FIT5124 Advanced Topics in Security

Lecture 3: Lattice-Based Crypto. III

Ron Steinfeld
Clayton School of IT
Monash University

March 2016

Plan for this lecture

- **How to construct lattice-based encryption schemes?**
  - Learning with Errors (LWE) Problem
  - Symmetric-key encryption from LWE
  - Public-key encryption from LWE: Regev's cryptosystem (2005).

Learning with Errors (LWE) Problem

Small Integer Solution (SIS) problem useful for hash functions and digital signatures, but seems not sufficient for encryption

- Many to one function — not invertible!

Q: What lattice-based problem can we use for encryption?

- **A:** Learning with Errors (LWE) Problem (Regev, 2005) – one-to-one and invertible!
- Idea: add some 'small' noise to a lattice point.

Learning with Errors (LWE) Problem - Search Variant

LWE – Setup:

- Fix integer q, and integers m, n.
- Let

$$A = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m,1} & a_{m,2} & \cdots & a_{m,n} \end{bmatrix}$$

  be an $m \times n$ matrix with entries independent and uniformly random in $\mathbb{Z}_q$ (as in SIS).
- Let $\vec{s}^T = [s_1 s_2 \cdots s_n]$ be a vector of independent uniformly random elements of $\mathbb{Z}_q$. (the "secret").
- Let $\vec{e}^T = [e_1 e_2 \cdots e_n \cdots e_m]$ be a vector of independent 'small' integers, each sampled from a probablity distribution $\chi_{\alpha q}$ (the "error").

What does 'small' e_i mean?

- $|e_i| \leq \alpha \cdot q$ with high probability, for some parameter $\alpha < 1$.
- Typically, $\chi_{\alpha q}$ = Normal (Gaussian) distribution with standard deviation $\approx \alpha \cdot q$, rounded to $\mathbb{Z}$.

Learning with Errors (LWE) Problem - Search Variant

LWE – Setup (cont.)

- Let

$$\vec{y} = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m,1} & a_{m,2} & \cdots & a_{m,n} \end{bmatrix} \cdot \begin{bmatrix} s_1 \\ s_2 \\ \vdots \\ s_n \end{bmatrix} + \begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_n \\ \vdots \\ e_m \end{bmatrix} \bmod q$$

Problem

Search Learning with Errors (Search-LWE) Problem –
Search $- LWE_{q,m,n,\alpha}$: *Given* q, m, n, α, *a matrix* $A \hookleftarrow U(\mathbb{Z}_q^{m\times n})$ *and* $\vec{y} = A \cdot \vec{s} + \vec{e} \bmod q$ *(with* $\vec{e} \hookleftarrow \chi_{\alpha q}^m$ *and* $\vec{s} \hookleftarrow U(\mathbb{Z}_q^n)$*), find* $\vec{s}$.

Learning with Errors (LWE) Problem - Decision Variant

To construct efficient cryptosystems, search variant is not sufficient. Need a decision variant of LWE.

Problem

Decision Learning with Errors (Decision-LWE) Problem –
Decision $-$ $LWE_{q,m,n,\alpha}$*: Given* $q, m, n, \alpha, A \hookleftarrow U(\mathbb{Z}_q^{m \times n}), \vec{y}$*, distinguish between the following two scenarios:*

- *'Real' Scenario:* $\vec{y} = A \cdot \vec{s} + \vec{e} \bmod q$ *(with* $\vec{e} \hookleftarrow \chi_{\alpha q}^m$ *and* $\vec{s} \hookleftarrow U(\mathbb{Z}_q^n)$*) (exactly as in search LWE).*
- *'Random' Scenario:* $\vec{y} \hookleftarrow U(\mathbb{Z}_q^m)$*.*

Q: What 2^λ security level mean?
Possible Ans: No Decision-LWE algorithm D exists that runs in time $T(D) \leq 2^\lambda$ and has distinguishing advantage $\mathrm{Adv}(D) \geq 2^{-\lambda}$, where:

- $\mathrm{Adv}(D) \stackrel{\mathrm{def}}{=} \left|\Pr_{\vec{y} \hookleftarrow \mathrm{Real}}[D(A, \vec{y}) = \mathrm{Real}] - \Pr_{\vec{y} \hookleftarrow \mathrm{Random}}[D(A, \vec{y}) = \mathrm{Real}]\right|$.

