**CSC 2419: Lattice-based Cryptography** **Fall 2025**

# Lecture 8: Attribute-based Encryption

*Instructor: Akshayaram Srinivasan* *Scribe: Ismael Gharbi*

**Date: 2025-11-10**

## 8.1 Recap

In the last lecture, we discussed *secure computation*: Alice holds an input $x$, Bob holds an input $y$, and the goal is to design two two-round secure protocol such that Alice learns $f(x, y)$ and no other information about Bob's input, and Bob learns no information about Alice's input. We then compared two lattice-based approaches for designing secure protocols with communication complexity succinct in the size of function.

## 8.2 Attribute-Based Encryption

We can observe that in both public-key encryption (PKE) and fully homomorphic encryption (FHE), semantic security guarantees an "all-or-nothing" property: without the secret key, a ciphertext reveals no information about the underlying message, and with the secret key, one recovers the entire message. In other words, access to the message is tied completely to possession of the single decryption key.

In contrast, Attribute-Based Encryption (ABE) offers a more flexible notion of access. The message are encrypted with respect to an attribute $x$. Secret keys $sk_f$ are generated with respect to function $f$ (this can be thought of as embedding *policies* in the secret key). A user, holding the secret key $sk_f$ can decrypt the message encrypted with respect to attribute $x$ iff the attribute satisfies the policy i.e. $f(x) = 0$. This variant is also referred to as *Key-Policy Attribute Based Encryption* (KP-ABE)

Let $x \in \{0,1\}^l$ and $\mathcal{F}$ be a class of function $\mathcal{F} \subseteq \left\{f : \{0,1\}^l \to \{0,1\}\right\}$ An Attribute-Based Encryption scheme for the class of functions $\mathcal{F}$ is defined by a tuple of efficient algorithms $(\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$ where

- $\mathsf{Setup}(1^\lambda) \to (mpk, msk)$

  Generate a public key $\mathsf{mpk}$ and a master secret key $msk$, which is used to issue policy-specific secret keys.

- $\mathsf{Enc}(\mathsf{mpk}, x, m) \to ct$

  On input the public key $\mathsf{mpk}$, message $m \in \{0,1\}$, and attribute $x \in \{0,1\}^l$, the encryption algorithm returns a ciphertext $ct$.

- $\mathsf{KeyGen}(msk, f) \to sk_f$

  Given a master secret key $msk$ and a function $f \in \mathcal{F}$, the key generation algorithm returns a decryption key $sk_f$ with respect to function $f$

- $\mathsf{Dec}(sk_f, ct) \to m$ On input the secret key $sk_f$ and a ciphertext $ct$. the decryption algorithm returns $m \in \{0,1\}$.

**Correctness.** For all $x \in \{0,1\}^l$, $f \in \mathcal{F}$, and $m \in \{0,1\}$ such that $f(x) = 0$

$$\Pr\left[\mathsf{Dec}(sk_f, ct) = m \middle| \begin{array}{c}(mpk, msk) \leftarrow \mathsf{Gen}(1^\lambda) \\ ct \leftarrow \mathsf{Enc}(mpk, x, m) \\ sk_f \leftarrow \mathsf{KeyGen}(msk, f)\end{array}\right] = 1$$

**Security.** Consider the following selective security game between adversary and challenger:

| **Adversary** | | **Challenger** |
|---|---|---|
| $x^* \leftarrow \mathcal{A}(1^{\text{sec}})$ | | $(mpk, msk) \leftarrow \mathsf{Setup}(1^\lambda)$ |
| | $\xrightarrow{x^*}$ | |
| | $\xleftarrow{mpk}$ | |
| Adversary has oracle access to $\mathsf{KeyGen}(msk, x^*, \cdot)$ where $KG(msk, x^*, f) = \begin{cases} sk_f = \mathsf{KeyGen}(msk, f), & \text{if } f(x^*) \neq 0 \\ \perp, & \text{otherwise} \end{cases}$ | | |
| $(m_0, m_1, state) \leftarrow \mathcal{A}^{\mathsf{KeyGen}(msk, x^*, \cdot)}(mpk)$ | | |
| | $\xrightarrow{x^*}$ | |
| | | $b \xleftarrow{\$} \{0,1\}$ |
| | | $ct \leftarrow \mathsf{Enc}(mpk, x^*, m_b)$ |
| | $\xleftarrow{ct}$ | |
| $b' \leftarrow \mathcal{A}^{\mathsf{KeyGen}(msk, x^*, \cdot)}(ct, state)$ | | |
| | $\xrightarrow{b'}$ | |

