6.876 Advanced Topics in Cryptography: Lattices October 26, 2015

Lecture 13

Lecturer: Vinod Vaikuntanathan *Scribe: Sunoo Park*

1 Introduction

So far, we have seen:

- An average-case hard problem on lattices: Short Integer Solution (SIS)
- Worst-case to average reduction (for SIS)
- Cryptographic applications of hardness of SIS: one-way functions, collision-resistant hash function families, etc.

Today, we plan to cover:

- Another average-case hard problem: Learning with Errors (LWE)
- Public-key encryption (PKE) and fully homomorphic encryption (FHE) from LWE

Next time, we will begin with:

- Worst-case to average reduction (namely, if there is an efficient solver for LWE, then there is an efficient solver for worst-case SIVP)

1.1 Background

Cryptographic work over the past decade has built many primitives based on the hardness of the Learning with Errors (LWE) [Reg05] problem. Today, LWE is known to imply essentially everything you could want from crypto, apart from a few notable exceptions: e.g. it is not known how to construct program obfuscation, one-way permutations, or non-interactive zero knowledge based on LWE.

Features of LWE that make it advantageous for use in cryptography include:

- LWE seems to be resilient to partial leakage of secrets, as we will see.
- No quantum attacks against LWE are known (unlike the other major cryptographic hardness assumptions such as factoring or discrete logarithm).

Notation PPT stands for *probabilistic polynomial time*. For a set S, we write $s \leftarrow S$ to mean that s is sampled uniformly at random from S. $\text{negl}(\cdot)$ denotes an arbitrary negligible function. For a natural number n, we write $[n]$ to denote the set $\{1, \dots, n\}$. We write $||\cdot||$ for the ℓ_2-norm.

2 Learning with Errors

A learning with errors instance $\mathsf{LWE}_{n,q,\chi}$ is parametrized by:

- $n \in \mathbb{N}$,
- $q \in$ Primes, and
- χ, a probability distribution over $\mathbb{Z}/q\mathbb{Z}$.

χ is known as the *noise distribution* and we would like it to generate "short" elements, i.e. $||e|| \leq B$ with high probability for some bound $B << q$, when $e \leftarrow \chi$. In practice, χ is usually a discrete Gaussian over $\mathbb{Z}$.

2.1 Search LWE

Suppose we are given an oracle $\mathcal{O}_{\mathbf{s}}^n$ which outputs samples of the form $(\mathbf{a}, \langle \mathbf{a}, \mathbf{s} \rangle + e)$,

- $\mathbf{a} \leftarrow \mathbb{Z}_q^n$ is chosen freshly at random for each sample.
- $\mathbf{s} \in \mathbb{Z}_q^n$ is the "secret" (and it is the same for every sample).
- $e \leftarrow \chi$ is chosen freshly according to χ for each sample.

The *search-LWE* problem is to find the secret $\mathbf{s}$ given access to $\mathcal{O}_{\mathbf{s}}^n$. The $\mathsf{LWE}_{n,q,\chi}$ *assumption* is the assumption that the search-LWE problem is computationally hard: this is formalized in Definition 1 below.

Remark Note that without the "noise" bit e, the problem would be trivial: if we get n samples of the form $(\mathbf{a}_1, \langle \mathbf{a}_1, \mathbf{s} \rangle), \ldots, (\mathbf{a}_n, \langle \mathbf{a}_n, \mathbf{s} \rangle)$, we can solve for $\mathbf{s}$ by Gaussian elimination.

Definition 1 ($\mathsf{LWE}_{n,q,\chi}$ assumption)**.** *For any PPT algorithm $\mathcal{A}$, it holds that:*

$$\Pr_{\mathbf{s} \leftarrow \mathbb{Z}_q^n}\left[\mathcal{A}^{\mathcal{O}_{\mathbf{s}}^n}(1^n) = \mathbf{s}\right] = \mathrm{negl}(n).$$

The *search* version of the LWE problem is not very suitable for cryptography: intuitively, if we are constructing an encryption scheme, then we want the adversary not to be able to get *any* information about the encrypted message, not that he just cannot guess it exactly. For example, the $\mathsf{LWE}_{n,q,\chi}$ assumption allows for the possibility that an adversary could reliably guess the first half of the secret $\mathbf{s}$. For cryptography, the *decisional* variant of the LWE assumption described in the next subsection is preferable.

