), (h, h') \in \text{DLOG}_G(x)$ *for some* x*, then* (u, v) *is uniformly random in* $\text{DLOG}_G(x)$*.*
- *If* $(g, g') \in \text{DLOG}_G(x)$ *and* $(h, h') \in \text{DLOG}_G(y)$ *for some* $x \neq y$*, then* (u, v) *is uniformly random in* G^2*.*

[1] To be completely formal, the respective ensembles of the two distributions, indexed by the security parameter n, are indistinguishable.

Proof. Define $\mathsf{Randomize}(g, h, g', h')$ to do the following: Choose $s, t \leftarrow \mathbb{Z}_p$ independently and let $u = g^s h^t$ and $v = (g')^s (h')^t$. Output (u, v).

Since g and h are generators of G, we can write $h = g^r$ for some nonzero $r \in \mathbb{Z}_p$. First suppose (g, g') and (h, h') belong to $\textsc{dlog}_G(x)$ for some x. Now, $u = g^s h^t = g^{s+rt}$ is uniformly random in G, since g is a generator of G and s is random in $\mathbb{Z}_p$. Furthermore, $v = (g')^s (h')^t = (g^s h^t)^x = u^x$ and thus, $(u, v) \in \textsc{dlog}_G(x)$.

Now suppose $(g, g') \in \textsc{dlog}_G(x)$ and $(h, h') \in \textsc{dlog}_G(y)$ for some $x \neq y$. Then $u = g^s h^t = g^{s+rt}$ and $v = g^{xs+ryt}$. Because $r(x - y) \neq 0 \in \mathbb{Z}_p$, the expressions $s + rt$ and $xs + ryt$ are linearly independent combinations of s and t. Therefore, over the choice of $s, t \in \mathbb{Z}_p$, u and v are uniform and independent in G. □

We now describe the basic cryptosystem.

- $\mathsf{DDHKeyGen}(1^n)$: Choose $\mathbb{G} = (G, p, g) \leftarrow \mathcal{G}(1^n)$. The message space of the system is G.

  Choose another generator $h \leftarrow G$ and exponent $x \leftarrow \mathbb{Z}_p$. Let $pk = (g, h, g^x, h^x)$ and $sk = x$. Output (pk, sk).

- $\mathsf{DDHEnc}(pk, m)$: Parse pk as (g, h, g', h'). Let $(u, v) \leftarrow \mathsf{Randomize}(g, h, g', h')$. Output the ciphertext $(u, v \cdot m)$.

- $\mathsf{DDHDec}(sk, c)$: Parse c as (c_0, c_1). Output c_1 / c_0^{sk}.

Now consider a public key pk of the form (g, h, g^x, h^y) for distinct $x, y \in \mathbb{Z}_p$ (and where g, h are generators of G). It follows directly from Lemma 5.1 that $\mathsf{DDHEnc}(pk, \cdot)$ is message-lossy. Namely, for every two messages $m_0, m_1 \in \mathbb{Z}_p$, $\mathsf{DDHEnc}(pk, m_0) \stackrel{s}{\approx} \mathsf{DDHEnc}(pk, m_1)$.

5.3 Dual-Mode Cryptosystem

We now construct a dual-mode encryption scheme based on the hardness of DDH.

- Both $\mathsf{SetupExt}$ and $\mathsf{SetupDec}$ start by choosing $\mathbb{G} = (G, p, g) \leftarrow \mathcal{G}(1^n)$.

  $\mathsf{SetupExt}(1^n)$: Choose random generators $g_0, g_1 \in G$. Choose *distinct nonzero* exponents $x_0, x_1 \leftarrow \mathbb{Z}_p$. Let $h_b = g_b^{x_b}$ for $b \in \{0, 1\}$. Let $\mathsf{crs} = (g_0, h_0, g_1, h_1)$ and $t = (x_0, x_1)$. Output (crs, t).

  $\mathsf{SetupDec}(1^n)$: Choose a random generator $g_0 \in G$, a random nonzero $y \in \mathbb{Z}_p$, and let $g_1 = g_0^y$. Choose a random nonzero exponent $x \in \mathbb{Z}_p$. Let $h_b = g_b^x$ for $b \in \{0, 1\}$, let $\mathsf{crs} = (g_0, h_0, g_1, h_1)$ and $t = y$. Output (crs, t).

  In the following, all algorithms are implicitly provided the crs and parse it as (g_0, h_0, g_1, h_1).

- $\mathsf{KeyGen}(\sigma)$: Choose $r \leftarrow \mathbb{Z}_p$. Let $g = g_\sigma^r$, $h = h_\sigma^r$ and $pk = (g, h)$. Let $sk = r$. Output (pk, sk).

- $\mathsf{Enc}(pk, b, m)$: Parse pk as (g, h). Let $pk_b = (g_b, h_b, g, h)$. Output $\mathsf{DDHEnc}(pk_b, m)$ as the encryption of m on branch b.

- $\mathsf{Dec}(sk, c)$: Output $\mathsf{DDHDec}(sk, c)$.

- $\mathsf{FindLossy}(t, pk)$: Parse the extraction mode trapdoor t as (x_0, x_1) where $x_0 \neq x_1$. Parse the public key pk as (g, h). If $h \neq g^{x_0}$, then output $b = 0$ as a (candidate) message-lossy branch. Otherwise, we have $h = g^{x_0} \neq g^{x_1}$ because $x_0 \neq x_1$, so output $b = 1$ as a (candidate) message-lossy branch.

- $\mathsf{TrapKeyGen}(t)$: Parse the decryption mode trapdoor t as a nonzero $y \in \mathbb{Z}_p$. Pick a random $r \leftarrow \mathbb{Z}_p$ and compute $pk = (g_0^r, h_0^r)$ and output $(pk, r, r/y)$.

We remark that $\mathsf{SetupExt}$ actually produces a crs that is statistically close to a common *random* (not reference) string, because it consists of four generators that do not comprise a DDH tuple.

Theorem 5.2. *The above scheme is a dual-mode cryptosystem, assuming that DDH is hard for $\mathcal{G}$.*

Proof. Completeness follows by inspection from the correctness of the basic DDH cryptosystem.

We now show indistinguishability of the two modes. In extraction mode, $\mathsf{crs} = (g_0, h_0 = g_0^{x_0}, g_1, h_1 = g_1^{x_1})$, where g_0, g_1 are random generators of G and x_0, x_1 are distinct and nonzero in $\mathbb{Z}_p$. Let $a = \log_{g_0} g_1$, which is nonzero but otherwise uniform in $\mathbb{Z}_p$. Then $b = \log_{h_0}(h_1) = a \cdot x_1/x_0$ is nonzero and distinct from a, but otherwise uniform. Therefore crs is statistically close to a random DDH non-tuple (g_0, h_0, g_0^a, h_0^b), where $a, b \leftarrow \mathbb{Z}_p$ are distinct but otherwise uniform. Now in decryption mode, $\mathsf{crs} = (g_0, h_0 = g_0^x, g_1, h_1 = g_1^x)$, where x is nonzero and random in $\mathbb{Z}_p$. Since $\log_{h_0}(h_1) = \log_{g_0}(g_1) = y$ is nonzero and random in $\mathbb{Z}_p$, crs is statistically close to a random DDH tuple. Under the DDH assumption, the two modes are indistinguishable.

We now demonstrate extraction of a message-lossy branch. By inspection, $\mathsf{FindLossy}(t, pk)$ computes a branch b for which (g_b, h_b, g, h) (the key used when encrypting under pk on branch b) is not a DDH tuple. By Lemma 5.1, this b is therefore a message-lossy branch.

We conclude with trapdoor key generation. Let $(\mathsf{crs}, y) \leftarrow \mathsf{SetupDec}(1^n)$. Note that crs is a DDH tuple of the form $(g_0, h_0 = g_0^x, g_1 = g_0^y, h_1 = g_1^x)$, where x and y are nonzero. $\mathsf{TrapKeyGen}(\mathsf{crs}, y)$ outputs $(pk, sk_0, sk_1) = ((g_0^r, h_0^r), r, r/y)$. The output of $\mathsf{KeyGen}(\sigma)$, on the other hand, is $((g_\sigma^r, h_\sigma^r), r)$. We will now show that (pk, sk_σ) and $\mathsf{KeyGen}(\sigma)$ are identically distributed.