An ABE scheme is (selectively) secure if in the game above,

$$\Pr[b' = b] \leq 1/2 + \mathsf{negl}(\lambda)$$

## 8.3 Attribute Based Encryption from LWE

We recall some preliminaries on lattice trapdoors and the key equation.

### 8.3.1 Building Blocks

**Trapdoor Sampling.** We recall the notion of Trapdoor generator. A trapdoor generator is defined by the tuple of algorithms $(\mathsf{TrapSamp}, \mathsf{ExtendRight})$ where

- $\mathsf{TrapSamp}(1^\lambda, 1^m, 1^n) \to (\mathbf{A}, \mathbf{T_A})$: The Trapdoor generator algorithm returns the tuple $(\mathbf{A}, \mathbf{T_A})$ where $\mathbf{A} \in \mathbb{Z}^{n \times m}$ is a full rank matrix that is $negl(\lambda)$-close to a uniform matrix and $\mathbf{T_A} \in \mathbb{Z}^{m \times m}$ such that

  - $\mathbf{A}\mathbf{T}_{\mathbf{A}} = \mathbf{0}^{m\times m}$
  - The column of $\mathbf{T}_{\mathbf{A}}$ have low norm
  - $\mathbf{T}_{\mathbf{A}}$ has full rank over $\mathbb{Z}$

- $\mathsf{ExtendRight}(\mathbf{A}, \mathbf{T}_{\mathbf{A}}, \mathbf{B}) \to \mathbf{T}_{\mathbf{A}||\mathbf{B}}$: Given full rank matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ and $\mathbf{B} \in \mathbb{Z}_q^{n\times k}$ and the trapdoor $\mathbf{T}_{\mathbf{A}}$ for matrix $\mathbf{A}$, the deterministic algorithm returns the trapdoor $\mathbf{T}_{\mathbf{A}||\mathbf{B}}$ for the full-rank matrix $\mathbf{A}||\mathbf{B}$.

- $\mathsf{ExtendLeft}(\mathbf{A}, \mathbf{S}, \mathbf{T}_{\mathbf{G}}) \to \mathbf{T}_{\mathbf{A}\|\mathbf{A}\mathbf{S}-\mathbf{G}}$: Given a full rank matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, a low-norm matrix $\mathbf{S}$ and the trapdoor $\mathbf{T}_{\mathbf{G}}$, this deterministic algorithm returns a trapdoor $\mathbf{T}_{\mathbf{A}\|\mathbf{A}\mathbf{S}-\mathbf{G}}$. Specifically, given the trapdoor $\mathbf{T}_{\mathbf{G}}$ for the matrix $\mathbf{G}$, we can construct the trapdoor $\mathbf{T}_{\mathbf{A}\|\mathbf{A}\mathbf{S}-\mathbf{G}}$ as $\begin{pmatrix} \mathbf{S}\mathbf{T}_{\mathbf{G}} & \mathbf{I}+\mathbf{S}\mathbf{R} \\ -\mathbf{T}_{\mathbf{G}} & -\mathbf{R} \end{pmatrix}$ where $\mathbf{R}$ is a low norm matrix such that $\mathbf{G}\mathbf{R} = -\mathbf{A}$. It is easy to observe that $\mathbf{T}_{\mathbf{A}\|\mathbf{A}\mathbf{S}-\mathbf{G}}$ is low-norm and has full rank.