2.2 Decisional LWE

Let $\mathcal{O}_{\mathbf{s}}^n$ be the oracle described in the previous subsection, and let $\mathcal{R}$ be an oracle which outputs uniformly random samples $(\mathbf{a}, b) \leftarrow \mathbb{Z}_q^n \times \mathbb{Z}_q$. The *decisional LWE* problem is to "guess" which oracle you are interacting with, when given access to an unknown oracle which is either $\mathcal{O}_{\mathbf{s}}^n$ or R. This is formalized in Definition 2.

Definition 2 (Decisional $\mathsf{LWE}_{n,q,\chi}$ assumption)**.** *For any PPT algorithm $\mathcal{A}$, it holds that:*

$$\left|\Pr\left[\mathcal{A}^{\mathcal{O}_{\mathbf{s}}^n}(1^n) = 1\right] - \Pr\left[\mathcal{A}^{\mathcal{R}}(1^n) = 1\right]\right| = \mathrm{negl}(n).$$

It is easy to see that if the decisional $\mathsf{LWE}_{n,q,\chi}$ assumption holds, then the (search) $\mathsf{LWE}_{n,q,\chi}$ assumption holds too. Interestingly, the opposite implication also holds (although we lose a little in the parameters), as will be shown in subsection 2.7.

2.3 A variant definition with fixed number of samples

We define $\mathsf{LWE}_{n,m,q,\chi}$ with an additional parameter $m \in \mathbb{N}$ which represents the number of samples that the adversary is given. That is, the adversary no longer has oracle access to $\mathcal{O}_{\mathbf{s}}^n$ (or $\mathcal{R}$), but instead receives as input m samples from $\mathcal{O}_{\mathbf{s}}^n$ (or $\mathcal{R}$). Note that m samples from $\mathcal{O}_{\mathbf{s}}^n$ have the following form:

$$(\mathbf{a}_1, \langle \mathbf{a}_1, \mathbf{s} \rangle + e_1), \ldots, (\mathbf{a}_m, \langle \mathbf{a}_m, \mathbf{s} \rangle + e_m),$$

where for each $i \in [m]$, $\mathbf{a}_i \leftarrow \mathbb{Z}_q^n$ and $e_i \leftarrow \chi$. Thus, these samples can equivalently be expressed as:

$$(\mathbf{A}, \mathbf{s}^T\mathbf{A} + \mathbf{e}^T),$$

where $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}$ is a matrix that has columns $\mathbf{a}_1, \ldots, \mathbf{a}_m$, and $\mathbf{e} \leftarrow \chi^m$ has entries $e_1, \ldots, e_m$.

Definitions 3 and 4 formally describe the search and decisional $\mathsf{LWE}_{n,m,q,\chi}$ assumptions, respectively.

Definition 3 ($\mathsf{LWE}_{n,m,q,\chi}$ assumption)**.** *For any PPT algorithm $\mathcal{A}$, it holds that:*

$$\Pr_{\substack{\mathbf{s}\leftarrow\mathbb{Z}_q^n\\ \mathbf{A}\leftarrow\mathbb{Z}_q^{n\times m}\\ \mathbf{e}\leftarrow\chi^m}}\left[\mathcal{A}(1^n,(\mathbf{A},\mathbf{s}^T\mathbf{A}+\mathbf{e}^T))=\mathbf{s}\right]=\mathrm{negl}(n).$$

Definition 4 (Decisional $\mathsf{LWE}_{n,m,q,\chi}$ assumption)**.** *For any PPT algorithm $\mathcal{A}$, it holds that:*

$$\left|\Pr_{\substack{\mathbf{s}\leftarrow\mathbb{Z}_q^n\\ \mathbf{A}\leftarrow\mathbb{Z}_q^{n\times m}\\ \mathbf{e}\leftarrow\chi^m}}\left[\mathcal{A}(1^n,(\mathbf{A},\mathbf{s}^T\mathbf{A}+\mathbf{e}^T))=1\right]-\Pr_{\substack{\mathbf{A}\leftarrow\mathbb{Z}_q^{n\times m}\\ \mathbf{b}\leftarrow\mathbb{Z}_q^m}}\left[\mathcal{A}(1^n,(\mathbf{A},\mathbf{b}))=1\right]\right|=\mathrm{negl}(n).$$

Note that $\mathsf{LWE}_{n,m,q,\chi}$ is more restricted that $\mathsf{LWE}_{n,q,\chi}$ in that the adversary gets only a predetermined number of samples, rather than being able to oracle-query however many times he wants. We will find the $\mathsf{LWE}_{n,m,q,\chi}$ definition to be useful for the reductions we show in the rest of the lecture.

