Lattices and Homomorphic Encryption, Spring 2013	Instructors: Shai Halevi, Tal Malkin

Learning with Errors (LWE)

February 26, 2013 — Scribe: Clément Canonne

1 Learning with Errors (LWE) [Reg05]

Parameters and Setting. We have three parameters:
- n (security parameter)
- $\alpha = \frac{1}{\mathrm{poly}(n)}$ (noise parameter)
- $q = \Omega(\mathrm{poly}(n))$, sometimes exponential in n (modulus)

For a fixed $s \in \mathbb{Z}_q^n$, define the distribution

$$\mathrm{LWE}_s \stackrel{\text{def}}{=} \left\{ (a,b) \in \mathbb{Z}_q^n \times \mathbb{Z}_q \;\middle|\; a \sim \mathcal{U}_{\mathbb{Z}_q^n},\ \rho \sim \Phi_{\alpha q},\ b \stackrel{\text{def}}{=} \langle s, a\rangle + \rho \mod q \right\} \tag{1}$$

where $\Phi_{\alpha q}$ is a distribution with "good" properties (for instance a continuous[1] gaussian $\mathcal{N}(0, \alpha q)$).

1.1 Computational problems

Definition 1 (Search problem)**.** *In SearchLWE[n, α, q], the goal is, given oracle access to* LWE_s *for some fixed $s \sim \mathcal{U}_{\mathbb{Z}_q^n}$, to find and output s.*

Definition 2 (Decision problem)**.** *In DecisionLWE[n, α, q], given oracle access to some oracle $\mathcal{O}$ along with the promise that it either outputs samples (a) from* LWE_s *(for some fixed $s \sim \mathcal{U}_{\mathbb{Z}_q^n}$) or (b) drawn uniformly at random in $\mathbb{Z}_q^n \times \mathbb{Z}_q$, the goal is to decide which one of these two cases hold.*

A distinguisher D for LWE_s is said to have *advantage* ε if $|\mathbb{P}_{\mathrm{LWE}_s}\{ D = 1 \} - \mathbb{P}_{\mathcal{U}}\{ D = 1 \}| = \varepsilon$.

Theorem 1. *Given a distinguisher D for DecisionLWE[n, α, q] with advantage ε, one can obtain a D' that, for* every *s distinguishes* LWE_s *from uniform with advantage $1 - e^{-n}$ and runs in time* $\mathrm{poly}(n, 1/\varepsilon)$.

Proof. For any fixed $r \in \mathbb{Z}_q^n$, consider the mapping $\psi_r \colon (a,b) \in \mathbb{Z}_q^n \times \mathbb{Z}_q \mapsto (a, b + \langle a, r\rangle) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$. It is easy to check that if $(a,b) \sim \mathrm{LWE}_s$, then $\psi_r(a,b) \sim \mathrm{LWE}_{s+r}$; while if $(a,b) \sim \mathcal{U}$, then so does $\psi_r(a,b)$.

Reduction (distinguisher D')

1. Use sampling to find a threshold τ such that $\mathbb{P}_{\mathrm{LWE}_s}\{ D = 1 \} \geq \tau + \frac{\varepsilon}{4}$ and $\mathbb{P}_{\mathcal{U}}\{ D = 1 \} \leq \tau - \frac{\varepsilon}{4}$.
2. Repeat $N = \mathrm{poly}(n, 1/\varepsilon)$ times:
   (a) draw $r \sim \mathcal{U}_{\mathbb{Z}_q^n}$;
   (b) run D, answering each query by drawing (a,b) from the oracle and giving $\psi_r(a,b)$ to D;
   (c) record the final decision of D as a vote $v_i \in \{0,1\}$.
3. return 1 if $\frac{1}{N}\sum_{i=1}^N v_i > \tau$, and 0 otherwise.

[1]In which case the second component b belongs to $\mathbb{R}_q = \mathbb{R}/\mathbb{Z}_q = [0, q)$ instead of $\mathbb{Z}_q$, and the modulo is defined similarly as in the discrete case. In general, all the results below still hold for $b \in \mathbb{R}_q$.

