CS 294. (Hierarchical) Identity-based Encryption

1 Identity-based Encryption

Let us think first about deploying a public-key encryption scheme on a large scale. We need a mechanism to maintain a directory of (ID, PK) pairs where ID is the identifying information of a person, say Alice's e-mail address or phone number, that other people use to send her a message. Then, when you wish to send an email to Alice, you look up her public key in the directory and encrypt to the public key.

The directory, which forms part of a public-key infrastructure (PKI), has to be authenticated and trusted. For example, an adversary should not be able to insert an entry of the form (ID_A, PK'_A), where she presumably knows SK'_A, into the directory.

Identity-based encryption (IBE) solves the problem of having to maintain an authenticated PKI. In an IBE:

- there is a master authority who generates a master public key MPK together with a master secret key MSK, and publishes the MPK.
- To encrypt a message μ, one needs to know MPK and the identity ID (e.g., the e-mail address) of the recipient.
- Each user goes to the master authority and receives SK_{ID} after authenticating that they indeed are the owner of ID.
- Using SK_{ID}, the user can decrypt ciphertexts encrypted to the identity ID.

Let us now define the syntax of an IBE, formalizing the discussion above.

- $\mathsf{Setup}(1^\lambda)$: is a probabilistic algorithm that generates a master public key MPK and a master secret key MSK.
- $\mathsf{Enc}(MPK, ID, \mu)$: is a probabilistic algorithm that generates a ciphertext C of a message μ (for simplicity, we will encrypt bits but that is largely irrelevant) w.r.t. identity ID.
- $\mathsf{KeyGen}(MSK, ID \in \{0,1\}^*)$: is a probabilistic algorithm that generates a secret key SK_{ID}.
- $\mathsf{Dec}(SK_{ID}, C)$: is a deterministic decryption algorithm.

You may have noticed that the master authority can decrypt *all* the ciphertexts generated in this system and is therefore *very powerful*.

Application: Access Delegation across Space. I can act as the master authority and use an IBE to delegate decryption of certain subsets of messages to other people (e.g., my administative assistant). For example, all messages are tagged with a keyword $ID = \texttt{CS294}$, and I can issue the SK_{ID} to my assistant that lets him decrypt only those messages tagged with ID.

Application: Access Delegation across Time. Imagine that I go on (virtual) vacation to Cancun and want to take my laptop. However, I am worried that it will be stolen. So, I ask folks encrypting messages to me to use an IBE and tag the messages with an ID which is the current date. This allows me to generate a small set of secret keys, corresponding to the days that I am away, which allows me to decrypt only the corresponding small subset of messages. IBE lets me enjoy my vacation worry-free!