Indeed, $(pk, sk_0) = (g_0^r, h_0^r, r)$ is identically distributed to $\mathsf{KeyGen}(0)$, by definition of KeyGen. By a renaming of variables letting $r = r'y$, we have that $(pk, sk_1) = (g_0^r, h_0^r, r/y)$ is identical to $(g_0^{r'y}, h_0^{r'y}, r') = (g_1^{r'}, h_1^{r'}, r')$, which is distributed identically to $\mathsf{KeyGen}(1)$, since $r' = r/y \in \mathbb{Z}_p$ is uniformly distributed. ☐

Larger branch sets. We briefly outline how the dual-mode cryptosystem is modified for larger branch sets $\{0,1\}^k$. Essentially, the scheme involves k parallel and independent copies of the one above, but all using the same group $\mathbb{G}$. The encryption algorithm Enc computes a k-wise secret sharing of the message, and encrypts each share under the corresponding copy of the scheme. This ensures that decryption succeeds only for the one specific branch selected to be message-preserving. The $\mathsf{FindLossy}$ algorithm includes a branch $b \in \{0,1\}^k$ in its output list of lossy branches if *any* branch b_i is message-lossy for its corresponding scheme.

6 Realization from QR

6.1 Cryptosystem Based on QR

We start by describing a (non-identity-based) variant of Cocks' cryptosystem [Coc01], which is based on the conjectured hardness of the Quadratic Residuosity problem.

For $N \in \mathbb{N}$, let $\mathbb{J}_N$ denote the set of all $x \in \mathbb{Z}_N^*$ with Jacobi symbol 1. Let $\mathbb{QR}_N \subset \mathbb{J}_N$ denote the set of all quadratic residues (squares) in $\mathbb{Z}_N^*$. The message space is $\{\pm 1\}$. Let $\left(\frac{t}{N}\right)$ denote the Jacobi symbol of t in $\mathbb{Z}_N^*$.

- $\mathsf{CKeyGen}(1^n)$: Choose two random n-bit safe primes[2] p and q and let $N = pq$. Choose $r \leftarrow \mathbb{Z}_N^*$ and let $y \leftarrow r^2$. Let $pk = (N, y)$, and $sk = r$. Output (pk, sk).
- $\mathsf{CEnc}(pk, m)$: Parse pk as (N, y). Choose $s \leftarrow \mathbb{Z}_N^*$ at random such that $\left(\frac{s}{N}\right) = m$, and output $c = s + y/s$.
- $\mathsf{CDec}(sk, c)$: Output the Jacobi symbol of $c + 2 \cdot sk$.

The following lemma is implicit in the security proof of the Cocks cryptosystem.

Lemma 6.1 (Message-Lossy Characterization)**.** *Let N be a product of two random n-bit safe primes p and q, let $y \in \mathbb{Z}_N^*$ and let $pk = (N, y)$. If $y \notin \mathbb{QR}_N$, then $\mathsf{CEnc}(pk, b, \cdot)$ is message-lossy. Namely,*

$$\mathsf{CEnc}(pk, b, +1) \stackrel{s}{\approx} \mathsf{CEnc}(pk, b, -1).$$

Proof. If $y \notin \mathbb{QR}_N$, then at least one of $\alpha_1 = \left(\frac{y}{p}\right)$ or $\alpha_2 = \left(\frac{y}{q}\right)$ is -1. Consider the equation $c = s + y/s \bmod N$, and say s_0 is one of the solutions. Then we have $c = s_0 + y/s_0 \bmod p$ and $c = s_0 + y/s_0 \bmod q$. The other solutions are s_1, s_2, and s_3, where

$$\begin{array}{lll} s_1 = s_0 \bmod p & s_2 = y/s_0 \bmod p & s_3 = y/s_0 \bmod p \\ s_1 = y/s_0 \bmod q & s_2 = s_0 \bmod q & s_3 = y/s_0 \bmod q. \end{array}$$

Then $\left(\frac{s_1}{N}\right) = \alpha_2\left(\frac{s_0}{N}\right)$, $\left(\frac{s_2}{N}\right) = \alpha_1\left(\frac{s_0}{N}\right)$ and $\left(\frac{s_3}{N}\right) = \alpha_1\alpha_2\left(\frac{s_0}{N}\right)$. Thus, two of these are $+1$ and the other two are -1. It follows that c hides $\left(\frac{s}{N}\right)$ perfectly. $\square$

6.2 Dual-Mode Cryptosystem

We now describe a dual-mode cryptosystem that is based on the Cocks cryptosystem.

- $\mathsf{SetupExt}(1^n)$: Choose two random n-bit safe primes p and q and let $N = pq$. Choose $y \leftarrow \mathbb{J}_N \setminus \mathbb{QR}_N$. Let $\mathsf{crs} = (N, y)$, and $t = (p, q)$. Output (crs, t).

  $\mathsf{SetupDec}(1^n)$: Let $N = pq$ for random n-bit safe primes as above. Choose $s \leftarrow \mathbb{Z}_N^*$, and let $y = s^2 \bmod N$. Let $\mathsf{crs} = (N, y)$, amd $t = s$. Output (crs, t).

  In the following, all algorithms are implicitly provided the crs and parse it as (N, y), and all operations are performed in $\mathbb{Z}_N^*$.
- $\mathsf{KeyGen}(\sigma)$: Choose $r \leftarrow \mathbb{Z}_N^*$, and let $pk = r^2/y^\sigma$. Let $sk = r$. Output (pk, sk).
- $\mathsf{Enc}(pk, b, m)$: Let $pk_b = (N, pk \cdot y^b)$. Output $\mathsf{CEnc}(pk_b, m)$.
- $\mathsf{Dec}(sk, c)$: Output $\mathsf{CDec}(sk, c)$.
- $\mathsf{FindLossy}(t, pk)$: Parse the extraction trapdoor t as (p, q) where $N = pq$. If $pk \in \mathbb{QR}_N$ (this can be checked efficiently using p and q), then output $b = 1$ as the (candidate) message-lossy branch; otherwise, output $b = 0$.

[2]Safe primes are primes p such that $\frac{p-1}{2}$ is also prime.

- $\mathsf{TrapKeyGen}(t)$: Choose a random $r \leftarrow \mathbb{Z}_N^*$ and let $pk = r^2$ and $sk_b = r \cdot t^b$ for each $b \in \{0,1\}$. Output (pk, sk_0, sk_1).

Theorem 6.2. *The above scheme is a dual-mode cryptosystem, assuming the hardness of the quadratic residuosity problem.*

Proof. We first show completeness. Say $(pk, sk) \leftarrow \mathsf{KeyGen}(\sigma)$. Thus, $pk = r^2 y^{-\sigma}$ for some r. $\mathsf{Enc}(pk, \sigma, m)$ runs $\mathsf{CEnc}(pk \cdot y^\sigma, m) = \mathsf{CEnc}(r^2, m)$. Thus, the public key used in the Cocks encryption algorithm is a quadratic residue. By the completeness of the Cocks cryptosystem, the decryption algorithm recovers m.

We now show indistinguishability of the two modes. In extraction mode, $\mathsf{crs} = (N, y)$, where y is a uniform element in $\mathbb{J}_N \setminus \mathbb{QR}_N$. In decryption mode, $\mathsf{crs} = (N, y)$, where y is a uniform element in $\mathbb{QR}_N$. By the QR assumption, these are indistinguishable.

We now demonstrate extraction of a message-lossy branch. Let pk be the (possibly malformed) public key. Since $y \notin \mathbb{QR}_N$, either pk or $pk \cdot y$ is not a quadratic residue. Lemma 6.1 implies that one of the branches of pk is message-lossy; it can be found using the factorization $t = (p, q)$ of N.

We conclude with trapdoor key generation. Let $y = t^2$. $\mathsf{TrapKeyGen}(\mathsf{crs}, t)$ outputs $(r^2, r, r \cdot t)$. The output of $\mathsf{KeyGen}(\sigma)$, on the other hand, is $(r^2 y^{-\sigma}, r)$. Now, $(pk, sk_0) = (r^2, r)$ is distributed identically to $\mathsf{KeyGen}(0)$, by definition of KeyGen. By a renaming of variables letting $r = r'/t$, we have $(pk, sk_1) = ((r')^2/t^2, r') = ((r')^2/y, r')$, which is distributed identically to $\mathsf{KeyGen}(1)$, since $r' = r/t \in \mathbb{Z}_N^*$ is uniformly distributed. □

For larger branch sets $\{0,1\}^k$, the scheme is modified in a manner similar to the one from Section 5.2, where all k parallel copies of the scheme use the same modulus N.

7 Realization from Lattices

Here we construct a dual-mode cryptosystem based on the hardness of the *learning with error* (LWE) problem, which is implied by the (quantum) hardness of standard *worst-case* lattice problems, as shown by Regev [Reg05]. We stress that although the underlying lattice assumption relates to quantum algorithms, our cryptosystems are entirely classical.