Symmetric-Key Encryption from LWE

As a first step, we construct symmetric-key encryption from LWE.

Definition

LWE-based Symmetric-Key Encryption:

- Key Generation – KG: Fix integers q, n. Pick secret key $\vec{s} \hookleftarrow U(\mathbb{Z}_q^n)$.
- Encryption – Enc: Fix integers t, ℓ. Given message $\vec{m} \in \mathbb{Z}_t^\ell$,
  - Pick $A \hookleftarrow U(\mathbb{Z}_q^{\ell \times n})$ and 'small' noise $\vec{e} \hookleftarrow \chi_{\alpha \cdot q}^\ell$.
  - Compute $\vec{c} = A \cdot \vec{s} + \vec{e} + \lceil q/t \rfloor \cdot \vec{m} \bmod q$.
  - Return ciphertext $(A, \vec{c})$.
- Decryption – Dec: Given ciphertext $(A, \vec{c})$ and secret key $\vec{s}$,
  - Compute $\vec{c'} = \vec{c} - A \cdot \vec{s} \bmod q$.
  - Compute $\vec{c''}$ by rounding coordinates of $\vec{c'}$ to the nearest multiple of $\lceil q/t \rfloor \bmod q$.
  - Return plaintext $\vec{m} = \frac{\vec{c''}}{\lceil q/t \rfloor}$.

Symmetric-Key Encryption from LWE: Correctness

Decryption recovers $\vec{c'} = \vec{e} + \lceil q/t \rfloor \cdot \vec{m} \bmod q$. Rounding succeeds to recover the ith coordinate m_i of $\vec{m}$ if the ith noise coordinate e_i is sufficiently small:

$$e_i < \frac{1}{2} \cdot \lceil q/t \rfloor \approx \frac{q}{2t}.$$

If noise distribution $\chi_{\alpha q}$ is (rounded) normal distribution with std. dev. αq, error probability per coordinate p_e is $\approx$ probability that a standard normal distributed random variable (mean 0, std. dev 1) exceeds $\frac{1}{2t\alpha}$ in magnitude:

$$p_e \approx 2 \cdot \left(1 - \Phi\left(\frac{1}{2t\alpha}\right)\right),$$

where Φ is the cumulative distribution function of normal distribution.

So: p_e 'small' when the following correctness condition holds:

$$t << \frac{1}{2\alpha}.$$

Symmetric-Key Encryption from LWE: Security

Q: Why is it secure, assuming that Decision-LWE is hard?
A: Security Reduction from Decision-LWE

- Show how to build an efficient Dec-LWE algorithm D, given an efficient attack algorithm B breaking encryption scheme.

Q: What do we mean by 'B breaks the encryption scheme'?

Possible A: B breaks standard definition of Indistinguishability security against Chosen Plaintext Attack (IND-CPA) IND-CPA Attack model: A 'game' between a challenger and the attacker B against the encryption scheme:

- Challenger runs Key Gen. algorithm of encryption scheme, obtains a secret key $\vec{s}$.
- Attacker B is given access to an 'encryption oracle': B can submit a query chosen plaintext $\vec{m}$ and receive ciphertext $(A, C) = \mathrm{Enc}(\vec{s}, \vec{m})$. After several queries, B outputs a pair of 'challenge messages' $\vec{m}_0^*$, $\vec{m}_1^*$.
- Challenger picks a random bit $b \hookleftarrow U(\{0,1\})$, computes 'challenge ciphertext' $(A^*, C^*) = \mathrm{Enc}(\vec{s}, \vec{m}_b^*)$ for the challenge message selected by b, and gives (A^*, C^*) to B.
- Attacker B continues running with query access to the 'encryption oracle'.
- Attacker B outputs a guess b' for the bit b chosen by the challenger. Attacker 'wins' game if $b' = b$.

