Lecture 14: Lattice-Based Cryptography and the Learning with Error (LWE) Assumption

Notes by Yael Kalai

MIT - 6.5620
Lecture 14 (October 22, 2025)

> ***Warning:*** This document is a rough draft, so it may contain bugs. Please feel free to email me with corrections.

Recap

So far we saw several constructions of CPA-secure public-key encryption schemes:

- The El-Gamal scheme, based on the DDH (or CDH) assumption.
- The scheme obtained from Rabin's trapdoor permutation $f_N : \mathbb{QR}_N \to \mathbb{QR}_N$ (where N is a Blum integer)[1], based on the hardness of Factoring.
- The Goldwasser-Micali scheme, based on the Quadratic Residuosity assumption.
- The RSA scheme from the RSA trapdoor permutation, based on the RSA assumption.

[1] A Blum integer is a number of the form $N = p \cdot q$, whee p and q are primes such that $p = 3 \bmod 4$ and $q = 3 \bmod 4$.

All these assumptions (and schemes) can be broken using a quantum computer. In particular, Shor [3] presented a polynomial-time quantum algorithm for factoring large integers and computing discrete logarithms modulo a prime p.

Today: Lattice-Based Cryptography

Before we begin the topic of today's lecture, let me start with correcting a mistake I made in the other lecture.

Correction from last lecture: Computing square roots in $\mathbb{Z}_p^*$

- **Case 1:** $p = 3 \bmod 4$**.** In this case $p - 1 = 2 \cdot k$ for some odd number $k \in \mathbb{N}$.
  Given such prime p and $y \in \mathbb{QR}_p$,[2], the square-root of y in

[2] Recall that $\mathbb{QR}_p = \{x^2 : x \in \mathbb{Z}_p^*\}$.

$\mathbb{Z}_p^*$ is simply $a = y^{\frac{p+1}{4}}$. To see why this is the case note that

$$a^2 = y^{\frac{p+1}{2}} = y^{\frac{p-1}{2}} \cdot y = y$$

where the last equality follows from the fact that $y \in \mathbb{QR}_p$ and hence is of the form $x^2 \bmod p$ for some $x \in \mathbb{Z}_p^*$.

Remark. Note that in the above, we relied on the fact that we know the order of the group $\mathbb{Z}_p^*$, which is $p-1$. We also relied on the fact that $\frac{p+1}{4} \in \mathbb{N}$, which holds only if $p = 3 \bmod 4$.

- **General case: $p - 1 = 2^s \cdot k$, for odd $k \in \mathbb{N}$.**[3] In this case if $y^k = 1 \bmod p$, then finding the square-root can be done as above, by setting it to be $a = y^{\frac{k+1}{2}}$. Note that

  $$a^2 = y^k \cdot y = y.$$

  More generally, we will find z such that $y^k \cdot z^2 = 1$. Then we can set $a = y^{\frac{k+1}{2}} \cdot z$, where

  $$a^2 = y^{k+1} \cdot z^2 = y.$$

  It remains to find such z, which is a bit complicated, and is done as follows:

  1. Find any element $w \in \mathbb{Z}_p^*$ which is not a quadratic residue. This can be done by sampling a random $w \leftarrow \mathbb{Z}_p^*$ and checking if $w^{\frac{p-1}{2}} = -1$. If this is the case then w is not a quadratic residue. Otherwise, try again. Since half of the elements are not quadratic residues we expect to find one in constant time.
  2. Let $W = w^k \bmod p$.
  3. Set $a = 0$.
  4. Note that the facts that $a = 0$ and y is a quadratic residue, imply that

     $$\left(y^k \cdot W^{2a}\right)^{2^{s-1}} = \left(y^k\right)^{2^{s-1}} = 1$$

  5. Find the minimal $s' \leq s - 1$ such that

     $$\left(y^k \cdot W^{2a}\right)^{2^{s'}} = 1$$

  6. If $s' = 0$ then output $z = W^a$.
  7. Otherwise, let $a := a + 2^{s-s'-1}$ and go back to Item 5.

[3] In the case above, where $p = 3 \bmod 4$, it holds that$p - 1 = 2 \cdot k$ for an odd $k \in \mathbb{N}$.