2.4 Reduction from SIS to LWE

Recall the Short Integer Solution ($\mathsf{SIS}_{n,m,q,\beta}$) problem: given $\mathbf{A}\leftarrow\mathbb{Z}^{n\times m}$, find a "short" non-zero vector $\mathbf{r}\in\mathbb{Z}^m$ such that $\mathbf{Ar}=0 \mod q$ and $||\mathbf{r}||\leq\beta$.

Claim 5. *If there is an efficient algorithm that solves $\mathsf{SIS}_{n,m,q,\beta}$, then there is an efficient algorithm that solves decisional $\mathsf{LWE}_{n,m,q,\chi}$, provided that $\beta\cdot B << q$.*

Proof. Let $\mathcal{A}_{\mathsf{SIS}}$ be an efficient solver for $\mathsf{SIS}_{n,m,q,\beta}$. We build an efficient solver $\mathcal{A}_{d\mathsf{LWE}}$ for decisional LWE as follows. On input $(\mathbf{A},\mathbf{b}^T)\in\mathbb{Z}_q^{n\times m}\times\mathbb{Z}_q^m$, $\mathcal{A}_{d\mathsf{LWE}}$ runs $\mathcal{A}_{\mathsf{SIS}}(\mathbf{A})=\mathbf{r}$ and obtains a short vector $\mathbf{r}$. Now, if $(\mathbf{A},\mathbf{b}^T)$ is an LWE sample, then

$$\mathbf{b}^T\mathbf{r}=(\mathbf{s}^T\mathbf{A}+\mathbf{e}^T)\mathbf{r}=\mathbf{e}^T\mathbf{r},$$

which is small (specifically, it is at most $\beta\cdot B$) since both $\mathbf{e}$ and $\mathbf{r}$ are short. On the other hand, if $(\mathbf{A},\mathbf{b}^T)$ is random in $\mathbb{Z}_q^{n\times m}\times\mathbb{Z}_q^m$, then $\mathbf{b}^T\mathbf{r}$ is random in $\mathbb{Z}_q$. Hence, if our solver $\mathcal{A}_{d\mathsf{LWE}}$ outputs 1 when $||\mathbf{b}^T\mathbf{r}||\leq\beta\cdot B$ and outputs 0 otherwise, it will distinguish with non-negligible advantage between the case when $(\mathbf{A},\mathbf{b}^T)$ is an LWE sample and the case when $(\mathbf{A},\mathbf{b}^T)$ is random. □

For the next claim, we invoke a *strong* SIS solver. A strong SIS solver is one which, when run many times, will output many *independent, random* short vectors $\mathbf{r}$ satisfying the requirements of the SIS problem.

Claim 6. *If there is an efficient algorithm that strongly solves $\mathsf{SIS}_{n,m,q,\beta}$, then there is an efficient algorithm that solves (search) $\mathsf{LWE}_{n,m,q,\chi}$.*

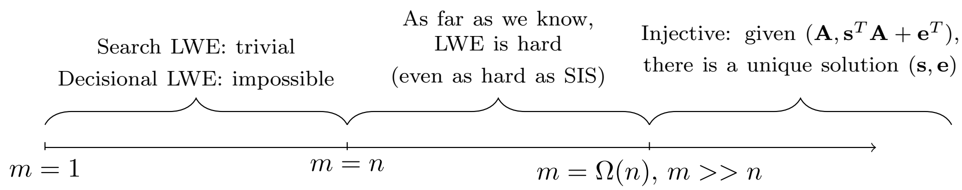
Proof. Let $\mathcal{A}^*_{\mathsf{SIS}}$ be an efficient algorithm which strongly solves $\mathsf{SIS}_{n,m,q,\beta}$. We build an efficient solver $\mathcal{A}_{\mathsf{LWE}}$ for search LWE as follows: on input $(\mathbf{A},\mathbf{b}^T)$, $\mathcal{A}_{\mathsf{LWE}}$ runs $\mathcal{A}^*_{\mathsf{SIS}}(\mathbf{A})$ m times to obtain short vectors $\mathbf{r}_1,\ldots,\mathbf{r}_m$. Note that for each $i\in[m]$, our algorithm $\mathcal{A}_{\mathsf{LWE}}$ can efficiently compute

$$\mathbf{b}^T\mathbf{r}_i=(\mathbf{s}^T\mathbf{A}+\mathbf{e}^T)\mathbf{r}_i=\mathbf{e}^T\mathbf{r}_i.$$