As an independent contribution and a building block for our efficient OT protocol, we present a much more efficient version of Regev's CPA-secure cryptosystem [Reg05] based on LWE. Our cryptosystem encrypts an n-bit message at essentially the same cost (in both space and time) as a single-bit message in the system from [Reg05]. Specifically, our ciphertexts can be made only a constant factor larger than the message, as opposed to an $\tilde{O}(n)$ factor. Our encryption and decryption algorithms require only $\tilde{O}(n)$ bit operations per encrypted bit, as opposed to $\tilde{O}(n^2)$. Our overall public key size remains asymptotically the same at $\tilde{O}(n^2)$ bits; however, in the CRS model, an optimization from [Reg05] allows the user-specific part of the public key to made only $\tilde{O}(n)$ bits, while ours remains $\tilde{O}(n^2)$ bits.

7.1 Background

We start by introducing the notation and computational problems that are relevant to this section, for the most part following [Reg05].

For any $x, y \in \mathbb{R}$ with $y > 0$ we define $x \bmod y$ to be $x - \lfloor x/y \rfloor y$. For $x \in \mathbb{R}$, $\lfloor x \rceil = \lfloor x + 1/2 \rfloor$ denotes the nearest integer to x (with ties broken upward). We define $\mathbb{T} = \mathbb{R}/\mathbb{Z}$, i.e., the group of reals $[0, 1)$ with modulo 1 addition.

Probability distributions. The *normal distribution* with mean 0 and variance σ^2 (or standard deviation σ) is the distribution on $\mathbb{R}$ having density function $\frac{1}{\sigma \cdot \sqrt{2\pi}} \exp(-x^2/2\sigma^2)$. It is a standard fact that the sum of two independent normal variables with mean 0 and variances σ_1^2 and σ_2^2 (respectively) is a normal variable with mean 0 and variance $\sigma_1^2 + \sigma_2^2$. We will also need a standard tail inequality: a normal variable with variance σ^2 is within distance $t \cdot \sigma$ (i.e., t standard deviations) of its mean, except with probability at most $\frac{1}{t} \cdot \exp(-t^2/2)$. Finally, it is possible to efficiently sample from a normal variable to any desired level of accuracy.

For $\alpha \in \mathbb{R}^+$ we define Ψ_α to be the distribution on $\mathbb{T}$ of a normal variable with mean 0 and standard deviation $\alpha/\sqrt{2\pi}$, reduced modulo 1. For any probability distribution $\phi : \mathbb{T} \to \mathbb{R}^+$ and an integer $q \in \mathbb{Z}^+$ (often implicit) we define its *discretization* $\bar{\phi} : \mathbb{Z}_q \to \mathbb{R}^+$ to be the discrete distribution over $\mathbb{Z}_q$ of the random variable $\lfloor q \cdot X_\phi \rceil \bmod q$, where X_ϕ has distribution ϕ.

For an integer $q \geq 2$ and some probability distribution $\chi : \mathbb{Z}_q \to \mathbb{R}^+$, an integer dimension $n \in \mathbb{Z}^+$ and a vector $\mathbf{s} \in \mathbb{Z}_q^n$, define $A_{\mathbf{s},\chi}$ as the distribution on $\mathbb{Z}_q^n \times \mathbb{Z}_q$ of the variable $(\mathbf{a}, \langle \mathbf{a}, \mathbf{s} \rangle + e)$ where $\mathbf{a} \leftarrow \mathbb{Z}_q^n$ is uniform and $e \leftarrow \chi$ are independent, and all operations are performed in $\mathbb{Z}_q$.

Learning with error (LWE). For an integer $q = q(n)$ and a distribution χ on $\mathbb{Z}_q$, the goal of the (average-case) *learning with error* problem $\mathsf{LWE}_{q,\chi}$ is to distinguish (with nonnegligible probability) between an oracle that returns independent samples from $A_{\mathbf{s},\chi}$ for some uniform $\mathbf{s} \leftarrow \mathbb{Z}_q^n$, and an oracle that returns independent samples from the uniform distribution on $\mathbb{Z}_q^n \times \mathbb{Z}_q$.

The conjectured hardness of LWE is parameterized chiefly by the dimension n. Therefore we let all other parameters (e.g., q, χ, and others) be functions of n, often omitting the explicit dependence for notational clarity.

Regev [Reg05] demonstrated that for certain moduli q and error distributions χ, $\mathsf{LWE}_{q,\chi}$ is as hard as solving several standard *worst-case* lattice problems *using a quantum algorithm.* We state a version of his result here:

Proposition 7.1 ([Reg05]). *Let $\alpha = \alpha(n) \in (0, 1)$ and let $q = q(n)$ be a* prime *such that $\alpha \cdot q > 2\sqrt{n}$. If there exists an efficient (possibly quantum) algorithm that solves $\mathsf{LWE}_{q,\bar{\Psi}_\alpha}$, then there exists an efficient* quantum *algorithm for solving the following worst-case lattice problems in the ℓ_2 norm:*

- *SIVP: In any lattice Λ of dimension n, find a set of n linearly independent lattice vectors of length within at most $\tilde{O}(n/\alpha)$ of optimal.*
- *GapSVP: In any lattice Λ of dimension n, approximate the length of a shortest nonzero lattice vector to within a $\tilde{O}(n/\alpha)$ factor.*

Peikert [Pei07] extended this result to hold for lattice problems in *any* ℓ_p norm, $p \geq 2$, for the same $\tilde{O}(n/\alpha)$ approximation factors.

We will define our cryptosystems purely in relation to the LWE problem, without explicitly taking into account the connection to lattices (or their parameter restrictions). We will then instantiate the parameters appropriately, invoking Proposition 7.1 to ensure security assuming the (quantum) hardness of lattice problems.

7.2 Efficient Cryptosystem Based on LWE

Our efficient cryptosystem closely resembles Regev's [Reg05]. In Regev's scheme, public keys consist of an $m \times (n+1)$ matrix $(\mathbf{A}|\mathbf{b})$. The m rows $(\mathbf{a}_i, b_i)$ of the matrix are samples from the LWE distribution $A_{\mathbf{s},\chi}$, where $\mathbf{s}$ is the uniform secret key. A ciphertext encrypting a single bit is of the form $(\mathbf{rA}, \langle \mathbf{r}, \mathbf{b} \rangle + c)$ for a suitable random $1 \times m$ row vector $\mathbf{r}$ and a value c determined by the message bit. The bulk of the public key and the ciphertext, as well as the encryption and decryption time, is therefore devoted to the large matrix $\mathbf{A}$. Our main new idea is that both the $\mathbf{A}$ component of the public key and the encryption randomness $\mathbf{r}$ can be *securely amortized* (reused) over as many as $\ell = O(n)$ independent secrets $\mathbf{s}_i$, without increasing the asymptotic complexity of the scheme. This yields an asymptotic improvement by a factor of n in nearly all aspects of the system. We now proceed more formally.

We first describe the various parameters of the scheme and their roles, but defer giving concrete values for them until later. The message space is $\mathbb{Z}_p^\ell$ for some integers $p = \mathrm{poly}(n) \geq 2$ and $\ell = \mathrm{poly}(n) \geq 1$. Let $q = \mathrm{poly}(n) > p$ be a prime, so that $\mathbb{Z}_q$ is a field; our public keys and ciphertexts will consist of matrices/vectors over $\mathbb{Z}_q$. For every $v \in \mathbb{Z}_p$ (i.e., one coordinate of a message), define the "offset" for v to be $c(v) = \lfloor v \cdot \frac{q}{p} \rceil \in \mathbb{Z}_q$. Let χ denote an efficiently sampleable error distribution over $\mathbb{Z}_q$. Let $R \leq q$ be an integer, and define a set of row vectors $\mathcal{R} = [0, R-1]^{1 \times m} \subseteq \mathbb{Z}_q^{1 \times m}$.

Before describing the cryptosystem in detail, we give some intuition about how the parameters affect the correctness, security, and efficiency. The moduli p and q will give a trade-off between the ciphertext expansion and the underlying concrete lattice assumption. The amortization factor ℓ will affect the overall message space and efficiency of the system. The error distribution χ will be chosen to ensure correctness of decryption, and will determine the concrete underlying lattice assumption. Finally, the dimension m and the bound R on the entries of $\mathbf{r} \in \mathcal{R}$ will determine the density of message-lossy public keys; larger values of R and m will yield a larger density of lossy keys. This density will play an especially important role in our OT application.

We now describe the cryptosystem more formally. All operations are performed over $\mathbb{Z}_q$.

