ACM India Summer School 2024 on Theoretical Foundations of Cryptography[†]

Venkata Koppula[‡]

June 12th, 2024

Contents

1 Advanced Cryptographic Primitives

So far in the course, we have seen cryptographic primitives such as secret key encryption, public key encryption, digital signatures etc. These are widely used in practice. Today, we will discuss encryption primitives which go beyond public key encryption. We start with identity based encryption (IBE), then move on to a richer generalization called attribute based encryption (ABE), and finally discuss functional encryption (FE), which captures IBE, ABE and much more.

1.1 *Definitions and Basic Properties*

Identity Based Encryption (IBE)

Consider the following scenario: there is a large organization, and every person at this organization has a unique identity. Typically, if anyone wants to email a particular person at this organization, they would need the person's public key. As a result, the organization needs to maintain everyone's public keys, which could be a key management headache. Using identity based encryption, one can encrypt using only the person's identity (and some master public key which is common for the whole organization). Each person will have a secret key corresponding to his/her identity, which will allow him/her to decrypt only the ciphertexts meant for his/her identity. This requires a master authority who

[†]These notes are meant to be a self-contained summary of the talks on advanced encryption primitives. I have not proof-read the notes, and therefore it might contain some minor typos and mistakes. Please feel free to contact in case something is unclear/looks incorrect.

[‡]kvenkata@cse.iitd.ac.in

samples a master secret key and a master public key. The master secret key can be used to generate a secret key for any identity.
Below we introduce the syntax formally.

Syntax. An IBE scheme for identity space $\mathcal{ID}$ and message space $\mathcal{M}$ consists of four algorithms with the following syntax.

- $\mathsf{Setup}\left(1^\lambda\right)$: The setup algorithm outputs the master public key mpk and the master secret key msk.
- $\mathsf{Enc}\left(\mathsf{mpk}, \mathsf{id}, m\right)$: The encryption algorithm takes as input the master public key, identity $\mathsf{id} \in \mathcal{ID}$, and message $m \in \mathcal{M}$. It outputs a ciphertext $\mathsf{ct}_{\mathsf{id}}$.
- $\mathsf{KeyGen}\left(\mathsf{msk}, \mathsf{id}\right)$: The key generation algorithm takes as input the master secret key msk, and an identity $\mathsf{id} \in \mathcal{ID}$, and outputs a secret key $\mathsf{sk}_{\mathsf{id}}$.
- $\mathsf{Dec}\left(\mathsf{sk}_{\mathsf{id}}, \mathsf{ct}_{\mathsf{id}}\right)$: The decryption algorithm takes as input a secret key for identity id, ciphertext $\mathsf{ct}_{\mathsf{id}}$, and outputs $y \in \mathcal{M} \cup \{\perp\}$.

Correctness. Here, we define perfect correctness (statistical correctness can be defined analogously). We require that for all $\mathsf{id} \in \mathcal{ID}$, $m \in \mathcal{M}$, $\lambda \in \mathbb{N}$, $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}\left(1^\lambda\right)$, $\mathsf{ct}_{\mathsf{id}} \leftarrow \mathsf{Enc}\left(\mathsf{mpk}, \mathsf{id}, m\right)$, $\mathsf{sk}_{\mathsf{id}} \leftarrow \mathsf{KeyGen}\left(\mathsf{msk}, \mathsf{id}\right)$,

$$\mathsf{Dec}\left(\mathsf{sk}_{\mathsf{id}}, \mathsf{ct}_{\mathsf{id}}\right) = m.$$

Security. First, let us informally discuss what properties we need from a secure IBE system. Suppose a ciphertext $\mathsf{ct}_{\mathsf{id}}$ is meant for identity id. If an attacker does not have the secret key for identity id, then it should learn no information about the underlying message. Note that the attacker may collude with other users, and have secret keys for many other identities. Naturally, we require that given many secret keys, it should be hard to derive a secret key for a new identity.
We will now discuss how to formalize the above intuition. The following are some candidate definitions that were proposed during the lecture, we discuss the shortcomings in each of these definitional attempts. Let us assume the message space is $\{0,1\}$.

1. For every identity id, the following distributions are computationally indistinguishable:

$$\begin{aligned} &\{\mathsf{Enc}(\mathsf{mpk}, \mathsf{id}, 0) \ : \ (\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}\} \\ &\qquad\qquad \approx_c \\ &\{\mathsf{Enc}(\mathsf{mpk}, \mathsf{id}, 1) \ : \ (\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}\} \end{aligned}$$

   This definition does not capture the secret keys that the adversary can obtain, and hence is too weak to be used.

2. For every identity id, the following distributions are computationally in-

distinguishable:

$$\left\{\begin{pmatrix} \mathsf{msk}, \\ \mathsf{Enc}(\mathsf{mpk}, \mathsf{id}, 0) \end{pmatrix} : (\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}\right\}$$
$$\approx_c$$
$$\left\{\begin{pmatrix} \mathsf{msk}, \\ \mathsf{Enc}(\mathsf{mpk}, \mathsf{id}, 1) \end{pmatrix} : (\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}\right\}$$

In this definition, the adversary gets the master secret key together with the challenge ciphertext. This definition is too strong, as the adversary can use msk to derive a secret key for id, and therefore decrypt the challenge ciphertext.

3. For every identity id, the following distributions are computationally indistinguishable:

$$\left\{\begin{pmatrix} \{\mathsf{sk}_{\mathsf{id}'}\}_{\mathsf{id}' \neq \mathsf{id}}, \\ \mathsf{Enc}(\mathsf{mpk}, \mathsf{id}, 0) \end{pmatrix} : \begin{array}{l} (\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}\left(1^\lambda\right) \\ \mathsf{sk}_{\mathsf{id}'} \leftarrow \mathsf{KeyGen}\left(\mathsf{msk}, \mathsf{id}'\right) \end{array}\right\}$$
$$\approx_c$$
$$\left\{\begin{pmatrix} \{\mathsf{sk}_{\mathsf{id}'}\}_{\mathsf{id}' \neq \mathsf{id}}, \\ \mathsf{Enc}(\mathsf{mpk}, \mathsf{id}, 1) \end{pmatrix} : \begin{array}{l} (\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}\left(1^\lambda\right) \\ \mathsf{sk}_{\mathsf{id}'} \leftarrow \mathsf{KeyGen}\left(\mathsf{msk}, \mathsf{id}'\right) \end{array}\right\}$$

This definition is almost correct, the only issue here is that the identity space can be exponential in the security parameter, in which case the adversary receives an exponential-sized input. Instead of giving secret keys for all identities other than id, we can allow the adversary to query for secret keys corresponding to any identity of its choice. We formalize this definition below, using a security game between a challenger and an adversary.

Definition 1.1. *An IBE scheme is said to be CPA secure if, for any p.p.t. adversary $\mathcal{A}$, there exists a negligible function* negl *such that for all $\lambda \in \mathbb{N}$, the advantage of $\mathcal{A}$ in the IBE security game (defined in Figure 1) is at most* $\mathrm{negl}(\lambda)$.
An IBE scheme is said to be selectively secure if any p.p.t. adversary has negligible advantage in the selective CPA security game, *which is identical to the IBE security game, except that the adversary must send its challenge identity at the start of the experiment (before receiving the master public key).* ◊

Note that the above definition only guarantees that the ciphertext hides the message, the ciphertext may not hide the identity.

> **Exercise 1.1.** *Propose a security game which captures IBE schemes where the ciphertext hides both the message and identity. Such IBE schemes are called* anonymous *IBE schemes.*

An application of IBE: CCA secure PKE. Canetti, Halevi and Katz [CHK04] showed that any IBE scheme can be used to build a CCA secure PKE scheme. Below, we first present a CCA-1 secure PKE scheme using an IBE scheme as a building block.