Since $\mathcal{A}^*_{\mathsf{SIS}}$ strongly solves $\mathsf{SIS}_{n,m,q,\beta}$, the vectors $\mathbf{r}_1,\ldots,\mathbf{r}_m$ are independent and random subject to $||\mathbf{r}_i||\leq\beta$. It follows that from the pairs $(\mathbf{r}_i,\mathbf{e}^T\mathbf{r}_i)$, it is possible for $\mathcal{A}_{\mathsf{LWE}}$ to compute $\mathbf{e}$ by Gaussian elimination. Once $\mathbf{e}$ is known, $\mathcal{A}_{\mathsf{LWE}}$ can compute $\mathbf{s}$ as $\mathbf{s}=(\mathbf{b}-\mathbf{e})^T\mathbf{A}'$ where $\mathbf{A}'$ is the right-inverse of $\mathbf{A}$. □

2.5 For which values of m is the LWE problem hard?

The search LWE problem is actually easy if m is much smaller than n. For example, if you only get one sample $(\mathbf{a}, \langle \mathbf{a}, \mathbf{s} \rangle + e)$, then there is only one constraint, and there are many solutions which satisfy this constraint: in other words, $\mathbf{s}$ is not uniquely defined.



2.6 Random self-reducibility

Claim 7. *If there is an efficient decider $\mathcal{D}_{\text{avg}}$ for average-case decisional LWE (i.e. where the secret $\mathbf{s}$ is chosen at random), then there is an efficient decider $\mathcal{D}_{\text{worst}}$ for worst-case decisional LWE (i.e. where the secret $\mathbf{s}$ may be chosen from an arbitrary distribution).*

Proof. Given an average-case decider $\mathcal{D}_{\text{avg}}$, we can build a worst-case decider $\mathcal{D}_{\text{worst}}$ as follows: on input $(\mathbf{A}, \mathbf{b}^T)$, our decider $\mathcal{D}_{\text{worst}}$ chooses a fresh random $\mathbf{s}' \leftarrow \mathbb{Z}_q^n$ and runs $\mathcal{D}_{\text{avg}}$ on input $(\mathbf{A}, \mathbf{b}^T + \mathbf{s}'^T\mathbf{A})$. Notice that if $(\mathbf{A}, \mathbf{b}^T)$ was an LWE sample, i.e. $\mathbf{b}^T = \mathbf{s}^T\mathbf{A} + \mathbf{e}^T$, then the input given to $\mathcal{D}_{\text{avg}}$ can be written as

$$(\mathbf{A}, \mathbf{b}^T + \mathbf{s}'^T\mathbf{A}) = (\mathbf{A}, (\mathbf{s}^T + \mathbf{s}'^T)\mathbf{A} + \mathbf{e}^T),$$

and this is an LWE sample with secret $\mathbf{s} + \mathbf{s}'$. Since $\mathbf{s}'$ was chosen at random, this new secret $\mathbf{s} + \mathbf{s}'$ is also distributed uniformly at random – that is, the input to $\mathcal{D}_{\text{avg}}$ is an *average-case* LWE sample. On the other hand, if $(\mathbf{A}, \mathbf{b})$ was random, then the input $(\mathbf{A}, \mathbf{b}^T + \mathbf{s}'^T\mathbf{A})$ given to $\mathcal{D}_{\text{avg}}$ is also random. Hence, the decider $\mathcal{D}_{\text{avg}}$ will succeed (with non-negligible advantage) at distinguishing between the case where its input $(\mathbf{A}, \mathbf{b}^T + \mathbf{s}'^T\mathbf{A})$ is an LWE sample and the case where it is random. Moreover, we have shown that these two cases exactly correspond to the case where the input $(\mathbf{A}, \mathbf{b}^T)$ to $\mathcal{D}_{\text{worst}}$ is an LWE sample and the case where it is random, respectively. Therefore, if $\mathcal{D}_{\text{worst}}$ runs $\mathcal{D}_{\text{avg}}$ on input $(\mathbf{A}, \mathbf{b}^T + \mathbf{s}'^T\mathbf{A})$, and outputs the value outputted by $\mathcal{D}_{\text{avg}}$, then it will be an efficient decider for worst-case decisional LWE. □

2.7 Reduction from search to decisional LWE

We now show a reduction from search to decisional LWE.