Definition

IND-CPA security (at 2^λ security level): Any attack algorithm B with run-time $T(\mathrm{B}) \le 2^\lambda$ wins game with prob. $\le 1/2 + 1/2^\lambda$.

Symmetric-Key Encryption from LWE: Security

Security Reduction from hardness of Decision-LWE

Suppose there was an efficient IND-CPA attack B, breaking 2^λ security of the LWE encryption scheme:

- B runs in time T_B and wins IND-CPA game with probability $1/2 + \varepsilon_B$ (with $T_B < 2^\lambda$ and $\varepsilon_B > 1/2^\lambda$).
- B makes Q encryption queries overall (including the challenge ciphertext).

Then, given a $Decision - LWE_{q,m=Q\cdot\ell,n,\alpha}$ instance $(q, n, A, \vec{y})$, we build a Dec-LWE algorithm D that runs as follows:

- D runs attacker B. When B makes its ith encryption oracle query $\vec{m}_i$, D uses the ith block $A_i \in \mathbb{Z}_q^{\ell \times n}$ of ℓ consecutive rows of A and corresponding ith block $\vec{y}_i \in \mathbb{Z}_q^{\ell}$ of ℓ consecutive rows of $\vec{y}$ to answer the oracle query with $(A_i, \vec{c}_i)$ where:
$$\vec{c}_i = \vec{y}_i + \lceil q/t \rfloor \cdot \vec{m}_i \bmod q.$$
- Similarly, when B makes its challenge query $(\vec{m}_0^*, \vec{m}_1^*)$, D chooses a random bit b and uses the next (not yet used) blocks $A_{i^*}, \vec{y}_{i^*}$ of A and $\vec{y}$ to respond with $(A^* = A_{i^*}, \vec{c}^* = \vec{y}_{i^*} + \lceil q/t \rfloor \cdot \vec{m}_b^* \bmod q)$.
- Rest of encryption oracle queries of B answered as above.
- When B returns a guess b' for b, D returns 'Real' if $b' = b$, and 'Rand' if $b' \neq b$.

Symmetric-Key Encryption from LWE: Security

Q: Why does D work? Consider two LWE scenarios for $\vec{y}$:

- 'Real' LWE scenario, $\vec{y} = A \cdot \vec{s} + \vec{e}$ – all ciphertexts returned by D to B are computed exactly as in the real IND-CPA game, so B wins game with good probability $1/2 + \varepsilon_B$, hence D returns 'Real' with prob. $1/2 + \varepsilon_B$.
- 'Random' LWE scenario, $\vec{y}$ is independent and uniformly random in $\mathbb{Z}_q^{\ell \cdot Q}$ – in challenge ciphertext, $\vec{c}_i$ is uniformly random in $\mathbb{Z}_q^{\ell}$, independent of bit b — B gets no information on b, and wins the game with probability $1/2$. Hence D returns 'Real' with prob. $1/2$.

So: Distinguishing advantage of D $= \varepsilon_B > 1/2^{\lambda}$.
Also, run-time of D is (approx.) run-time of B, i.e. $< 2^{\lambda}$.
Conclusion: Contradiction with 2^{λ} security of Decision-LWE!

Theorem

IND-CPA security of LWE encryption (Q encryption queries) is at least as hard as $Decision - LWE_{q,m=Q\cdot\ell,n,\alpha}$.

Public-Key Encryption from LWE

Now we convert from symmetric-key to public-key encryption - Regev's cryptosystem (2005).

Ideas (take $\ell = 1$):