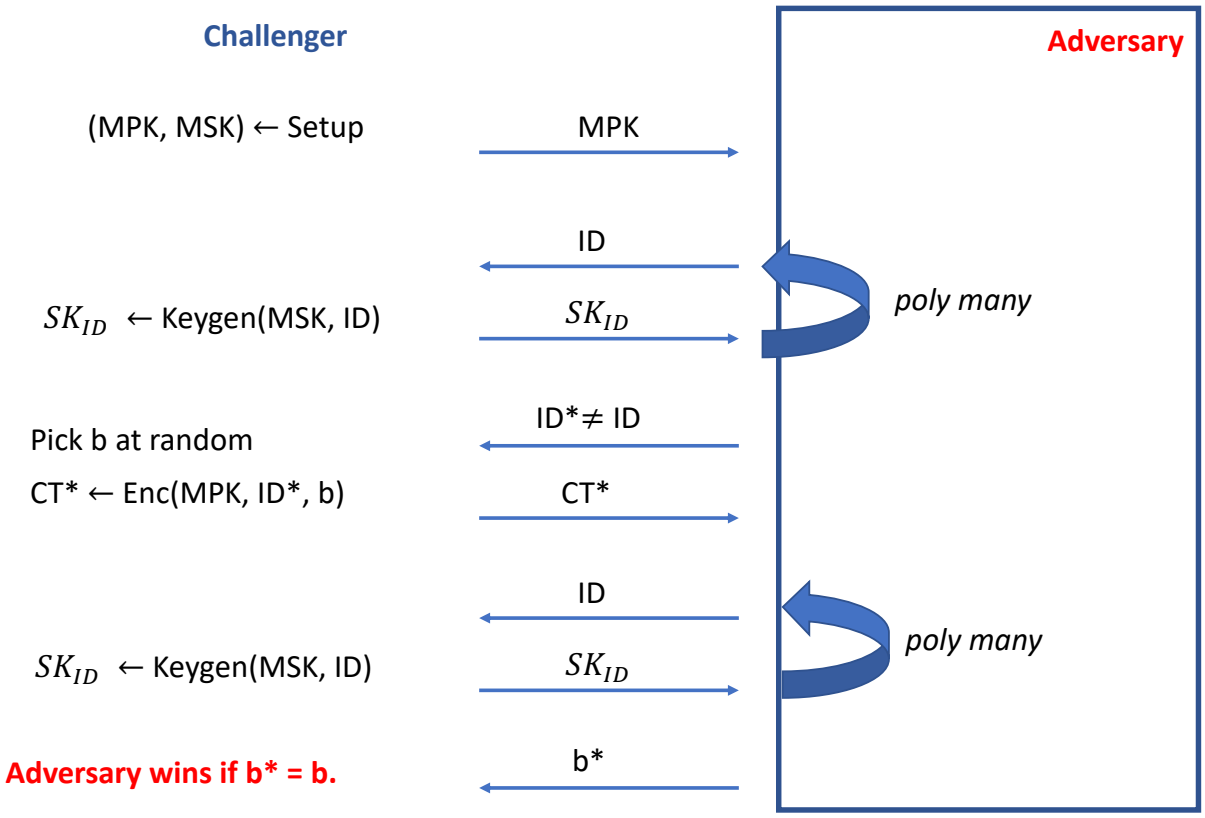
Application: Chosen-Ciphertext Security. IBE can be used in surprisingly non-trivial ways to construct other cryptographic systems, e.g., chosen ciphertext secure public-key encryption schemes and digital signature schemes (that we will describe later in this lecture).

Constructions. The first constructions used bilinear maps on elliptic curves (Boneh-Franklin'00) and quadratic residuosity (Cocks'00). We will present the third IBE scheme from LWE (Gentry-Peikert-Vaikuntanathan'08) and several variants today. Recently, Garg and Dottling have come up with a completely different scheme that relies on Diffie-Hellman groups (no need for bilinear maps!) Following up, Brakerski-Lombardi-Segev-Vaikuntanathan came up with a scheme based on learning parity with very low noise.

1.1 Definitions of Security

We imagine a PPT adversary that plays the following game with a challenger. This captures the requirement that encryptions relative to ID^* should be secure even to an adversary that can obtain secret keys for polynomially many *different* identities $ID \neq ID^*$. This is called the *adaptive security* or *full security* definition. The weaker *selective* security definition restricts the adversary to pick the identity it is attacking at the very beginning of the game (before it receives MPK).

Selectively secure IBE schemes can be generically proven to be fully secure under a sub-exponentially stronger assumption. Therefore, we will not attempt to optimize the strength of the assumption and focus on selective security for this lecture.



1.2 IBE=Signatures+Public-Key Encryption

Moni Naor observed that any IBE scheme gives us *for free* a digital signature scheme. **The intuition is that the identity secret key SK_{ID} can act as a signature for the "message" ID.** How so?

- It can be generated using the master secret key MSK (which will serve as the secret signing key.)
- It can be verified using the master public key MPK – indeed, encrypt a bunch of random messages using MPK and attempt to use the "signature" to decrypt. If decryption produces the correct message, accept the signature. Otherwise, reject.
- after receiving signatures SK_{ID} on polynomially many messages ID, being able to produce the "signature" on a different message ID^* constitutes a signature forgery; but being able to do that breaks IBE security. Conversely, in a signature scheme derived from a secure IBE scheme, it should be infeasible to do that.

Indeed, turning this around, we will use the GPV signature scheme we saw in the last class as a starting point to build an IBE scheme.