**Key-Equation.** We summarise the notion of *Key-Equation* from Lecture 7. There exist algorithms $(\mathsf{EvalPK}, \mathsf{EvalCT})$ such that:

- $\mathsf{EvalPK}(\mathbf{A}_1, \ldots, \mathbf{A}_l, f)$: On input $\mathbf{A}_i \in \mathbb{Z}_q^{n\times m}$ and function $fL\{0,1\}^l \to \{0,1\}$, the (deterministic algorithm) returns $\mathbf{A}_f \in \mathbb{Z}_q^{n\times m}$

- $\mathsf{EvalCT}(\mathbf{c}_1^T, \ldots, \mathbf{c}_l^T, \mathbf{A}_1, \ldots, \mathbf{A}_l, f, x)$: On input vectors $\mathbf{c}_i^T = \mathbf{s}^T(\mathbf{A}_i + x_i\mathbf{G}) + \mathbf{e}_i^T$, function $f$ and input $x$, the deterministic algorithm returns a LWE sample $\mathbf{c}_f^T = \mathbf{s}^T(\mathbf{A}_f + f(x)\mathbf{G}) + \mathbf{e}^T$ such that

$$(\mathbf{c}_1^T||\ldots||\mathbf{c}^T)\mathbf{H}_{x,f} = \mathbf{s}^T(\mathbf{A}_f + f(x)\mathbf{G}) + \mathbf{e}^T$$

  and $\mathbf{H}_{x,f}$ is a low-norm matrix

- $\mathsf{EvalSim}(f, x, \{\mathbf{A}_i\})$: If we can write $\mathbf{A}_i = \mathbf{A}\cdot\mathbf{S}_i - x_i\mathbf{G}$ where $\mathbf{S}_i \in \{\pm 1\}^{m\times m}$, then, this algorithm returns a low norm matrix $\mathbf{S}_f$ such that $\mathbf{A}_f = \mathbf{A}\mathbf{S}_f - f(x)\mathbf{G}$. Note that $\mathsf{EvalSim}$ is essentially how we do GSW homomorphic evaluation.

### 8.3.2 Construction

We provide a construction of the Attribute-Based Encryption scheme assuming the hardness of LWE.

- $\mathsf{Setup}(1^\lambda)$:
  1. Sample $(\mathbf{A}, \mathbf{T}_{\mathbf{A}}) \leftarrow \mathsf{TrapSamp}(1^\lambda, 1^m, 1^n)$.
  2. Sample $\mathbf{A}_1, \ldots, \mathbf{A}_l \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$ and $\mathbf{v} \xleftarrow{\$} \mathbb{Z}_q^m$.
  3. Return $(msk, mpk)$ where

$$mpk = (\mathbf{A}, \mathbf{A}_1, \ldots, \mathbf{A}_l, \mathbf{v})$$
$$msk = \mathbf{T}_{\mathbf{A}}$$

- $\mathsf{KeyGen}(msk, f)$
  1. Compute $\mathbf{A}_f = \mathsf{EvalPK}(\mathbf{A}_1, \ldots, \mathbf{A}, f)$

2. Sample trapdoor extension $\mathbf{T}_{\mathbf{A}||\mathbf{A}_f} = \mathsf{ExtendRight}(\mathbf{A}, \mathbf{T}_{\mathbf{A}}, \mathbf{A}_f)$. Use the trapdoor to sample low norm vector $\mathbf{r}^T$ such that $(\mathbf{A}||\mathbf{A}_f) \cdot \mathbf{r} = \mathbf{v}$.
3. Define the secret key $sk_f = \mathbf{r}$

- $\mathsf{Enc}(mpk, x, m)$

1. Compute $\mathbf{A}_x = [\mathbf{A}||\mathbf{A}_1 + x_1\mathbf{G}||\ldots||\mathbf{A}_l + x_l\mathbf{G}]$.
2. Sample LWE secret $\mathbf{s}^T \leftarrow \mathbb{Z}_q^n$, $\mathbf{e} \leftarrow \chi^{m(\ell+1)}$ and $e_1 \leftarrow \chi$.
3. Compute the ciphertext as

$$ct = (\mathbf{s}^T\mathbf{A}_x + \mathbf{e}^T, \mathbf{s}^T\mathbf{v} + e_1 + m\lfloor q/2 \rfloor)$$