- Observation: $\mathsf{Enc}(\vec{s}, m) = \mathsf{Enc}(\vec{s}, 0) + [\vec{0}^T, m] \bmod q$.
  - Recall: $[\vec{a}^T, \vec{a}^T \cdot \vec{s} + e + m] = [\vec{a}^T, \vec{a}^T \cdot \vec{s} + \vec{e}] + [\vec{0}^T, m]$.
- **Attempt 1:** Publish $\vec{p} = \mathsf{Enc}(\vec{s}, 0)$ in public key, add $[\vec{0}^T, m]$ during encryption.
  - But... is it secure???
- **Attempt 2:** Publish several $\vec{p}_i = \mathsf{Enc}(\vec{s}, 0)$'s in public key. Combine them linearly with random coefficients r_i during encryption to a 'fresh' $c = \mathsf{Enc}(\vec{s}, 0)$!
  - Observation: For small r_i's, $r_1 \cdot \mathsf{Enc}(\vec{s}, 0) + r_2 \cdot \mathsf{Enc}(\vec{s}, 0) = \mathsf{Enc}(\vec{s}, 0)$
    - $r_1 \cdot [\vec{a}_1^T, \vec{a}_1^T \cdot \vec{s} + e_1] + r_2 \cdot [\vec{a}_2^T, \vec{a}_2^T \cdot \vec{s} + e_2] = [\vec{a}^T, \vec{a}^T \cdot \vec{s} + e]$, where $\vec{a} = r_1 \cdot \vec{a}_1 + r_2 \cdot \vec{a}_2$, $e = r_1 \cdot e_1 + r_2 \cdot e_2$.
  - Correctness: $|e| > |e_1|, |e_2|$, but 'small' if r_1, r_2 'small'.
  - Security: $\vec{a}$ is $\approx$ uniformly random if r_i's have enough entropy!

Public-Key Encryption from LWE

Definition

Regev's LWE-based Public-Key Encryption:

- Key Generation – KG: Fix integers q, m, n. Pick secret key $\vec{s} \hookleftarrow U(\mathbb{Z}_q^n)$. Publish public key $(A, \vec{p})$, where:
  - $A \hookleftarrow U(\mathbb{Z}_q^{m \times n})$.
  - $\vec{p} = A \cdot \vec{s} + \vec{e} \bmod q$ with $\vec{e} \hookleftarrow \chi_{\alpha q}^m$.
- Encryption – Enc: Fix integers t, B_r. Given message $m \in \mathbb{Z}_t$ and public key $(A, \vec{p})$,
  - Pick coefficient vector $\vec{r} \hookleftarrow U(\{-B_r, \ldots, B_r\}^m)$.
  - Compute:
  $$\vec{a}^T = \vec{r}^T \cdot A, c = \vec{r}^T \cdot \vec{p} + \lceil q/t \rfloor \cdot m \bmod q.$$
  - Return ciphertext $(\vec{a}^T, c)$.
- Decryption – Dec: Given ciphertext $(\vec{a}^T, c)$ and secret key $\vec{s}$,
  - Compute $c' = c - \vec{a}^T \cdot \vec{s} \bmod q$.
  - Compute $c'' \in \mathbb{Z}_q$ by rounding c' to the nearest multiple of $\lceil q/t \rfloor \bmod q$.
  - Return plaintext $m = \frac{\vec{c''}}{\lceil q/t \rfloor}$.

Public-Key Encryption from LWE: Correctness

- In ciphertext, $c = \vec{r}^T \cdot A \cdot \vec{s} + \vec{r}^T \cdot \vec{e} + \lceil q/t \rfloor \cdot m \bmod q = \vec{a}^T \cdot \vec{s} + e + \lceil q/t \rfloor \cdot m \bmod q$, where $e = \vec{r}^T \cdot \vec{e}$.
- Decryption recovers $c' = e + \lceil q/t \rfloor \cdot \vec{m} \bmod q$. As in symmetric-key scheme, rounding succeeds to recover m if the 'new' noise e is sufficiently small:

$$e < \frac{1}{2} \cdot \lceil q/t \rfloor \approx \frac{q}{2t}.$$

- If noise distribution $\chi_{\alpha q}$ of $\vec{e}$ coordinates is (rounded) normal distribution with std. dev. αq, distribution of 'new' noise $e = \vec{r}^T \cdot \vec{e}$ (neglecting rounding) is, for a fixed $\vec{r}$, also normal distributed with std. dev. $\alpha q \cdot \|\vec{r}\|$. And the expected value of $\|\vec{r}\|$ is $\approx \sqrt{B_r(B_r+1)m/3}$, which is a good approximation to $\|\vec{r}\|$ with high probability.
- Hence error probability per coordinate p_e is probability that a standard normal distributed random variable (mean 0, std. dev 1) exceeds $\frac{1}{2t\alpha}$ in magnitude:

$$p_e \approx 2 \cdot \left(1 - \Phi\left(\frac{1}{2t\alpha} \cdot \sqrt{\frac{3}{B_r(B_r+1)m}}\right)\right),$$

  where Φ is the cumulative distribution function of normal distribution.

