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

Lecture 4: Digital Signatures and Fully Homomorphic Encryption

Instructor: Akshayaram Srinivasan *Scribe: Hunter Richards*

Date: 2025-09-29

4.1 Recap

In the last lecture, we saw that injective trapdoor functions imply public-key encryption (PKE), but the other direction does not always hold. Thus, trapdoor functions are a strictly stronger primitive than PKE.

We also observed that solving LWE, i.e., recovering both the secret and the error from an LWE sample, is equivalent to recovering just one of them. Because of the linear structure of LWE, once either the secret or the error is known, the other can be efficiently determined.

Finally, we saw how to construct a trapdoor function from LWE. Specifically, we learned how to sample an LWE matrix and its corresponding trapdoor $(\mathbf{A}, \mathbf{T}) \leftarrow \mathsf{TrapGen}(1^n)$ such that the marginal distribution of $\mathbf{A}$ is statistically close to uniform. This ensures that the hardness of LWE holds with respect to $\mathbf{A}$.

4.2 Digital Signatures

4.2.1 Trapdoor Presampling Lemma

Lemma 4.1 *Given $q \in \mathbb{Z}$ and $m, n \in \mathbb{N}$, define the following two distributions over the triple $(\mathbf{A}, \mathbf{e}, \mathbf{v})$, where $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, $\mathbf{v} \in \mathbb{Z}_q^n$, and $\mathbf{e} \in \mathbb{Z}_q^m$.*

1. ***Forward sampling:*** *Let D be a distribution[1] over $\mathbb{Z}_q^m$. Output*

$$(\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}, \mathbf{e} \leftarrow D^m, \mathbf{v} := \mathbf{A} \cdot \mathbf{e} \pmod q).$$

2. ***Trapdoor presampling:*** *Sample $(\mathbf{A}, \mathbf{T}) \leftarrow \mathsf{TrapGen}(1^n)$. Output*

$$(\mathbf{A}, \mathbf{e} \leftarrow \mathsf{TrapPreSamp}(\mathbf{A}, \mathbf{T}, \mathbf{v}), \mathbf{v} \leftarrow \mathbb{Z}_q^n).$$

The distributions over $(\mathbf{A}, \mathbf{e}, \mathbf{v})$ in the forward and trapdoor presampling directions are statistically close.

Proof idea: It is a property of the distribution D that $\mathbf{v} := \mathbf{A} \cdot \mathbf{e} \pmod q$ is statistically close to uniform, where $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}$ and $\mathbf{e} \leftarrow D^m$. We also know that $\mathbf{A}$ is statistically close to uniform when sampled according to $(\mathbf{A}, \mathbf{T}) \leftarrow \mathsf{TrapGen}(1^n)$. Therefore, the marginal distributions of $(\mathbf{A}, \mathbf{v})$ are statistically close to uniform over $\mathbb{Z}_q^{n \times m} \times \mathbb{Z}_q^n$.

Lastly, $\mathsf{TrapPreSamp}(\cdot)$ efficiently samples $\mathbf{e}$ from D^m conditioned on $\mathbf{A} \cdot \mathbf{e} \equiv \mathbf{v} \pmod q$ such that the marginal distribution of $\mathbf{e}$ is statistically close to D^m. See [GPV08] for details. ■

[1] D is a B-bounded discrete Gaussian distribution. See [GPV08].

Let's compare Lemma 4.1 to the definition of a trapdoor function (TDF) discussed in the last lecture. Like a TDF, we can efficiently evaluate the forward direction with only the public key, which in the case of Lemma 4.1 is $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}$. If we sample a domain point $\mathbf{e} \leftarrow D^m$, then we can efficiently determine the codomain point $\mathbf{v} = \mathbf{A} \cdot \mathbf{e} \pmod{q}$. Where Lemma 4.1 and a TDF differ is that trapdoor presampling is neither injective nor deterministic in its inverse.