We next argue that s' always decreases by at least one in each step, so that the algorithm terminates after at most $s \leq \log p$ steps. Namely, we claim that

$$\left(y^k \cdot W^{2\cdot\left(a+2^{s-s'-1}\right)}\right)^{2^{s'-1}} = 1$$

This follows from the following calculations:

$$\left(y^k \cdot W^{2\cdot\left(a+2^{s-s'-1}\right)}\right)^{2^{s'-1}} =$$
$$\left(y^k \cdot W^{2a}\right)^{2^{s'-1}} \cdot W^{2^{s-1}} =$$
$$(-1)\cdot(w^k)^{2^{s-1}} = (-1)\cdot(-1) = 1,$$

as needed.

Lattice-Based Cryptography

Lattice-based cryptography is extremely useful for several reasons:

- **Post-quantum security:** Lattice assumptions are believed to be post-quantum secure. This is in contrast to Factoring and Discrete-Log which are known to be broken in polynomial time using a quantum computer [3].
  Indeed, lattice-based cryptography has recently been standardized, with the intention of replacing many of the uses of factoring and discrete-log based crypto systems, with lattice-based ones.
- **Exponential hardness:** Lattice assumptions are believed to be exponentially hard. This is in contrast with Factoring and Discrete-Log over $\mathbb{Z}_p^*$, for which we have sub-exponential algorithms that run in time roughly $2^{\lambda^{1/3}}$.
- **Worst-case hardness:** We can build crypto-systems whose security can be based on a worst-case hardness assumption. This is in contrast to what we have seen so far, where all our assumptions were average-case assumptions.
- **Surprising capabilities:** As we will see next week we can build *fully homomorphic encryption* from lattices.
- **Simple and efficient!**

Learning with Error (LWE) Assumption

We know that linear equations can be efficiently solved by Gaussian elimination. Namely, given a random matrix $\mathbf{A} \leftarrow \mathbb{Z}_q^{m \times n}$ and given $\mathbf{A} \cdot \mathbf{s}$, where $\mathbf{s} \leftarrow \mathbb{Z}_q^n$ is a random vector, and where $m > n$, one can efficiently recover $\mathbf{s}$.

However, we believe that solving *noisy* linear equations is hard. Namely, we believe that there is a "noise distribution" χ over $\mathbb{Z}_q$, such that

$$(\mathbf{A}, \mathbf{A}\mathbf{s} + \mathbf{e}) \approx (\mathbf{A}, \mathbf{u})$$

where $\mathbf{e} \leftarrow \chi^m$ and $\mathbf{u} \leftarrow \mathbb{Z}_q^*$. This is precisely the LWE **assumption**!

Remark. As Vinod Vaikuntanathan says: to make solving linear equations hard we need to chop both the head (i.e., take the linear equations modulo some q) and the tail (i.e., add noise). If we only chop the head then we can solve the linear system using Gaussian elimination over $\mathbb{Z}_q$ and if we only chop the tail (by adding noise) then we can solve by linear regression. It turns out that if we chop both the head and the tail then solving the system becomes very hard!

Which noise distribution should we use? One can take χ to be the uniform distribution over $\mathbb{Z}_q$, in which case

$$(\mathbf{A}, \mathbf{A}\mathbf{s} + \mathbf{e}) \equiv (\mathbf{A}, \mathbf{u});$$

i.e., the two distributions are *identical*, where $\mathbf{A} \leftarrow \mathbb{Z}_q^{m \times n}$, $\mathbf{s} \leftarrow \mathbb{Z}_q^n$, $\mathbf{e} \leftarrow \chi^m$ and $\mathbf{u} \leftarrow \mathbb{Z}_q^m$. But this is not useful. For cryptographic applications, we need the error distribution χ to produce elements of bounded size, say in the interval $[-B, B]$ where $B << q$.

For this class, think of χ as being the uniform distribution on $[-B, B]$. Though, usually, we choose χ to be a "discrete Gaussian" distribution, since with such a distribution we know how to reduce the LWE assumption to a worst-case hardness assumption on lattices [2, 1].

Linearization Attacks

One needs to be careful with the choice of B. As we saw above, if B is unbounded, say $B = q/2$, and χ is uniform in $[-q/2, q/2]$ then we get something which is useless. However, it turns out that if we choose B to be too small then there are attacks. Specifically, if B is any constant, and we have enough equations, i.e., $m \geq n^{2B+1}$, then there are attacks. These are called linearization attacks.