. Output ciphertext $ct$

- $\mathsf{Dec}(sk_f, x, ct)$

1. Parse

$$\begin{aligned} ct &= (ct_1, ct_2) \\ ct_1 &= \mathbf{c}_{in}^T||\mathbf{c}_1^T||\mathbf{c}_2^T||\ldots||\mathbf{c}_l^T \\ sk_f &= \mathbf{r} \end{aligned}$$

2. Compute $\mathbf{c}_f^T = \mathsf{EvalCT}(\{\mathbf{c}_i^T\}, \{\mathbf{A}_i\}, f, x)$
3. Return $\mathsf{Round}(ct_2 - (\mathbf{c}_{in}^T||\mathbf{c}_f^T) \cdot \mathbf{r})$

**Correctness** Observe that a valid ciphertext $ct = (ct_1, ct_2)$ is of the form

$$\begin{aligned} ct_1 &= \mathbf{s}^T(A||\mathbf{A}_1 + x_1\mathbf{G}||\ldots||\mathbf{A}_l + x_l\mathbf{G}) + \mathbf{e}^T) \\ &= \mathbf{c}_{in}^T||\mathbf{c}_1^T||\ldots||\mathbf{c}_l^T \end{aligned}$$

where $\mathbf{c}_{in}^T = \mathbf{s}^T(\mathbf{A}) + \mathbf{e}_0^T$ and $\mathbf{c}_i^T = \mathbf{s}^T(\mathbf{A}_i + x_i\mathbf{G}) + \mathbf{e}_0^T\mathbf{S}_i = \mathbf{s}^T(\mathbf{A}_i + x_i\mathbf{G}) + \mathbf{e}_i^{T\prime}$ where $\mathbf{e}_i^{T\prime}$ is a low-norm error. If we have $(f, x)$ such that $f(x) = 0$, then, from the correctness of $\mathsf{EvalCT}$, the decryption algorithm computes $\mathbf{c}_f^T = \mathbf{s}^T(\mathbf{A}_f + f(x)\mathbf{G}) + \mathbf{e}^T = \mathbf{s}^T\mathbf{A}_f + \mathbf{e}^T$

$$\begin{aligned} &\implies \mathbf{c}_{in}^T||\mathbf{c}_f^T = \mathbf{s}^T(\mathbf{A}||\mathbf{A}_f) + \mathbf{e}_0^T||\mathbf{e}^T \\ &\implies (\mathbf{c}_{in}^T||\mathbf{c}_f^T)\mathbf{r} = \mathbf{s}^T(\mathbf{A}||\mathbf{A}_f)\mathbf{r} + (\mathbf{e}_0^T||\mathbf{e}^T)\mathbf{r} \\ &\qquad\qquad = \mathbf{s}^T\mathbf{v} + e' \\ &\implies ct_2 - (\mathbf{c}_{in}^T||\mathbf{c}_f^T)\mathbf{r} = m\lfloor q/2 \rfloor + \overline{e} \\ &\implies \mathsf{Round}(ct_2 - (\mathbf{c}_{in}^T||\mathbf{c}_f^T)\mathbf{r}) = m \end{aligned}$$

where the last equality holds with high probability.

