From linear algebra to post-quantum cryptography

Dr. Ir. Fré Vercauteren

frederik.vercauteren@gmail.com
Open Security Research (China)
ESAT/COSIC - KU Leuven (Belgium)

Quantum computers and factoring

Learning with errors

Cryptography from LWE

Post-quantum public key cryptography

- Currently only two types PK are popular
- Factoring based: given $n = p \cdot q$, find p and q
- Discrete logarithm based: given g and $h = g^a \bmod p$, find a

Post-quantum public key cryptography

- Currently only two types PK are popular
- Factoring based: RSA
- Discrete logarithm based: DSA, ECDSA

Post-quantum public key cryptography

- Currently only two types PK are popular
- Factoring based: RSA
- Discrete logarithm based: DSA, ECDSA
- **Shor (1994)**: quantum algorithm for factoring and dlog in time $\tilde{O}((\log N)^2)$

Post-quantum public key cryptography

- Currently only two types PK are popular
- Factoring based: RSA
- Discrete logarithm based: DSA, ECDSA
- **Shor (1994)**: quantum algorithm for factoring and dlog in time $\tilde{O}((\log N)^2)$
- Need for new constructions for the post-quantum era
  - Lattice based
  - Multivariate polynomial based
  - Code based
  - Hash based
  - Supersingular isogenies

Quantum computers

- Classical computer: bits, either 0 or 1
- Quantum computer: quantum bit (qubit)
- Qubit: superposition of two basic states $|0\rangle$ and $|1\rangle$

$$|\phi\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle\,, \quad \alpha_0, \alpha_1 \in \mathbb{C}, \quad |\alpha_0|^2 + |\alpha_1|^2 = 1$$

Quantum computers

- Classical computer: bits, either 0 or 1
- Quantum computer: quantum bit (qubit)
- Qubit: superposition of two basic states $|0\rangle$ and $|1\rangle$

$$|\phi\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle\,, \quad \alpha_0, \alpha_1 \in \mathbb{C}, \quad |\alpha_0|^2 + |\alpha_1|^2 = 1$$

- α_i is called amplitude of $|i\rangle$ in $|\phi\rangle$
- Impossible to "see" the superposition itself
- Measurement: quantum state collapses into basic state $|i\rangle$ with probability $|\alpha_i|^2$

Quantum computers

- Quantum register: n qubits can be in superposition of $N = 2^n$ basic states

$$|00\ldots0\rangle, |00\ldots1\rangle, \ldots, |11\ldots1\rangle$$

- Quantum state: $|\phi\rangle = \sum_{i=0}^{N-1} \alpha_i |i\rangle$ with $\sum_{i=0}^{N-1} |\alpha_i|^2 = 1$

Quantum computation

- Quantum mechanics only allows linear operations applied to quantum state
- A state $|\phi\rangle = \sum_{i=0}^{N-1} \alpha_i |i\rangle$ with "coordinates" $(\alpha_0, \ldots, \alpha_{N-1})$ get mapped to

$$U \begin{pmatrix} \alpha_0 \\ \alpha_1 \\ \vdots \\ \alpha_{N-1} \end{pmatrix} = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_{N-1} \end{pmatrix}$$

Quantum computation

- Quantum mechanics only allows linear operations applied to quantum state
- A state $|\phi\rangle = \sum_{i=0}^{N-1} \alpha_i |i\rangle$ with "coordinates" $(\alpha_0, \ldots, \alpha_{N-1})$ get mapped to

$$U \begin{pmatrix} \alpha_0 \\ \alpha_1 \\ \vdots \\ \alpha_{N-1} \end{pmatrix} = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_{N-1} \end{pmatrix}$$

- Since RHS has norm 1 as well, U has to be unitary
- Note general U has exponential size ...

Quantum computation

- Quantum gate: unitary matrix on small number of qubits
- Main example: 1-qubit Hadamard transform H given by

$$(\alpha_0, \alpha_1) \mapsto \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} \begin{pmatrix} \alpha_0 \\ \alpha_1 \end{pmatrix}$$

- Maps basic state $|0\rangle$ into superposition $\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$

Quantum computation

- Quantum gate: unitary matrix on small number of qubits
- Main example: 1-qubit Hadamard transform H given by

$$(\alpha_0, \alpha_1) \mapsto \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} \begin{pmatrix} \alpha_0 \\ \alpha_1 \end{pmatrix}$$

- Maps basic state $|0\rangle$ into superposition $\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$
- Hadamard on each qubit of n-bit register gives ($N = 2^n$)

$$\frac{1}{\sqrt{N}}|0\rangle + \frac{1}{\sqrt{N}}|1\rangle + \ldots + \frac{1}{\sqrt{N}}|N-1\rangle$$

- Matrix U is n-fold tensor product of 2×2 above

Quantum parallelism

- Given function $f : \{0,1\}^n \to \{0,1\}^m$, make quantum circuit U that maps $|x\rangle|0\rangle$ into $|x\rangle|f(x)\rangle$
- Apply U to a superposition gives

$$U\left(\frac{1}{\sqrt{2^n}}\sum_{x\in\{0,1\}^n}|x\rangle|0\rangle\right) = \frac{1}{\sqrt{2^n}}\sum_{x\in\{0,1\}^n}|x\rangle|f(x)\rangle$$

Quantum parallelism

- Given function $f : \{0,1\}^n \to \{0,1\}^m$, make quantum circuit U that maps $|x\rangle|0\rangle$ into $|x\rangle|f(x)\rangle$
- Apply U to a superposition gives

$$U\left(\frac{1}{\sqrt{2^n}}\sum_{x\in\{0,1\}^n}|x\rangle|0\rangle\right) = \frac{1}{\sqrt{2^n}}\sum_{x\in\{0,1\}^n}|x\rangle|f(x)\rangle$$