CPA security game for IBE

- (Setup Phase) Challenger samples $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}\left(1^\lambda\right)$ and sends mpk to the adversary.
- (Pre-challenge Query Phase) Adversary makes polynomially many secret key queries. For each queried identity id, the challenger sends $\mathsf{sk}_{\mathrm{id}} \leftarrow \mathsf{KeyGen}(\mathsf{msk}, \mathrm{id})$.
- (Challenge Phase) Adversary sends a challenge identity id^* (not equal to any of the identities queried in the pre-challenge phase), and two messages m_0, m_1. Challenger samples $b \leftarrow \{0,1\}$ and sends $\mathsf{ct}^* \leftarrow \mathsf{Enc}\left(\mathsf{mpk}, \mathrm{id}^*, m_b\right)$.
- (Post-challenge Query Phase) Adversary makes polynomially many secret key queries. For each queried identity $\mathrm{id} \neq \mathrm{id}^*$, the challenger sends $\mathsf{sk}_{\mathrm{id}} \leftarrow \mathsf{KeyGen}(\mathsf{msk}, \mathrm{id})$.
- Adversary finally sends it guess b'. The adversary wins if $b = b'$, and the advantage of the adversary is $\Pr[\text{Adversary wins}] - 1/2$.

Figure 1: IBE security is captured using a security game between a challenger and an adversary. The adversary receives polynomially many secret keys, corresponding to identities of its choice. At the end, it must distinguish between encryption of m_0 and m_1 for the challenge identity.

Construction 1.2 (CCA-1 secure PKE using IBE [CHK04])**.** *Let $\mathcal{ID} = \{0,1\}^\lambda$, and let* $(\mathsf{IBE.Setup}, \mathsf{IBE.Enc}, \mathsf{IBE.KeyGen}, \mathsf{IBE.Dec})$ *be an IBE scheme for identity space $\mathcal{ID}$ and message space $\mathcal{M}$. We will construct a CCA-1 secure encryption scheme with message space $\mathcal{M}$.*

- $\mathsf{Setup}\left(1^\lambda\right)$: *The setup algorithm samples* $(\mathsf{ibe.mpk}, \mathsf{ibe.msk}) \leftarrow \mathsf{IBE.Setup}\left(1^\lambda\right)$*, and sets* $\mathsf{pk} = \mathsf{ibe.mpk}$, $\mathsf{sk} = \mathsf{ibe.msk}$.
- $\mathsf{Enc}(\mathsf{pk} = \mathsf{ibe.mpk}, m)$: *The encryption algorithm samples* $\mathrm{id} \leftarrow \{0,1\}^\lambda$*, and computes* $\mathsf{ibe.ct} \leftarrow \mathsf{IBE.Enc}(\mathsf{ibe.mpk}, \mathrm{id}, m)$*. The ciphertext is* $(\mathrm{id}, \mathsf{ibe.ct})$.
- $\mathsf{Dec}(\mathsf{sk} = \mathsf{ibe.msk}, \mathsf{ct} = (\mathrm{id}, \mathsf{ibe.ct}))$: *The decryption algorithm first computes* $\mathsf{ibe.sk} \leftarrow \mathsf{IBE.KeyGen}(\mathsf{ibe.msk}, \mathrm{id})$*. It outputs* $\mathsf{IBE.Dec}(\mathsf{ibe.sk}, \mathsf{ibe.ct})$.

$\Diamond$

Correctness of the PKE scheme follows from the correctnes of the IBE scheme. Below, we describe a reduction algorithm, which proves CCA-1 security of the scheme assuming the IBE scheme is CPA secure.

Claim 1.3. *Suppose there exists a p.p.t. adversary that breaks CCA-1 security of the above scheme. Then there exists a p.p.t. reduction algorithm that breaks the CPA security of the IBE scheme.*

Proof. The reduction algorithm receives the master public key from the IBE challenger, which it forwards to the CCA-1 adversary. Next, it receives polynomially

many decryption queries. For each decryption query of the form $(\mathsf{id}_i, \mathsf{ibe.ct}_i)$, the reduction algorithm sends a secret key query to the IBE challenger, and receives $\mathsf{ibe.sk}_i$. It uses $\mathsf{ibe.sk}_i$ to decrypt $\mathsf{ibe.ct}_i$. Finally, after all decryption queries, the CCA-1 adversary sends two challenge messages m_0, m_1. The reduction algorithm samples a uniformly random identity id^*. With overwhelming probability, this identity is not equal to any of the queried identities. It sends id^* together with challenge messages m_0, m_1, and receives a challenge ciphertext ct^*. It forwards $(\mathsf{id}^*, \mathsf{ct}^*)$ to the adversary, and forwards the adversary's response to the IBE challenger. □

The above scheme can be upgraded to a full CCA-secure PKE scheme using one-time signatures. The only difference is that the encryption algorithm samples a signing and verification key, and the verification key is used as the identity for encryption. After computing the IBE ciphertext, the encryption algorithm computes a signature on the ciphertext. The signature ensures that in all post-challenge decryption queries, the verification key is not the same as the verification key in the challenge ciphertext.

Exercise 1.2. *Show that any IBE scheme can be used to build a secure signature scheme.*

Attribute Based Encryption (ABE)

Attribute Based Encryption is a generalization of IBE which captures more *fine-grained access control* on the data. Here, we can encrypt messages using any *policy*. Users will have certain attributes, and they will get secret keys corresponding to their attributes. The secret key for attribute att can decrypt a ciphertext for policy f if $f(\mathsf{att}) = 1$.
As an example, consider a university where everyone has the following attributes: $\mathtt{name}, \mathtt{department} \in \{\mathtt{CSE}, \mathtt{EE}, \mathtt{Math}\}, \mathtt{category} \in \{\mathtt{student}, \mathtt{staff}, \mathtt{faculty}\}$. Suppose we want to encrypt a message that can be decrypted by any student in the CSE department, or any faculty member. Then, we can encrypt this message using the policy

$$(\mathtt{department} = \mathtt{CSE} \wedge \mathtt{category} = \mathtt{student}) \vee (\mathtt{category} = \mathtt{faculty}).$$

Note that IBE is a special case of ABE where the policies are implemented using equality check. An ABE scheme is defined for a class of policies. For simplicity, we will think of policies as circuits, and attributes as bit-vectors. One can modify the syntax appropriately for other classes of policies and attributes. When policies are tied to ciphertexts, we call it *ciphertext-policy* ABE (CP-ABE). One can also define ABE where policies are tied to the secret keys, this is called *key-policy* ABE (KP-ABE). Below, we define the syntax for CP-ABE.

Syntax. An ciphertext-policy ABE scheme for attribute space $\{0,1\}^\lambda$, policy space $\mathcal{C} = \left\{C : \{0,1\}^\lambda \to \{0,1\}\right\}$, and message space $\mathcal{M}$ consists of four algorithms with the following syntax.

- $\mathsf{Setup}\left(1^\lambda\right)$: The setup algorithm outputs the master public key mpk and the master secret key msk.

- $\mathsf{Enc}(\mathsf{mpk}, C, m)$: The encryption algorithm takes as input the master public key, circuit $C \in \mathcal{C}$, and message $m \in \mathcal{M}$. It outputs a ciphertext ct_C.
- $\mathsf{KeyGen}(\mathsf{msk}, \mathsf{att})$: The key generation algorithm takes as input the master secret key msk, and an attribute vector $\mathsf{att} \in \{0,1\}^\lambda$, and outputs a secret key $\mathsf{sk}_{\mathsf{att}}$.
- $\mathsf{Dec}(\mathsf{sk}_{\mathsf{att}}, \mathsf{ct}_C)$: The decryption algorithm takes as input a secret key for attribute att, ciphertext ct_C, and outputs $y \in \mathcal{M} \cup \{\perp\}$.