**Security** We want to prove that the ciphertext of message $m$ and attribute $x$ reveals no information about the message given access to secret keys $sk_f$ such that $f(x) = 1$. We provide a brief security proof sketch below:

- *Hybrid 0*: This is the security game defined between adversary and challenger.
- *Hybrid 1*: In this hybrid, the challenger samples the public matrices $\mathbf{A}_i$ as follows: it samples $\mathbf{A}_i^* \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$ and sets $\mathbf{A}_i = \mathbf{A}_i^* - x_i\mathbf{G}$. This hybrid is indistinguishable from *Hybrid 0* since the public matrices are still random.
- *Hybrid 2:* In this hybrid, the matrices are samples as follows: $\mathbf{A}_i^* = \mathbf{A}\cdot\mathbf{S}_i$ where $\mathbf{S}_i \xleftarrow{\$} \{\pm 1\}^{m\times m}$ and set $\mathbf{A}_i = \mathbf{A}\cdot\mathbf{S}_i - x_i\mathbf{G}$. Using Leftover Hash Lemma, we can conclude that this hybrid is statistically indistinguishable from the previous hybrid.
- *Hybrid 3:* In this hybrid, we generate $ct_1$ as follows:

$$\begin{aligned} ct_1 =& \mathbf{s}^T\mathbf{A}_x + \mathbf{e}_0^T(\mathbf{S}_0||\ldots\mathbf{S}_l)) + \mathbf{e}^T \\ =& \mathbf{s}^T\mathbf{A}(\mathbf{I}||\mathbf{S}_0||\ldots||\mathbf{S}_l) + \mathbf{e}_0^T(\mathbf{S}_0||\ldots\mathbf{S}_l)) + \mathbf{e}^T \\ =& (\mathbf{s}^T\mathbf{A} + \mathbf{e}_0^T)\cdot(\mathbf{I}||\mathbf{S}_0||\ldots||\mathbf{S}_l) + \mathbf{e}^T \end{aligned}$$

  This hybrid is statistically close to the previous hybrid from the Gaussian noise smudging.
- *Hybrid 4* In this hybrid, whenever the adversary queries the oracle $\mathsf{KeyGen}(msk, x^*, f)$, the oracle response is as follows:
  - Compute $\mathbf{S}_f \leftarrow \mathsf{EvalSim}(f, x, \{\mathbf{S}_i\})$ such that $\mathbf{A}\mathbf{S}_f - f(x)\mathbf{G} = \mathbf{A}\mathbf{S}_f - \mathbf{G} = \mathbf{A}_f$.
  - The oracle can now sample the secret key $\mathbf{r}$ given that we can compute the trapdoor of matrix $(\mathbf{A}||\mathbf{A}\mathbf{S}_f - \mathbf{G})$ from $\mathsf{ExtendLeft}$ without requiring the trapdoor for $\mathbf{A}$. If $\mathbf{T}_{\mathbf{A}||\mathbf{A}\mathbf{S}_f-\mathbf{G}}$ be the trapdoor, then $\mathbf{r}$ is sampled such that

$$(\mathbf{A}||\mathbf{A}\mathbf{S}_f - \mathbf{G})\mathbf{r} = \mathbf{v}$$

  This approach of sampling the secret key is statistically close to the normal way from the trapdoor presampling lemma. Subsequently, we note that the ciphertext is of the form:

$$\begin{aligned} ct_1 =& (\mathbf{s}^T\mathbf{A} + \mathbf{e}_0^T)\cdot(\mathbf{I}||\mathbf{S}_0||\ldots||\mathbf{S}_l) + \mathbf{e}^T \\ ct_2 =& \mathbf{s}^T\mathbf{v} + e_1 + m\lfloor q/2\rfloor \end{aligned}$$

  we can equivalently think of this ciphertext as an LWE sample $ct = \mathbf{s}^T\mathbf{A}' + \mathbf{e}^T + [0, 0, \ldots, m\lfloor q/2\rfloor]$ where $\mathbf{A}' = \mathbf{A}||\mathbf{v}$ and $\mathbf{e}^{T'} = \mathbf{e}^T||e_1$. By invoking the hardness of LWE, we can see that this ciphertext reveals no information about $m$

**Remark 8.1** *It is important to note that we cannot directly invoke hardness of LWE and have to go through the hybrids described above. This is because we want to show that information that adversary gains in the security game, namely $sk_f$ can be simulated without the adversary learning the trapdoor for matrix $\mathbf{A}$.*