- This by itself is totally useless since observing the above state gives a random $|x\rangle|f(x)\rangle$

(Quantum) Fourier Transform

- Set $N = 2^n$, and set $\omega_N = \exp(2\pi i/N)$ a primitive N-th root of unity
- QFT: maps standard basis $|x\rangle$ into state

$$\frac{1}{\sqrt{N}} \sum_{y=0}^{N-1} \omega_N^{xy} |y\rangle$$

- 2^n-QFT can be computed by composition of $n(n-1)/2$ quantum gates

Factoring via period finding

- Given an N one wants to factor, fix m coprime to N
- Define $f : \mathbb{N} \to \mathbb{Z}/N\mathbb{Z} : k \mapsto m^k \bmod N$,
- $f(x) = f(x + r)$ with period r order of m modulo N

Factoring via period finding

- Given an N one wants to factor, fix m coprime to N
- Define $f : \mathbb{N} \to \mathbb{Z}/N\mathbb{Z} : k \mapsto m^k \bmod N$,
- $f(x) = f(x + r)$ with period r order of m modulo N
- Assume r is even then

$$m^r - 1 \equiv (m^{r/2} + 1)(m^{r/2} - 1) = kN$$

- Compute $\gcd(m^{r/2} - 1, N)$ as factor of N
- Probability $> 1/4$ the above is non-trivial

Shor's algorithm = period finding

- **1**: two quantum registers:
  - n-qubit register with $N^2 < 2^n \leq 2N^2$
  - $\lceil \log_2 N \rceil$ qubit register

Shor's algorithm = period finding

- **1**: two quantum registers:
  - n-qubit register with $N^2 < 2^n \leq 2N^2$
  - $\lceil \log_2 N \rceil$ qubit register
- **2**: use Hadamard n times to create superposition

$$\frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |x\rangle |0\rangle$$

Shor's algorithm = period finding

- **1**: two quantum registers:
  - n-qubit register with $N^2 < 2^n \leq 2N^2$
  - $\lceil \log_2 N \rceil$ qubit register
- **2**: use Hadamard n times to create superposition

$$\frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |x\rangle |0\rangle$$

- **3**: Apply function $f(x) = m^x \bmod N$ to the above state

$$\frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |x\rangle |m^x \bmod N\rangle$$

Shor's algorithm = period finding

- **4**: measure second register, so it collapses to one value, say m^s mod N (ignore second register)

$$\star \sum_{s+jr<2^n} |s + jr\rangle$$

Shor's algorithm = period finding

- **4**: measure second register, so it collapses to one value, say m^s mod N (ignore second register)

$$\star \sum_{s+jr<2^n} |s+jr\rangle$$

- **5**: apply 2^n point QFT which gives

$$\sum_x \alpha_x |x\rangle$$

where $\alpha_x \simeq 0$ for all x not close to a multiple of q/r

Shor's algorithm = period finding

- **4**: measure second register, so it collapses to one value, say m^s mod N (ignore second register)

$$\star \sum_{s+jr<2^n} |s + jr\rangle$$

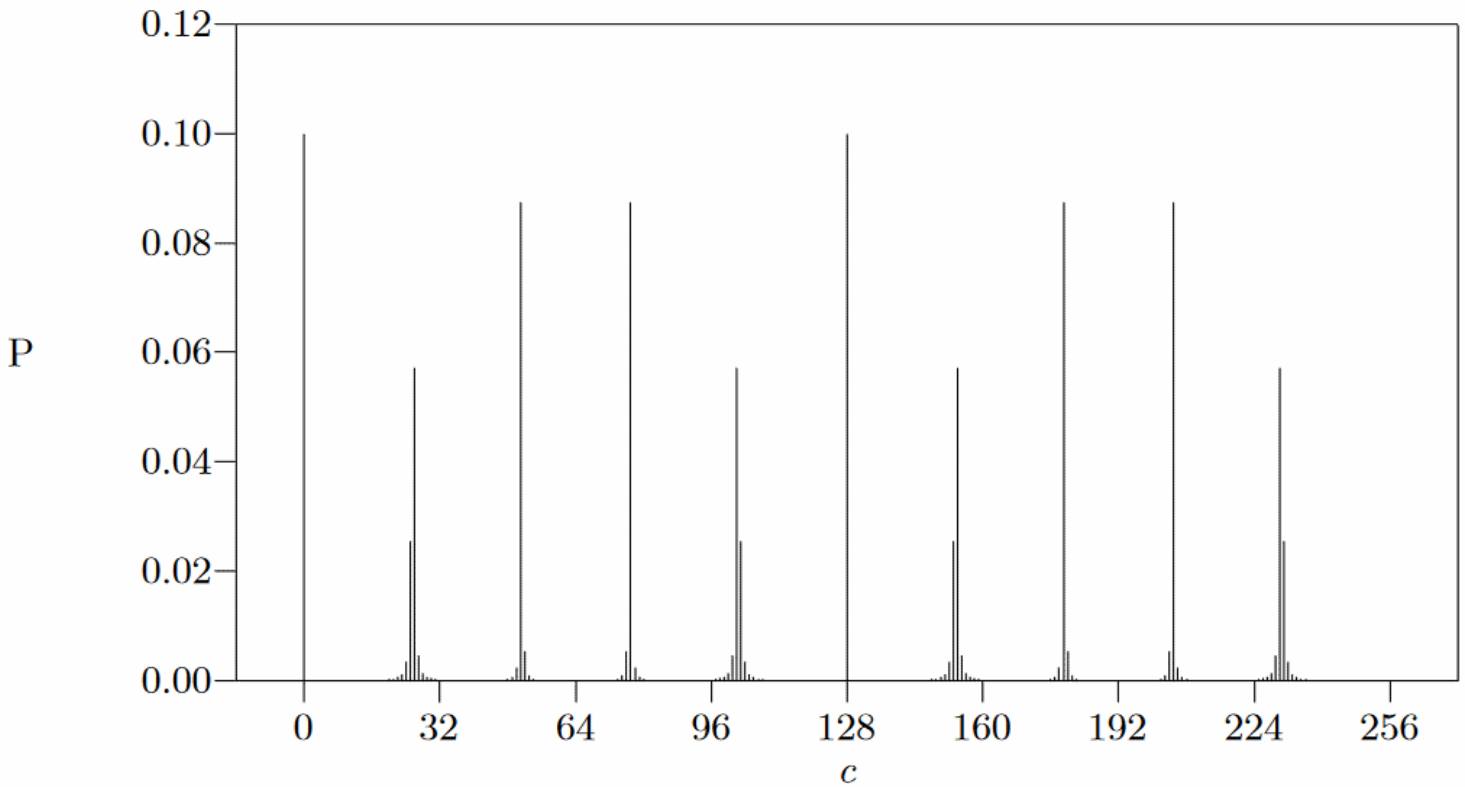
- **5**: apply 2^n point QFT which gives