Correctness. Here, we define perfect correctness (statistical correctness can be defined analogously). We require that for all $\lambda \in \mathbb{N}$, $C \in \mathcal{C}$, $\mathsf{att} \in \{0,1\}^\lambda$, $m \in \mathcal{M}$, $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}\left(1^\lambda\right)$, $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{mpk}, C, m)$, $\mathsf{sk} \leftarrow \mathsf{KeyGen}(\mathsf{msk}, \mathsf{att})$, if $C(x) = 1$, then $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) = m$.

Security. The security definition is similar to the IBE security definition.

CPA security game for CP-ABE

- (Setup Phase) Challenger samples $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}\left(1^\lambda\right)$ and sends mpk to the adversary.
- (Pre-challenge Query Phase) Adversary makes polynomially many secret key queries. For each queried identity att, the challenger sends $\mathsf{sk}_{\mathsf{att}} \leftarrow \mathsf{KeyGen}(\mathsf{msk}, \mathsf{att})$.
- (Challenge Phase) Adversary sends a challenge policy C (such that $C(\mathsf{att}) = 0$ for any attribute att queried in the pre-challenge phase), and two messages m_0, m_1. Challenger samples $b \leftarrow \{0,1\}$ and sends $\mathsf{ct}^* \leftarrow \mathsf{Enc}(\mathsf{mpk}, C, m_b)$.
- (Post-challenge Query Phase) Adversary makes polynomially many secret key queries. For each queried attribute att, $C(\mathsf{att}) = 0$, the challenger sends $\mathsf{sk}_{\mathsf{att}} \leftarrow \mathsf{KeyGen}(\mathsf{msk}, \mathsf{att})$.
- Adversary finally sends it guess b'. The adversary wins if $b = b'$, and the advantage of the adversary is $\Pr[\text{Adversary wins}] - 1/2$.

Figure 2: The adversary receives polynomially many secret keys, corresponding to attributes of its choice. At the end, it must distinguish between encryption of m_0 and m_1 for the challenge policy C, where $C(x) = 0$ for all attributes x queried by the adversary.

Definition 1.4. *An ABE scheme is said to be CPA secure if, for any p.p.t. adversary $\mathcal{A}$, there exists a negligible function* negl *such that for all $\lambda \in \mathbb{N}$, the advantage of $\mathcal{A}$ in the ABE security game (defined in Figure 2) is at most* $\mathrm{negl}(\lambda)$.
An ABE scheme is said to be selectively secure if any p.p.t. adversary has negligible advantage in the selective CPA security game, *which is identical to the ABE security game, except that the adversary must send its challenge policy at the start of the experiment (before receiving the master public key).* ◊

Functional Encryption (FE)

Finally, we discuss the notion of functional encryption. This is the most expressive encryption primitive. Here, anyone can encrypt a message using the master public key. Different users can learn different functions of the message by decrypting the ciphertext using a secret key corresponding to a function f. As in IBE and ABE, the secret keys are generated by a master authority using a master secret key. Informally, we require that if an adversary has secret keys for functions $f_1, f_2, \ldots, f_t$, and has an encryption of x, then it should learn only $f_1(x), f_2(x), \ldots, f_t(x)$ and nothing else.

Syntax. An FE scheme for input space $\mathcal{D}_\lambda$, function space $\mathcal{C}_\lambda = \{C : \mathcal{D}_\lambda \to \mathcal{R}_\lambda\}$ consists of four algorithms with the following syntax.

- $\mathsf{Setup}\left(1^\lambda\right)$: The setup algorithm outputs the master public key mpk and the master secret key msk.
- $\mathsf{Enc}\left(\mathsf{mpk}, x\right)$: The encryption algorithm takes as input the master public key, input $x \in \mathcal{D}_\lambda$. It outputs a ciphertext ct.
- $\mathsf{KeyGen}\left(\mathsf{msk}, C\right)$: The key generation algorithm takes as input the master secret key msk, and a function C, and outputs a secret key sk_C.
- $\mathsf{Dec}\left(\mathsf{sk}_C, \mathsf{ct}\right)$: The decryption algorithm takes as input a secret key for function C, ciphertext ct, and outputs $y \in \mathcal{R}_\lambda \cup \{\perp\}$.

Correctness. We require that for all $\lambda \in \mathbb{N}$, $C \in \mathcal{C}_\lambda$, $x \in \mathcal{D}_\lambda$, $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}\left(1^\lambda\right)$, $\mathsf{ct} \leftarrow \mathsf{Enc}\left(\mathsf{mpk}, x\right)$, $\mathsf{sk}_C \leftarrow \mathsf{KeyGen}\left(\mathsf{msk}, C\right)$, $\mathsf{Dec}\left(\mathsf{sk}_C, \mathsf{ct}\right) = C(x)$.

Security. The indistinguishability-based definition for functional encryption is very similar to the ABE definition. The adversary sends two strings x_0, x_1 as the challenge messages, and receives encryption of one of them. It is allowed secret key queries for any function f, as long as $f(x_0) = f(x_1)$. Finally, after polynomially many queries, the adversary must guess whether x_0 or x_1 was encrypted. Note that we only require that the ciphertext hides the message x, the secret key need not hide the circuit.

Exercise 1.3. *Show that FE for circuits implies ABE for circuits.*

Exercise 1.4. *Single-query FE security: consider a weaker security game where the adversary sends two messages x_0, x_1 and a circuit C such that $C(x_0) = C(x_1)$. It receives the master public key mpk, encryption of x_b and a secret key for circuit C. The adversary must guess whether x_0 was encrypted or x_1.*
Propose an FE scheme that is secure with regard to this security game.
(Hint: use garbled circuits and public key encryption.)

1.2 *Constructions*

In this section, we present various constructions of IBE/ABE schemes from lattice-based assumptions. We will first present a *toolkit* for lattice-based cryptography, and then see how to use these tools for building advanced encryption primitives (we will start with public key encryption, then discuss two constructions of IBE, and finally an ABE scheme for inner products).

Toolkit for Lattice-Based Cryptography

Throughout this section, all computations are modulo q, where q is a large modulus. We will use the following parameters for our discussion. Let n denote the security parameter, $m = n^2$ and $q = \Theta\left(2^{\sqrt{n}}\right)$. We say that a number $x \in \mathbb{Z}_q$ is *small* if $x \leq \sqrt{q}$, otherwise we say that x is *large*.

The Leftover Hash Lemma. Consider the following experiment: sample a uniformly random matrix $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}$, a uniformly random binary vector $\mathbf{r} \leftarrow \{0,1\}^m$. Given $\mathbf{A}$ and $\mathbf{A} \cdot \mathbf{r}$, can we learn any information about $\mathbf{r}$? The following lemma says that $\mathbf{A}$ and $\mathbf{A} \cdot \mathbf{r}$ are statistically indistinguishable from a uniformly random matrix and a uniformly random vector (and therefore there is negligible information about $\mathbf{r}$).

