Quantum Cyber Security
Lecture 17: Post-Quantum Cryptography II

Petros Wallden

University of Edinburgh

19th March 2024
Lattice Problems:

1. Learning-With-Errors (LWE)
2. Shortest-Vector Problem (SVP)

LWE-based Public-Key Encryption (Regev’s)
Lattice Problems:

1. Learning-With-Errors (LWE)

2. Shortest-Vector Problem (SVP)

LWE-based Public-Key Encryption (Regev’s)

**Notation colour code:** parameters and functions: public (blue), private (red), secret but not used later (brown)
The Learning-With-Errors (LWE) Problem

- **Parameters:**
  - Vectors in $\mathbb{Z}_q^n$
  - $n \in \mathbb{N}$ dimension of vectors
  - $q$ is a prime number where additions are carried over $\mod q$
  - Coefficients are **integers**

LWE Problem (Search)

Given $m$ pairs $(\vec{a}_i, b_i)$ find the secret vector $\vec{s}$

Where we have:

- Random public vectors $\vec{a}_i$
- A single random secret vector $\vec{s}$ that we want to find
- Small error terms $e_i$ (sampled from a distribution that w.h.p. is small, i.e. $e_i \ll q$) that are kept secret
- Public scalars $b_i := \vec{a}_i \cdot \vec{s} + e_i$ $i \in \{1, 2, \ldots, m\}$

Petros Wallden
Lecture 17: Post-Quantum Cryptography II
The Learning-With-Errors (LWE) Problem

- **Parameters:**
  - Vectors in $\mathbb{Z}_q^n$
  - $n \in \mathbb{N}$ dimension of vectors
  - $q$ is a prime number where additions are carried over $\mod q$
  - Coefficients are integers

**LWE Problem (Search)**

Given $m$ pairs $(\vec{a}_i, b_i)$ find the secret vector $\vec{s}$
The Learning-With-Errors (LWE) Problem

- **Parameters:**
  - Vectors in $\mathbb{Z}_q^n$
  - $n \in \mathbb{N}$ dimension of vectors
  - $q$ is a prime number where additions are carried over $\mod q$
  - Coefficients are integers

**LWE Problem (Search)**

Given $m$ pairs $(\vec{a}_i, b_i)$ find the secret vector $\vec{s}$

Where we have:
- Random public vectors $\vec{a}_i$
- A single random secret vector $\vec{s}$ that we want to find
- Small error terms $e_i$ (sampled from a distribution that w.h.p. is small, i.e. $e \ll q$) that are kept secret
- Public scalars $b_i := \vec{a}_i \cdot \vec{s} + e_i$
- $i \in \{1, 2, \ldots, m\}$
The Learning-With-Errors (LWE) Problem: Example

- Let $n = 4$, $q = 17$, $\vec{s} = (s_1, s_2, s_3, s_4)$
Let \( n = 4 \), \( q = 17 \), \( \vec{s}' = (s_1, s_2, s_3, s_4) \)

**Given** \( m \) equations:

\[
\begin{align*}
14s_1 + 15s_2 + 5s_3 + 2s_4 & \approx 8 \pmod{17} \\
13s_1 + 14s_2 + 14s_3 + 6s_4 & \approx 16 \pmod{17} \\
6s_1 + 10s_2 + 13s_3 + 1s_4 & \approx 3 \pmod{17} \\
\vdots \\
9s_1 + 5s_2 + 9s_3 + 6s_4 & \approx 9 \pmod{17}
\end{align*}
\]

where \( \vec{a}_1 = (14, 15, 5, 2) \) and \( b_1 = 8 \), etc.
The Learning-With-Errors (LWE) Problem: Example

- Let \( n = 4 \), \( q = 17 \), \( \vec{s} = (s_1, s_2, s_3, s_4) \)

- Given \( m \) equations:

  \[
  14s_1 + 15s_2 + 5s_3 + 2s_4 \approx 8 \pmod{17} \\
  13s_1 + 14s_2 + 14s_3 + 6s_4 \approx 16 \pmod{17} \\
  6s_1 + 10s_2 + 13s_3 + 1s_4 \approx 3 \pmod{17} \\
  \vdots \\
  9s_1 + 5s_2 + 9s_3 + 6s_4 \approx 9 \pmod{17}
  \]

where \( \vec{a}_1 = (14, 15, 5, 2) \) and \( b_1 = 8 \), etc.