$$\sum_x \alpha_x |x\rangle$$

  where $\alpha_x \simeq 0$ for all x not close to a multiple of q/r
- **6**: measuring gives b very close to kq/r for some k
- **7**: recover r, k from b and q using continued fractions

Shor's algorithm example

- Factor $N = 33$
- Choose $m = 2$ which has order $r = 10$
- Measurement will give integer close to multiple of $256/10$



Quantum algorithms

- Shor's algorithm: $\tilde{O}((\log N)^2)$ steps on a quantum computer, needs $3 \log N$ qubits
- Discrete logarithms: compute a given $h = g^a \bmod p$ and g
  - Function $f : \mathbb{Z} \times \mathbb{Z} \to \mathbb{G} : (x, y) \mapsto h^{-x} \cdot g^y$
  - Note that $f(x, y) = f(x + 1, y + a)$, so f has period $(1, a)$

Quantum algorithms

- Shor's algorithm: $\tilde{O}((\log N)^2)$ steps on a quantum computer, needs $3 \log N$ qubits
- Discrete logarithms: compute a given $h = g^a \bmod p$ and g
  - Function $f : \mathbb{Z} \times \mathbb{Z} \to \mathbb{G} : (x, y) \mapsto h^{-x} \cdot g^y$
  - Note that $f(x, y) = f(x + 1, y + a)$, so f has period $(1, a)$
- Grover's algorithm: find pre-image of function
  - given $f : A \to B$, and $b \in B$, find x s.t. $f(x) = b$
  - If $N = |A|$, Grover only requires $O(\sqrt{N})$ steps

Quantum algorithms

- Shor's algorithm: $\tilde{O}((\log N)^2)$ steps on a quantum computer, needs $3 \log N$ qubits
- Discrete logarithms: compute a given $h = g^a \bmod p$ and g
  - Function $f : \mathbb{Z} \times \mathbb{Z} \to \mathbb{G} : (x, y) \mapsto h^{-x} \cdot g^y$
  - Note that $f(x, y) = f(x + 1, y + a)$, so f has period $(1, a)$
- Grover's algorithm: find pre-image of function
  - given $f : A \to B$, and $b \in B$, find x s.t. $f(x) = b$
  - If $N = |A|$, Grover only requires $O(\sqrt{N})$ steps
- General belief: major speed-up only for problems that are not in P nor NP-complete

Linear algebra over $\mathbb{Z}_q$

- Let q be a prime and $\mathbb{Z}_q \simeq \mathbb{Z}/q\mathbb{Z}$ the field with q elements
- System of m linear equations in n unknowns ($m \geq n$)

$$\begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{m,1} & a_{m,2} & \cdots & a_{m,n} \end{pmatrix} \cdot \begin{pmatrix} s_1 \\ s_2 \\ \vdots \\ s_n \end{pmatrix} = \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \\ \vdots \\ c_m \end{pmatrix}$$

- Given matrix A and vector C, Gaussian elimination finds s_i

Distorting right hand side

- Instead of exact vector C, only given vector B with

$$\begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \\ \vdots \\ b_m \end{pmatrix} = \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \\ \vdots \\ c_m \end{pmatrix} + \begin{pmatrix} e_1 \\ e_2 \\ \vdots \\ e_n \\ \vdots \\ e_m \end{pmatrix}$$

- Error terms e_i are small wrt. q

Distorting right hand side

- Instead of exact vector C, only given vector B with

$$\begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \\ \vdots \\ b_m \end{pmatrix} = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{m,1} & a_{m,2} & \cdots & a_{m,n} \end{pmatrix} \cdot \begin{pmatrix} s_1 \\ s_2 \\ \vdots \\ s_n \end{pmatrix} + \begin{pmatrix} e_1 \\ e_2 \\ \vdots \\ e_n \\ \vdots \\ e_m \end{pmatrix}$$

- Error terms e_i are small wrt. q

Learning With Errors (LWE) Problem: search

Regev '05: On lattices, learning with errors, random linear codes, and cryptography