- $\mathsf{LWEKeyGen}(1^n)$: Choose a matrix $\mathbf{S} \leftarrow \mathbb{Z}_q^{n \times \ell}$ uniformly at random; $\mathbf{S}$ is the secret key.

  To create the public key, choose a matrix $\mathbf{A} \leftarrow \mathbb{Z}_q^{m \times n}$ uniformly at random. Choose a matrix $\mathbf{E} \in \mathbb{Z}_q^{m \times \ell}$ where each entry $e_{i,j} \leftarrow \chi$ is chosen independently from the error distribution χ. The public key is the pair $(\mathbf{A}, \mathbf{B} = \mathbf{AS} + \mathbf{E}) \in \mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^{m \times \ell}$.

  Note that the (i,j)th entry of $\mathbf{B}$ is $b_{i,j} = \langle \mathbf{a}_i, \mathbf{s}_j \rangle + e_{i,j}$, where $\mathbf{a}_i$ is the (uniform and public) ith row of $\mathbf{A}$ and $\mathbf{s}_j$ is the (uniform and secret) jth column of $\mathbf{S}$. That is, $(\mathbf{a}_i, b_{i,j})$ is a sample from the LWE distribution $A_{\mathbf{s}_j,\chi}$.

- $\mathsf{LWEEnc}(pk = (\mathbf{A}, \mathbf{B}), \mathbf{v})$: To encrypt a message written as a row vector $\mathbf{v} \in \mathbb{Z}_p^{1 \times \ell}$, let the row vector $\mathbf{c} = c(\mathbf{v}) \in \mathbb{Z}_q^{1 \times \ell}$ be the "offset" for $\mathbf{v}$, where $c(\cdot)$ is applied coordinate-wise. Choose a row vector $\mathbf{r} \leftarrow \mathcal{R} \subset \mathbb{Z}_q^{1 \times m}$.

  The encryption of $\mathbf{v}$ under public key $(\mathbf{A}, \mathbf{B})$ is the pair $(\mathbf{t}_1, \mathbf{t}_2) = (\mathbf{rA}, \mathbf{rB} + \mathbf{c}) \in \mathbb{Z}_q^{1 \times n} \times \mathbb{Z}_q^{1 \times \ell}$.

- $\mathsf{LWEDec}(sk = \mathbf{S}, (\mathbf{t}_1, \mathbf{t}_2))$: Compute $\mathbf{d} = \mathbf{t}_2 - \mathbf{t}_1 \mathbf{S} \in \mathbb{Z}_q^{1 \times \ell}$. Output the plaintext $\mathbf{v} \in \mathbb{Z}_p^{1 \times \ell}$, where each v_i is such that $d_i - c(v_i) \in \mathbb{Z}_q$ is closest to 0 mod q.

7.2.1 Some Supporting Lemmas

We now prove a few lemmas that will help establish correctness and security (for appropriate choices of parameters), both for the above system and for our dual-mode cryptosystem later on. At first, the reader may wish to just read the statements of the lemmas and then skip directly to Section 7.2.2, where we instantiate the parameters, analyze the efficiency, and prove security.

Lemma 7.2 (Completeness)**.** *Let $q \geq 4R \cdot p \cdot m$, let $\alpha \leq 1/(R \cdot p \cdot \sqrt{m} \cdot g)$ for some $g = \omega(\sqrt{\log n})$, and let $\chi = \bar{\Psi}_\alpha$. Then* LWEDec *decrypts correctly with overwhelming probability (over the random choice of* $\mathbf{E}$ *by* $\mathsf{LWEKeyGen}$*).*

Proof. The proof uses a standard tail bound on the sum of (rounded-off) Gaussians.

Consider some secret key $\mathbf{S}$ and associated public key $(\mathbf{A}, \mathbf{B} = \mathbf{AS} + \mathbf{E})$ where $\mathbf{A}$ and $\mathbf{S}$ are arbitary, and $\mathbf{E}$ is random according to the prescribed distribution. Now take a ciphertext $(\mathbf{t}_1, \mathbf{t}_2) = (\mathbf{rA}, \mathbf{rB} + \mathbf{c})$ generated by LWEEnc using some $\mathbf{r} \in \mathcal{R}$, where $\mathbf{c} = c(\mathbf{v})$ is the offset vector for the message $\mathbf{v}$. The decryption algorithm LWEDec computes the row vector

$$\mathbf{d} = \mathbf{t}_2 - \mathbf{t}_1\mathbf{S} = \mathbf{r}(\mathbf{AS} + \mathbf{E}) + \mathbf{c} - \mathbf{rAS} = \mathbf{rE} + \mathbf{c} \in \mathbb{Z}_q^{1\times\ell}.$$

Now consider any coordinate $j \in [\ell]$. The distance from d_j to c_j (modulo q) is given by $(\mathbf{rE})_j = \langle \mathbf{r}, \mathbf{e}_j \rangle$, where e_j is the jth column of $\mathbf{E}$. Therefore it suffices to show that for every j, $\langle \mathbf{r}, \mathbf{e}_j \rangle$ is at distance at most $q/4p$ from 0 (modulo q) with overwhelming probability, because this guarantees that d_j is closest to $c(v_j)$.

Now by definition of $\chi = \bar{\Psi}_\alpha$, $e_{i,j} = \lfloor q \cdot y_{i,j} \rceil \bmod q$, where $y_{i,j}$ are independent normal variables with mean 0 and variance α^2. Then by the triangle inequality, $(\mathbf{rE})_j$ is at most $Rm/2 \leq q/8p$ away from $q(\langle \mathbf{r}, \mathbf{y}_j \rangle \bmod 1)$, where $\mathbf{y}_j$ is the jth column of $\mathbf{Y} = (y_{i,j})$. Therefore it suffices to show that $|\langle \mathbf{r}, \mathbf{y}_j \rangle| \leq 1/8p$ with overwhelming probability.

Because the $y_{i,j}$ are independent, $\langle \mathbf{r}, \mathbf{y}_j \rangle$ is distributed as a normal variable with mean 0 and standard deviation $\|\mathbf{r}\| \cdot \alpha \leq R\sqrt{m} \cdot \alpha \leq 1/(p \cdot g)$. Therefore by the tail inequality on normal variables,

$$\Pr_{\mathbf{y}_j}[|\langle \mathbf{r}, \mathbf{y}_j \rangle| > 1/8p] \leq \tfrac{8}{g} \cdot \exp(-g^2/128).$$

Because $g = \omega(\sqrt{\log n})$, this probability is negligible, and we are done. □

Lemma 7.3 (Pseudorandom public keys)**.** *The distribution of the public key $pk = (\mathbf{A}, \mathbf{B})$ generated by* $\mathsf{LWEKeyGen}$ *is computationally indistinguishable from uniform over $\mathbb{Z}_q^{m\times n} \times \mathbb{Z}_q^{m\times\ell}$, assuming $\mathsf{LWE}_{q,\chi}$ is hard.*

Proof. The proof is virtually identical to the one from [PW07]. Consider hybrid distributions $H_0, \ldots, H_\ell$ for the pair $(\mathbf{A}, \mathbf{B})$: in distribution H_k, the entire matrix $\mathbf{A}$ and the first k columns of $\mathbf{B}$ are all uniform, while the remaining columns of $\mathbf{B}$ are chosen exactly according to $\mathsf{LWEKeyGen}$, using independent secrets $\mathbf{s}_j$ and error terms $e_{i,j}$ for all $j > k$ and all $i \in [m]$. Then H_0 is the distribution generated by $\mathsf{LWEKeyGen}$, and H_ℓ is completely uniform.

We now show that distributions H_{k-1} and H_k are computationally indistinguishable (assuming $\mathsf{LWE}_{q,\chi}$ is hard), from which the lemma follows. Consider a simulator $\mathcal{S}^{\cdot}$, which is given an oracle $\mathcal{O}$ that returns samples either from $A_{\mathbf{s},\chi}$ (for some secret $\mathbf{s} \leftarrow \mathbb{Z}_q^n$) or from the uniform distribution over $\mathbb{Z}_q^n \times \mathbb{Z}_q$. First, $\mathcal{S}$ makes m queries to $\mathcal{O}$, yielding samples $(\mathbf{a}_i, b_i)$ for $i \in [m]$. Then for all $j > k$ and $i \in [m]$, $\mathcal{S}$ chooses independent $\mathbf{s}_j \leftarrow \mathbb{Z}_q^n$ and error terms $e_{i,j} \leftarrow \chi$. $\mathcal{S}$ outputs the pair

$(\mathbf{A}, \mathbf{B})$ constructed in the following way: the ith row of $\mathbf{A}$ is the vector $\mathbf{a}_i$, the first $k-1$ columns of $\mathbf{B}$ are uniform, the kth column of $\mathbf{B}$ consists of the entries $b_{i,k} = b_i$, and the remaining entries of $\mathbf{B}$ are $b_{i,j} = \langle \mathbf{a}_i, \mathbf{s}_j \rangle + e_{i,j}$ for $i \in [m]$ and $j > k$.