- Find the secret vector \( \vec{s} = (s_1, s_2, s_3, s_4) \)
The Learning-With-Errors (LWE) Problem

- **Without the errors it is simple** (if we had exact equality)
- Once $m = n$, we have $n$ equations with $n$ unknowns
  (Can be solved efficiently with Gaussian elimination)
The Learning-With-Errors (LWE) Problem

- **Without the errors it is simple** (if we had exact equality)

- Once $m = n$, we have $n$ equations with $n$ unknowns
  (Can be solved efficiently with Gaussian elimination)

- Instead we try to learn $\vec{s}$ from noisy samples

- Errors accumulate with operations trying to solve the system of equations (even with small initial errors).
  Large final uncertainty

- It relates with **hard lattice problems** (see next)
The Learning-With-Errors (LWE) Problem

- **Without the errors it is simple** (if we had exact equality)

- Once \( m = n \), we have \( n \) equations with \( n \) unknowns
  (Can be solved efficiently with Gaussian elimination)

- Instead we try to learn \( \vec{s} \) from noisy samples

- Errors accumulate with operations trying to solve the system of equations (even with small initial errors).
  Large final uncertainty

- It relates with **hard lattice problems** (see next)

- One can always (in exponential time) solve it by brute-force
The Learning-With-Errors (LWE) Problem

- **Without the errors it is simple** (if we had exact equality)

- Once \( m = n \), we have \( n \) equations with \( n \) unknowns (Can be solved efficiently with Gaussian elimination)

- Instead we try to learn \( \vec{s} \) from noisy samples

- Errors accumulate with operations trying to solve the system of equations (even with small initial errors). Large final uncertainty

- It relates with **hard lattice problems** (see next)

- One can always (in exponential time) solve it by brute-force

**Alternative Version:**

**Decisional LWE Problem**

Can a (quantum) poly-time adversary distinguish between LWE samples \((\vec{a}_i, b_i)\) and random samples \((\vec{a}_i, r_i) ; r_i \leftarrow \mathbb{Z}_q\)?
The Shortest Vector Problem (SVP)

Parameters:
- \( n \) dimension vector space
- \( k \) linearly independent vectors (with integer coefficients)
  \[ B = \{ \vec{b}_1, \ldots, \vec{b}_k \} \]
- (Euclidean) norm \( \|\vec{a}\| := \sqrt{\vec{a} \cdot \vec{a}} \)

SVP Problem
Find the shortest (non-zero) integer linear combination of basis vectors:
\[ \vec{SV} = \vec{b}_1 x_1 + \ldots + \vec{b}_k x_k \]
where \((x_1, \ldots, x_k) \in \mathbb{Z}^k \setminus \{0\} \).

We define \( \lambda(L) := \|\vec{SV}\| \).

SVP \( \beta \) Problem (Approximate)
Find a non-zero integer vector with length \( \beta \lambda(L) \).

GapSVP \( \beta \) Problem
Determine whether the shortest vector has \( \lambda(L) \leq 1 \) or \( \lambda(L) \geq \beta \).
The Shortest Vector Problem (SVP)

Parameters:
- \( n \) dimension vector space
- \( k \) linearly independent vectors (with integer coefficients)
  \[ \mathbf{B} = \{ \mathbf{b}_1, \ldots, \mathbf{b}_k \} \]
- (Euclidean) norm \( \| \mathbf{a} \| := \sqrt{\mathbf{a} \cdot \mathbf{a}} \)