2 Recap: GPV Signatures

- $\mathsf{KeyGen}(1^\lambda)$: Generate a random matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and its trapdoor $\mathbf{T}$ by running $\mathsf{TrapSamp}$.
- $\mathsf{Sign}(\mu)$: first compute $\mathbf{v} = H(\mu) \in \mathbb{Z}_q^n$ where H is treated as a random oracle in the analysis. Then, use Gaussian sampling (via the GPV algorithm) to compute a Gaussian solution $\mathbf{e} \in \mathbb{Z}^m$ to the equation

  $$\mathbf{Ae} = \mathbf{v} \pmod q$$

  Let $\Lambda^\perp(\mathbf{A})$ denote the lattice

  $$\{\mathbf{e} \in \mathbb{Z}^m : \mathbf{Ae} = \mathbf{0} \pmod q\}$$

  and let $\Lambda^\perp_{\mathbf{v}}(\mathbf{A})$ denote a coset of $\Lambda^\perp(\mathbf{A})$ indexed by $\mathbf{v}$. That is,

  $$\Lambda^\perp_{\mathbf{v}}(\mathbf{A}) = \{\mathbf{e} \in \mathbb{Z}^m : \mathbf{Ae} = \mathbf{v} \pmod q\}$$

  Note that the distribution of $\mathbf{e}$ is $D_{\Lambda^\perp_{\mathbf{v}}(\mathbf{A}),\sigma}$ where $\sigma \approx ||\mathbf{T}|| \cdot \omega(\sqrt{\log n})$. (The $\omega(\sqrt{\log n})$ is so that the sampling algorithm can achieve negligible statistical distance from a true discrete Gaussian.)
- $\mathsf{Verify}(\mathbf{A}, \mathbf{e}, \mu)$: check that (1) $\mathbf{e}$ is short, that is $||\mathbf{e}|| \leq ||T|| \cdot \omega(\sqrt{n \log n})$; and (2) $\mathbf{Ae} = H(\mu) \pmod q$.

The key question now is how to we build an encryption algorithm whose public key is $\mathbf{v}$ (which will be treated as $H(ID)$) and the corresponding private key is $\mathbf{e}$ as above. Indeed, we have seen precisely such a scheme in the first lecture (cf. lecture notes) called the GPV encryption scheme or more commonly, the dual-Regev encryption scheme.

But before we get there, the scheme as stated above is insecure – do you see why? Bonus points if you see how to fix it.

3 The Dual Regev Encryption Scheme

- KeyGen: the public key is an LWE matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and a random vector $\mathbf{v} \in \mathbb{Z}_q^n$. The private key is a short vector $\mathbf{e}$ such that $\mathbf{A}\mathbf{e} = \mathbf{v} \pmod q$.

$$pk = (\mathbf{A}, \mathbf{v}) \qquad sk = \mathbf{e}$$

- $\mathsf{Enc}(pk, \mu)$: pick an LWE secret $\mathbf{s} \in \mathbb{Z}_q^n$ and output

$$(\mathbf{c}_1^T, c_2) := \left(\mathbf{s}^T\mathbf{A} + \mathbf{x}^T, \mathbf{s}^T\mathbf{v} + x' + m\lfloor q/2 \rfloor\right)$$

  as the ciphertext. We will call this ciphertext the dual Regev encryption of μ relative to $\mathbf{A}$ and $\mathbf{v}$.

- $\mathsf{Dec}(sk, (\mathbf{c}_1^T, c_2))$: Compute

$$\tilde{\mu} := \mathsf{Round}(c_2 - \mathbf{c}_1^T\mathbf{e})$$

  where $\mathsf{Round}(\alpha)$ outputs 1 if $|\alpha - q/2| \leq q/4$ and 0 otherwise.

We will leave the correctness and security as an exercise. (Alternatively, look at lecture 1.)

4 The GPV IBE Scheme

- $\mathsf{Setup}(1^\lambda)$: Pick the right $n = n(\lambda)$ for a security level of λ bits. Generate a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and its trapdoor $\mathbf{T} \in \mathbb{Z}^{m \times m}$ by running the trapdoor sampling algorithm.

$$(\mathbf{A}, \mathbf{T}) \leftarrow \mathsf{TrapSamp}(1^n)$$

  (The parameters m and q are picked internally by the trapdoor sampling algorithm.) The master public key is $\mathsf{mpk} = \mathbf{A}$ and the master secret key is $\mathsf{msk} = \mathbf{T}$.

- $\mathsf{KeyGen}(\mathsf{msk}, ID)$: Compute $\mathbf{v} := H(ID) \in \mathbb{Z}_q^n$ where $H : \{0,1\}^* \to \mathbb{Z}_q^n$ is a hash function (which, in the security analysis, will be treated as a random oracle.) Generate a short vector

$$\mathbf{e} \leftarrow \mathsf{DGSamp}(\mathbf{A}, \mathbf{T}, \mathbf{v})$$

  by running the discrete Gaussian sampling algorithm. Recall that $\mathbf{Ae} = \mathbf{v} \pmod q$. Output the secret key $sk_{ID} = \mathbf{e}$.