One may verify that if $\mathcal{O}$ samples from $A_{\mathbf{s},\chi}$, then $\mathcal{S}$'s output is distributed according to H_{k-1}, whereas if $\mathcal{O}$ samples from the uniform distribution, $\mathcal{S}$'s output is distributed according to H_k. It follows that H_{k-1} and H_k are indistinguishable, and we are done. □

For an arbitrary fixed public key $pk = (\mathbf{A}, \mathbf{B}) \in \mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^{m \times \ell}$, define $\delta(pk)$ to be the statistical distance between the distribution of $(\mathbf{rA}, \mathbf{rB}) \in \mathbb{Z}_q^{1 \times n} \times \mathbb{Z}_q^{1 \times \ell}$, where $\mathbf{r} \leftarrow \mathcal{R}$, and the uniform distribution over $\mathbb{Z}_q^{1 \times n} \times \mathbb{Z}_q^{1 \times \ell}$. Then if $\delta(pk)$ is negligible, pk is a message-lossy public key; i.e., for any two messages $\mathbf{v}_0, \mathbf{v}_1 \in \mathbb{Z}_q^{1 \times \ell}$, $\mathsf{LWEEnc}(pk, \mathbf{v}_0) \stackrel{s}{\approx} \mathsf{LWEEnc}(pk, \mathbf{v}_1)$, because both distributions are statistically close to uniform. (Of course, the correctness of LWEDec implies that the public keys pk generated by $\mathsf{LWEKeyGen}$ have large $\delta(pk)$.)

In contrast to our DDH- and QR-based cryptosystems in Sections 5.2 and 6.1, we do not have an explicit characterization of the message-lossy public keys. Instead, we can show that, for appropriate choices of parameters, an overwhelming fraction of all keys are message-lossy. This is sufficient for proving security of the basic cryptosystem, and will also play a crucial role in proving the correctness of $\mathsf{FindLossy}$ in our dual-mode cryptosystem of Section 7.3.

Lemma 7.4 (Density of message-lossy keys)**.** *Let $\epsilon = \sqrt{1/q^{n+\ell}} = \mathsf{negl}(n)$. Then the fraction of public keys $pk = (\mathbf{A}, \mathbf{B}) \in \mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^{m \times \ell}$ such that $\delta(pk) > \epsilon$ is at most $q^{n+\ell}/R^m$. In other words, all but an (at most) $q^{n+\ell}/R^m$ fraction of public keys are message-lossy.*

Proof. We use techniques from [IZ89]. Let $G = \mathbb{Z}_q^{n+\ell}$. For any distribution D over G, define its collision probability to be $\sum_{g \in G} D^2(g) = \|D\|_2^2$, where $\|\cdot\|_2$ denotes the ℓ_2 norm of the distribution. For any distribution D, we have $\|D\|_2^2 \geq 1/|G|$. Further, by the relationship between ℓ_1 and ℓ_2 norm, (twice) the statistical distance between D and uniform is

$$\begin{aligned} \sum_{g \in G} |D(g) - 1/|G|| &\leq \sqrt{|G|} \cdot \left(\sum_{g \in G} (D(g) - 1/|G|)^2 \right)^{1/2} \\ &= \sqrt{|G|} \cdot \left(\|D\|_2^2 - 1/|G| \right). \end{aligned}$$

Therefore it will suffice to focus our attention on the quantity $\|D\|_2^2 - 1/|G|$.

For each public key $pk = (\mathbf{A}, \mathbf{B})$, let D_{pk} be the distribution of $(\mathbf{rA}, \mathbf{rB}) \in G$ where $\mathbf{r} \leftarrow \mathcal{R}$. The collision probability of D_{pk} is

$$\begin{aligned} \sum_{g \in G} D_{pk}^2(g) &= \Pr_{\mathbf{r}, \mathbf{r}' \leftarrow \mathcal{R}} \left[(\mathbf{rA}, \mathbf{rB}) = (\mathbf{r}'\mathbf{A}, \mathbf{r}'\mathbf{B}) \right] \\ &\leq \frac{1}{R^m} + \Pr_{\mathbf{r}, \mathbf{r}'} \left[(\mathbf{rA}, \mathbf{rB}) = (\mathbf{r}'\mathbf{A}, \mathbf{r}'\mathbf{B}) \mid \mathbf{r} \neq \mathbf{r}' \right]. \end{aligned}$$

Now because q is prime, $\mathbb{Z}_q$ is a field, and we get that for any fixed $\mathbf{r} \neq \mathbf{r}'$,

$$\Pr_{pk = (\mathbf{A}, \mathbf{B})} \left[\left((\mathbf{r} - \mathbf{r}')\mathbf{A}, (\mathbf{r} - \mathbf{r}')\mathbf{B} \right) = \mathbf{0} \in G \right] = 1/|G|.$$

Taking an expectation over uniform pk, we get

$$\mathop{\mathrm{E}}_{pk}\left[\|D_{pk}\|_2^2 - 1/|G|\right] \quad = \quad \mathop{\mathrm{E}}_{pk}\left[\sum_g D_{pk}^2(g)\right] - 1/|G| \quad \leq \quad 1/R^m.$$

Therefore by Markov's inequality, we have

$$\Pr_{pk}\left[\left(\|D_{pk}\|_2^2 - 1/|G|\right) > 1/|G|\right] \leq |G|/R^m.$$

In other words, an at most $q^{n+\ell}/R^m$ fraction of public keys pk have $\delta(pk) > \sqrt{1/q^{n+\ell}} = \epsilon$. $\square$

7.2.2 Instantiation and Efficiency

We now instantiate all the parameters of our LWE cryptosystem to ensure correctness and security. Other instantiations are also possible, yielding slight gains in efficiency at the expense of stronger underlying lattice assumptions. It is also possible to slightly improve the constant factors.

Recall that $\mathbb{Z}_p$ is the message space. Let $p = n^c$ for some positive c that is bounded above by a constant (e.g., c may itself be a constant, or a decreasing function of n, such as $c(n) = 1/\lg n$). Let the amortization factor $\ell = n$ (we could let ℓ be any $O(n)$ without affecting the asymptotics). Let $R = 2$, and let $m = (4+2c)(n+\ell)\lg n = O(n \lg n)$. Let q be a prime in $[10, 20] \cdot p \cdot m \lg n = \tilde{O}(n^{c+1})$. Finally, let the error distribution $\chi = \bar{\Psi}_\alpha$ for $\alpha = 1/(R \cdot p \cdot \sqrt{m} \cdot \lg n) = 1/\tilde{O}(n^{c+1/2})$.

We now analyze the efficiency of the system. For concreteness, say that $c > 0$ is a constant. Then with the parameters above, a public key contains $m(n+\ell)$ elements of $\mathbb{Z}_q$, for a total size of $\tilde{O}(n^2)$ bits. The message space is $\mathbb{Z}_p^\ell$. Then to encrypt $\ell \lg p = \Theta(n \lg n)$ bits requires $O(m(n+\ell))$ operations in $\mathbb{Z}_q$, costing $O(n^2 \lg^2 n)$ bit operations in total, or $O(n \lg n)$ bit operations per message bit. The ciphertext contains $(n+\ell)$ elements of $\mathbb{Z}_q$, so its size is only an $O(1)$ factor larger than the message.

Theorem 7.5. *For the parameters described above, our LWE cryptosystem is secure under chosen plaintext attack, assuming that either of the lattice problems described in Proposition 7.1 are hard for any efficient quantum algorithm to approximate to within some $\tilde{O}(n^{c+3/2})$ factor.*

Proof. We first show completeness. By Lemma 7.2, LWEDec decrypts correctly (with overwhelming probability) for our choice of α, as long as $q \geq 4R \cdot p \cdot m$. Indeed, $4R \cdot p \cdot m = 8pm < 10pm \lg n \leq q$.