As a consequence of Lemma 4.1, the marginal distribution of $\mathbf{v}$ is statistically close to uniform in the forward direction. Thus, in the inverse direction we can sample a uniform codomain point $\mathbf{v} \leftarrow \mathbb{Z}_q^n$ and, with the trapdoor $\mathbf{T}$, efficiently sample a preimage $\mathbf{e}$ from the conditional distribution

$$\mathbf{e} \sim D^m \mid \mathbf{A} \cdot \mathbf{e} \equiv \mathbf{v} \pmod{q}.$$

In other words, the trapdoor does not recover a unique preimage (since many $\mathbf{e}$ map to the same $\mathbf{v}$), but rather samples one according to the distribution over all valid preimages.

This is a stronger property than that of injective TDFs, because it yields a randomized preimage, rather than a unique one. As a result, it supports more advanced cryptographic constructions, such as digital signatures.

4.2.2 Definition of a Digital Signature

Digital signatures are the digital analogue of handwritten signatures. A valid scheme must satisfy two properties: correctness, meaning anyone can verify that a signature was generated by the legitimate signer, and unforgeability, meaning no one else can produce a valid signature on a new message.

Definition 4.2 (Digital Signature Scheme) *A digital signature scheme consists of three PPT algorithms:*

- $\mathsf{KeyGen}(1^\lambda) \to (\mathsf{vk}, \mathsf{sk})$*: Outputs a public verification key* vk *and a secret signing key* sk*.*
- $\mathsf{Sign}(\mathsf{sk}, \mathsf{msg}) \to \sigma$*: Outputs a signature* σ *on a message* msg*.*
- $\mathsf{Verify}(\mathsf{vk}, \mathsf{msg}, \sigma) \to b$*: Outputs* $b \in \{\mathsf{accept}, \mathsf{reject}\}$*.*

Definition 4.3 (Correctness) *For all key pairs* $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$*, all messages* msg*, and all signatures* $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, \mathsf{msg})$*, it holds that*

$$\mathsf{Verify}(\mathsf{vk}, \mathsf{msg}, \sigma) = \mathsf{accept}.$$

Definition 4.4 (Unforgeability) *A digital signature scheme is chosen-message unforgeable if, for all PTT adversaries* $\mathcal{A}$*, the following has a negligible probability of success:*

1. $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$ *and* vk *is given to* $\mathcal{A}$*.*
2. $\mathcal{A}$ *queries a signing oracle on messages* $\mathsf{msg}^{(1)}, \ldots, \mathsf{msg}^{(q)}$*; receiving* $\sigma^{(i)} = \mathsf{Sign}(\mathsf{sk}, \mathsf{msg}^{(i)})$ *for each.*
3. *Eventually,* $\mathcal{A}$ *outputs* $(\mathsf{msg}^*, \sigma^*)$*.*

$\mathcal{A}$ *wins if* $\mathsf{msg}^* \notin \{\mathsf{msg}^{(1)}, \ldots, \mathsf{msg}^{(q)}\}$ *and* $\mathsf{Verify}(\mathsf{vk}, \mathsf{msg}^*, \sigma^*) = \mathsf{accept}$*. For any polynomial* $q(\lambda)$*, we require* $\Pr[\mathcal{A} \text{ wins}] \leq \mu(\lambda)$ *for some negligible function* $\mu(\lambda)$*.*

The adversary tries to achieve accept on a message msg^* for which it did not receive a valid signature. Note that q can be any arbitrary polynomial, it does not have to be fixed at the time of setup, and it can run in polynomial time larger than that of KeyGen.

4.2.3 Digital Signatures from Trapdoor Presampling Lemma

Now, let's use the trapdoor presampling lemma to construct a candidate digital signature scheme:

- $\mathsf{KeyGen}(1^n)$: Sample $(\mathbf{A}, \mathbf{T}) \leftarrow \mathsf{TrapGen}(1^n)$. Output $\mathsf{vk} := \mathbf{A}$ and $\mathsf{sk} := (\mathbf{A}, \mathbf{T})$.
- $\mathsf{Sign}(\mathsf{sk} = (\mathbf{A}, \mathbf{T}), \mathsf{msg} \in \mathbb{Z}_q^n)$: Set $\mathbf{v} := \mathsf{msg}$. Output $\sigma := \mathbf{e} \leftarrow \mathsf{TrapPreSamp}(\mathbf{A}, \mathbf{T}, \mathbf{v})$.
- $\mathsf{Verify}(\mathsf{vk} = \mathbf{A}, \mathsf{msg} = \mathbf{v}, \sigma = \mathbf{e})$: If $\|\mathbf{e}\| \leq B$ and $\mathbf{A} \cdot \mathbf{e} \equiv \mathbf{v} \pmod q$ output accept. Else, output reject.