Analysis We deal here with the case where the oracle answers according to LWE_s for an arbitrary s; the uniform distribution case is similar.
Since $\forall i \in [N], \mathbb{P}\{ v_i = 1 \} \geq \tau + \frac{\varepsilon}{4}$, an (additive) Chernoff bound yields that $\mathbb{P}\left\{ \frac{1}{N} \sum_{i=1}^{N} v_i \leq \tau \right\} \leq e^{-n}$, as long as $N = \Omega\left(\frac{n}{\varepsilon^2}\right)$. $\square$

Theorem 2. *Given a distinguisher D for DecisionLWE[n, α, q] with advantage $1 - \mathrm{negl}(n)/q$, one can construct a solver S for SearchLWE[n, α, q] that succeeds w.p. $1 - \mathrm{negl}(n)$ and runs in time $q \cdot \mathrm{poly}(n)$.*

Proof. For $i \in [n]$ and $\kappa, \gamma \in \mathbb{Z}_q$, consider the transformation

$$\varphi_{i,\kappa,\gamma} \colon (a, b) \in \mathbb{Z}_q^n \times \mathbb{Z}_q \mapsto (\underbrace{a + \gamma e_i}_{a'}, \underbrace{b + \gamma\kappa}_{b'}) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$$

where $e_i \stackrel{\text{def}}{=} (0, \dots, 0, 1, 0, \dots, 0)$.

- if $b = \sum_{j=1}^{n} s_j a_j + \rho$ and $s_i = \kappa$, then $b' = \sum_{j=1}^{n} s_j a_j + \gamma\kappa + \rho = \sum_{j=1}^{n} s_j a'_j + \rho$
- if $b = \sum_{j=1}^{n} s_j a_j + \rho$ and $s_i = \kappa' \neq \kappa$, then $b' = \sum_{j=1}^{n} s_j a'_j + \rho + \underbrace{\gamma(\kappa - \kappa')}_{\text{u.a.r. if } \gamma \sim \mathcal{U}}$

so, for any fixed i and κ, choosing γ u.a.r. changes the distribution of (a, b) to $\varphi_{i,\kappa,\gamma}(a, b)$ according to:

$$\mathrm{LWE}_s \underset{s_i = \kappa}{\longmapsto} \mathrm{LWE}_s$$
$$\mathrm{LWE}_s \underset{s_i \neq \kappa}{\longmapsto} \mathcal{U}$$

The idea is then to try for each possible values of i, κ, repeating for each couple $\mathrm{poly}(n)$ times the following: draw γ u.a.r. each time, and call D to detect if the current simulated oracle is uniform or not. If not, then the i^{th} component of s has been found – it is κ. $\square$

Remark 1. Theorem 2 has been extended to other classes of moduli ([Pei09]): if $q = \prod_{j=1}^{\ell} q_j$ where each q_j is $\mathrm{poly}(n)$, and all are distinct primes, the resulting solver can run in time $\mathrm{poly}(n, q_1 + \dots + q_\ell)$. Instead of running in time proportional to q (which may be exponential), the algorithm will run in time proportional to $\sum q_i$ (which is much smaller, maybe even polynomial).

Theorem 3. *DecisionLWE[n, α, q] remains hard even when s is drawn from the* error *distribution, that is if $s \sim \lceil \Phi_{\alpha q} \rfloor \mod q$.*

Proof. We show that a distinguisher D for the error distribution can be turned into a distinguisher D' for uniform.