We now calculate the density of message-lossy keys. Observe that, for all sufficiently large n,

$$2(n+\ell)\lg q \quad \leq \quad 2(n+\ell)\lg(n^{c+2}) \quad = \quad (4+2c)(n+\ell)\lg n \quad \leq \quad m.$$

Therefore, we have

$$R^m \geq 2^{2(n+\ell)\lg q} = q^{2(n+\ell)}.$$

Lemma 7.4 thus implies that an all but $1/q^{n+\ell} = \mathsf{negl}(n)$ fraction of public keys are message-lossy. Additionally, Lemma 7.3 says that as long as $\mathsf{LWE}_{q,\chi}$ is hard (which we show below), a public key generated by $\mathsf{LWEKeyGen}$ is indistinguishable from uniform.

The last two facts together imply security of the cryptosystem: because a uniform public key is message-lossy with overwhelming probability, no adversary (even an unbounded one) has more than a negligible advantage in a chosen plaintext attack when the public key is selected uniformly. Then because the public keys generated by $\mathsf{LWEKeyGen}$ are indistinguishable from uniform, it follows that any bounded adversary also has negligible advantage when attacking the real system.

We conclude by justifying the hardness of $\mathsf{LWE}_{q,\chi}$. Using the fact that $q \geq 4R \cdot p \cdot m \cdot \lg n$, we have (as required by Proposition 7.1),

$$q \cdot \alpha \geq \frac{4R \cdot p \cdot m \cdot \lg n}{R \cdot p \cdot \sqrt{m} \cdot \lg n} = 4\sqrt{m} > 2\sqrt{n}.$$

Therefore we may apply Proposition 7.1, which for our choice of α means that hardness of $\mathsf{LWE}_{q,\chi}$ is implied by quantum hardness of lattice problems to within some factor $\tilde{O}(n/\alpha) = \tilde{O}(n^{c+3/2})$. $\square$

7.3 Dual-Mode Cryptosystem

Here we construct a dual-mode cryptosystem based on the hardness of LWE. We do not know how to construct a scheme that exactly satisfies the requirements of Definition 3.1; however, we can construct one that satisfies a slightly relaxed definition, and which suffices for composing a *bounded* number of efficient oblivious transfers using a single common string.

We will relax the notion of trapdoor key generation (in decryption mode) in two ways. The first difference is that the public/secret keypairs generated by $\mathsf{TrapKeyGen}$ are only *computationally* indistinguishable from those generated by KeyGen. The second is that indistinguishability applies to the outputs of a predetermined number of calls to $\mathsf{TrapKeyGen}$, *together with* the common string.

More formally, let $\ell = \mathrm{poly}(n)$ be some fixed polynomial in the security parameter n. We add an additional parameter $i \in [\ell]$ to the inputs of all the algorithms (except Setup), which is used to specify their ith executions. We then relax Property 4 of Definition 3.1 to the following:

4′. Let $(\mathsf{crs}, t) \leftarrow \mathsf{SetupDec}(1^n)$ and $(pk^{(i)}, sk_0^{(i)}, sk_1^{(i)}) \leftarrow \mathsf{TrapKeyGen}(t, i)$ for each $i \in [\ell]$. Then for every $\sigma_i \in \{0, 1\}$, we have

$$(\mathsf{crs}, (pk^{(1)}, sk_{\sigma_1}^{(1)}), \ldots, (pk^{(\ell)}, sk_{\sigma_\ell}^{(\ell)})) \overset{c}{\approx} (\mathsf{SetupDec}_1(1^n), \mathsf{KeyGen}(\sigma_1, 1), \ldots, \mathsf{KeyGen}(\sigma_\ell, \ell)).$$

Our dm protocol for emulating the multi-session OT functionality $\hat{\mathcal{F}}_{\mathrm{OT}}$ and its proof of security from Section 4 can be easily modified to use this relaxed definition. The protocol simply limits itself to ℓ separate uses of a single crs; it then uses a new crs by invoking another copy of the $\mathcal{F}_{\mathrm{CRS}}$ functionality. The proof of security follows similarly, though it provides only computational security for *both* the sender and receiver when using a crs generated according to $\mathsf{SetupDec}_1$. Therefore it is strictly preferable to use a crs generated according to $\mathsf{SetupExt}$, which in our instantiation will also have the advantage of being a *uniform random* (not reference) string.

To efficiently implement $\mathsf{FindLossy}$ in extraction mode, our scheme also depends on a concurrent work by Peikert and Vaikuntanathan [PV07] on sampling a random lattice together with a trapdoor for distinguishing points close to the lattice from those far from the lattice. In particular, the trapdoor allows one to detect that a given public key for the LWE cryptosystem is message-lossy, as long as it meets certain conditions (which will be guaranteed by our dual-mode system). We direct the reader to [PV07] for details.

We now give our construction of a dual-mode cryptosystem. For clarity, we present the construction for $\ell = 1$ and omit the extra parameter $i \in [\ell]$, noting the changes required for the general case. We retain all the notation from Section 7.2. However, we note that the dual-mode cryptosystem always has message space $\mathbb{Z}_p$, even when $\ell > 1$. (The amortization is over ℓ individual OT executions, which each transfer a message in $\mathbb{Z}_p$.)

- $\mathsf{SetupExt}(1^n)$: choose a matrix $\mathbf{A} \leftarrow \mathbb{Z}_q^{m \times n}$ uniformly, together with a trapdoor t as described in [PV07]. For $b \in \{0,1\}$, choose column vector $\mathbf{u}_b \leftarrow \mathbb{Z}_q^{m \times 1}$ uniformly and indepedently. Let $\mathsf{crs} = (\mathbf{A}, \mathbf{u}_0, \mathbf{u}_1)$, and output (crs, t).

  $\mathsf{SetupDec}(1^n)$: choose a matrix $\mathbf{A} \leftarrow \mathbb{Z}_q^{m \times n}$ uniformly. Choose a column vector $\mathbf{w} \leftarrow \mathbb{Z}_q^{m \times 1}$ uniformly. For $b \in \{0,1\}$, choose secret column vector $\mathbf{s}_b \leftarrow \mathbb{Z}_q^{n \times 1}$ uniformly and error column vector $\mathbf{e}_b \leftarrow \chi^{m \times 1}$ (i.e., the m entries are chosen independently from error distribution χ). Let $\mathbf{u}_b = \mathbf{A}\mathbf{s}_b + \mathbf{e}_b - \mathbf{w}$. Let $\mathsf{crs} = (\mathbf{A}, \mathbf{u}_0, \mathbf{u}_1)$, let $t = (\mathbf{w}, \mathbf{s}_0, \mathbf{s}_1)$, and output (crs, t).

  In the following, all algorithms are implicitly provided the crs and parse it as $(\mathbf{A}, \mathbf{u}_0, \mathbf{u}_1)$.

- $\mathsf{KeyGen}(\sigma)$: choose a secret column vector $\mathbf{s} \leftarrow \mathbb{Z}_q^{n \times 1}$ uniformly and error column vector $\mathbf{e} \leftarrow \chi^{m \times 1}$. Let $pk = \mathbf{A}\mathbf{s} + \mathbf{e} - \mathbf{u}_\sigma$, let $sk = \mathbf{s}$, and output (pk, sk).

- $\mathsf{Enc}(pk, b, m)$: output $c \leftarrow \mathsf{LWEEnc}((\mathbf{A}, pk + \mathbf{u}_b), m)$.[3]

- $\mathsf{Dec}(sk, c)$: output $m \leftarrow \mathsf{LWEDec}(sk, c)$.

- $\mathsf{FindLossy}(t, pk)$: As described in [PV07], use the extraction trapdoor t to find a $b \in \{0,1\}$ such that $(\mathbf{A}, pk + \mathbf{u}_b)$ is a message-lossy public key for LWEEnc, i.e., such that $\delta(\mathbf{A}, pk + \mathbf{u}_b)$ is negligible. (Lemma 7.7 shows that at least one of the two public keys is indeed message-lossy.)

  Alternately, we may let $\mathsf{FindLossy}$ be an exponential-time algorithm that computes $\delta(\mathbf{A}, pk + \mathbf{u}_b)$ by brute-force enumeration for each $b \in \{0,1\}$. In the OT protocol, this translates to an exponential-time statistically-close simulation for a cheating receiver. While not sufficient for UC security, this is good enough for (say) concurrent composition of many OT executions (and no other protocols) between the same sender and receiver.

- $\mathsf{TrapKeyGen}(t)$: Parse the trapdoor t as $(\mathbf{w}, \mathbf{s}_0, \mathbf{s}_1)$, and output $(pk, sk_0, sk_1) = (\mathbf{w}, \mathbf{s}_0, \mathbf{s}_1)$.