Theorem 4.5 (Correctness) *For all key pairs $(\mathbf{A}, (\mathbf{A}, \mathbf{T})) \leftarrow \mathsf{KeyGen}(1^n)$, all messages $\mathbf{v} \in \mathbb{Z}_q^n$, and all signatures $\mathbf{e} \leftarrow \mathsf{Sign}((\mathbf{A}, \mathbf{T}), \mathbf{v})$, it holds that*

$$\mathsf{Verify}(\mathbf{A}, \mathbf{v}, \mathbf{e}) = \mathsf{accept}.$$

Proof idea: By construction, $\mathbf{e}$ is sampled from the distribution

$$\mathbf{e} \sim D^m \mid \mathbf{A} \cdot \mathbf{e} \equiv \mathbf{v} \pmod q.$$

Therefore, every signature output $\mathbf{e}$ satisfies $\mathbf{A} \cdot \mathbf{e} \equiv \mathbf{v} \pmod q$. Moreover, since D is a B-bounded distribution, $\|\mathbf{e}\| \leq B$ with overwhelming probability. ■

Theorem 4.6 (Unforgeability) *Assuming the hardness of SIS, the above scheme is chosen-message unforgeable.*

Proof attempt: Recall that $\mathcal{A}$ has oracle access to $\mathsf{Sign}(\mathsf{sk}, \cdot)$. Each signature $\mathbf{e}^{(i)}$ is distributed statistically close to D^m independent of the trapdoor $\mathbf{T}$. Thus, $\mathcal{A}$ learns nothing about $\mathbf{T}$ from its queries; it only sees noise.

To produce a forgery $(\mathbf{v}^*, \mathbf{e}^*)$, $\mathcal{A}$ must find short vector $\mathbf{e}^*$ such that $\mathbf{A} \cdot \mathbf{e}^* \equiv \mathbf{v}^* \pmod q$ for some new message $\mathbf{v}^*$. Since $\mathbf{A}$ is statistically close to uniform, this corresponds to solving an instance of SIS. By the hardness assumption, the success probability of $\mathcal{A}$ is negligible. ■

Herein lies the importance of the trapdoor presampling lemma and the randomness of the preimage $\mathbf{e}$. If our trapdoor were injective, an adversary could learn enough information about $\mathbf{T}$ after a polynomial number of queries to recover the signing key. Instead, preimages look like random noise. Thus, the adversary is forced to solve a problem indistinguishable from solving SIS, which we know to be hard.

However, there is still a critical flaw with this candidate that disproves unforgeability, which we must fix. Consider the following counter-example:

The adversary $\mathcal{A}$ makes two queries to the signing oracle with $\mathbf{v}, \mathbf{v}'$ and receives $\mathbf{e}, \mathbf{e}'$ such that

$$\begin{aligned} \mathbf{A} \cdot \mathbf{e} &\equiv \mathbf{v} \pmod q \\ \mathbf{A} \cdot \mathbf{e}' &\equiv \mathbf{v}' \pmod q. \end{aligned}$$

Then, $\mathcal{A}$ produces $(\mathbf{v}^*, \mathbf{e}^*) = (\mathbf{v} + \mathbf{v}', \mathbf{e} + \mathbf{e}')$. It follows that

$$\begin{aligned} \mathbf{A} \cdot \mathbf{e} + \mathbf{A} \cdot \mathbf{e}' &\equiv \mathbf{v} + \mathbf{v}' \pmod q \\ \mathbf{A} \cdot (\mathbf{e} + \mathbf{e}') &\equiv \mathbf{v} + \mathbf{v}' \pmod q \\ \mathbf{A} \cdot \mathbf{e}^* &\equiv \mathbf{v}^* \pmod q. \end{aligned}$$

By standard tail bounds, $\mathbf{e}^* = \mathbf{e} + \mathbf{e}'$ will also be short. Therefore, $\mathsf{Verify}(\mathbf{A}, \mathbf{v}^*, \mathbf{e}^*) = \mathsf{accept}$ and $\mathcal{A}$ wins.