- Secret vector $\mathbf{s} \in \mathbb{Z}_q^n$ for some fixed n and q
- An oracle generates random $\mathbf{a} \in \mathbb{Z}_q^n$ and a small error e
- The oracle outputs $\mathbf{a}, b := \langle \mathbf{a}, \mathbf{s} \rangle + e \bmod q$
- Process is repeated many times for fresh $\mathbf{a}$ and e

Learning With Errors (LWE) Problem: search

Regev '05: On lattices, learning with errors, random linear codes, and cryptography

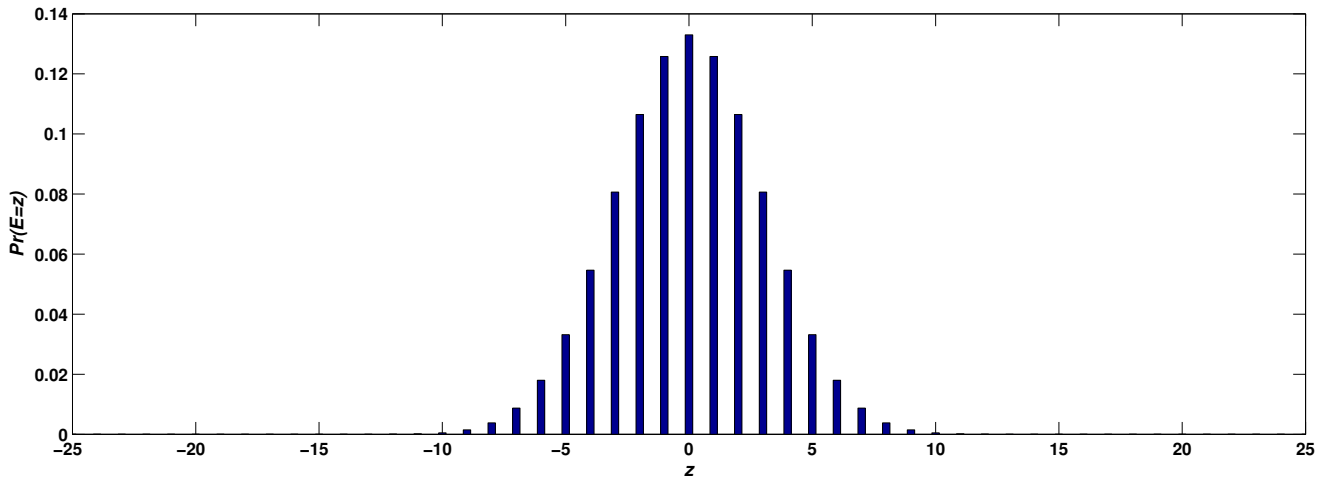
- Secret vector $\mathbf{s} \in \mathbb{Z}_q^n$ for some fixed n and q
- An oracle generates random $\mathbf{a} \in \mathbb{Z}_q^n$ and a small error e
- The oracle outputs $\mathbf{a}, b := \langle \mathbf{a}, \mathbf{s} \rangle + e \bmod q$
- Process is repeated many times for fresh $\mathbf{a}$ and e

$$\begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{m,1} & a_{m,2} & \cdots & a_{m,n} \end{pmatrix} \cdot \begin{pmatrix} s_1 \\ s_2 \\ \vdots \\ s_n \end{pmatrix} + \begin{pmatrix} e_1 \\ e_2 \\ \vdots \\ e_n \\ \vdots \\ e_m \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \\ \vdots \\ b_m \end{pmatrix}$$

Discrete Gaussian distribution

- The error distribution χ is typically discrete Gaussian distribution χ_σ on $\mathbb{Z}$
- Definition = discretization of continuous Gaussian distribution: for $z \in \mathbb{Z}$

$$\chi_\sigma(z) \sim \exp\left(\frac{-z^2}{2 \cdot \sigma^2}\right)$$



Learning With Errors (LWE) Problem: decision

Distinguish between two distributions:

LWE distribution	Uniform distribution
Fixed $\mathbf{s} \in \mathbb{Z}_q^n$	
$\mathbf{a}_i$ uniform random in $\mathbb{Z}_q^n$	$\mathbf{a}_i$ uniform random in $\mathbb{Z}_q^n$
e_i small random error from χ	b_i uniform random in $\mathbb{Z}_q$
$(\mathbf{a}_1, b_1 := \langle \mathbf{a}_1, \mathbf{s} \rangle + e_1 \bmod q)$	$(\mathbf{a}_1, b_1)$
$(\mathbf{a}_2, b_2 := \langle \mathbf{a}_2, \mathbf{s} \rangle + e_2 \bmod q)$	$(\mathbf{a}_2, b_2)$
$\vdots$	$\vdots$
$(\mathbf{a}_m, b_m := \langle \mathbf{a}_m, \mathbf{s} \rangle + e_m \bmod q)$	$(\mathbf{a}_m, b_m)$

Learning With Errors (LWE) Problem: decision

Distinguish between two distributions:

LWE distribution	Uniform distribution
Fixed $\mathbf{s} \in \mathbb{Z}_q^n$ $\mathbf{a}_i$ uniform random in $\mathbb{Z}_q^n$ e_i small random error from χ $(\mathbf{a}_1, b_1 := \langle \mathbf{a}_1, \mathbf{s} \rangle + e_1 \bmod q)$ $(\mathbf{a}_2, b_2 := \langle \mathbf{a}_2, \mathbf{s} \rangle + e_2 \bmod q)$ $\vdots$ $(\mathbf{a}_m, b_m := \langle \mathbf{a}_m, \mathbf{s} \rangle + e_m \bmod q)$	$\mathbf{a}_i$ uniform random in $\mathbb{Z}_q^n$ b_i uniform random in $\mathbb{Z}_q$ $(\mathbf{a}_1, b_1)$ $(\mathbf{a}_2, b_2)$ $\vdots$ $(\mathbf{a}_m, b_m)$