SVP Problem
Find the shortest (non-zero) integer linear combination of basis vectors:
\[ \mathbf{SV} := \mathbf{b}_1 x_1 + \ldots + \mathbf{b}_k x_k, \text{ where } (x_1, \ldots, x_k) \in \mathbb{Z}^k \setminus \{0\}. \]

We define \( \lambda(L) := \| \mathbf{SV} \| \).
The Shortest Vector Problem (SVP)

Parameters:
- $n$ dimension vector space
- $k$ linearly independent vectors (with integer coefficients)
  $B = \{\vec{b}_1, \ldots, \vec{b}_k\}$
- (Euclidean) norm $\|\vec{a}\| := \sqrt{\vec{a} \cdot \vec{a}}$

SVP Problem

Find the shortest (non-zero) integer linear combination of basis vectors:
$S\vec{V} := \vec{b}_1 x_1 + \ldots + \vec{b}_k x_k$, where $(x_1, \ldots, x_k) \in \mathbb{Z}^k \setminus \{0\}$.
We define $\lambda(L) := \|S\vec{V}\|$. 

SVP$_\beta$ Problem (Approximate)

Find a non-zero integer vector with length $\beta \lambda(L)$. 

6/11
The Shortest Vector Problem (SVP)

Parameters:
- $n$ dimension vector space
- $k$ linearly independent vectors (with integer coefficients)
  $B = \{ \vec{b}_1, \ldots, \vec{b}_k \}$
- (Euclidean) norm $\| \vec{a} \| := \sqrt{\vec{a} \cdot \vec{a}}$

SVP Problem
Find the shortest (non-zero) integer linear combination of basis vectors:
$\vec{S}V := \vec{b}_1 x_1 + \ldots + \vec{b}_k x_k$, where $(x_1, \ldots, x_k) \in \mathbb{Z}^k \setminus \{0\}$.
We define $\lambda(L) := \| \vec{S}V \|$. 

SVP$_\beta$ Problem (Approximate)
Find a non-zero integer vector with length $\beta \lambda(L)$.

GapSVP$_\beta$ Problem
Determine whether the shortest vector has $\lambda(L) \leq 1$ or $\lambda(L) \geq \beta$. 

Lecture 17: Post-Quantum Cryptography II
Average case LWE implies worst case $\text{SVP}_\beta$

In cryptography we need proven average case hardness!
Why is LWE good for post-quantum cryptography

- **Average case LWE implies worst case SVP$_\beta$**

- In cryptography we need proven average case hardness!

- The exact SVP is NP-hard and thus **hard for quantum computers** (unless crazy things happen!)
**Average case LWE implies worst case SVP$_\beta$**

- In cryptography we need proven average case hardness!
- The exact SVP is NP-hard and thus **hard for quantum computers** (unless crazy things happen!)
- Approximate versions are also **believed to be hard** (but not proven – hardness depends on the approximation $\beta$)
Why is LWE good for post-quantum cryptography

- **Average case** LWE implies **worst case** SVP\(\beta\)

- In cryptography we need proven average case hardness!

- The exact SVP is NP-hard and thus **hard for quantum computers** (unless crazy things happen!)

- Approximate versions are also **believed to be hard** (but not proven – hardness depends on the approximation \(\beta\))

- Regev’s encryption scheme (next) is secure provided that the decisional-LWE is hard.
LWE-based Encryption Scheme (Regev’s)

Parameters: $n$ security parameters, $m \neq$ equations, $q$ modulus, $\alpha$ error parameter
LWE-based Encryption Scheme (Regev’s)

Parameters: \( n \) security parameters, \( m \) \# equations, \( q \) modulus, \( \alpha \) error parameter

Conditions on Parameters: Essential for security
\[ n^2 \leq q \leq 2n^2 \; ; \; m = 1.1n \log q \; ; \; \alpha = 1/ (\sqrt{n} \log^2 n) \]
LWE-based Encryption Scheme (Regev’s)

Parameters: $n$ security parameters, $m$ # equations, $q$ modulus, $\alpha$ error parameter