Lemma 1.5. *Let $m = n^2$, and let $q = \Theta\left(2^{\sqrt{n}}\right)$ be a prime. Then the following distributions are statistically indistinguishable:*

$$\left\{ (\mathbf{A}, \mathbf{A} \cdot \mathbf{r}) \ : \ \begin{array}{c} \mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m} \\ \mathbf{r} \leftarrow \{0,1\}^m \end{array} \right\} \approx_s \left\{ (\mathbf{A}, \mathbf{u}) \ : \ \begin{array}{c} \mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m} \\ \mathbf{u} \leftarrow \mathbb{Z}_q^n \end{array} \right\}$$

The above lemma holds as long as $\mathbf{r}$ is sampled from a distribution with sufficient min-entropy.

The Learning with Errors Assumption. We will use the following version of the Learning with Errors (LWE) assumption.

Computational Problem 1. *Let $m = n^2$, let $q = \Theta\left(2^{\sqrt{n}}\right)$ be a prime, and $B = O\left(q^{1/3}\right)$. Then the following distributions are computationally indistinguishable:*

$$\left\{ \left(\mathbf{A}, \mathbf{s}^\top \cdot \mathbf{A} + \mathbf{e}\right) \ : \ \begin{array}{c} \mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m} \\ \mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{e} \leftarrow [-B, B]^m \end{array} \right\} \approx_c \left\{ (\mathbf{A}, \mathbf{u}) \ : \ \begin{array}{c} \mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m} \\ \mathbf{u} \leftarrow \mathbb{Z}_q^m \end{array} \right\}$$

A few observations about the LWE problem/assumption:

- if the 'error vector' $\mathbf{e}$ was sampled uniformly at random from $\mathbb{Z}_q^m$, then these two distributions would be identical.
- Without the error vector, one can easily distinguish between these two distributions by using Gaussian elimination. Somewhat surprisingly, when the error vector is added, we don't have any polynomial time algorithm for this problem. Any such algorithm would end up resolving decades old computational problems.

However, a computationally unbounded adversary **can** distinguish between these two distributions.

- The LWE problem seems to be resilient against quantum algorithms too! This makes it one of the leading candidates for *post-quantum cryptography* (cryptography that is secure in the presence of quantum adversaries).

Finding short preimages. Let $\mathbf{v} \in \mathbb{Z}_q^n$ be any vector, and $\mathbf{A}$ a uniformly random matrix. Given $\mathbf{A}$ and $\mathbf{v}$, one can easily find a vector $\mathbf{w}$ such that $\mathbf{A} \cdot \mathbf{w} = \mathbf{v}$ (there are several such vectors $\mathbf{w}$). However, the problem becomes interesting if we also require the preimage vector to have small entries. We will call such a vector $\mathbf{w}$ a short preimage of $\mathbf{v}$ wrt $\mathbf{A}$. The following exercise shows that it is computationally hard to find short preimages (assuming the LWE assumption holds) wrt uniformly random matrices.

Exercise 1.5. *Consider the following experiment between a challenger and an adversary $\mathcal{A}$, with security parameter n, $m = n^2$ and $q = \Theta\left(q^{\sqrt{n}}\right)$:*
The adversary sends a vector $\mathbf{v} \leftarrow \mathbb{Z}_q^n$ to the challenger. The challenger samples $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}$ and sends $\mathbf{A}$ to $\mathcal{A}$. The adversary sends a vector $\mathbf{w}$, and wins if $\mathbf{A} \cdot \mathbf{w} = \mathbf{v}$ and all entries of $\mathbf{w}$ are smaller than $q^{1/4}$.
Show that if there exists a p.p.t. adversary that can win the above experiment with non-negligible probability, then there exists a p.p.t. adversary $\mathcal{B}$ that breaks the LWE assumption with non-negligible probability.

However, it is easy to find short preimages wrt special, structured matrices. One such structured matrix (which will be very useful in our lattice-based construction) is the *gadget matrix* $\mathbf{G}$ defined below. Let $\mathbf{g}$ denote the vector $\begin{bmatrix} 1 & 2 & 2^2 & \dots & 2^{\lceil \log(q) \rceil} \end{bmatrix}$, and let $\mathbf{G} = \mathbf{I} \otimes \mathbf{g}$.

$$\mathbf{G} = \begin{bmatrix} \mathbf{g} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{g} & \dots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & \mathbf{g} \end{bmatrix}$$

The above matrix $\mathbf{G}$ has dimensions $n \times n \log(q)$. We will pad it with columns of zeroes to make it an $n \times m$ matrix. Note that it is easy to find short preimages wrt $\mathbf{G}$.

Exercise 1.6. *Show that there exists an efficient algorithm such that, for any $\mathbf{v} \in \mathbb{Z}_q^n$, the algorithm outputs a vector $\mathbf{w} \in \mathbb{Z}_q^m$ such that $\mathbf{G} \cdot \mathbf{w} = \mathbf{v}$ and $\mathbf{w}$ has binary entries.*

Using $\mathbf{G}$, we can sample random looking matrix, together with some secret information that we call a *trapdoor*, such that the trapdoor allows us to compute a short preimage for any vector. This sampler outputs matrices of the form $[\mathbf{A} \mid \mathbf{A} \cdot \mathbf{R} + \mathbf{G}]$, where $\mathbf{R}$ is a matrix with binary entries and is the trapdoor for matrix $\mathbf{A}$.

Exercise 1.7. *Let* $\mathbf{R}$ *be a binary matrix, and* $\mathbf{A} = [\mathbf{A}' \mid \mathbf{A}' \cdot \mathbf{R} + \mathbf{G}]$. *Show that there exists an efficient algorithm that, given* $\mathbf{R}$ *and any* $\mathbf{v} \in \mathbb{Z}_q^n$, *samples a vector* $\mathbf{w}$ *such that* $\mathbf{A} \cdot \mathbf{w} = \mathbf{v}$ *and* $\mathbf{w}$ *has small entries.*

The trapdoor for matrix $\mathbf{A}$ can also be used to sample short preimages of any vector $\mathbf{v}$, wrt any matrix of the form $[\mathbf{A} \mid \mathbf{B}]$. Similarly, it can also be used to sample short preimages of any $\mathbf{v}$, wrt any matrix of the form $[\mathbf{B} \mid \mathbf{B} \cdot \mathbf{S} + \mathbf{A}]$ where $\mathbf{S}$ is a matrix with binary entries.

Exercise 1.8. *Let* $\mathbf{A} = [\mathbf{A}' \mid \mathbf{A}' \cdot \mathbf{R} + \mathbf{G}]$. *Show that there exists an efficient algorithm that uses* $\mathbf{R}$, *and for any* $\mathbf{B} \in \mathbb{Z}_q^{n \times m}$, $\mathbf{S} \in \{0,1\}^{m \times m}$ *and* $\mathbf{v} \in \mathbb{Z}_q^n$, *it samples vectors* $\mathbf{w}_1$ *and* $\mathbf{w}_2$ *with small entries such that* $[\mathbf{A} \mid \mathbf{B}] \cdot \mathbf{w}_1 = \mathbf{v}$ *and* $[\mathbf{B} \mid \mathbf{B} \cdot \mathbf{S} + \mathbf{A}] \cdot \mathbf{w}_2 = \mathbf{v}$.

With a little more work, we can show that the preimages for a random vector looks like a random low-norm vector from an appropriate (fixed) distribution. These results are formally summarised in the following lemma.