Our candidate digital signature scheme is inherently malleable because of the underlying SIS construction. We must break the linear relationship between the signature $\mathbf{e}$ and the message $\mathbf{v}$.

4.2.4 Achieving Unforgeability with Random Oracles

To eliminate exploitable correlations, we replace the message by a hash. Specifically, let H be a hash function, and require

$$\mathbf{A} \cdot \mathbf{e} \equiv H(\mathsf{msg}) \pmod{q}.$$

It is not sufficient for H to be merely collision resistant. For Lemma 4.1 to apply, we require $\mathbf{v}$ to be statistically close to uniform. Therefore, H must behave as a *random oracle*, meaning that for every new input msg, $H(\mathsf{msg})$ is an independent uniform sample from $\mathbb{Z}_q^n$. Under this assumption, the scheme becomes unforgeable.

4.3 Fully Homomorphic Encryption

One of the landmark achievements in modern cryptography is fully homomorphic encryption (FHE), which brought lattice-based cryptography to the forefront decades after Ajtai's 1996 worst-case to average-case reduction [A96]. Breakthroughs by Gentry, Brakerski and Vaikuntanathan showed how to build encryption schemes that allow computation on encrypted data, earning them the Gödel Prize in 2022.

Recall that a public-key encryption scheme consists of algorithms $\mathsf{KeyGen}(1^n) \to (\mathsf{pk}, \mathsf{sk})$, $\mathsf{Enc}(\mathsf{pk}, \mathsf{msg}) \to \mathsf{ct}$, and $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) \to \mathsf{msg}$. In practice, this lets a user encrypt private data before outsourcing it to a cloud provider, ensuring (by semantic security) that the provider learns nothing about the plaintext. But if the user later wants to, say, search their encrypted emails, the naive solution of downloading and decrypting everything locally is inefficient. FHE overcomes this by enabling the server to apply a *homomorphic evaluation operation* directly to ciphertexts and return a compact result that the client can decrypt.

Given a public key pk, a function $f : \{0,1\}^\ell \to \{0,1\}$, and ciphertexts $\mathsf{ct}^{(i)} = \mathsf{Enc}(\mathsf{pk}, x^{(i)})$, a homomorphic evaluation operation computes

$$\mathsf{ct}^* \leftarrow \mathsf{Eval}(\mathsf{pk}, f, \mathsf{ct}^{(1)}, \ldots, \mathsf{ct}^{(\ell)}),$$

such that decryption yields

$$\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}^*) = f(x^{(1)}, \ldots, x^{(\ell)}).$$

Here, f represents the function you want to apply to the underlying data (e.g., searching emails for keywords). Crucially, evaluation does not depend on the secret key. Due to semantic security, the ciphertexts along with the public key provide no information about the underlying message.

The idea of homomorphic encryption was first proposed in 1978 by Rivest, Adleman and Dertouzos [RAD78], but it was not until Gentry's PhD thesis in 2009 [G09] that the first candidate scheme capable of evaluating general Boolean circuits was introduced. However, Gentry's construction relied on a novel lattice-based assumption. Later, Brakerski and Vaikuntanathan would give a construction based on the standard LWE assumption [BV14].

We will study an especially simple LWE-based scheme developed by Gentry, Sahai and Waters [GSW13] that supports general Boolean circuit evaluation.

4.3.1 Homomorphisms in LWE-based PKE

Recall the LWE-based PKE scheme from last lecture:

- $\mathsf{KeyGen}(1^n)$: Sample $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}$, $\mathbf{s} \leftarrow \mathbb{Z}_q^n$, $\mathbf{e} \leftarrow \chi^m$. Output $\mathsf{pk} := (\mathbf{A}, \mathbf{b}^\top := \mathbf{s}^\top \mathbf{A} + \mathbf{e}^\top)$ and $\mathsf{sk} := \mathbf{s}$.
- $\mathsf{Enc}(\mathsf{pk}, x \in \{0,1\})$: Sample $\mathbf{r} \leftarrow \{0,1\}^m$. Output $(\mathsf{ct}_1, \mathsf{ct}_2) := (\mathbf{A}\mathbf{r}, \mathbf{b}^\top \mathbf{r} + x \cdot q/2) \pmod{q}$.