For simplicity, let me illustrate the attack for the case where $B = 1$. The attack consists of two steps. First, get rid of the noise, by converting each noisy linear equation $b = \mathbf{a} \cdot \mathbf{s} + e$, where $e \in \{-1, 0, 1\}$, to

the following *noiseless polynomial* equation:

$$(b - \mathbf{a} \cdot \mathbf{s}) \cdot (b - \mathbf{a} \cdot \mathbf{s} - 1) \cdot (b - \mathbf{a} \cdot \mathbf{s} + 1) = 0.$$

Thankfully, even solving systems of equations of degree 2 is known to be NP-complete.

However, one can attempt the following *linearization attack*: Replace each monomial $s_i \cdot s_j \cdot s_k$ with a single variable $s_{i,j,k}$, and solve the system of linear equations. In general the solution may not satisfy that $s_i \cdot s_j \cdot s_k = s_{i,j,k}$, but it turns out that if m is large enough, in particular, if $m > n^{2B+1}$, then there is a unique solution which indeed satisfies that $s_i \cdot s_j \cdot s_k = s_{i,j,k}$.

Parameter setting: Typically we set the parameters as follows:

- $n = \lambda$.
- $m = \text{poly}(n)$.
- $B \approx \sqrt{n}$.
- q must be significantly larger than B. We can set $q = \text{poly}(n)$ or even set q to be sub-exponential in n; e.g., $q = 2^{n^{0.99}}$. As we will see, it is often beneficial to set $q = n^{\omega(1)}$.

Definition 1. The Decisional LWE assumption with parameters (n, m, q, χ), denoted by $\mathsf{LWE}_{n,m,q,\chi}$, where $n = n(\lambda)$ and $m = m(n)$, $\chi = \chi(n)$, and $q = q(n)$, asserts that

$$(\mathbf{A}, \mathbf{A} \cdot \mathbf{s} + \mathbf{e}) \approx (\mathbf{A}, \mathbf{u})$$

where $\mathbf{A} \leftarrow \mathbb{Z}_q^{m \times n}$, $\mathbf{s} \leftarrow \mathbb{Z}_q^n$, $\mathbf{e} \leftarrow \chi^m$, and $\mathbf{u} \leftarrow \mathbb{Z}_q^m$.
The computational LWE assumption with parameters (n, m, q, χ), as above, asserts that the following function is a one-way:

$$F(\mathbf{A}, \mathbf{s}, \mathbf{e}) = (\mathbf{A}, \mathbf{A} \cdot \mathbf{s} + \mathbf{e}). \tag{1}$$

Namely, the computational LWE assumption asserts that the function in Equation (1) is one-way, while the decisional LWE assumption asserts that this function is a PRG.

Remark. It turns out that that one can convert an efficient distinguisher for the decisional LWE assumption into an efficient solver for the computational LWE assumption (for $q = \text{poly}(\lambda)$). We therefore often do not distinguish between the two, and simply refer to them as the LWE assumption.

The fact that F it is a PRG implies that we can use F to construct a secret-key CPA-secure encryption, by using F to construct a PRF F' and using F' to encrypt messages by $\mathsf{Enc}(k, m) = (r, F'(k, r) \oplus m)$.

We will present a much simpler construction due to Regev [2], from his paper where he introduced the LWE assumption. Moreover, in that same paper he constructs a public-key encryption scheme under the LWE assumption! We will see both constructions starting from the secret-key one.

Secret-Key Encryption Scheme from LWE

We next present Regev's secret-key encryption scheme for message space $\mathcal{M} = \{0,1\}$. In what follows let $n = \lambda$, let q be much larger than n (it can be polynomial in n or can be super-polynomial in n), χ error distribution that produces elements $x \in \mathbb{Z}_q$ of size $|x| \leq B$, and).

- The secret key is $\mathbf{s} \leftarrow \mathbb{Z}_q^n$.
- $\mathsf{Enc}(\mathbf{s}, m)$ does the following:
  1. Sample $\mathbf{a} \leftarrow \mathbb{Z}_q^n$ and $e \leftarrow \chi$.
  2. Output $\mathsf{ct} = (\mathbf{a}, \mathbf{a} \cdot \mathbf{s} + e + \lfloor q/2 \rfloor \cdot m)$.
- $\mathsf{Dec}(\mathbf{s}, \mathsf{ct})$ does the following:
  1. Parse $\mathsf{ct} = (\mathbf{a}, c)$.
  2. Output $m = 0$ if and only if $|c - \mathbf{a} \cdot \mathbf{s}| < q/4$, and output 1 otherwise.