Lemma 1.6. *There exist efficient algorithms* TrapGen, SamplePre, ExtendRight, ExtendLeft, *polynomial* p *and an efficiently samplable distribution* $\mathcal{D}$, *with the following syntax and properties.*

- $\mathsf{TrapGen}(1^n, 1^m)$*: takes as input the matrix dimensions* n, m *and outputs* $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ *together with trapdoor* $T_{\mathbf{A}}$.
- $\mathsf{SamplePre}(\mathbf{A}, T_{\mathbf{A}}, \mathbf{v})$*: takes as input a matrix* $\mathbf{A}$ *together with its trapdoor* $T_{\mathbf{A}}$, *and a vector* $\mathbf{v} \in \mathbb{Z}_q^n$. *It outputs a vector* $\mathbf{w} \in \mathbb{Z}^m$ *such that* $\mathbf{A} \cdot \mathbf{w} = \mathbf{v}$ *and* $\|\mathbf{w}\|_\infty \leq p(n)$
- $\mathsf{ExtendRight}(\mathbf{A}, T_{\mathbf{A}}, \mathbf{B}, \mathbf{v})$*: takes as input a matrix* $\mathbf{A}$ *with its trapdoor* $T_{\mathbf{A}}$, *a matrix* $\mathbf{B}$ *and a vector* $\mathbf{v}$. *Let* $\mathbf{C} = [\mathbf{A} \mid \mathbf{B}]$. *It outputs a vector* $\mathbf{w} \in \mathbb{Z}^m$ *such that* $\mathbf{C} \cdot \mathbf{w} = \mathbf{v}$ *and* $\|\mathbf{w}\|_\infty \leq p(n)$
- $\mathsf{ExtendLeft}(\mathbf{A}, T_{\mathbf{A}}, \mathbf{B}, \mathbf{S}, \mathbf{v})$*: takes as input a matrix* $\mathbf{A}$ *with its trapdoor* $T_{\mathbf{A}}$, *a matrix* $\mathbf{B}$, *a binary matrix* $\mathbf{S}$ *and a vector* $\mathbf{v}$. *Let* $\mathbf{C} = [\mathbf{B} \mid \mathbf{B} \cdot \mathbf{S} + \mathbf{A}]$. *It outputs a vector* $\mathbf{w} \in \mathbb{Z}^m$ *such that* $\mathbf{C} \cdot \mathbf{w} = \mathbf{v}$ *and* $\|\mathbf{w}\|_\infty \leq p(n)$.

These algorithms satisfy the following properties:

1. $\{\mathbf{A} : (\mathbf{A}, T_{\mathbf{A}}) \leftarrow \mathsf{TrapGen}(1^n, 1^m)\} \approx_s \left\{\mathbf{A} : \mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}\right\}$
2. *For any* $(\mathbf{A}, T_{\mathbf{A}}) \leftarrow \mathsf{TrapGen}(1^n, 1^m)$,
$$\left\{\mathbf{w} : \mathbf{v} \leftarrow \mathbb{Z}_q^n, \mathbf{w} \leftarrow \mathsf{SamplePre}(\mathbf{A}, T_{\mathbf{A}}, \mathbf{v})\right\} \approx_s \{\mathbf{w} \ : \ \mathbf{w} \leftarrow \mathcal{D}\}$$
3. *For any* $(\mathbf{A}, T_{\mathbf{A}}) \leftarrow \mathsf{TrapGen}(1^n, 1^m)$, $\mathbf{B} \in \mathbb{Z}_q^{n \times m}$,
$$\left\{\mathbf{w} : \mathbf{v} \leftarrow \mathbb{Z}_q^n, \mathbf{w} \leftarrow \mathsf{ExtendRight}(\mathbf{A}, T_{\mathbf{A}}, \mathbf{B}, \mathbf{v})\right\} \approx_s \{\mathbf{w} \ : \ \mathbf{w} \leftarrow \mathcal{D}\}$$

4. *For any* $(\mathbf{A}, T_{\mathbf{A}}) \leftarrow \mathsf{TrapGen}(1^n, 1^m)$, $\mathbf{B} \in \mathbb{Z}_q^{n \times m}$, $\mathbf{S} \in \{0,1\}^{m \times m}$,

$$\left\{\mathbf{w} : \mathbf{v} \leftarrow \mathbb{Z}_q^n, \mathbf{w} \leftarrow \mathsf{ExtendLeft}(\mathbf{A}, T_{\mathbf{A}}, \mathbf{B}, \mathbf{S}, \mathbf{v})\right\} \approx_s \{\mathbf{w} \ : \ \mathbf{w} \leftarrow \mathcal{D}\}$$

Public Key Encryption

Construction 1.7 (PKE Construction: dual-Regev)**.** *The dual-Regev encryption scheme is a bit-encryption scheme, closely related to Regev's PKE scheme* [Reg09]*. The algorithms are described below.*

- $\mathsf{Setup}(1^n)$*: The setup algorithm samples* $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}$, $\mathbf{r} \leftarrow \{0,1\}^m$*. The public key is* $(\mathbf{A}, \mathbf{b} = \mathbf{A} \cdot \mathbf{r})$ *and the secret key is* $\mathbf{r}$*.*
- $\mathsf{Enc}(\mathsf{pk} = (\mathbf{A}, \mathbf{b}), m \in \{0,1\})$*: The encryption algorithm samples* $\mathbf{s} \leftarrow \mathbb{Z}_q^n$, $\mathbf{e} \leftarrow [-B, B]^m$, $e' \leftarrow [-B, B]$*, and sets* $\mathsf{ct} = \mathbf{s}^\top \cdot \mathbf{A} + \mathbf{e}$, $\mathsf{ct}' = \mathbf{s}^\top \cdot \mathbf{b} + e' + m \cdot (q/2)$.
- $\mathsf{Dec}(\mathsf{sk} = \mathbf{r}, \mathsf{ct} = (\mathsf{ct}_1, \mathsf{ct}'))$ *: The decryption algorithm computes* $z = \mathsf{ct}' - \mathsf{ct} \cdot \mathbf{r}$*. If* $|z - q/2| \leq \sqrt{q}$*, then decryption outputs* 1*, else it outputs* 0*.*

$\Diamond$

Correctness is immediate. For proving security, we first use Lemma 1.5 to switch the public key to a uniformly random matrix $\mathbf{A}$ and a uniformly random vector $\mathbf{b}$. Next, we use the LWE assumption to argue that the ciphertext components $(\mathsf{ct}, \mathsf{ct}')$ look uniformly random.

Identity Based Encryption

Next, we present two lattice-based IBE constructions. The first construction is by Gentry, Peikert and Vaikuntanathan [GPV08], and is proven secure in the random oracle model. The second one is by Cash, Hofheinz, Peikert and Kiltz [CHKP10], and is proven secure in the standard model. Both these constructions have some common structure. In both schemes, the encryption algorithm uses the master public key mpk and identity id to derive a dual-Regev public key $\mathsf{pk}_{\mathrm{id}}$, which is then used for dual-Regev encrypting the message.

Construction 1.8 (IBE Construction [GPV08])**.** *This lattice-based IBE construction is proven secure in the random oracle model. Let* $H : \{0,1\}^* \to \mathbb{Z}_q^n$ *be a hash function (to be modeled as a random oracle in the proof).*

- $\mathsf{Setup}(1^n)$*: The setup algorithm samples* $(\mathbf{A}, T_{\mathbf{A}}) \leftarrow \mathsf{TrapGen}(1^n, 1^m)$*, and sets* $\mathsf{mpk} = \mathbf{A}$, $\mathsf{msk} = T_{\mathbf{A}}$.
- $\mathsf{Enc}(\mathsf{mpk} = \mathbf{A}, \mathrm{id}, m)$*: The encryption algorithm computes* $\mathbf{v} = H(\mathrm{id})$*, and sets* $\mathsf{pk}_{\mathrm{id}} = (\mathbf{A}, \mathbf{v})$*. It then uses* $\mathsf{pk}_{\mathrm{id}}$ *for dual-Regev encryption of* m*. That is, it chooses* $\mathbf{s} \leftarrow \mathbb{Z}_q^n$, $\mathbf{e} \leftarrow [-B, B]^m$, $e' \leftarrow [-B, B]$ *and sets* $\mathsf{ct} = \mathbf{s}^\top \cdot \mathbf{A} + \mathbf{e}$, $\mathsf{ct}' = \mathbf{s}^\top \cdot \mathbf{v} + e' + m \cdot (q/2)$.