- $\mathsf{Dec}(\mathsf{sk}, (\mathsf{ct}_1, \mathsf{ct}_2))$: If $|\mathsf{ct}_2 - \mathbf{s}^\top \mathsf{ct}_1| < q/4$, output 0. Else, output 1.

For two ciphertexts $(\mathsf{ct}_1, \mathsf{ct}_2)$ and $(\mathsf{ct}'_1, \mathsf{ct}'_2)$,

$$\begin{aligned}(\mathsf{ct}_1, \mathsf{ct}_2) + (\mathsf{ct}'_1, \mathsf{ct}'_2) &\equiv (\mathsf{ct}_1 + \mathsf{ct}'_1, \mathsf{ct}_2 + \mathsf{ct}'_2) && \pmod q \\ &\equiv (\mathbf{Ar} + \mathbf{Ar}',\ \mathbf{b}^\top \mathbf{r} + x \cdot q/2 + \mathbf{b}^\top \mathbf{r}' + x' \cdot q/2) && \pmod q \\ &\equiv (\mathbf{A}(\mathbf{r} + \mathbf{r}'),\ \mathbf{b}^\top (\mathbf{r} + \mathbf{r}') + (x + x') \cdot q/2) && \pmod q.\end{aligned}$$

Since $(x + x') \cdot q/2 \equiv q/2 \pmod q$ iff exactly one of x, x' is 1, addition homomorphically evaluates XOR:

$$(\mathsf{ct}_1, \mathsf{ct}_2) + (\mathsf{ct}'_1, \mathsf{ct}'_2) \equiv (\mathbf{A}(\mathbf{r} + \mathbf{r}'),\ \mathbf{b}^\top (\mathbf{r} + \mathbf{r}') + (x \oplus x') \cdot q/2) \pmod q.$$

Moreover,

$$(\mathsf{ct}_2 + \mathsf{ct}'_2) - \mathbf{s}^\top (\mathsf{ct}_1 + \mathsf{ct}'_1) \equiv \mathbf{e}^\top (\mathbf{r} + \mathbf{r}') + (x \oplus x') \cdot q/2 \pmod q.$$

Therefore, decryption succeeds provided $|\mathbf{e}^\top (\mathbf{r} + \mathbf{r}')| < q/4$. That is, the noise stays within some threshold. However, XOR is not enough to evaluate general Boolean circuits. One must also be able to evaluate AND.

4.3.2 FHE Scheme Inspired by GSW

There is no natural way to multiply vectors, so to achieve homomorphisms with respect to both addition and multiplication we turn to matrices. The following is a candidate LWE-based PKE construction that demonstrates homomorphic properties.

Encryption. Assume $m = n + 1$. Rearrange the public and secret keys as

$$\mathbf{C} := \begin{bmatrix} \mathbf{A} \\ \mathbf{b}^\top \end{bmatrix} \in \mathbb{Z}_q^{(n+1) \times m}, \qquad \begin{bmatrix} -\mathbf{s} \\ 1 \end{bmatrix} \in \mathbb{Z}_q^{n+1}.$$

Sample $\mathbf{R} \leftarrow \{0,1\}^{m \times (n+1)}$, and let $\tilde{x} := x \cdot q/2$. Encryption is then

$$\mathbf{C} \cdot \mathbf{R} + \tilde{x} \cdot \mathbf{I} \pmod q,$$

where $\mathbf{I}$ denotes the $(n+1) \times (n+1)$ identity matrix.