- $\mathsf{Enc}(\mathsf{mpk}, ID, \mu)$: Run the dual Regev encryption algorithm with $pk := (\mathbf{A}, \mathbf{v} = H(ID))$ and message μ and output the resulting ciphertext.

- $\mathsf{Dec}(sk_{ID}, c)$: Run the dual Regev decryption algorithm with $sk := sk_{ID} = \mathbf{e}$.

4.1 Proof of (Full) Security

We will come up with alternate algorithms called $\mathsf{Setup}^*, \mathsf{KeyGen}^*$ and Enc^* (Dec^* will be the same as Dec) which the challenger will run. Our goal will be to show that (1) the adversary cannot distinguish between the challenger running Algorithm *vs* Algorithm* and (2) Algorithms* do not need the master secret key and moreover, a challenger using Algorithm* can use a successful adversary to break LWE.

A crucial advantage of Algorithm* for the GPV scheme is that it can use the programmability of the random oracle as we will see below. We will for simplicity first create algorithms for the selective security game.

- $\mathsf{Setup}^*(ID^*, 1^\lambda)$: Sample random $\mathbf{A}^*$ which forms the MPK^* (no need for trapdoor).
- $\mathsf{Hash}^*(ID)$: Set $H(ID^*) = \mathbf{v}^*$, a random vector in $\mathbb{Z}_q^n$. For all other IDs, set $H(ID) = \mathbf{A}^*\mathbf{e}_{ID}$ where $\mathbf{e}_{ID}$ is chosen from a Gaussian. Remember $\mathbf{e}_{ID}$.
- $\mathsf{KeyGen}^*(ID)$: We know that $ID \neq ID^*$. So, we know the $\mathbf{e}_{ID}$ by construction! This is a consequence of working in the random oracle model!
- $\mathsf{Enc}^*(MPK^*, ID^*, \mu)$: return the dual Regev encryption of μ relative to $\mathbf{A}^*$ and $\mathbf{v}^*$.

The Algorithm* produce the same distribution as the original algorithms. Thus, an adversary will break the challenge ciphertext when interacting with Algorithm* just as well as with Algorithms. By embedding the dual-Regev challenge matrix $\mathbf{A}$ as the master public key and the dual-Regev public key $\mathbf{v}^*$ as the hash of ID^*, we can easily turn the IBE adversary into an attack against the dual Regev public key encryption scheme.

A Note on Full Security. Since Setup^* does not know ID^*, it guesses which of the (polynomially many) hash queries will be for ID^*. (1) any adversary that succeeds has to know $H(ID^*)$ which it can only find out by making a hash query; and (2) if the guess is correct (happens with probability $1/Q$) we can translate an IBE breaker into a dual-Regev breaker just as above.

5 The CHKP IBE Scheme

The CHKP Trick: Trapdoor Extension.
Given the trapdoor for a matrix $\mathbf{A}$, can you generate a trapdoor for $[\mathbf{A}||\mathbf{B}]$ where $\mathbf{B}$ is an arbitrary matrix?

5.1 The Scheme

- $\mathsf{Setup}(1^\lambda)$: Pick the right $n = n(\lambda)$ for a security level of λ bits. Generate matrices

$$\mathbf{A}_{1,0}, \mathbf{A}_{1,1}, \ldots, \mathbf{A}_{\ell,0}, \mathbf{A}_{\ell,1} \in \mathbb{Z}_q^{n \times m}$$

  where ℓ is the length of the identities. The master public key is

$$\mathsf{mpk} = \left(\mathbf{A}_{i,b}\right)_{i \in [\ell], b \in \{0,1\}}, \mathbf{v}$$

  where $\mathbf{v} \in \mathbb{Z}_q^n$ is a random vector, and the master secret key is

$$\mathsf{msk} = (\mathbf{T}_{\mathbf{A}_0}, \mathbf{T}_{\mathbf{A}_1})$$

  We will never use the trapdoors for the other matrices (except in the security proof.)