For general ℓ, we make the following simple modifications: the setup algorithms independently choose $\mathbf{u}_b^{(i)}$ (in the case of $\mathsf{SetupExt}$) or $\mathbf{w}^{(i)}, \mathbf{s}_b^{(i)}$ (in the case of $\mathsf{SetupDec}$) for each $i \in [\ell]$ and $b \in \{0,1\}$ as above, placing the appropriate values in crs and t. On their ith executions, the algorithms use $\mathbf{w}^{(i)}$ instead of $\mathbf{w}$, etc., but use the same matrix $\mathbf{A}$ throughout. We stress that the algorithms and resulting OT protocol only reuse $\mathbf{A}$, and *not* the randomness of LWEEnc (as is done in the basic LWE cryptosystem). The reason is that we do not know how to prove security when reusing randomness in the context of an OT protocol, because a cheating receiver controls the public keys that are used for encryption. This may allow it to introduce correlations between the ciphertexts from different executions. Using fresh randomness in every execution preserves security.

We can also generalize the above system to a larger branch set $\{0,1\}^k$ for a 1-out-of-2^k OT protocol, by including vectors $\mathbf{u}_b$ for every $b \in \{0,1\}^k$ in the crs, chosen in an analogous way. The only difference in the proof of security is in Lemma 7.7, where the probability of the "bad" event increases by a polynomial factor (but remains negligible), due to a union bound over all pairs of branches.

[3] Technically, in order to guarantee that $\mathsf{FindLossy}$ works properly, we need to use a slight variant of LWEEnc from [PV07], whose essential properties are nonetheless the same as those of our system from Section 7.2.

7.3.1 Proof of Security

The proof that the above system comprises a dual-mode cryptosystem (according to our relaxed definition) is somewhat involved. Therefore we break it into separate lemmas relating to the indistinguishability of the two modes (Lemma 7.6), the guaranteed existence of message-lossy branches in extraction mode (Lemma 7.7), and trapdoor generation of keys in decryption mode (Lemma 7.8).

Lemma 7.6. *In the above system, the extraction and decryption modes are indistinguishable, i.e.,* $\mathsf{SetupExt}_1(1^n) \stackrel{c}{\approx} \mathsf{SetupDec}_1(1^n)$*, assuming* $\mathsf{LWE}_{q,\chi}$ *is hard.*

Proof. Consider the output of $\mathsf{SetupDec}_1$, which is of the form $(\mathbf{A}, (\mathbf{A}\mathbf{s}_0 + \mathbf{e}_0) - \mathbf{w}, (\mathbf{A}\mathbf{s}_1 + \mathbf{e}_1) - \mathbf{w})$. Now by the hardness of $\mathsf{LWE}_{q,\chi}$, the pairs $(\mathbf{A}, \mathbf{A}\mathbf{s}_1 + \mathbf{e}_1)$ and $(\mathbf{A}, \mathbf{w}')$, where $\mathbf{w}' \leftarrow \mathbb{Z}_q^{m\times 1}$ is uniform and independent, are computationally indistinguishable. Therefore the output of $\mathsf{SetupDec}_1$ is indistinguishable from a tuple $(\mathbf{A}, (\mathbf{A}\mathbf{s}_0 + \mathbf{e}_0) - \mathbf{w}, \mathbf{w}' - \mathbf{w})$. This tuple is totally uniform, because $\mathbf{w}$ and $\mathbf{w}'$ are uniform and independent. Because $\mathsf{SetupExt}_1$ produces a uniformly random output, the claim follows. □

Lemma 7.7. *Let* $R \geq 2\sqrt{q}$ *and let* $m \geq 2(n+1)\lg q$*. Let* $\epsilon = \sqrt{1/q^{n+1}}$*. Then with all but* 2ϵ *probability over the uniform choice of* $(\mathbf{A}, \mathbf{u}_0, \mathbf{u}_1)$ *by* $\mathsf{SetupExt}$, *every* $pk \in \mathbb{Z}_q^{m\times 1}$ *has a message-lossy branch. Specifically, for every* pk*,* $\delta(\mathbf{A}, pk + \mathbf{u}_b) \leq \epsilon$ *for some branch* $b \in \{0,1\}$*.*

Proof. We will first show that, for any *fixed* pk, the probability that pk lacks a message-lossy branch is *extremely* small. The claim will follow by inverting the quantifiers using a union bound.

First, say that a matrix $\mathbf{A} \in \mathbb{Z}_q^{m\times n}$ is "good" if the fraction of $\mathbf{b} \in \mathbb{Z}_q^{m\times 1}$ such that $\delta(\mathbf{A}, \mathbf{b}) > \epsilon$ is at most $q^{3(n+1)/2}/R^m$. Then by Lemma 7.4 (where $\ell = 1$) and Markov's inequality, all but an ϵ-fraction of $\mathbf{A}$'s are good. From now on, we assume that $\mathbf{A}$ is good, which happens with all but ϵ probability over the randomness of $\mathsf{SetupExt}$.

Now consider some fixed $pk \in \mathbb{Z}_q^{m\times 1}$. Because $\mathbf{u}_0, \mathbf{u}_1$ are uniform and independent, $pk + \mathbf{u}_0$ and $pk + \mathbf{u}_1$ are uniform and independent over $\mathbb{Z}_q^{m\times 1}$. Then by the assumption that $\mathbf{A}$ is good, we have

$$\Pr_{\mathbf{u}_0,\mathbf{u}_1}[\delta(\mathbf{A}, pk + \mathbf{u}_0) > \epsilon \text{ and } \delta(\mathbf{A}, pk + \mathbf{u}_1) > \epsilon] = \left(\Pr_{\mathbf{b}\leftarrow\mathbb{Z}_q^{m\times 1}}[\delta(\mathbf{A}, \mathbf{b}) > \epsilon]\right)^2 \leq \frac{q^{3(n+1)}}{R^{2m}}.$$

Now by a union bound over all $pk \in \mathbb{Z}_q^{m\times 1}$, we have

$$\Pr_{\mathbf{u}_0,\mathbf{u}_1}[\exists\, pk : \ \delta(\mathbf{A}, pk + \mathbf{u}_0) > \epsilon \text{ and } \delta(\mathbf{A}, pk + \mathbf{u}_1) > \epsilon] \leq \frac{q^m \cdot q^{3(n+1)}}{R^{2m}}. \tag{1}$$

By assumption on R and m, we have

$$R^{2m} \geq (2\sqrt{q})^{2m} = q^m \cdot 2^{2m} > q^m \cdot q^{4(n+1)}.$$

Therefore the probability in (1) is at most $1/q^{n+1} < \epsilon$, and we are done. □

Lemma 7.8. *The above system satisfies our relaxed Property* $4'$*, assuming* $\mathsf{LWE}_{q,\chi}$ *is hard.*

Proof. For simplicity, we will prove the lemma for $\ell = 1$; the general case follows by a hybrid argument over the ℓ independent sets of variables in the crs and the ℓ calls to $\mathsf{KeyGen}/\mathsf{TrapKeyGen}$

that use these variables. We also prove the lemma just for the case $\sigma = 0$; the other case follows symmetrically.

Our goal is to prove

$$(\mathsf{SetupDec}_1(1^n), \mathsf{KeyGen}(0)) \overset{c}{\approx} (\mathsf{crs}, (pk, sk_0)), \tag{2}$$

where in the right-hand side $(\mathsf{crs}, t) \leftarrow \mathsf{SetupDec}(1^n)$ and $(pk, sk_0, sk_1) \leftarrow \mathsf{TrapKeyGen}(t)$. We do so via a sequence of games.

First consider a hybrid game that produces $(\mathsf{SetupExt}_1(1^n), \mathsf{KeyGen}(0))$; this is indistinguishable from the left-hand side of (2) by indistinguishability of modes (which we prove below in Theorem 7.9 for our specific construction). The output of this game decomposes as

$$(\mathbf{A}, \quad \mathbf{u}_0, \quad \mathbf{u}_1, \quad \mathbf{As} + \mathbf{e} - \mathbf{u}_0, \quad \mathbf{s}),$$

where $\mathbf{A}$, $\mathbf{u}_0$, $\mathbf{u}_1$, and $\mathbf{s}$ are uniform (in their respective domains) and $\mathbf{e} \leftarrow \chi^{m \times 1}$.