Description of D'

1. choose n samples $(a_i, b_i)_{i\in[n]}$ according to LWE_s (recall that $s \sim \mathcal{U}_{\mathbb{Z}_q^n}$), and consider the matrix $A \stackrel{\text{def}}{=} \big(a_1|\dots|a_n\big)$ (assume that A is invertible)

2. Set $b \stackrel{\text{def}}{=} (b_1, \dots, b_n)$ (so that we have $b = A^{\mathrm{T}}s + x$ for some $x \sim \lceil\Phi_{\alpha q}\rfloor$), and define the mapping
$$f_{A,b}\colon (\alpha, \beta) \in \mathbb{Z}_q^n \times \mathbb{Z}_q \mapsto \Big(\underbrace{-\left(A^{-1}\right)^{\mathrm{T}}\alpha}_{\alpha'},\ \underbrace{\beta - \Big\langle \left(A^{-1}\right)^{\mathrm{T}}\alpha, b\Big\rangle}_{\beta'}\Big)$$

3. Run D to distinguish LWE_x from uniform, answering the queries by sampling (α, β) from the oracle and providing D with $f_{A,b}(\alpha, \beta)$.

Analysis

- if $(\alpha, \beta) \sim \mathcal{U}_{\mathbb{Z}_q^n\times\mathbb{Z}_q}$, then so is $f_{A,b}(\alpha, \beta)$ for *every* A (full-rank);
- if $(\alpha, \beta) \sim \mathrm{LWE}_s$, it holds that
$$\begin{aligned}\beta' &= \beta - \Big\langle \left(A^{-1}\right)^{\mathrm{T}}\alpha, b\Big\rangle = (\langle\alpha, s\rangle + \rho) - \left\langle -\alpha', A^{\mathrm{T}}s + x\right\rangle \\ &= \langle\alpha, s\rangle + \rho - \Big\langle \left(A^{-1}\right)^{\mathrm{T}}\alpha, A^{\mathrm{T}}s\Big\rangle + \left\langle\alpha', x\right\rangle \\ &= \cancel{\langle\alpha, s\rangle} + \rho - \cancel{\langle\alpha, s\rangle} + \left\langle\alpha', x\right\rangle \\ &= \left\langle\alpha', x\right\rangle + \rho\end{aligned}$$

  with $\rho \sim \lceil\Phi_{\alpha q}\rfloor$; and therefore $(\alpha', \beta') \sim \mathrm{LWE}_x$.

□

2 Application: Secret-Key encryption scheme

Recall that a *public-key encryption scheme* is a tuple of (possibly randomized) algorithms ($\mathsf{Keygen}, \mathsf{Enc}, \mathsf{Dec}$) working as below – n being a security parameter given as input to the generation algorithm:

$$s_k \leftarrow \mathsf{Keygen}_n, \quad c \leftarrow \mathsf{Enc}(m, s_k), \quad m \leftarrow \mathsf{Dec}(c, s_k)$$

where $s_k \in \mathcal{K}$ (key space), $m \in \mathcal{M}$ (message space), $c \in \mathcal{C}$ (cyphertext space), and such that

$$\forall s_k \in \mathcal{K}, m \in \mathcal{M}, c \in \mathcal{C}, \qquad \mathbb{P}(\,\mathsf{Dec}(c, s_k) = m \mid \mathsf{Enc}(m, s_k) = c\,) = 1 \qquad \text{(Correctness guarantee)}$$

Security against Chosen-Plaintext Attacks (CPA) This is a "game" between and attacker $\mathcal{A}$ and a challenger $\mathcal{B}$, where, for an arbitrary fixed n,

1. A (secret) key s_k is generated by $\mathcal{B}$, running Keygen_n;

2. $\mathcal{A}$ is given 1^n as input, and oracle access to $\mathsf{Enc}(\cdot, s_k)$, and must output a pair of messages m_0, m_1 of same length;

3. $\mathcal{B}$ chooses a random bit $\sigma \sim \mathcal{U}_{\{0,1\}}$ and computes the *challenge cyphertext* $c \leftarrow \mathsf{Enc}(m_\sigma, s_k)$;

4. $\mathcal{A}$ is then given c, and continues to have oracle access to $\mathsf{Enc}(\cdot, s_k)$; it must output a guess $\sigma' \in \{0,1\}$;

5. the output of the game is 1 is $\mathcal{A}$ wins (i.e., if $\sigma = \sigma'$), 0 otherwise.