Conditions on Parameters: Essential for security

\[ n^2 \leq q \leq 2n^2 ; \quad m = 1.1n \log q ; \quad \alpha = 1/ (\sqrt{n} \log^2 n) \]

KeyGen:

- Private Key: $\vec{s} \leftarrow Z_q^n$
LWE-based Encryption Scheme (Regev’s)

Parameters: $n$ security parameters, $m$ equations, $q$ modulus, $\alpha$ error parameter

Conditions on Parameters: Essential for security

$n^2 \leq q \leq 2n^2$ ; $m = 1.1n \log q$ ; $\alpha = 1/ (\sqrt{n} \log^2 n)$

KeyGen:

- **Private Key:** $\vec{s} \leftarrow Z_q^n$

- **Public Key:** $m$ LWE samples $(\vec{a}_i, b_i)$, where:
  - $b_i = \vec{a}_i \cdot \vec{s} + e_i$
  - $\vec{a}_i \leftarrow Z_q^n \forall i$
  - $e_i$ random small numbers (sampled from normal distribution with standard deviation $\alpha q$).
Enc((\vec{a}_i, b_i), \mu):

- For single bit message \( \mu \in \{0, 1\} \)
- Choose a random subset \( S \) of indices \( \{1, \ldots, m\} \) (out of the \( 2^m \) possible subsets).
- Compute \( \vec{a} := \sum_{i \in S} a_i \) and \( b := \sum_{i \in S} b_i \)
- Output pair \((\vec{a}, c)\), where \( c := b + \mu \left\lfloor \frac{a}{2} \right\rfloor \)
2 Enc((\vec{a}_i, b_i), \mu):
   - For single bit message \( \mu \in \{0, 1\} \)
   - Choose a random subset \( S \) of indices \( \{1, \ldots, m\} \) (out of the \( 2^m \) possible subsets).
   - Compute \( \vec{a} := \sum_{i \in S} \vec{a}_i \) and \( b := \sum_{i \in S} b_i \)
   - Output pair \((\vec{a}, c)\), where \( c := b + \mu \lfloor \frac{q}{2} \rfloor \)

3 Dec((\vec{a}, c), \vec{s}):
   - Compute \( c - \vec{a} \cdot \vec{s} \)
   - Check whether outcome is closer to 0 or \( \frac{q}{2} \) (oper. done mod \( q \))
   - Output \( \mu = 0 \) if closer to zero, and \( \mu = 1 \) otherwise
**Correctness:** We consider \( \text{Dec}(\text{Enc}((\vec{a}_i, b_i), \mu), \vec{s}) \).

\[
c - \vec{a} \cdot \vec{s} = b + \mu \left\lfloor \frac{q}{2} \right\rfloor - \vec{a} \cdot \vec{s}
\]

\[
= \sum_{i \in S} (\vec{a}_i \cdot \vec{s} + e_i) + \mu \left\lfloor \frac{q}{2} \right\rfloor - \left( \sum_{i \in S} \vec{a}_i \right) \cdot \vec{s}
\]

\[
= \sum_{i \in S} e_i + \mu \left\lfloor \frac{q}{2} \right\rfloor
\]
**Correctness:** We consider \( \text{Dec}(\text{Enc}(\langle \vec{a}_i, b_i \rangle, \mu), \vec{s}) \).

\[
c - \vec{a} \cdot \vec{s} = b + \mu \left\lfloor \frac{q}{2} \right\rfloor - \vec{a} \cdot \vec{s}
\]

\[
= \sum_{i \in S} (\vec{a}_i \cdot \vec{s} + e_i) + \mu \left\lfloor \frac{q}{2} \right\rfloor - \left( \sum_{i \in S} \vec{a}_i \right) \cdot \vec{s}
\]

\[
= \sum_{i \in S} e_i + \mu \left\lfloor \frac{q}{2} \right\rfloor
\]

Provided \( e_i \)'s are small enough, this is closer to 0 when \( \mu = 0 \) and to \( \frac{q}{2} \) otherwise.
LWE-based Encryption Scheme (Regev’s)

- **Security:**
  - Dec works since we “cancel” $\vec{a} \cdot \vec{s}$ term (unknown to adversary).