- $\mathsf{KeyGen}(\mathsf{msk} = T_{\mathbf{A}}, \mathrm{id})$: *The key generation algorithm computes* $\mathbf{v} = H(\mathrm{id})$, *then samples* $\mathbf{r} \leftarrow \mathsf{SamplePre}(\mathbf{A}, T_{\mathbf{A}}, \mathbf{v})$. *It outputs* $\mathsf{sk}_{\mathrm{id}} = \mathbf{r}$.
- $\mathsf{Dec}(\mathsf{sk}_{\mathrm{id}}, \mathsf{ct}_{\mathrm{id}} = (\mathsf{ct}, \mathsf{ct}'))$: *The decryption is identical to dual-Regev decryption. It computes* $z = \mathsf{ct}' - \mathsf{ct} \cdot \mathsf{sk}_{\mathrm{id}}$. *If* $|z - q/2| \leq \sqrt{q}$, *then it outputs* 1, *else it outputs* 0.

◊

Correctness is identical to the dual-Regev correctness. The security proof is in the random oracle model, where the hash function is modeled as a random oracle. Let us work with a simplified security game which conveys the main ideas involved in the proof. In this simplified security game, the adversary always submits the all-zeroes string $\mathbf{0}$ as the challenge identity, and makes a single secret key query for the all-ones string $\mathbf{1}$.

Simplified game for IBE in random oracle model

- Adversary sends the challenge identity $\mathbf{0}$.
- Challenger samples $(\mathbf{A}, T_{\mathbf{A}})$, samples $\mathbf{v}$ and sets $H(\mathbf{0}) = \mathbf{v}$. It then samples a bit b, $\mathbf{s} \leftarrow \mathbb{Z}_q^n$, noise $\mathbf{e}, e'$ and sets $\mathsf{ct} = \left(\mathbf{s}^\top \cdot \mathbf{A} + \mathbf{e}, \mathbf{s}^\top \cdot \mathbf{v} + e + b(q/2)\right)$. It sends $\mathsf{mpk} = \mathbf{A}$, $H(0)$ and ct.
- Adversary sends secret key query for $\mathbf{1}$. The challenger samples $\mathbf{v}' \leftarrow \mathbb{Z}_q^n$, sets $H(\mathbf{1}) = \mathbf{v}'$, and $\mathsf{sk} \leftarrow \mathsf{SamplePre}\left(\mathbf{A}, T_{\mathbf{A}}, \mathbf{v}'\right)$. It sends $H(\mathbf{1})$ and sk to the adversary.
- Adversary finally sends it guess b'.

Figure 3: In this simplified security game, the adversary makes only one secret key query. Moreover, both the challenge identity and secret key query are fixed in advance.

Claim 1.9. *Suppose there exists a p.p.t. adversary* $\mathcal{A}$ *that wins the simplified IBE security game (described in Figure 3) with non-negligible advantage. Then there exists a p.p.t. adversary* $\mathcal{B}$ *that breaks the CPA security of dual-Regev encryption scheme with non-negligible advantage.*

Proof idea: The reduction algorithm receives the public key $\mathsf{pk} = (\mathbf{A}, \mathbf{v})$ and the challenge ciphertext ct from the dual-Regev challenger. Since it must use the IBE adversary, it must derive the IBE master public key from pk, and the IBE challenge ciphertext from ct. A natural choice is to set $\mathsf{mpk} = \mathbf{A}$, $H(\mathbf{0}) = \mathbf{v}$ and challenge ciphertext as ct. However, the reduction also needs to give a secret key for $\mathbf{1}$, and this requires the trapdoor for $\mathbf{A}$ (which the reduction does not have). The main idea in this proof is to *program the random oracle*. Instead of picking a uniformly random vector for $H(\mathbf{1})$, the reduction can pick a short vector $\mathbf{w}'$ and set $H(\mathbf{1}) = \mathbf{A} \cdot \mathbf{w}'$. Using the properties of trapdoor sampling, we get that sampling a uniformly random vector $\mathbf{v}'$ and then sampling its preimage $\mathbf{w}'$ is indistinguishable from sampling a low-norm vector $\mathbf{w}'$ and then setting $\mathbf{v}' = \mathbf{A} \cdot \mathbf{w}'$.

Proof. The formal proof goes via a sequence of hybrids.

Hybrid 0: same as simplified security game.
Hybrid 1: This is similar to the simplified security game, except the response to secret key query. Here, the challenger samples $\mathbf{w}' \leftarrow \mathcal{D}$, sets $H(\mathbf{1}) = \mathbf{A} \cdot \mathbf{w}'$, and the secret key for $\mathbf{1}$ is $\mathbf{w}'$.

The distribution $\mathcal{D}$ is defined in Lemma 1.6.

Using Property 2 of trapdoor sampling, it follows that Hybrid 0 and Hybrid 1 are statistically indistinguishable. Therefore, if an adversary succeeds in Hybrid 0, then it also succeeds in Hybrid 1 with non-negligible advantage. The reduction algorithm, on receiving $\mathsf{pk} = (\mathbf{A}, \mathbf{v})$ and ct, sets $\mathsf{mpk} = \mathbf{A}$, $H(\mathbf{0}) = \mathbf{v}$ and the IBE challenge ciphertext as ct. On receiving the secret key query, it samples $\mathbf{w}' \leftarrow \mathcal{D}$, sets $H(\mathbf{1}) = \mathbf{A} \cdot \mathbf{w}'$, and the secret key for $\mathbf{1}$ is $\mathbf{w}'$. Finally, the adversary sends its guess, which the reduction forwards to the dual-Regev PKE challenger.

□

The above construction and proof crucially use the programming of random oracle. Next, we present an IBE construction in the standard model. The key ideas are similar: using the master public key and identity, we will derive a dual-Regev public key and use it for encrypting the message.

Construction 1.10 (IBE Construction in the standard model [CHKP10])**.** *This IBE construction is proven secure in the standard model. Let $\mathcal{ID} = \{0,1\}^\ell$ denote the identity space.*