Now rename variables in the above game, defining $\mathbf{w} = \mathbf{As} + \mathbf{e} - \mathbf{u}_0$ and renaming $\mathbf{s}$ and $\mathbf{e}$ as $\mathbf{s}_0$ and $\mathbf{e}_0$ (respectively). The above game is therefore equivalent to one that outputs

$$(\mathbf{A}, \quad \mathbf{As}_0 + \mathbf{e}_0 - \mathbf{w}, \quad \mathbf{u}_1, \quad \mathbf{w}, \quad \mathbf{s}_0),$$

where $\mathbf{w}$ is uniform. Now because $\mathbf{u}_1$ is uniform and independent of the other variables, the preceding game is equivalent to one that outputs

$$(\mathbf{A}, \quad \mathbf{As}_0 + \mathbf{e}_0 - \mathbf{w}, \quad \mathbf{u}_1 - \mathbf{w}, \quad \mathbf{w}, \quad \mathbf{s}_0).$$

Finally, the hardness of $\mathsf{LWE}_{q,\chi}$ implies that $(\mathbf{A}, \mathbf{u}_1)$ is indistinguishable from $(\mathbf{A}, \mathbf{As}_1 + \mathbf{e}_1)$, where $\mathbf{s}_1 \leftarrow \mathbb{Z}_q^{m \times 1}$ and $\mathbf{e}_1 \leftarrow \chi^{m \times 1}$. Therefore the prior game is indistinguishable from one that outputs

$$(\mathbf{A}, \quad \mathbf{As}_0 + \mathbf{e}_0 - \mathbf{w}, \quad \mathbf{As}_1 + \mathbf{e}_1 - \mathbf{w}, \quad \mathbf{w}, \quad \mathbf{s}_0).$$

This game is, by definition, equivalent to the right-hand side of (2), and we are done. □

7.3.2 Putting Everything Together

We now instantiate all the parameters of the dual-mode cryptosystem to satisfy the various constraints that have accumulated, and connect LWE to the (quantum) hardness of lattice problems.

First we instantiate the parameters (though we have made no effort to optimize the various constant or polylog(n) factors). Recall that $\mathbb{Z}_p$ is the message space. Let $p = n^c$ for some positive $c = c(n)$ bounded above by a constant. Let $m = (4c + 6)(n + 1) \lg n = O(n \lg n)$. Let q be a prime in $[200, 400] \cdot p^2 \cdot m^2 \cdot \lg^2 n = \tilde{O}(n^{2c+2})$, and let $R = 40p \cdot m \cdot \lg n = \tilde{O}(n^{c+1})$. Finally, let the error distribution $\chi = \bar{\Psi}_\alpha$ for $\alpha = 1/(R \cdot p \cdot \sqrt{m} \cdot \lg n) = 1/\tilde{O}(n^{2c+3/2})$.

Theorem 7.9. *The above system is a dual-mode cryptosystem (according to the relaxed definition), assuming that either of the lattice problems described in Proposition 7.1 are hard for any efficient quantum algorithm to approximate to within some $\tilde{O}(n^{2c+5/2})$ factor.*

Proof. As long as $\mathsf{LWE}_{q,\chi}$ is hard (which we show below), indistinguishability of modes follows directly from Lemma 7.6, and trapdoor key generation follows directly from Lemma 7.8.

We now show completeness of our dual-mode cryptosystem, which will follow directly from the correctness of LWEDec. By Lemma 7.2, LWEDec decrypts correctly (with overwhelming probability) for our choice of α, as long as $q \geq 4R \cdot p \cdot m$. Indeed, $4R \cdot p \cdot m = 160p^2m^2 \lg n < 200p^2m^2 \lg^2 n \leq q$.

Now we show extraction of a message-lossy branch. As required by Lemma 7.7, we have $R = 40pm \lg n = 2 \cdot \sqrt{400p^2m^2 \lg^2 n} \geq 2\sqrt{q}$. Also as required by Lemma 7.7, we have (for all sufficiently large n)

$$\begin{aligned} 2(n+1)\lg q &\leq 2(n+1) \cdot \lg(n^{2c+3}) \\ &\leq (4c+6) \cdot (n+1) \lg n = m. \end{aligned}$$

Therefore, with overwhelming probability over $\mathsf{SetupExt}$'s uniform choice of $\mathsf{crs} = (\mathbf{A}, \mathbf{u}_0, \mathbf{u}_1)$, every (possibly malformed) public key pk has some message-lossy branch b. Efficiently finding such a branch follows from [PV07].

We conclude by justifying the hardness of $\mathsf{LWE}_{q,\chi}$. Using the fact that $q \geq 4R \cdot p \cdot m \cdot \lg n$, we have (as required by Proposition 7.1),

$$q \cdot \alpha \geq \frac{4R \cdot p \cdot m \cdot \lg n}{R \cdot p \cdot \sqrt{m} \cdot \lg n} = 4\sqrt{m} > 2\sqrt{n}.$$

Therefore we may apply Proposition 7.1, which for our choice of α means that hardness of $\mathsf{LWE}_{q,\chi}$ is implied by quantum hardness of lattice problems to within some factor $\tilde{O}(n/\alpha) = \tilde{O}(n^{2c+5/2})$. $\square$

8 Acknowledgments

We thank Oded Regev for suggesting a cleaner and tighter proof of Lemma 7.4.

References

[AD97] Miklós Ajtai and Cynthia Dwork. A public-key cryptosystem with worst-case/average-case equivalence. In *STOC*, pages 284–293, 1997.

[BGW88] Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In *STOC*, pages 1–10, 1988.

[Can01] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *FOCS*, pages 136–145, 2001.

[CF01] Ran Canetti and Marc Fischlin. Universally composable commitments. In *CRYPTO*, pages 19–40, 2001.

[CLOS02] Ran Canetti, Yehuda Lindell, Rafail Ostrovsky, and Amit Sahai. Universally composable two-party and multi-party secure computation. In *STOC*, pages 494–503, 2002.

[CNS07] Jan Camenisch, Gregory Neven, and Abhi Shelat. Simulatable adaptive oblivious transfer. In *EUROCRYPT*, pages 573–590, 2007.

[Coc01] Clifford Cocks. An identity based encryption scheme based on quadratic residues. In *IMA Int. Conf.*, pages 360–363, 2001.

[CR03] Ran Canetti and Tal Rabin. Universal composition with joint state. In *CRYPTO*, pages 265–281, 2003.

[Cré87] Claude Crépeau. Equivalence between two flavours of oblivious transfers. In *CRYPTO*, pages 350–354, 1987.

[CS02] Ronald Cramer and Victor Shoup. Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In *EUROCRYPT*, pages 45–64, 2002.

[GH07] Matthew Green and Susan Hohenberger. Blind identity-based encryption and simulatable oblivious transfer. Cryptology ePrint Archive, Report 2007/235, 2007. `http://eprint.iacr.org/`.

[GMW87] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or a completeness theorem for protocols with honest majority. In *STOC*, pages 218–229, 1987.

[GOS06] Jens Groth, Rafail Ostrovsky, and Amit Sahai. Perfect non-interactive zero knowledge for np. In *EUROCRYPT*, pages 339–358, 2006.

[HK07] Omer Horvitz and Jonathan Katz. Universally-composable two-party computation in two rounds. In *CRYPTO*, 2007.

[IZ89] Russell Impagliazzo and David Zuckerman. How to recycle random bits. In *FOCS*, pages 248–253, 1989.

[Kal05] Yael Tauman Kalai. Smooth projective hashing and two-message oblivious transfer. In *EUROCRYPT*, pages 78–95, 2005.

[Kil88] Joe Kilian. Founding cryptography on oblivious transfer. In *STOC*, pages 20–31, 1988.

[LP07] Yehuda Lindell and Benny Pinkas. An efficient protocol for secure two-party computation in the presence of malicious adversaries. In *EUROCRYPT*, pages 52–78, 2007.

[MNPS04] Dahlia Malkhi, Noam Nisan, Benny Pinkas, and Yaron Sella. Fairplay - secure two-party computation system. In *USENIX Security Symposium*, pages 287–302, 2004.

[NP01] Moni Naor and Benny Pinkas. Efficient oblivious transfer protocols. In *SODA*, pages 448–457, 2001.

[Pei07] Chris Peikert. Limits on the hardness of lattice problems in ℓ_p norms. In *IEEE Conference on Computational Complexity*, pages 333–346, 2007.

[PV07] Chris Peikert and Vinod Vaikuntanathan. Manuscript., 2007.

[PW07] Chris Peikert and Brent Waters. Lossy trapdoor functions and their applications. Cryptology ePrint Archive, Report 2007/279, 2007. `http://eprint.iacr.org/`.

[Rab81] Michael O. Rabin. How to exchange secrets by oblivious transfer. Technical report, Harvard University, 1981.

[Reg04] Oded Regev. New lattice-based cryptographic constructions. *J. ACM*, 51(6):899–942, 2004.

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *STOC*, pages 84–93, 2005.

[Yao86] Andrew Chi-Chih Yao. How to generate and exchange secrets (extended abstract). In *FOCS*, pages 162–167, 1986.