- Basically says that b_i look completely random

Gaussian elimination for LWE?

$$\begin{pmatrix} b_1 \\ b_2 \\ \vdots \end{pmatrix} = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & & & \end{pmatrix} \cdot \begin{pmatrix} s_1 \\ s_2 \\ \vdots \end{pmatrix} + \begin{pmatrix} e_1 \\ e_2 \\ \vdots \end{pmatrix}$$

- Eliminate $a_{2,1}$ by computing $A[2] - a_{1,1}^{-1} a_{2,1} A[1]$
- Element $a_{1,1}^{-1} a_{2,1}$ is typically large so blows up error e_1

Gaussian elimination for LWE?

$$\begin{pmatrix} b_1 \\ b_2 \\ \vdots \end{pmatrix} = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & & & \end{pmatrix} \cdot \begin{pmatrix} s_1 \\ s_2 \\ \vdots \end{pmatrix} + \begin{pmatrix} e_1 \\ e_2 \\ \vdots \end{pmatrix}$$

- Eliminate $a_{2,1}$ by computing $A[2] - a_{1,1}^{-1} a_{2,1} A[1]$
- Element $a_{1,1}^{-1} a_{2,1}$ is typically large so blows up error e_1
- Only combine equations with equal $a_{j,1}$ and $a_{k,1}$
- Blum, Kalai, Wasserman '03:
  - combine equations with equal blocks of coefficients

Getting rid of the errors?

- Given $\mathbf{b} \in \mathbb{Z}_q^{m \times 1}$ and $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$ with $\mathbf{b} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e}$
- Errors are small when reduced in the interval $[-q/2, q/2]$
- $\rightsquigarrow$ global problem with natural notion of smallness

Getting rid of the errors?

- Given $\mathbf{b} \in \mathbb{Z}_q^{m\times 1}$ and $\mathbf{A} \in \mathbb{Z}_q^{m\times n}$ with $\mathbf{b} = \mathbf{A}\cdot\mathbf{s} + \mathbf{e}$
- Errors are small when reduced in the interval $[-q/2, q/2]$
- $\rightsquigarrow$ global problem with natural notion of smallness
- Consider the set of vectors in $\mathbb{Z}^m$

$$\mathcal{L}(\mathbf{A}) = \{\mathbf{z} \in \mathbb{Z}^m \mid \mathbf{z} = \mathbf{A}\cdot\mathbf{x} \bmod q \text{ and } \mathbf{x} \in \mathbb{Z}_q^n\}$$

- Note that if $\mathbf{z}_1, \mathbf{z}_2 \in \mathcal{L}(\mathbf{A})$ we have $\mathbf{z}_1 - \mathbf{z}_2 \in \mathcal{L}(\mathbf{A})$
- If $\mathbf{e} \neq 0$, then $\mathbf{b} \notin \mathcal{L}(\mathbf{A})$ but still quite close to it