  The Decisional-LWE states that adversary cannot distinguish $(\vec{a}, b)$ and $(\vec{a}, r)$ where $r$ is random. Thus $c = b + \mu q^2$ looks like $r + \mu q^2$ to the adversary.

  The message $\mu$ is masked by the random $r$.

  See an example at Tutorial 6

Efficiency: The parameters required to ensure security and correctness imply public key of $O(n^2)$ (not efficient).

Using “Ring-LWE” instead can bring this to linear. We define rings and give another ring lattice-based cryptosystem at the next lecture.
LWE-based Encryption Scheme (Regev’s)

- **Security:**
  - Dec works since we “cancel” $\vec{a} \cdot \vec{s}$ term (unknown to adversary).
  - The Decisional-LWE states that adversary cannot distinguish between $(\vec{a}, b)$ and $(\vec{a}, r)$ where $r$ is random.
  - Thus $c = b + \mu \left\lfloor \frac{q}{2} \right\rfloor$ looks like $r + \mu \left\lfloor \frac{q}{2} \right\rfloor$ to the adversary.
LWE-based Encryption Scheme (Regev’s)

Security:

- Dec works since we “cancel” $\vec{a} \cdot \vec{s}$ term (unknown to adversary).
- The Decisional-LWE states that adversary cannot distinguish between $(\vec{a}, b)$ and $(\vec{a}, r)$ where $r$ is random.
- Thus $c = b + \mu \left\lfloor \frac{a}{2} \right\rfloor$ looks like $r + \mu \left\lfloor \frac{a}{2} \right\rfloor$ to the adversary.
- The message $\mu$ is masked by the random $r$
LWE-based Encryption Scheme (Regev’s)

- **Security:**
  - Dec works since we “cancel” $\vec{a} \cdot \vec{s}$ term (unknown to adversary).
  - The **Decisional-LWE** states that adversary cannot distinguish between $(\vec{a}, b)$ and $(\vec{a}, r)$ where $r$ is random.
  - Thus $c = b + \mu \left\lfloor \frac{a}{2} \right\rfloor$ looks like $r + \mu \left\lfloor \frac{a}{2} \right\rfloor$ to the adversary.
  - The message $\mu$ is masked by the random $r$

- See an example at Tutorial 6
Security:
- Dec works since we “cancel” $\vec{a} \cdot \vec{s}$ term (unknown to adversary).
- The Decisional-LWE states that adversary cannot distinguish between $(\bar{a}, b)$ and $(\bar{a}, r)$ where $r$ is random.
- Thus $c = b + \mu \left\lfloor \frac{q}{2} \right\rfloor$ looks like $r + \mu \left\lfloor \frac{q}{2} \right\rfloor$ to the adversary.
- The message $\mu$ is masked by the random $r$

See an example at Tutorial 6

Efficiency: The parameters required to ensure security and correctness imply public key of $O(n^2)$ (not efficient)

Using “Ring-LWE” instead can bring this to linear.
LWE-based Encryption Scheme (Regev’s)

- **Security:**
  - Dec works since we “cancel” \( \vec{a} \cdot \vec{s} \) term (unknown to adversary).
  - The Decisional-LWE states that adversary cannot distinguish between \((\vec{a}, b)\) and \((\vec{a}, r)\) where \(r\) is random.
  - Thus \( c = b + \mu \left\lfloor \frac{q}{2} \right\rfloor \) looks like \( r + \mu \left\lfloor \frac{q}{2} \right\rfloor \) to the adversary.
  - The message \(\mu\) is masked by the random \(r\)

- See an example at Tutorial 6

- **Efficiency:** The parameters required to ensure security and correctness imply public key of \(O(n^2)\) (not efficient)

  Using “Ring-LWE” instead can bring this to linear.

- We define rings and give another ring lattice-based cryptosystem at the next lecture.