- $\mathsf{Setup}(1^n)$*: The setup algorithm samples* $(\mathbf{A}, T_{\mathbf{A}}) \leftarrow \mathsf{TrapGen}(1^n, 1^m)$*, samples* 2ℓ *matrices* $\{\mathbf{A}_{i,b}\}_{i\in[\ell],b\in\{0,1\}}$*, vector* $\mathbf{v} \leftarrow \mathbb{Z}_q^n$ *and sets* $\mathsf{mpk} = \left(\mathbf{A}, \{\mathbf{A}_{i,b}\}_{i,b}, \mathbf{v}\right)$*,* $\mathsf{msk} = T_{\mathbf{A}}$*.*
- $\mathsf{Enc}(\mathsf{mpk} = \mathbf{A}, \mathsf{id}, m)$*: The encryption algorithm sets* $\mathbf{A}_{\mathsf{id}} = \left[\mathbf{A} \mid \mathbf{A}_{1,\mathsf{id}_1} \mid \ldots \mid \mathbf{A}_{\ell,\mathsf{id}_\ell}\right]$*, and sets* $\mathsf{pk}_{\mathsf{id}} = (\mathbf{A}_{\mathsf{id}}, \mathbf{v})$*. It then uses* $\mathsf{pk}_{\mathsf{id}}$ *for dual-Regev encryption of* m*. That is, it chooses* $\mathbf{s} \leftarrow \mathbb{Z}_q^n$*,* $\mathbf{e} \leftarrow [-B, B]^m$*,* $e' \leftarrow [-B, B]$ *and sets* $\mathsf{ct} = \mathbf{s}^\top \cdot \mathbf{A}_{\mathsf{id}} + \mathbf{e}$*,* $\mathsf{ct}' = \mathbf{s}^\top \cdot \mathbf{v} + e' + m \cdot (q/2)$*.*
- $\mathsf{KeyGen}\left(\mathsf{mpk} = \left(\mathbf{A}, \{\mathbf{A}_{i,b}\}_{i,b}, \mathbf{v}\right), \mathsf{msk} = T_{\mathbf{A}}, \mathsf{id}\right)$*: The key generation algorithm sets* $\mathbf{B}_{\mathsf{id}} = \left[\mathbf{A}_{1,\mathsf{id}_1} \mid \ldots \mid \mathbf{A}_{\ell,\mathsf{id}_\ell}\right]$*, then samples* $\mathbf{r} \leftarrow \mathsf{ExtendRight}(\mathbf{A}, T_{\mathbf{A}}, \mathbf{B}_{\mathsf{id}}, \mathbf{v})$*. It outputs* $\mathsf{sk}_{\mathsf{id}} = \mathbf{r}$*.*

  Note that $[\mathbf{A} \mid \mathbf{B}_{\mathsf{id}}] \cdot \mathbf{r} = \mathbf{v}$ *and* $\mathbf{r}$ *has small entries.*
- $\mathsf{Dec}(\mathsf{sk}_{\mathsf{id}}, \mathsf{ct}_{\mathsf{id}} = (\mathsf{ct}, \mathsf{ct}'))$*: The decryption is identical to dual-Regev decryption. It computes* $z = \mathsf{ct}' - \mathsf{ct} \cdot \mathsf{sk}_{\mathsf{id}}$*. If* $|z - q/2| \le \sqrt{q}$*, then it outputs* 1*, else it outputs* 0*.*

◊

Correctness follows from the correctness of dual-Regev PKE scheme. For security, we will again work with a simplified security game where the adversary sends only one secret key query, and both the challenge identity and secret key query are fixed in advance. This security game is described in Figure 4 below.

Simplified game for IBE in standard model

- Adversary sends the challenge identity $\mathbf{0}$.
- Challenger samples $(\mathbf{A}, T_{\mathbf{A}})$, matrices $\{\mathbf{A}_{i,b}\}_{i,b}$, samples $\mathbf{v}$ and sets $\mathsf{mpk} = \left(\mathbf{A}, \{\mathbf{A}_{i,b}\}_{i,b}, \mathbf{v}\right)$. It sets $\mathbf{A_0} = [\mathbf{A} \mid \mathbf{A}_{1,0} \mid \ldots \mid \mathbf{A}_{\ell,0}]$. It then samples a bit b, $\mathbf{s} \leftarrow \mathbb{Z}_q^n$, noise $\mathbf{e}, e'$ and sets $\mathsf{ct} = \left(\mathbf{s}^\top \cdot \mathbf{A_0} + \mathbf{e}, \mathbf{s}^\top \cdot \mathbf{v} + e + b(q/2)\right)$.
  It sends mpk and ct.
- Adversary sends secret key query for $\mathbf{1}$. The challenger sets $\mathbf{B} = [\mathbf{A}_{1,1} \mid \ldots \mid \mathbf{A}_{\ell,1}]$, computes $\mathbf{r} \leftarrow \mathsf{ExtendRight}(\mathbf{A}, T_{\mathbf{A}}, \mathbf{B}, \mathbf{v})$ and sends $\mathbf{r}$ as the secret key for $\mathbf{1}$.
- Adversary finally sends it guess b'.

Figure 4: In this simplified security game, the adversary makes only one secret key query. Moreover, both the challenge identity and secret key query are fixed in advance.

Claim 1.11. *Suppose there exists a p.p.t. adversary $\mathcal{A}$ that wins the simplified IBE security game (described in Figure 4) with non-negligible advantage. Then there exists a p.p.t. adversary $\mathcal{B}$ that breaks the CPA security of dual-Regev encryption scheme with non-negligible advantage.*

Proof idea: The reduction algorithm receives a public key and challenge ciphertext from the PKE challenger. Using this, the reduction can set the IBE challenge ciphertext. However, it must set the master public key in such a manner that it can respond to the secret key query. Note that half the master public key is already set (given the public key sent by the PKE challenger). In the construction, the secret keys are generated using ExtendRight. In the proof, these will be generated using ExtendLeft, and this is where it will be useful to set the master public key appropriately.

Proof. The reduction algorithm receives $\mathsf{pk} = ([\mathbf{A} \mid \mathbf{B}_1 \mid \ldots \mid \mathbf{B}_\ell], \mathbf{v})$ and ct from the PKE challenger. For each $i \in [\ell]$, it sets $\mathbf{A}_{i,0} = \mathbf{B}_i$, samples $\mathbf{R}_i \leftarrow \{0,1\}^{m \times m}$ and sets $\mathbf{A}_{i,1} = \mathbf{A} \cdot \mathbf{R}_i + \mathbf{G}$. The reduction sends $\mathsf{mpk} = \left(\mathbf{A}, \{\mathbf{A}_{i,b}\}, \mathbf{v}\right)$ and challenge ciphertext ct to the IBE adversary. For the secret key, the reduction must produce a short vector $\mathbf{r}$ such that

$$[\mathbf{A} \mid \mathbf{A} \cdot \mathbf{R}_1 + \mathbf{G} \mid \ldots \mid \mathbf{A} \cdot \mathbf{R}_\ell + \mathbf{G}] \cdot \mathbf{r} = \mathbf{v}.$$

The reduction can compute such a vector using the trapdoor of $\mathbf{G}$, and this is indistinguishable from using ExtendRight. This completes our proof. □

Exercise 1.9. *Using the above ideas, show that Construction 1.10 satisfies selective IBE security (with unbounded secret key queries).*

Attribute Based Encryption for Inner-Products

Next, we present an ABE scheme, where the attributes and policies are vectors, and decryption works only if their inner product is zero. The ABE scheme, pro-

posed by Agrawal, Freeman and Vaikuntanathan [AFV11] has a neat structure that can be extended to obtain ABE for general circuits (this was shown by Boneh et al. [BGG+14].