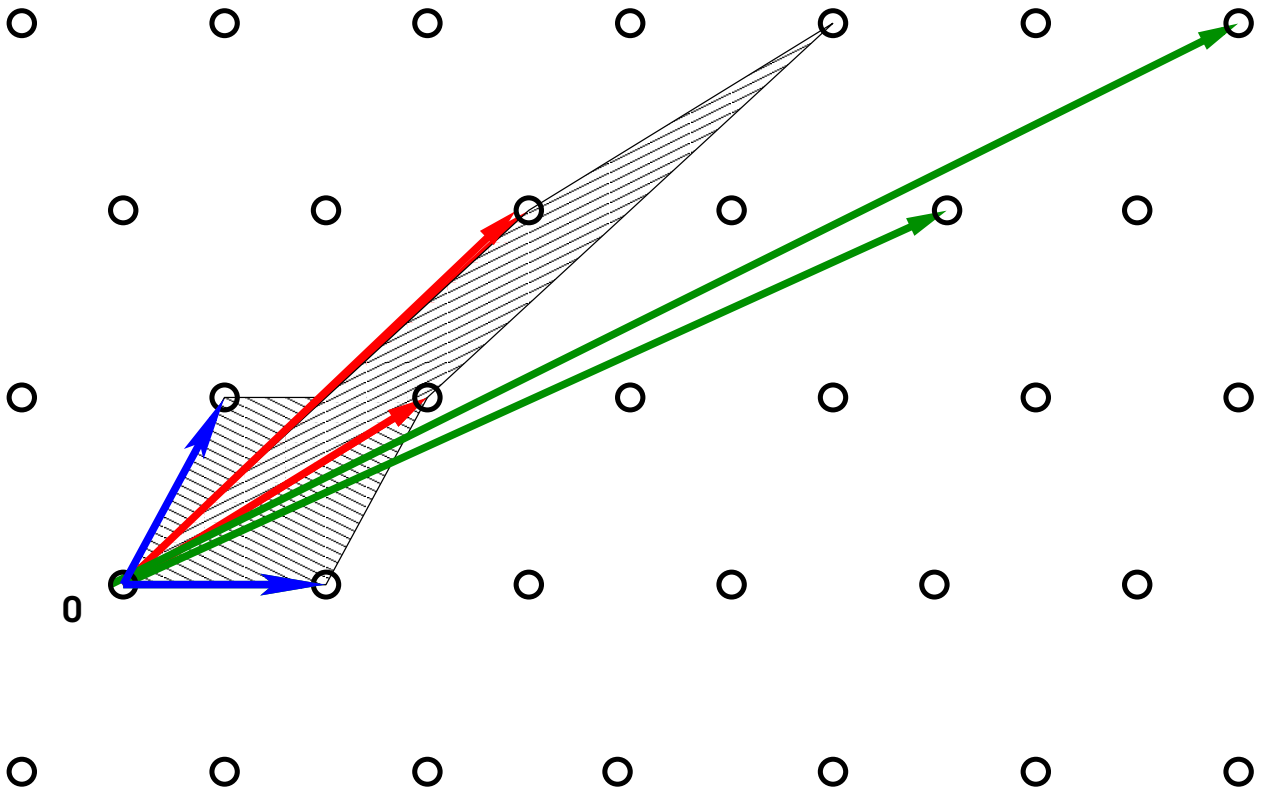
Lattices

A lattice $L \subset \mathbb{R}^m$ generated by $\mathbb{R}$-linearly independent vectors $\vec{b}_1, \ldots, \vec{b}_n \in \mathbb{R}^m$

$$L = L(\vec{b}_1, \ldots, \vec{b}_n) = \{\sum_{i=1}^{n} x_i \vec{b}_i \mid x_i \in \mathbb{Z}\},$$

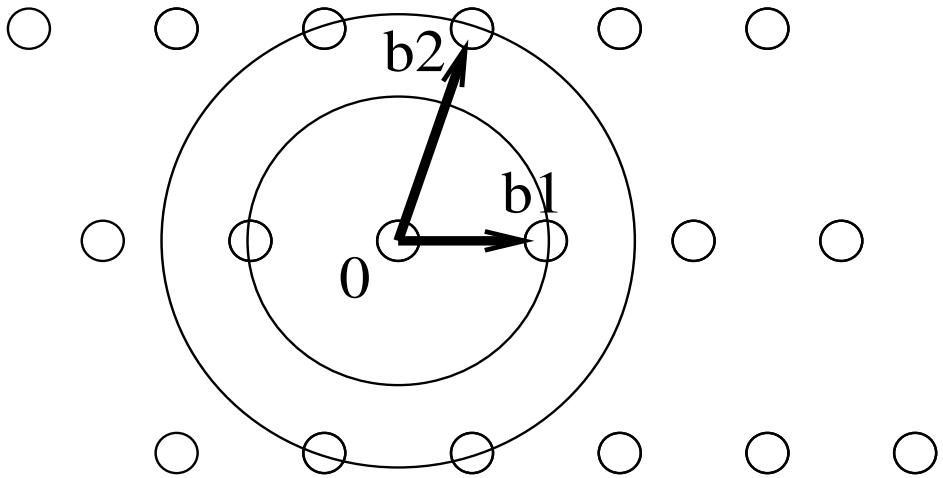
- Lattice dimension: $n = \dim(L)$
- Embedding dimension: m
- $\vec{b}_1, \ldots, \vec{b}_n$ is a lattice basis (not unique)

Infinitely many bases



Lattice minima: $\lambda_i(L)$

- There exists a shortest non-zero vector, its length is $\lambda_1(L)$
- For $i \leq d$, $\lambda_i(L)$ is the minimum radius r for which $B(\vec{0}, r)$ contains i linearly independent lattice vectors



The shortest vector problem: SVP and SIVP

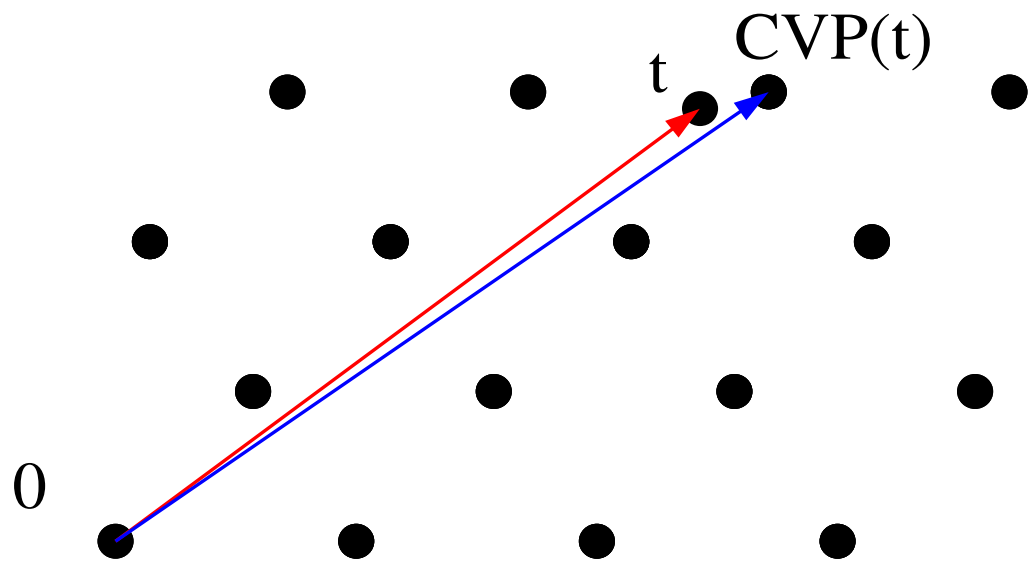
- SVP: given a basis of L, compute a vector of length $\lambda_1(L)$
- γ-SVP: compute a vector of length $\leq \gamma \cdot \lambda_1(L)$
- γ-GapSVP: decide if $\lambda_1(L) \leq 1$ or $\lambda_1(L) > \gamma$