So: p_e 'small' when the following correctness condition holds:

$$t << \frac{1}{2\alpha} \cdot \sqrt{\frac{3}{B_r(B_r+1)m}}.$$

Since B_r can be 1, lose a factor of $O(\sqrt{m})$ in t (or q for a given t) versus the symmetric-key case.

Public-Key Encryption from LWE: Security

Q: Why is it secure, assuming that Decision-LWE is hard?
A: As in symmetric-key case, a security reduction!

- Build an efficient Dec-LWE algorithm D, given an efficient attack algorithm B breaking encryption scheme.

Q: What do we mean by 'B breaks the encryption scheme'?
Possible A: Similar to symmetric-key case – IND-CPA definition for public-key encryption IND-CPA Attack model in the public-key case for attacker B:

- Challenger runs Key Gen. algorithm of encryption scheme, obtains a secret key $\vec{s}$ and a public key $(A, \vec{p})$. The public key is given to B.
- No need to give B access to an 'encryption oracle': B can simulate such an oracle by itself, using the public key. B outputs a pair of 'challenge messages' $\vec{m}_0^*$, $\vec{m}_1^*$.
- Challenger picks a random bit $b \longleftarrow U(\{0,1\})$, computes 'challenge ciphertext' $(\vec{a^*}^T, c^*) = \mathrm{Enc}((A, \vec{p}), \vec{m}_b^*)$ for the challenge message selected by b, and gives $(\vec{a^*}^T, c^*)$ to B.
- Attacker B outputs a guess b' for the bit b chosen by the challenger. Attacker 'wins' game if $b' = b$.

Definition

IND-CPA security (at 2^λ security level): Any attack B with run-time $T(\mathrm{B}) \leq 2^\lambda$ wins game with prob. $\leq 1/2 + 1/2^\lambda$.

Public-Key Encryption from LWE: Security

In security reduction, we need a way of measuring closeness of probability distributions. In crypto., usually use statistical distance between distributions.

Definition

For two probability distributions D_1 and D_2 on a discrete set S, the statistical distance $\Delta(D_1, D_2)$ is defined as:

$$\Delta(D_1, D_2) \stackrel{\text{def}}{=} \frac{1}{2} \cdot \sum_{x \in S} |D_1(x) - D_2(x)|.$$

- Δ is always between 0 ($D_1 = D_2$) and 1 (D_1 and D_2 never output the same value).

Why is stat. distance useful? Because no attack algorithm (function) can increase it!

Lemma

Let D_1, D_2 be any two distributions, and A be any algorithm. Then:

$$|\Pr_{x \hookleftarrow D_1}[A(x) = 1] - \Pr_{x \hookleftarrow D_2}[A(x) = 1]| \leq \Delta(D_1, D_2).$$

Public-Key Encryption from LWE: Security

Security Reduction from Decision-LWE

Suppose there was an efficient IND-CPA attack algorithm B, breaking 2^{λ} security of Regev's encryption scheme:

- B runs in time T_B and wins IND-CPA game with probability $1/2 + \varepsilon_B$ (with $T_B < 2^{\lambda}$ and $\varepsilon_B > 1/2^{\lambda}$).

Then, given a $Decision - LWE_{q,m,n,\alpha}$ instance $(q, n, A, \vec{y})$, Dec-LWE algorithm D works as follows:

- D runs attacker B on input public key $(A, \vec{p} = \vec{y})$.
- When B makes its challenge query $(\vec{m}_0^*, \vec{m}_1^*)$, D behaves like the real challenger: chooses a random bit b, picks coefficient vector $\vec{r} \hookleftarrow U(\{-B_r, \ldots, B_r\}^m)$ and computes:

  $$\vec{a^*}^T = \vec{r}^T \cdot A, c^* = \vec{r}^T \cdot \vec{y} + \lceil q/t \rfloor \cdot m_b \bmod q.$$

  D returns challenge ciphertext $(\vec{a^*}^T, c^*)$.
- When B returns a guess b' for b, D returns 'Real' if $b' = b$, and 'Rand' if $b' \neq b$.