Theorem 8. *If there is an efficient solver for decisional $\mathsf{LWE}_{n,m,q,\chi}$, then there is an efficient solver for search $\mathsf{LWE}_{n,m',q,\chi}$, where $m' = O(nmq/\varepsilon^2)$.*

Proof. Let $\mathcal{D}$ be an efficient solver for decisional $\mathsf{LWE}_{n,m,q,\chi}$. Without loss of generality, assume that

$$\Pr_{\substack{\mathbf{s} \leftarrow \mathbb{Z}_q^n \\ \mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m} \\ \mathbf{e} \leftarrow \chi^m}} \left[\mathcal{A}(1^n, (\mathbf{A}, \mathbf{s}^T\mathbf{A} + \mathbf{e}^T)) = 1\right] - \Pr_{\substack{\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m} \\ \mathbf{b} \leftarrow \mathbb{Z}_q^m}} [\mathcal{A}(1^n, (\mathbf{A}, \mathbf{b})) = 1] = \varepsilon(n). \quad (1)$$

where ε is polynomial in n. Our approach to solve search $\mathsf{LWE}_{n,m',q,\chi}$ will be to "guess" the secret, one coordinate at a time. Let $s_1, \ldots, s_n \in \mathbb{Z}_q$ denote the coordinates of $\mathbf{s}$, that is, $\mathbf{s} = (s_1, \ldots, s_n)$. Consider the algorithm which, on input $(\mathbf{A}, \mathbf{s}^T\mathbf{A} + \mathbf{e}^T)$, for each $i \in [m]$, guesses the i^{th} coordinate of $\mathbf{s}$ as described in Algorithm 1 below.

Algorithm 1 "Guess" the i^{th} coordinate of $\mathbf{s}$

For $j = 0, \ldots, q-1$:

- Let $g_i := j$.
- For $\ell = 1, \ldots, L = O(1/\varepsilon)$:
  - Sample a random vector $\mathbf{c}_\ell \leftarrow \mathbb{Z}_q^m$, and let $\mathbf{C}_\ell \in \mathbb{Z}_q^{n \times m}$ be the matrix whose top row is $\mathbf{c}_\ell$, and whose other entries are all zero.
  - Let $\mathbf{A}'_\ell := \mathbf{A} + \mathbf{C}_\ell$, and $\mathbf{b}'_\ell = \mathbf{b} + g_i \cdot \mathbf{c}_\ell$.
  - Run $\mathcal{D}$ on input $(\mathbf{A}'_\ell, \mathbf{b}'_\ell)$ and let the output of $\mathcal{D}$ be called d_ℓ.
- If majority$(d_1, \ldots, d_\ell) = 1$ then output g_i. Else, continue to the next iteration of the loop.

If a guess g_i is correct, i.e. $s_i = g_i$, then the inputs $(\mathbf{A}'_\ell, \mathbf{b}'_\ell)$ given to $\mathcal{D}$ are LWE samples, since

$$\begin{aligned}
\mathbf{b}'_\ell = \mathbf{b} + s_i \cdot \mathbf{c}_\ell &= \mathbf{s}^T\mathbf{A} + \mathbf{e}^T + s_i \cdot \mathbf{c}_\ell && \text{(expanding } \mathbf{b}\text{)} \\
&= (\mathbf{s}^T\mathbf{A} + s_i \cdot \mathbf{c}_\ell) + \mathbf{e}^T && \text{(rearranging)} \\
&= \mathbf{s}^T(\mathbf{A} + \mathbf{C}_\ell) + \mathbf{e}^T && \text{(by construction of } \mathbf{C}_\ell\text{)} \\
&= \mathbf{s}^T\mathbf{A}'_\ell + \mathbf{e}^T. && \text{(by definition of } \mathbf{A}'_\ell\text{)}
\end{aligned}$$