- SIVP: shortest independent vector problem
- γ-SIVP: compute n linearly independent vectors $\mathbf{v}_i$ with

$$||\mathbf{v}_i|| \leq \gamma \cdot \lambda_n(L)$$

The closest vector problem: CVP and BDD

- Given L and a vector $\mathbf{t}$, compute a lattice vector closest to $\mathbf{t}$
- γ-CVP: Given a basis of L and a target vector $\mathbf{t}$, compute a lattice vector $\mathbf{v}$ such that $\|\mathbf{v} - \mathbf{t}\| \leq \gamma \cdot \min_{\mathbf{b} \in L} \|\mathbf{b} - \mathbf{t}\|$
- BDD_d: CVP where $\mathbf{t}$ is closer than a given bound d



Hardness results

Solving these problems for a small γ is infeasible

- CVP: NP-hard under deterministic reductions, even with preprocessing (van Emde Boas, Micciancio).
- γ-SVP: NP-hard under randomized reductions for $\gamma < 2$ (Ajtai, Micciancio, Khot).
- γ-SVP: not NP-hard for $\gamma \geq \frac{\sqrt{n}}{\log n}$ under a reasonable assumption (Goldreich & Goldwasser).
- Random instances of n^c-SVP are not easier than worst-case instances when c is larger than some constant (Ajtai, Regev).

Getting rid of the errors?

- Recall that LWE samples can be written as $\mathbf{b} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e}$ where $\mathbf{A}$ is an $m \times n$ matrix over $\mathbf{Z}_q$
- Consider the lattice

$$\mathcal{L}(\mathbf{A}) = \{\mathbf{z} \in \mathbb{Z}^m \mid \mathbf{z} = \mathbf{A} \cdot \mathbf{x} \bmod q \text{ and } \mathbf{x} \in \mathbb{Z}_q^n\}$$

- Note that the vector $\mathbf{b}$ is at distance $||\mathbf{e}||$ of $\mathcal{L}(\mathbf{A})$
- Solving BDD_d with $d \geq ||\mathbf{e}||$ removes errors

Solving decision-LWE via lattices

- Recall that LWE samples can be written as $\mathbf{b} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e}$ where $\mathbf{A}$ is an $m \times n$ matrix over $\mathbf{Z}_q$
- Let $\mathbf{w} \in \mathbb{Z}^{1 \times m}$ be a vector with $\mathbf{w} \cdot \mathbf{A} = \mathbf{0}$
- Then $\mathbf{w} \cdot \mathbf{b} = \mathbf{w} \cdot \mathbf{A} \cdot \mathbf{s} + \mathbf{w} \cdot \mathbf{e} = \mathbf{w} \cdot \mathbf{e}$
- If $\mathbf{w}$ is short, then $\mathbf{w} \cdot \mathbf{e}$ will not be uniform if sample is LWE

Solving decision-LWE via lattices

- Recall that LWE samples can be written as $\mathbf{b} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e}$ where $\mathbf{A}$ is an $m \times n$ matrix over $\mathbf{Z}_q$
- Let $\mathbf{w} \in \mathbb{Z}^{1 \times m}$ be a vector with $\mathbf{w} \cdot \mathbf{A} = \mathbf{0}$
- Then $\mathbf{w} \cdot \mathbf{b} = \mathbf{w} \cdot \mathbf{A} \cdot \mathbf{s} + \mathbf{w} \cdot \mathbf{e} = \mathbf{w} \cdot \mathbf{e}$
- If $\mathbf{w}$ is short, then $\mathbf{w} \cdot \mathbf{e}$ will not be uniform if sample is LWE
- Consider the lattice

$$\mathcal{L}^{\perp}(\mathbf{A}) = \{\mathbf{w} \in \mathbb{Z}^m \mid \mathbf{w} \cdot \mathbf{A} = 0 \bmod q\}$$

- Finding short vectors in $\mathcal{L}^{\perp}(\mathbf{A})$ breaks decision LWE

Properties of the LWE Problems

- LWE is proven to be as hard as worst-case lattice problems (GapSVP and SIVP)
  - Gaussian parameter should be large enough $\sqrt{2\pi} \cdot \sigma > \sqrt{n}$

Properties of the LWE Problems

- LWE is proven to be as hard as worst-case lattice problems (GapSVP and SIVP)
  - Gaussian parameter should be large enough $\sqrt{2\pi} \cdot \sigma > \sqrt{n}$
- Easy to test if a candidate $\mathbf{s}' \in \mathbb{Z}_q^n$ is a real solution
  - test whether $b_i - \langle \mathbf{a}_i, \mathbf{s}' \rangle$ is small for all i

Properties of the LWE Problems

- LWE is proven to be as hard as worst-case lattice problems (GapSVP and SIVP)
  - Gaussian parameter should be large enough $\sqrt{2\pi} \cdot \sigma > \sqrt{n}$
- Easy to test if a candidate $\mathbf{s}' \in \mathbb{Z}_q^n$ is a real solution
  - test whether $b_i - \langle \mathbf{a}_i, \mathbf{s}' \rangle$ is small for all i
- Given LWE problem with secret $\mathbf{s}$, can easily create LWE problem for secret $\mathbf{s} + \mathbf{t}$
  - Replace b_i with $b_i + < \mathbf{a}_i, \mathbf{t} >$
  - Random self reduction