Correctness: Note that

$$|c - \mathbf{a} \cdot \mathbf{s}| = |e + \lfloor q/2 \rfloor \cdot m|$$

which is less than B if and only if $m = 0$, assuming $B < q/4$.

CPA-*security:* Follows immediately from the LWE assumption.

Additive homomorphism: Note that this scheme is additively homomorphic! In particular, $\mathsf{Enc}(\mathbf{s}, b_1) + \mathsf{Enc}(\mathbf{s}, b_2)$ is an encryption of $b_1 \oplus b_2$ with a larger noise, where the noise is $e_1 + e_2$. In particular, if we set q to be super-polynomial in n then we can tolerate polynomially many additions.

Regev's Public-Key Encryption Scheme

The idea is simply beautiful and seems crazy at first! The secret key will remain $\mathbf{s}$, and the public-key will contain many encryptions of zero. Namely,

$$\mathsf{pk} = (\mathbf{A}, \mathbf{A} \cdot \mathbf{s} + \mathbf{e})$$

To encrypt a message $m \in \{0,1\}$ add a random subset of these ciphertext to get a "fresh" encryption of 0 and then add to it $\lfloor q/2 \rfloor \cdot m$. Namely, $\mathsf{Enc}(\mathsf{pk}, m)$ does the following:

1. Sample $\mathbf{r} \leftarrow \{0,1\}^m$.
2. Output $(\mathbf{r} \cdot \mathbf{A}, \mathbf{r} \cdot (\mathbf{A} \cdot \mathbf{s} + \mathbf{e}) + \lfloor q/2 \rfloor \cdot m)$.

The decryption is done exactly as in the secret-key setting.

Security follows from the Leftover Hash Lemma (and the LWE assumption). Specifically, the Leftover Hash Lemma asserts that

$$(\mathbf{A}, \mathbf{r} \cdot \mathbf{A}) \equiv (\mathbf{A}, \mathbf{u})$$

assuming that $m >> n \cdot \log q$. More formally:

Lemma 2 (Leftover Hash Lemma (special case):)**.** *Fix any $\epsilon \in (0,1)$, then for $\mathbf{A} \leftarrow \mathbb{Z}_q^{m \times n}$ and $\mathbf{r} \leftarrow \{0,1\}^m$ and $\mathbf{u} \leftarrow \{0,1\}^n$, it holds that*

$$(\mathbf{A}, \mathbf{r} \cdot \mathbf{A}) \stackrel{\epsilon}{\equiv} (\mathbf{A}, \mathbf{u})$$

if $m > n \log q + 2 \cdot \log(1/\epsilon)$.

Claim 1. The public-key encryption scheme described above is CPA-secure, assuming $m \geq n \log q + \lambda$

Proof. Suppose for the sake of contradiction that there exists a poly-size $\mathcal{A}$ that wins in the CPA-security game with a non-negligible advantage. By the LWE assumption, $\mathcal{A}$ should still win with non-negligible advantage even if we replace

$$\mathsf{pk} = (\mathbf{A}, (\mathbf{A} \cdot \mathbf{s} + \mathbf{e})) \in \mathbb{Z}_q^{m \times (n+1)}$$

with $\mathsf{pk} = \mathbf{B}$, where $\mathbf{B} \leftarrow \mathbb{Z}_q^{m \times (n+1)}$. By the Leftover Hash Lemma $\mathbf{r} \cdot \mathbf{B} \equiv \mathbf{u}$, and hence the ciphertext $\mathsf{Enc}(\mathsf{pk}, m)$ is statistically close to uniform in $\mathbb{Z}_q^{n+1}$ regardless of m, and hence even an all powerful adversary cannot win the CPA-security game with non-negligible advantage. □

References

[1] Chris Peikert. Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In Michael Mitzenmacher, editor, *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31 - June 2, 2009*, pages 333–342. ACM, 2009.

[2] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Harold N. Gabow and Ronald Fagin,

editors, *Proceedings of the 37th Annual ACM Symposium on Theory of Computing, Baltimore, MD, USA, May 22-24, 2005*, pages 84–93. ACM, 2005.

[3] Peter W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In *Proceedings of the 35th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 124–134. IEEE, 1994.