On the other hand, if the guess g_i is wrong, i.e. $s_i \neq g_i$, then the inputs $(\mathbf{A}'_\ell, \mathbf{b}'_\ell)$ given to $\mathcal{D}$ are uniformly random, since

$$\begin{aligned}
\mathbf{b}'_\ell = \mathbf{b} + g_i \cdot \mathbf{c}_\ell &= \mathbf{s}^T\mathbf{A} + \mathbf{e}^T + g_i \cdot \mathbf{c}_\ell \\
&= (\mathbf{s}^T\mathbf{A} + g_i \cdot \mathbf{c}_\ell) + \mathbf{e}^T \\
&= \mathbf{s}^T\mathbf{A}'_\ell + (g_i - s_i) \cdot \mathbf{c}_\ell + \mathbf{e}^T,
\end{aligned}$$

and the term $(g_i - s_i) \cdot \mathbf{c}_\ell$ is random since $g_i - s_i$ is nonzero and $\mathbf{c}_\ell$ is random. It follows, by (1), that $\mathcal{D}$ will output 1 with probability at least $1/2 + \varepsilon$, in the case that $s_i = g_i$. Since we run $\mathcal{D}$ many times ($L = O(1/\varepsilon)$ times, to be precise), it follows from a Chernoff bound that with overwhelming probability: if the majority of the outputs $d_1, \ldots, d_\ell$ from $\mathcal{D}$ are equal to 1, then we are in the case where $s_i = g_i$, and if not, we are in the case where $s_i \neq g_i$. Hence, with overwhelming probability, Algorithm 1 guesses each coordinate of $\mathbf{s}$ correctly. Therefore, applying Algorithm 1 to each coordinate of $\mathbf{s}$ will, with overwhelming probability, correctly output all coordinates $s_1, \ldots, s_n$ of $\mathbf{s}$. □

3 Encryption schemes

3.1 Secret-key encryption from LWE

In this subsection, we describe a secret-key encryption scheme SKE based on LWE, due to [Reg05]. For the correctness of the encryption scheme, we will require that the noise distribution χ is such that $||e|| \leq q/4$ with high probability, for $e \leftarrow \chi$. We can choose χ to be a discrete Gaussian distribution that satisfies this constraint.

- SKE.KeyGen(1^n) takes as input the security parameter n and outputs a secret key $sk = \mathbf{s} \leftarrow \mathbb{Z}_q^n$.
- SKE.Enc$(sk = \mathbf{s}, \mu)$ takes as input a secret key $\mathbf{s}$ and a message $\mu \in \{0, 1\}$, and outputs a ciphertext

  $$(\mathbf{a}, \langle \mathbf{a}, \mathbf{s} \rangle + e + \mu \cdot \lceil q/2 \rfloor),$$

  where $\mathbf{a} \leftarrow \mathbb{Z}_q^n$ and $e \leftarrow \chi$ are sampled afresh for each ciphertext.

- $\mathsf{SKE.Dec}(sk = \mathbf{s}, (\mathbf{a}, b))$ takes as input a secret key $\mathbf{s}$ and a ciphertext $(\mathbf{a}, b)$, and outputs a decryption:

$$\mu' := \begin{cases} 0 & \text{if } ||b - \langle \mathbf{a}, \mathbf{s} \rangle|| < q/4 \\ 1 & \text{otherwise.} \end{cases}$$

We now argue the correctness and security of this encryption scheme.

Correctness If $(\mathbf{a}, b)$ is a correctly formed ciphertext, then we have

$$b - \langle \mathbf{a}, \mathbf{s} \rangle = e + \mu \cdot \lceil q/2 \rceil .$$

Then, from the definition of the decryption algorithm, it is clear that correctness holds as long as $||e|| < q/4$. This holds with high probability, due to our choice of χ.

Security By the decisional $\mathsf{LWE}_{n,q,\chi}$ assumption, a sample of the form $(\mathbf{a}, \langle \mathbf{a}, \mathbf{s} \rangle + e)$ is computationally indistinguishable from a random sample $(\mathbf{a}, b) \leftarrow \mathbb{Z}_q^n \times \mathbb{Z}_q$. The ciphertexts of SKE are simply $\mathsf{LWE}_{n,q,\chi}$ samples with $\mu \cdot \lceil q/2 \rceil$ added to the second component, so it follows that the ciphertexts are also computationally indistinguishable from random samples $(\mathbf{a}, b) \leftarrow \mathbb{Z}_q^n \times \mathbb{Z}_q$. In particular, there is no efficient algorithm that distinguishes with non-negligible advantage between the cases where $\mu = 0$ and $\mu = 1$.