Public-Key Encryption from LWE: Security

Security Reduction from Decision-LWE (cont.)

Q: Why does D work? Consider two LWE scenarios for $\vec{y}$:

- 'Real' LWE scenario, $\vec{y} = A \cdot \vec{s} + \vec{e}$ – public key and challenge ciphertext returned by D to B are computed exactly as in the real IND-CPA game, so B wins game with good probability $1/2 + \varepsilon_B$, hence D returns 'Real' with prob. $1/2 + \varepsilon_B$.
- 'Random' LWE scenario, $\vec{p} = \vec{y}$ is independent and uniformly random in $\mathbb{Z}_q^m$. Use following 'Leftover Hash Lemma' (LHL):

Lemma

Let $C \hookleftarrow U(\mathbb{Z}_q^{m \times (n+1)})$ and $\vec{r} \hookleftarrow U(\{-B_r, \ldots, B_r\}^m)$. If the following LHL condition holds:

$$(2B_r + 1)^m >> q^{n+1}, (\textit{more precisely:} (2B_r + 1)^m \geq 2^{2(\lambda+1)} \cdot q^{n+1})$$

then the probability distribution P of the pair $(C, \vec{r}^T \cdot C \bmod q)$ is statistically indistinguishable from the uniform distribution $U = U(\mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^{n+1})$. More precisely, the statistical distance $\Delta(P, U)$ between the probability distributions P, U is at most

$$\frac{1}{2} \cdot \sqrt{\frac{q^{n+1}}{(2B_r + 1)^m}}.$$

Public-Key Encryption from LWE: Security

'Random' LWE scenario (cont.): $\vec{p} = \vec{y}$ is independent and uniformly random in $\mathbb{Z}_q^m$

- If the distribution P of $(A, \vec{y}, \vec{a^*}^T = \vec{r}^T \cdot A, \vec{r}^T \cdot \vec{y})$ was exactly $U = U(\mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^{n+1})$, then (as in symmetric-key case), ciphertext $(\vec{a^*}^T, c^* = \vec{r}^T \cdot \vec{y} + \lceil q/t \rfloor \cdot m_b)$ is independent of b and public key $\vec{y}$ (contains no information on b), and hence D returns 'Real' with prob. $1/2$.
- By LHL, $\Delta(P, U) \leq \frac{1}{2} \cdot \sqrt{\frac{q^{n+1}}{(2B_r+1)^m}} = \delta$. By LHL condition, $\delta \leq 1/2^{\lambda+1}$ is negligible, so from property of statistical distance (wk 4 tute), D returns 'Real' with probability $\leq 1/2 + \delta \leq 1/2 + 1/2^{\lambda+1}$.

So: Distinguishing advantage of D
$\geq \varepsilon_B - 1/2^{\lambda+1} \geq 1/2^{\lambda} - 1/2^{\lambda+1} \geq 1/2^{\lambda+1}$.
Also, run-time of D is (approx.) run-time of B, i.e. $< 2^{\lambda}$.
Conclusion: Contradiction with $2^{\lambda+1}$ security of Decision-LWE!

Theorem

If LHL condition holds, IND-CPA security of Regev's encryption scheme is at least as hard as Decision $- LWE_{q,m,n,\alpha}$.

Public-Key Encryption from LWE: Security

Choice of Parameters for Regev's Encryption Scheme

The LHL condition tells us how large m should be chosen:

$$(2B_r+1)^m \geq 2^{2(\lambda+1)} \cdot q^{n+1} \text{ implies } m \geq \frac{(n+1)\cdot \log q + 2\cdot(\lambda+1)}{\log(2B_r+1)}.$$

Q: How to choose the other parameters of Regev's scheme?

A: Based on the security level and LWE problem's relation to lattice problems (next lecture!)