Decryption. Compute

$$\begin{aligned}\begin{bmatrix} -\mathbf{s}^\top & 1 \end{bmatrix} (\mathbf{CR} + \tilde{x} \cdot \mathbf{I}) &\equiv (-\mathbf{s}^\top \mathbf{A} + \mathbf{b}^\top)\mathbf{R} + \tilde{x} \cdot \begin{bmatrix} -\mathbf{s}^\top & 1 \end{bmatrix} \mathbf{I} && \pmod q \\ &\equiv \mathbf{e}^\top \mathbf{R} + \tilde{x} \cdot \begin{bmatrix} -\mathbf{s}^\top & 1 \end{bmatrix} && \pmod q.\end{aligned}$$

From the last column this yields

$$\mathbf{e}^\top \mathbf{r}_{n+1} + x \cdot q/2 \pmod q,$$

where $\mathbf{r}_j \in \{0,1\}^m$ is the j-th column of $\mathbf{R}$. Since $\mathbf{e}^\top \mathbf{r}_{n+1}$ has small norm, decryption succeeds as long as $|\mathbf{e}^\top \mathbf{r}_{n+1}| < q/4$.

Homomorphic addition. For ciphertexts $\mathsf{ct}, \mathsf{ct}'$ encrypting x, x' we obtain

$$\mathsf{ct} + \mathsf{ct}' \equiv \mathbf{C}(\mathbf{R} + \mathbf{R}') + (\tilde{x} + \tilde{x}') \cdot \mathbf{I} \pmod q,$$

which corresponds to a homomorphic evaluation of $x \oplus x'$ with increased noise in $\mathbf{R} + \mathbf{R}' \in \{0,1,2\}^{m \times (n+1)}$. The noise will continue to grow as we do homomorphic operations.

Homomorphic multiplication (naïve). Multiplying two ciphertexts directly gives

$$(\mathbf{CR} + \tilde{x} \cdot \mathbf{I})(\mathbf{CR}' + \tilde{x}' \cdot \mathbf{I}) \equiv \mathbf{C}(\mathbf{RCR}' + \tilde{x} \cdot \mathbf{R}' + \tilde{x}' \cdot \mathbf{R}) + \tilde{x}\tilde{x}' \cdot \mathbf{I} \pmod{q}.$$

The problematic term $\mathbf{RCR}'$ has large norm, so correctness fails. To repair this, we introduce the gadget matrix from the last lecture.

Gadget matrix. Define

$$\mathbf{G} := \begin{bmatrix} \mathbf{g}^\top & & \\ & \ddots & \\ & & \mathbf{g}^\top \end{bmatrix}, \qquad \mathbf{g}^\top = \begin{bmatrix} 1 & 2 & \cdots & q/2 \end{bmatrix}.$$

Let $\mathbf{G}^{-1} : \mathbb{Z}_q \to \{0,1\}^{\log q}$ denote the bit-decomposition map. As before, $\mathbf{G} \cdot \mathbf{G}^{-1}(x) = x$.

Encryption. Assume $m \geq 2n \log q$. Sample $\mathbf{R} \leftarrow \{0,1\}^{m \times (n+1)\log q}$ and redefine encryption as

$$\mathbf{C} \cdot \mathbf{R} + x \cdot \mathbf{G} \pmod{q},$$

where $\mathbf{G}$ is the gadget matrix with dimension $(n+1) \times (n+1)\log q$.

Decryption. Compute

$$\begin{bmatrix} -\mathbf{s}^\top & 1 \end{bmatrix} (\mathbf{CR} + x \cdot \mathbf{G}) \equiv \mathbf{e}^\top \mathbf{R} + x \cdot \begin{bmatrix} -\mathbf{s}^\top & 1 \end{bmatrix} \mathbf{G} \pmod{q}.$$

From the last column this yields

$$\mathbf{e}^\top \mathbf{r}_{(n+1)\log q} + x \cdot \begin{bmatrix} -\mathbf{s}^\top & 1 \end{bmatrix} \begin{bmatrix} 0^n \\ q/2 \end{bmatrix} \equiv \mathbf{e}^\top \mathbf{r}_{(n+1)\log q} + x \cdot q/2 \pmod{q},$$

recovering x as before.