Construction 1.12 (Inner-product ABE in the standard model [AFV11])**.** *Let $T \in \mathbb{Z}$ be some constant. For this ABE scheme, the attribute and policy space is $[-T, T]^\ell$, and policy $\mathbf{y}$ accepts attribute $\mathbf{x}$ if $\langle \mathbf{x}, \mathbf{y} \rangle = 0$.*

- $\mathsf{Setup}(1^n)$*: The setup algorithm samples* $(\mathbf{A}, T_{\mathbf{A}}) \leftarrow \mathsf{TrapGen}(1^n, 1^m)$*, samples* ℓ *matrices* $\{\mathbf{A}_i\}_{i \in [\ell]}$*, vector* $\mathbf{v} \leftarrow \mathbb{Z}_q^n$ *and sets* $\mathsf{mpk} = \left(\mathbf{A}, \{\mathbf{A}_i\}_i, \mathbf{v}\right)$*,* $\mathsf{msk} = T_{\mathbf{A}}$*.*
- $\mathsf{Enc}(\mathsf{mpk} = \mathbf{A}, \mathbf{x}, m)$*: The encryption algorithm sets* $\mathbf{A}_{\mathbf{x}} = [\mathbf{A}_1 + x_1 \mathbf{G} \mid \ldots \mid \mathbf{A}_\ell + x_\ell \mathbf{G}]$*. It chooses* $\mathbf{s} \leftarrow \mathbb{Z}_q^n$*,* $\mathbf{e}_0 \leftarrow [-n, n]^m$*,* $\mathbf{e} \leftarrow \left[-q^{1/3}, q^{1/3}\right]^{\ell \cdot m}$*,* $e' \leftarrow [-B, B]$ *and sets* $\mathsf{ct}_0 = \mathbf{s}^\top \cdot \mathbf{A} + \mathbf{e}_0$*,* $\mathsf{ct} = \mathbf{s}^\top \cdot \mathbf{A}_{\mathbf{x}} + \mathbf{e}$*,* $\mathsf{ct}' = \mathbf{s}^\top \cdot \mathbf{v} + e' + m \cdot (q/2)$*.*

  Note that the noise for ct_0 *is drawn from* $[-n, n]^m$*, while the noise for* ct *is drawn from* $\left[-q^{1/3}, q^{1/3}\right]$*. This will be important for the security proof.*
- $\mathsf{KeyGen}\left(\mathsf{mpk} = \left(\mathbf{A}, \{\mathbf{A}_i\}_i, \mathbf{v}\right), \mathsf{msk} = T_{\mathbf{A}}, \mathbf{y}\right)$*: The key generation algorithm sets* $\mathbf{B}_{\mathbf{y}} = \sum_i y_i \cdot \mathbf{A}_i$*, then samples* $\mathbf{r} \leftarrow \mathsf{ExtendRight}\left(\mathbf{A}, T_{\mathbf{A}}, \mathbf{B}_{\mathbf{y}}, \mathbf{v}\right)$*. It outputs* $\mathsf{sk}_{\mathbf{y}} = \mathbf{r}$*.*

  Note that $\left[\mathbf{A} \mid \mathbf{B}_{\mathbf{y}}\right] \cdot \mathbf{r} = \mathbf{v}$ *and* $\mathbf{r}$ *has small entries.*
- $\mathsf{Dec}\left(\mathsf{sk}_{\mathbf{y}}, \mathsf{ct}_{\mathbf{x}} = (\mathsf{ct}_0, \mathsf{ct}, \mathsf{ct}')\right)$*: The decryption algorithm parses the ciphertext* $\mathsf{ct} = [\mathsf{ct}_1 \mid \ldots \mid \mathsf{ct}_\ell]$*. Next, it computes* $\tilde{\mathsf{ct}} = \sum_i y_i \cdot \mathsf{ct}_i$*,* $z = \mathsf{ct}' - [\mathsf{ct}_0 \mid \tilde{\mathsf{ct}}] \cdot \mathsf{sk}_{\mathbf{y}}$*. If* $|z - q/2| \leq \sqrt{q}$*, then it outputs* 1*, else it outputs* 0*.*

◊

For correctness, note that $([\mathsf{ct}_0 \mid \tilde{\mathsf{ct}}], \mathsf{ct}')$ is dual-Regev encryption of the message using public key $\left(\left[\mathbf{A} \mid \mathbf{B}_{\mathbf{y}}\right], \mathbf{v}\right)$ (here, it is important that $\mathbf{y}$ has small entries). Therefore, the correctness of this scheme follows from the correctness of dual-Regev PKE scheme.

Security proof sketch. For security, the reduction must use ExtendLeft, and therefore it is important to set the mpk matrices appropriately. The scheme is selectively secure (that is, the reduction receives the challenge attribute $\mathbf{x}$ before it sends the master public key).
The reduction algorithm receives $\mathsf{pk} = (\mathbf{A}, \mathbf{v})$ and challenge ciphertext $(\mathsf{ct}, \mathsf{ct}')$ from the PKE challenger, and $\mathbf{x}$ from the ABE adversary, and must set mpk and the challenge ciphertext appropriately. It samples ℓ binary matrices $\mathbf{R}_i \leftarrow \{0,1\}^{m \times m}$, sets $\mathbf{A}_i = \mathbf{A} \cdot \mathbf{R}_i - x_i \cdot \mathbf{G}$. The master public key is $\left(\mathbf{A}, \{\mathbf{A}_i\}_i, \mathbf{v}\right)$. For the challenge ciphertext, it samples error vectors $\mathbf{e}_0 \leftarrow [-n, n]^m$, $\mathbf{e}_i \leftarrow \left[-q^{1/3}, q^{1/3}\right]^m$ and sets $\mathsf{ct}_0^* = \mathsf{ct}$, $\mathsf{ct}_i^* = \mathsf{ct} \cdot \mathbf{R}_i + \mathbf{e}_i$, $\mathsf{ct}^* = \left[\mathsf{ct}_1 \mid \ldots \mid \mathsf{ct}_\ell^*\right]$ and sends $(\mathsf{ct}_0, \mathsf{ct}^*, \mathsf{ct}')$ as the ABE challenge ciphertext.
For the secret key queries, note that $\theta_{\mathbf{y}} = \langle \mathbf{x}^*, \mathbf{y} \rangle \neq 0$, therefore $\mathbf{B}_{\mathbf{y}} = \mathbf{A} \cdot (\sum_i y_i \cdot \mathbf{R}_i) + \theta_{\mathbf{y}} \cdot \mathbf{G}$. Hence, we can use ExtendLeft to sample a preimage of $\mathbf{v}$ wrt $\left[\mathbf{A} \mid \mathbf{B}_{\mathbf{y}}\right]$.

References

[AFV11] Shweta Agrawal, David Mandell Freeman, and Vinod Vaikuntanathan. Functional Encryption for Inner Product Predicates from Learning with Errors. In Dong Hoon Lee and Xiaoyun Wang, editors, *Advances in Cryptology - ASIACRYPT 2011 - 17th International Conference on the Theory and Application of Cryptology and Information Security, Seoul, South Korea, December 4-8, 2011. Proceedings*, volume 7073 of *Lecture Notes in Computer Science*, pages 21–40. Springer, 2011.

[BGG+14] Dan Boneh, Craig Gentry, Sergey Gorbunov, Shai Halevi, Valeria Nikolaenko, Gil Segev, Vinod Vaikuntanathan, and Dhinakaran Vinayagamurthy. Fully Key-Homomorphic Encryption, Arithmetic Circuit ABE, and Compact Garbled Circuits. *IACR Cryptol. ePrint Arch.*, page 356, 2014.

[CHK04] Ran Canetti, Shai Halevi, and Jonathan Katz. Chosen-Ciphertext Security from Identity-Based Encryption. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology - EUROCRYPT 2004, International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, May 2-6, 2004, Proceedings*, volume 3027 of *Lecture Notes in Computer Science*, pages 207–222. Springer, 2004.

[CHKP10] David Cash, Dennis Hofheinz, Eike Kiltz, and Chris Peikert. Bonsai Trees, or How to Delegate a Lattice Basis. In Henri Gilbert, editor, *Advances in Cryptology - EUROCRYPT 2010, 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Monaco / French Riviera, May 30 - June 3, 2010. Proceedings*, volume 6110 of *Lecture Notes in Computer Science*, pages 523–552. Springer, 2010.

[GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In Cynthia Dwork, editor, *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*, pages 197–206. ACM, 2008.

[Reg09] Oded Regev. On Lattices, Learning with Errors, Random Linear Codes, and Cryptography. *J. ACM*, 56(6), sep 2009.