- $\mathsf{KeyGen}(\mathsf{msk}, ID \in \{0,1\}^\ell)$: Let

$$\mathbf{A}_{ID} := [\mathbf{A}_{1,ID_1} || \mathbf{A}_{2,ID_2} || \ldots || \mathbf{A}_{\ell,ID_\ell}]$$

  where $ID_1, \ldots, ID_\ell$ are the bits of ID. Generate a short vector $\mathbf{e} \leftarrow \mathsf{DGSamp}(\mathbf{A}_{ID}, \mathbf{T}_{\mathbf{A}_{ID}}, \mathbf{v})$ by running the discrete Gaussian sampling algorithm. Recall that $\mathbf{A}_{ID} \cdot \mathbf{e} = \mathbf{v} \pmod q$. Output the secret key $sk_{ID} = \mathbf{e}$.

- $\mathsf{Enc}(\mathsf{mpk}, ID, \mu)$: Run the dual Regev encryption algorithm with $pk := (\mathbf{A}_{ID}, \mathbf{v})$ and message μ and output the resulting ciphertext.

- $\mathsf{Dec}(sk_{ID}, c)$: Run the dual Regev decryption algorithm with $sk := sk_{ID} = \mathbf{e}$.

5.2 Proof of (Selective) Security

As before, we will come up with alternate algorithms called $\mathsf{Setup}^*, \mathsf{KeyGen}^*$ and Enc^* (Dec^* will be the same as Dec) which the challenger will run. We will not be able to use random oracles here.

- $\mathsf{Setup}^*(ID^*, 1^\lambda)$: sample random $\mathbf{v}^*$. sample ℓ random matrices $\mathbf{B}_1, \ldots, \mathbf{B}_\ell$ and set

  $$\mathbf{A}_{i,ID_i^*} = \mathbf{B}_i$$

  sample ℓ matrices $\mathbf{B}'_1, \ldots, \mathbf{B}'_\ell$ together with their trapdoors and set

  $$\mathbf{A}_{i,1-ID_i^*} = \mathbf{B}'_i$$

  MPK^* consists of all the $\mathbf{A}_{i,b}$ and $\mathbf{v}^*$. MSK^* consists of the trapdoors of all $\mathbf{A}_{i,1-ID_i^*}$.

- $\mathsf{KeyGen}^*(ID)$: We know that $ID \neq ID^*$. Therefore, I know the trapdoor of the matrix

  $$\mathbf{A}_{ID} := [\mathbf{A}_{1,ID_1} || \ldots || \mathbf{A}_{\ell,ID_\ell}]$$

  (do you see why?)

- $\mathsf{Enc}^*(MPK^*, ID^*, \mu)$: return the dual Regev encryption of μ relative to $\mathbf{A}_{ID^*}$ and $\mathbf{v}^*$. (note that MSK^* does not tell us anything about a trapdoor for $\mathbf{A}_{ID^*}$.)

One can also prove full security with a more sophisticated proof. In one sentence, the idea is to set up $\mathbf{A}_{i,b}$ so that Algorithm* can generate secret keys for all the Q secret key queries and yet *not* be able to generate the secret key for ID^*.

5.3 CHKP: Pros and Cons

- PLUS: the scheme is secure without resorting to the random oracle model.
- MINUS: the public parameters are rather large, namely $O(nm \log q \cdot \ell)$ as opposed to GPV where it is $O(nm \log q)$. Consequently, also ciphertexts are large.
- PLUS: While we only showed selective security, one can augment the scheme to be adaptively (fully) secure.
- PLUS: The scheme naturally extends to a hierarchical IBE scheme, described next.

5.4 A Brief Note on Hierarchical IBE

Think of hierarchies in an organization. The CEO (the master key generator) can delegate access to the VP of Engineering who can in turn delegate to programmers and so forth (but not the other way round). In a hierarchical IBE, one can generate SK_{ID} using MSK; in turn, the owner of SK_{ID} can generate $SK_{ID||ID'}$ etc.

The CHKP scheme has a natural hierarchical structure. Namely, if you know the trapdoor for $\mathbf{A}_{ID}$, you can generate a trapdoor for $\mathbf{A}_{ID||ID'} = [\mathbf{A}_{ID}||\mathbf{A}_{ID'}]$. Constructing a HIBE scheme building off of this idea is left as an exercise.

6 The ABB IBE Scheme

The ABB Trick: Punctured Trapdoors.
Given the trapdoor for a matrix $\mathbf{A}_0$, a matrix $\mathbf{R}$ with small entries, and a trapdoor for $\mathbf{G}$, can you generate a trapdoor for

$$[\mathbf{A}_0||\mathbf{A}_0\mathbf{R} + \alpha \cdot \mathbf{G}]$$

for an arbitrary integer $\alpha \neq 0 \pmod{q}$?