3.2 Public-key encryption from LWE

Finally, we describe a public-key encryption scheme PKE based on LWE, again due to [Reg05]. We require for the correctness of the encryption scheme that the noise distribution χ is such that $||e|| \leq q/4m$ with high probability, for $e \leftarrow \chi$.

- $\mathsf{PKE.KeyGen}(1^n)$ takes as input the security parameter n, samples $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}$ and $\mathbf{e} \leftarrow \chi^m$, and outputs a key-pair (pk, sk) where $sk = \mathbf{s} \leftarrow \mathbb{Z}_q^n$ and $pk = (\mathbf{A}, \mathbf{s}^T \mathbf{A} + \mathbf{e}^T)$.
- $\mathsf{PKE.Enc}(pk = (\mathbf{A}, \mathbf{b}^T), \mu)$ takes as input a public key $(\mathbf{A}, \mathbf{b}^T)$ and a message $\mu \in \{0, 1\}$, samples a short vector $\mathbf{r} \leftarrow \{0, 1\}^m$, and outputs a ciphertext

$$(\mathbf{A}\mathbf{r}, \mathbf{b}^T \mathbf{r} + \mu \cdot \lceil q/2 \rceil).$$

- $\mathsf{PKE.Dec}(sk = \mathbf{s}, (\mathbf{u}, v))$ takes as input a secret key $\mathbf{s}$ and a ciphertext $(\mathbf{u}, v)$, and outputs a decryption:

$$\mu' := \begin{cases} 0 & \text{if } ||v - \mathbf{s}^T \mathbf{u}|| < q/4 \\ 1 & \text{otherwise.} \end{cases}$$

We now argue the correctness and security of this encryption scheme.

Correctness If $(\mathbf{u}, v)$ is a correctly formed ciphertext, then we have

$$v - \mathbf{s}^T \mathbf{u} = \mathbf{b}^T \mathbf{r} - \mu \cdot \lceil q/2 \rceil - \mathbf{s}^T \mathbf{A} \mathbf{r} = \mathbf{e}^T \mathbf{r} + \mu \cdot \lceil q/2 \rceil .$$

Note that if we have a bound B such that $||e|| \leq B$ with high probability for $e \leftarrow \chi$, then we have that $||\mathbf{e}^T \mathbf{r}|| \leq m \cdot B$ by a coordinate-wise bound. From the definition of the decryption algorithm, it is clear that correctness holds if $||\mathbf{e}^T \mathbf{r}|| < q/4$. This holds with high probability, due to our choice of χ with $B = q/4m$.

Security We want to prove that for any $k = \text{poly}(n)$,

$$(pk, \mathsf{PKE.Enc}(pk, \mu_1), \ldots, \mathsf{PKE.Enc}(pk, \mu_k)) \overset{c}{\approx} (pk, \mathsf{PKE.Enc}(pk, 0), \ldots, \mathsf{PKE.Enc}(pk, 0)), \tag{2}$$

where pk is a public key sampled by $\mathsf{PKE.KeyGen}$, and $\overset{c}{\approx}$ denotes computational indistinguishability. In fact, it is sufficient to show that[1]

$$(pk, \mathsf{PKE.Enc}(pk, 0)) \overset{c}{\approx} (pk, \mathsf{PKE.Enc}(pk, 1)). \tag{3}$$

We now show that (3) holds by considering the following hybrids. In the description of each hybrid, the part which changed from the previous hybrid is underlined in red.

Hybrid 1

- $pk = (\mathbf{A}, \mathbf{b}^T) = (\mathbf{A}, \mathbf{s}^T\mathbf{A} + \mathbf{e}^T)$ for $\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}$, $\mathbf{s} \leftarrow \mathbb{Z}_q^n$, $\mathbf{e} \leftarrow \chi^m$
- $ct = \mathsf{PKE.Enc}(pk, 0) = (\mathbf{Ar}, \mathbf{b}^T\mathbf{r})$ for random $\mathbf{r} \leftarrow \{0,1\}^m$