The scheme is *CPA-secure* if for any feasible attacker $\mathcal{A}$, $\mathbb{P}\{ \mathcal{A} \text{ wins} \} \leq \frac{1}{2} + \text{negl}(n)$.

"Regev-like" cryptosystem We now describe a secret-key encryption scheme based on the LWE hardness assumption; hereafter, n, α, q are fixed as in the LWE setting.

Definition 3. *Let $\mathcal{M} = \{0,1\}$ (messages are bits), and for key $s \in \mathcal{K} = \mathbb{Z}_q^n$, define the encryption algorithm*[2] Enc_s *as follows: on input $\sigma \in \{0,1\}$,*

- *choose $a \sim \mathcal{U}_{\mathbb{Z}_q^n}$ and $\rho \sim \Phi_{\alpha q}$*
- *output (a, b), where $b \stackrel{\text{def}}{=} \underbrace{\langle a, s \rangle + \rho}_{(*)} + \left\lceil \frac{q}{2} \right\rfloor \sigma$*

Remark 2. information theoretically, getting encryptions of 0 is sufficient to determine s. However, with the LWE assumption, distinguishing between $(*)$ and a uniform random bit is hard.

Theorem 4. *If an attacker $\mathcal{A}$ has advantage ε in guessing σ, it can be transformed into a* $\mathsf{DecisionLWE}[n, \alpha, q]$ *distinguisher D with advantage $\varepsilon/2$.*

Proof. D will draw many samples (a_i, b_i) from the oracle and use them to provide $\mathcal{A}$ with "encryptions of 0" and "encryptions of 1". Then, it chooses a random bit σ and another sample (a, b), and provides $\mathcal{A}$ with the cyphertext $(a, b' \stackrel{\text{def}}{=} b + \left\lceil \frac{q}{2} \right\rfloor \sigma)$. $\mathcal{A}$ then guesses σ', and D outputs "uniform" if $\sigma \neq \sigma'$, "LWE" otherwise.

Analysis we know that $\mathbb{P}_{\mathcal{A}} \{ \sigma = \sigma' \} \geq \frac{1}{2} + \varepsilon$, so when D has a LWE oracle it will output "LWE" w.p. at least $\frac{1}{2} + \varepsilon$.

When D has a uniform oracle, then the attacker receives a cyphertext $(a, b + \left\lceil \frac{q}{2} \right\rfloor \sigma)$ which is distributed u.a.r, regardless of σ – so $\mathbb{P}_{\mathcal{A}} \{ \sigma = \sigma' \} \leq \frac{1}{2}$. □

Remark 3 (Decryption). The scheme is actually slightly modified (without affecting the previous proof) – namely, the key will be $(n+1)$ bits long:

$$s_k \stackrel{\text{def}}{=} (s||1)$$
$$c \stackrel{\text{def}}{=} (a||-b) \qquad \text{(instead of } (a,b))$$

Given this small modification, the decryption works by computing $-\langle s_k, c \rangle = \left\lceil \frac{q}{2} \right\rfloor \sigma + \rho$, and outputting 1 if this quantity is closer to $\frac{q}{2}$ than to 0, and 0 otherwise. This succeeds w.h.p (over the draw of ρ in the encryption).

Remark 4 (Additive homomorphism). Note that if c_1 encrypts σ_1 and c_2 encrypts σ_2, then $c_1 + c_2$ mod q decrypts to $\sigma_1 \oplus \sigma_2$ (as long as the errors ρ_1, ρ_2 were not too large). Thus, albeit $c_1 + c_2$ might not be a valid cyphertext (not exactly distributed according to the output of Enc_s, as the errors are also summed), we do get what is called *additive homomorphism* "for free".

[2]The decryption algorithm will be described shortly after.

References

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*, STOC '05, pages 84–93, New York, NY, USA, 2005. ACM.

[Pei09] Chris Peikert. Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In *Proceedings of the 41st annual ACM symposium on Theory of computing*, STOC '09, pages 333–342, New York, NY, USA, 2009. ACM.