How about for $\alpha = 0 \pmod{q}$, that is, $[\mathbf{A}_0||\mathbf{A}_0\mathbf{R}]$?

6.1 The Scheme

- $\mathsf{Setup}(1^\lambda)$: Pick the right $n = n(\lambda)$ for a security level of λ bits. Generate matrices

$$\mathbf{A}_0, \mathbf{A}_1 \in \mathbb{Z}_q^{n \times m}$$

  The master public key is

$$\mathsf{mpk} = \mathbf{A}_0, \mathbf{A}_1, \mathbf{v}$$

  where $\mathbf{v} \in \mathbb{Z}_q^n$ is a random vector, and the master secret key is

$$\mathsf{msk} = \mathbf{T}_{\mathbf{A}_0}$$

  We will never use the trapdoor for $\mathbf{A}_1$.

- $\mathsf{KeyGen}(\mathsf{msk}, ID \in \{0,1\}^\ell)$: Let h be a collision-resistant hash function that maps identities to $\mathbb{Z}_q^*$. Define

$$\mathbf{A}_{ID} := [\mathbf{A}_0 || \mathbf{A}_1 + h(ID) \cdot \mathbf{G}]$$

  where $\mathbf{G}$ is the gadget matrix. Note that by trapdoor extension, KeyGen knows a trapdoor for $\mathbf{A}_{ID}$ for any ID.

  Generate a short vector $\mathbf{e} \leftarrow \mathsf{DGSamp}(\mathbf{A}_{ID}, \mathbf{T}_{\mathbf{A}_{ID}}, \mathbf{v})$ by running the discrete Gaussian sampling algorithm. Recall that $\mathbf{A}_{ID} \cdot \mathbf{e} = \mathbf{v} \pmod q$. Output the secret key $sk_{ID} = \mathbf{e}$.

- $\mathsf{Enc}(\mathsf{mpk}, ID, \mu)$: Run the dual Regev encryption algorithm with $pk := (\mathbf{A}_{ID}, \mathbf{v})$ and message μ and output the resulting ciphertext.

- $\mathsf{Dec}(sk_{ID}, c)$: Run the dual Regev decryption algorithm with $sk := sk_{ID} = \mathbf{e}$.

6.2 ABB: Proof of Selective Security

As before, we will come up with a bunch of alternate algorithms called $\mathsf{Setup}^*, \mathsf{KeyGen}^*$ and Enc^* (Dec^* will be the same as Dec) which the challenger will run. We will not be able to use random oracles here either.

- $\mathsf{Setup}^*(ID^*, 1^\lambda)$: sample random $\mathbf{v}^*$. sample a random matrix $\mathbf{A}_0$ and a matrix $\mathbf{R}$ with small entries. Set

$$\mathbf{A}_1 := [\mathbf{A}_0 || \mathbf{A}_0\mathbf{R} - h(ID^*)\mathbf{G}]$$

  MPK^* consists of $\mathbf{A}_0, \mathbf{A}_1$ and $\mathbf{v}^*$. MSK^* consists of $\mathbf{R}$ (and the trapdoor for $\mathbf{G}$.)

- $\mathsf{KeyGen}^*(ID)$: We know that $ID \neq ID^*$. Therefore, I know the trapdoor of the matrix

$$\mathbf{A}_{ID} := [\mathbf{A}_0 || \mathbf{A}_1 + h(ID)\mathbf{G}] = [\mathbf{A}_0 || \mathbf{A}_0\mathbf{R} + (h(ID) - h(ID^*)\mathbf{G}]$$

  (do you see why?)

- $\mathsf{Enc}^*(MPK^*, ID^*, \mu)$: given a dual Regev encryption of μ relative to $\mathbf{A}_0$ and $\mathbf{v}^*$, compute a dual Regev encryption of μ relative to

$$\mathbf{A}_{ID^*} = [\mathbf{A}_0 || (\mathbf{A}_0\mathbf{R} - h(ID^*)\mathbf{G}) + h(ID^*)\mathbf{G}] = [\mathbf{A}_0 || \mathbf{A}_0\mathbf{R}]$$

  and $\mathbf{v}^*$. (do you see how to do this?)