Hybrid 2

- $pk = (\mathbf{A}, \mathbf{b}^T)$ for $\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}$ and $\underline{\text{random } \mathbf{b} \leftarrow \mathbb{Z}_q^m}$
- $ct = \mathsf{PKE.Enc}(pk, 0) = (\mathbf{Ar}, \mathbf{b}^T\mathbf{r})$ for random $\mathbf{r} \leftarrow \{0,1\}^m$

Hybrid 3

- $pk = (\mathbf{A}, \mathbf{b}^T)$ for $\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}$ and random $\mathbf{b} \leftarrow \mathbb{Z}_q^m$
- $ct = \underline{(\mathbf{u}, v) \leftarrow \mathbb{Z}_q^n \times \mathbb{Z}_q}$

Hybrid 4

- $pk = (\mathbf{A}, \mathbf{b}^T)$ for $\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}$ and random $\mathbf{b} \leftarrow \mathbb{Z}_q^m$
- $ct = \underline{\mathsf{PKE.Enc}(pk, 1) = (\mathbf{Ar}, \mathbf{b}^T\mathbf{r} + \lceil q/2 \rceil) \text{ for random } \mathbf{r} \leftarrow \{0,1\}^m}$

Hybrid 5

- $pk = (\mathbf{A}, \mathbf{b}^T) = \underline{(\mathbf{A}, \mathbf{s}^T\mathbf{A} + \mathbf{e}^T) \text{ for } \mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}, \mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{e} \leftarrow \chi^m}$
- $ct = \mathsf{PKE.Enc}(pk, 1) = (\mathbf{Ar}, \mathbf{b}^T\mathbf{r} + \lceil q/2 \rceil)$ for random $\mathbf{r} \leftarrow \{0,1\}^m$

Hybrid 1 is computationally indistinguishable from Hybrid 2 by the decisional $\mathsf{LWE}_{n,m,q,\chi}$ assumption. Hybrids 2 and 3 are statistically indistinguishable by the Leftover Hash Lemma (see Lemma 9). Hybrids 3 and 4 are also statistically indistinguishable by Lemma 9. Finally, Hybrids 4 and 5 are computationally indistinguishable by the decisional $\mathsf{LWE}_{n,m,q,\chi}$ assumption.

Notice that Hybrid 1 corresponds exactly to the pair $(pk, \mathsf{PKE.Enc}(pk, 0))$ on the left-hand side of (3), and Hybrid 5 corresponds to the pair $(pk, \mathsf{PKE.Enc}(pk, 1))$ on the right-hand side of (3). We conclude that no PPT adversary can distinguish with non-negligible advantage between Hybrid 1 and Hybrid 5, and thus we have shown that (3) holds.

Lemma 9. *The distribution of* $(\mathbf{A}, \mathbf{Ar})$ *is statistically* ε*-close (see Definition 10) to the distribution of* $(\mathbf{A}, \mathbf{u})$ *where* $\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}$, $\mathbf{r} \leftarrow \{0,1\}^m$, *and* $\mathbf{u} \leftarrow \mathbb{Z}_q^n$.

Proof. By the Leftover Hash Lemma [ILL89], this holds as long as $m > n\log(q) + 2\log(1/\varepsilon)$. □

Definition 10. *Let* X *and* Y *be two random variables with range* U. *The statistical distance between* X *and* Y *is defined as follows:*

$$\Delta(X, Y) = \frac{1}{2}\sum_{u\in U} |\Pr[X = u] - \Pr[Y = u]|$$

. *For any* $\varepsilon > 0$, *we say* X *and* Y *are statistically* ε*-close if* $\Delta(X, Y) \leq \varepsilon$.

[1] From (3), we can prove (2) by a standard hybrid argument.

References

[ILL89] Russell Impagliazzo, Leonid A. Levin, and Michael Luby. "Pseudo-random Generation from one-way functions (Extended Abstracts)". In: *Proceedings of the 21st Annual ACM Symposium on Theory of Computing, May 14-17, 1989, Seattle, Washigton, USA*. Ed. by David S. Johnson. ACM, 1989, pp. 12–24. ISBN: 0-89791-307-8. DOI: 10.1145/73007.73009. URL: http://doi.acm.org/10.1145/73007.73009.

[Reg05] Oded Regev. "On lattices, learning with errors, random linear codes, and cryptography". In: *STOC*. Ed. by Harold N. Gabow and Ronald Fagin. ACM, 2005, pp. 84–93. ISBN: 1-58113-960-8.