Theorem 4.7 (Semantic Security) *The above homomorphic encryption scheme is semantically secure.*

Proof idea: Under the LWE assumption, we can replace $\mathbf{b}$ with a uniform random vector $\mathbf{u} \in \mathbb{Z}_q^m$. Each column of $\mathbf{R}$ is an independent random binary vector, so by the Leftover Hash Lemma both $\mathbf{A}\mathbf{r}_j$ and $\mathbf{u}^\top \mathbf{r}_j$ are statistically close to uniform. Consequently, $\mathbf{CR}$ is statistically close to uniform over $\mathbb{Z}_q^{(n+1)\times(n+1)\log q}$.

Adding the term $x \cdot \mathbf{G}$ merely shifts this uniform distribution by a fixed offset, leaving it statistically uniform. Hence, the ciphertexts for $x = 0$ and $x = 1$ are statistically indistinguishable, and the scheme achieves semantic security. ■

Homomorphic addition. For ciphertexts $\mathsf{ct}, \mathsf{ct}'$,

$$\mathsf{ct} + \mathsf{ct}' \equiv \mathbf{C}(\mathbf{R} + \mathbf{R}') + (x + x') \cdot \mathbf{G} \pmod{q}.$$

Homomorphic multiplication. Let $\mathbf{B} := \mathbf{CR}' + x' \cdot \mathbf{G}$. Define multiplication as

$$\begin{aligned}
\mathsf{ct} \times \mathsf{ct}' &\equiv (\mathbf{CR} + x \cdot \mathbf{G}) \cdot \mathbf{G}^{-1}(\mathbf{B}) && \pmod q \\
&\equiv \mathbf{CR} \cdot \mathbf{G}^{-1}(\mathbf{B}) + x \cdot \mathbf{B} && \pmod q \\
&\equiv \mathbf{CR} \cdot \mathbf{G}^{-1}(\mathbf{B}) + x \cdot \mathbf{CR}' + xx' \cdot \mathbf{G} && \pmod q \\
&\equiv \mathbf{C}(\mathbf{R} \cdot \mathbf{G}^{-1}(\mathbf{B}) + x \cdot \mathbf{R}') + xx' \cdot \mathbf{G} && \pmod q.
\end{aligned}$$

After a single homomorphic multiplication,

$$\mathbf{R}_1 := \mathbf{R} \cdot \mathbf{G}^{-1}(\mathbf{B}) + x \cdot \mathbf{R}',$$

whose entries have magnitude of order $O(m)$.

After a second homomorphic multiplication,

$$\mathbf{R}_2 := \mathbf{R}_1 \cdot \mathbf{G}^{-1}(\mathbf{B}') + x \cdot \mathbf{R}'_1,$$

the entries grow to order $O(m^2)$.

Each entry of $\mathbf{R}_2$ is a sum of $O(m)$ products, with each product having magnitude $O(\|\mathbf{R}_1\|_\infty) = O(m)$, yielding $\|\mathbf{R}_2\|_\infty = O(m^2)$. In general, after d levels, the infinity norm grows as $\|\mathbf{R}_d\|_\infty = O(m^d)$.

How large can d be before decryption fails? At depth d, the ciphertext has the form

$$\mathbf{CR}_d + f(x^{(1)}, \ldots, x^{(\ell)}) \cdot \mathbf{G} \pmod q,$$

where $f : \{0,1\}^\ell \to \{0,1\}$ is the Boolean function computed homomorphically on the message bits.

Decryption produces

$$\mathbf{e}^\top \mathbf{R}_d + f(x^{(1)}, \ldots, x^{(\ell)}) \cdot \begin{bmatrix} -\mathbf{s}^\top & 1 \end{bmatrix} \mathbf{G} \pmod q.$$

From the last column we obtain

$$\mathbf{e}^\top \mathbf{r}_{d,(n+1)\log q} + f(x^{(1)}, \ldots, x^{(\ell)}) \cdot q/2 \pmod q.$$

For correctness, the noise term must remain small:

$$|\mathbf{e}^\top \mathbf{r}_{d,(n+1)\log q}| < q/4.$$

Because $\|\mathbf{e}\| \le B$ and $\|\mathbf{R}_d\|_\infty = O(m^d)$, the magnitude of the noise term grows as

$$O(m^{d+1}) \cdot B < q/4.$$

Substituting $m \approx 2n \log q$, this becomes

$$2^{O(\log n)d} \cdot B < q/4.$$

Assume q has the form $q = 2^{n^\epsilon}$. Then, the largest supported depth is $d \approx n^{\epsilon'}$ for some $\epsilon' < \epsilon$.