6.3 ABB: Pros and Cons

- PLUS: the scheme is secure without resorting to the random oracle model.
- PLUS: the public parameters and ciphertexts are as small as GPV, namely $O(nm \log q)$.
- PLUS: Can be extended to full security.
- PLUS: Extensible to hierarchical IBE. A different ABB paper uses additional techniques to construct a "better" HIBE (where the lattice dimension stays the same regardless of the number of levels of delegation).

7 Application: Chosen Ciphertext Secure Public-key Encryption

We will now show a very simple construction of a chosen ciphertext secure (CCA2-secure) public-key encryption scheme from IBE. This is due to Canetti, Halevi and Katz [CHK04]. In fact, here we will describe a solution for the weaker notion of CCA1-security.

But first, the definition of CCA1-security. In the CCA1 game, the adversary gets the public-key PK of the encryption scheme, and can ask to get polynomially many ciphertexts decrypted. That is, a challenger will, on input c, run $\mathsf{Dec}(SK, c)$ and return the answer to the adversary. Note that c need not be distributed like an honestly generated ciphertext, and may not even live in the range of the encryption algorithm (i.e., may not be a valid ciphertext). Eventually, the adversary gets an encryption of a random bit b under PK and is asked to guess b. CCA1 security requires that no PPT adversary can guess b with probability better than $1/2 + \mathsf{negl}(\lambda)$.

Here is the construction.

- $\mathsf{KeyGen}(1^\lambda)$: run $\mathsf{IBE.Setup}(1^\lambda)$ to get an MPK_{IBE} and an MSK_{IBE}. The public key PK of the CCA scheme is MPK_{IBE} and the secret key SK is MSK_{IBE}.
- $\mathsf{Enc}(PK, \mu)$: pick a random string ID. Run $\mathsf{IBE.Enc}(PK = MPK_{IBE}, ID, \mu)$ and output ID together with the resulting ciphertext.
- $\mathsf{Dec}(SK, (ID, c))$: use $SK = MSK_{IBE}$ to create SK_{ID} and run the IBE decryption algorithm $\mu = \mathsf{IBE.Dec}(SK_{ID}, c)$.

The CCA security proof is super simple. The intuition?

- the decryption algorithm only uses SK_{ID} (and not the MSK per se) and
- the identity in the challenge ciphertext is random and hence different w.h.p. from the (adversarially chosen) identities in all the decryption queries.

Put together, IBE security should say that breaking the security of the challenge ciphertext is hard.

8 Registration-based Encryption

We will say just a few words about RBE here. Recall from the beginning of the lecture that a major disadvantage of IBE is the power of the master key authority to decrypt all ciphertexts.

A completely orthogonal approach which does not have this problem starts from the following strawman scheme: the master public key, curated by the authority, is the concatenation of all the users' public keys... Of course, this leads us back to exactly the PKI problem we wanted to solve. However, it is possible that the authority can publish a *short digest* of the concatenation of all public keys, which is nevertheless good enough for encryption (although it should not be clear exactly how yet!)

It turns out that this idea can be brought to fruition using the methodology of deferred encryption due to Garg et al. We refer the reader to the papers [GHMR18, GHM+19]. The construction proceeds in a completely different way from everything we saw today, and is quite inefficient. An open problem is to come up with an RBE that is as efficient as (or more efficient than!) the IBE schemes we saw here.

References

[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.

[GHM+19] Sanjam Garg, Mohammad Hajiabadi, Mohammad Mahmoody, Ahmadreza Rahimi, and Sruthi Sekar. Registration-based encryption from standard assumptions. In Dongdai Lin and Kazue Sako, editors, *Public-Key Cryptography - PKC 2019 - 22nd IACR International Conference on Practice and Theory of Public-Key Cryptography, Beijing, China, April 14-17, 2019, Proceedings, Part II*, volume 11443 of *Lecture Notes in Computer Science*, pages 63–93. Springer, 2019.

[GHMR18] Sanjam Garg, Mohammad Hajiabadi, Mohammad Mahmoody, and Ahmadreza Rahimi. Registration-based encryption: Removing private-key generator from IBE. In Amos Beimel and Stefan Dziembowski, editors, *Theory of Cryptography - 16th International Conference, TCC 2018, Panaji, India, November 11-14, 2018, Proceedings, Part I*, volume 11239 of *Lecture Notes in Computer Science*, pages 689–718. Springer, 2018.