In other words, decryption remains correct only while the accumulated noise, which grows exponentially with the number of multiplications, stays below $q/4$. The modulus q therefore determines the maximum homomorphic depth that the scheme can handle.

4.3.3 Bootstrapping

The leveled FHE scheme described so far supports only a fixed, bounded number of homomorphic operations before decryption fails due to noise growth. To evaluate circuits of arbitrary depth, we use a technique called *bootstrapping*.

Decryption in LWE-based schemes can be represented as a circuit of depth $O(\log n)$. Therefore, if our scheme can evaluate circuits of at least this depth, it can evaluate its own decryption circuit *homomorphically*.

Suppose we have a fresh encryption of the secret key, $\mathsf{Enc}(\mathsf{pk}, \mathsf{sk})$, under the same scheme. Given a ciphertext ct with large noise, we define a function $f_{\mathsf{ct}}(\mathsf{sk}) = \mathsf{Dec}(\mathsf{sk}, \mathsf{ct})$ that outputs a decryption of ct given sk. We can then evaluate this function homomorphically:

$$\mathsf{Eval}(\mathsf{pk}, f_{\mathsf{ct}}, \mathsf{Enc}(\mathsf{pk}, \mathsf{sk})) \to \mathsf{ct}'.$$

The resulting ciphertext ct' encrypts the same message, but its noise now depends only on the norm bound of $\mathsf{Enc}(\mathsf{pk}, \mathsf{sk})$ and the depth of f_{ct}.

The decryption circuit itself has depth $O(\log n)$. Because $\mathsf{Enc}(\mathsf{pk}, \mathsf{sk})$ is a freshly encrypted ciphertext with small norm, this leaves an overall noise bound of $O(m^{\log n})$. Hence, the refreshed ciphertext ct' has noise of order $O(m^{\log n})$.

From here, one can resume homomorphic operations until the ciphertext once again reaches the noise bound. At that point, bootstrapping can be applied again to refresh the ciphertext and reduce its noise. Repeating this process enables an unbounded number of homomorphic operations, achieving fully homomorphic encryption.

4.3.4 Circular Security

This approach assumes access to an encryption of the secret key under the same public key. This requires the *circular security* assumption: even when an adversary is given $(\mathsf{pk}, \mathsf{Enc}(\mathsf{pk}, \mathsf{sk}))$, the scheme remains semantically secure. Formally, we require that

$$(\mathsf{pk}, \mathsf{Enc}(\mathsf{pk}, \mathsf{sk}), \mathsf{Enc}(\mathsf{pk}, \mathsf{msg})) \approx_c (\mathsf{pk}, \mathsf{Enc}(\mathsf{pk}, \mathsf{sk}), \mathsf{Enc}(\mathsf{pk}, 1)).$$

Achieving bootstrapping, or any general method of noise reduction, from standard LWE assumptions without assuming circular security remains one of the central open problems in fully homomorphic encryption.

References

[A96] M. Ajtai. Generating hard instances of lattice problems. In *Proceedings of the 28th Annual ACM Symposium on Theory of Computing (STOC 1996)*, pages 99–108, 1996.

[RAD78] R. L. Rivest, L. Adleman, and M. L. Dertouzos. On data banks and privacy homomorphisms. In *Foundations of Secure Computation*, pages 169–180, 1978.

[G09] C. Gentry. A fully homomorphic encryption scheme. Ph.D. thesis, Stanford University, 2009.

[BV14] Z. Brakerski and V. Vaikuntanathan. Efficient fully homomorphic encryption from (standard) LWE. *SIAM Journal on Computing*, 43(2):831–871, 2014.

[GSW13] C. Gentry, A. Sahai, and B. Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In *Proceedings of CRYPTO 2013*, pages 75–92, 2013.

[GPV08] C. Gentry, C. Peikert, and V. Vaikuntanathan. How to use a short basis: Trapdoors for hard lattices and new cryptographic constructions. In *Proceedings of STOC 2008*, pages 197–206, 2008.