Properties of the LWE Problems

- LWE is proven to be as hard as worst-case lattice problems (GapSVP and SIVP)
  - Gaussian parameter should be large enough $\sqrt{2\pi} \cdot \sigma > \sqrt{n}$
- Easy to test if a candidate $\mathbf{s}' \in \mathbb{Z}_q^n$ is a real solution
  - test whether $b_i - \langle \mathbf{a}_i, \mathbf{s}' \rangle$ is small for all i
- Given LWE problem with secret $\mathbf{s}$, can easily create LWE problem for secret $\mathbf{s} + \mathbf{t}$
  - Replace b_i with $b_i + < \mathbf{a}_i, \mathbf{t} >$
  - Random self reduction
- Search and decision problems are equivalent (easy for q prime $O(\text{poly}(n))$

Properties of the LWE Problems

- LWE is proven to be as hard as worst-case lattice problems (GapSVP and SIVP)
  - Gaussian parameter should be large enough $\sqrt{2\pi} \cdot \sigma > \sqrt{n}$
- Easy to test if a candidate $\mathbf{s}' \in \mathbb{Z}_q^n$ is a real solution
  - test whether $b_i - \langle \mathbf{a}_i, \mathbf{s}' \rangle$ is small for all i
- Given LWE problem with secret $\mathbf{s}$, can easily create LWE problem for secret $\mathbf{s} + \mathbf{t}$
  - Replace b_i with $b_i + < \mathbf{a}_i, \mathbf{t} >$
  - Random self reduction
- Search and decision problems are equivalent (easy for q prime $O(\text{poly}(n))$
- Secret $\mathbf{s}$ can be taken from the error distribution χ

Search LWE $\leq_P$ Decision LWE

- Given an oracle that solves Decision LWE, we will solve the Search LWE
- Idea: use Decision oracle to deduce coefficients of $\mathbf{s}$ one at a time

Search LWE $\leq_P$ Decision LWE

- Given an oracle that solves Decision LWE, we will solve the Search LWE
- Idea: use Decision oracle to deduce coefficients of $\mathbf{s}$ one at a time
- Make guess g for the first coefficient of $\mathbf{s}$
- Change each sample $(\mathbf{a}, b)$ in $(\mathbf{a} + (r, 0, \ldots, 0), b + g \cdot r)$

Search LWE $\leq_P$ Decision LWE

- Given an oracle that solves Decision LWE, we will solve the Search LWE
- Idea: use Decision oracle to deduce coefficients of $\mathbf{s}$ one at a time
- Make guess g for the first coefficient of $\mathbf{s}$
- Change each sample $(\mathbf{a}, b)$ in $(\mathbf{a} + (r, 0, \ldots, 0), b + g \cdot r)$
- Submit new LWE instance to Decision oracle
  - If guess g is correct, then new instance has LWE distribution
  - If guess g is incorrect, then new instance has uniform distribution
- Repeat for other coefficients of $\mathbf{s}$

Cryptographic Applications of LWE

- LWE is as hard as worst case lattice problems, that are believed to be exponentially hard
- LWE has been used as the basis for:
  - Public key encryption
  - Key agreement
  - Digital signatures
  - Identity-based encryption
  - Many more exotic things . . .
- Main downside: inefficient both in space and time (see next slide)

Public Key Encryption based on LWE

- **Private key**: secret vector $\mathbf{s} \in \mathbb{Z}_q^n$ chosen uniform random
- **Public key**: m samples from LWE distribution with secret $\mathbf{s}$, given as $m \times n$ matrix A and $m \times 1$ matrix B

Public Key Encryption based on LWE

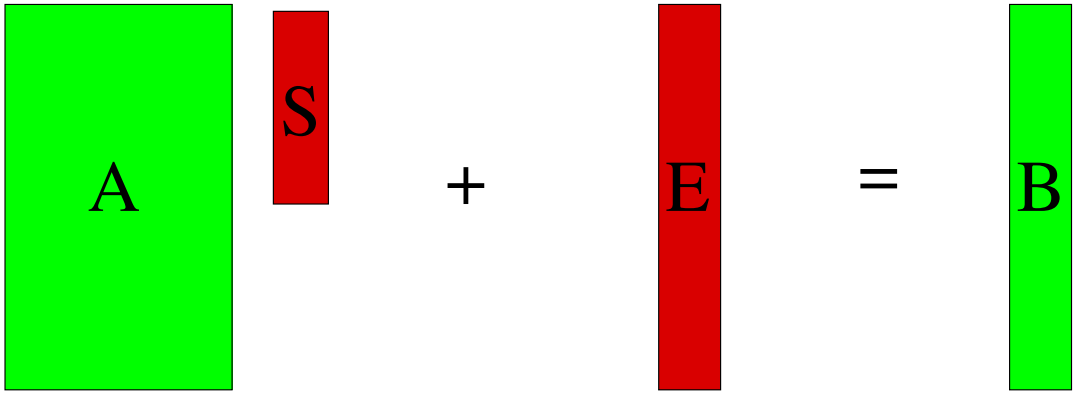
- **Private key**: secret vector $\mathbf{s} \in \mathbb{Z}_q^n$ chosen uniform random
- **Public key**: m samples from LWE distribution with secret $\mathbf{s}$, given as $m \times n$ matrix A and $m \times 1$ matrix B
- **Encryption**: for each bit b of message do
  - choose random vector $\mathbf{r} \in \mathbb{Z}_q^m$ with small coefficients
  - ciphertext = $(\mathbf{c}, d) = (\mathbf{r}^t \cdot A, \mathbf{r}^t \cdot B + b \cdot \lfloor \frac{q}{2} \rfloor) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$

Public Key Encryption based on LWE

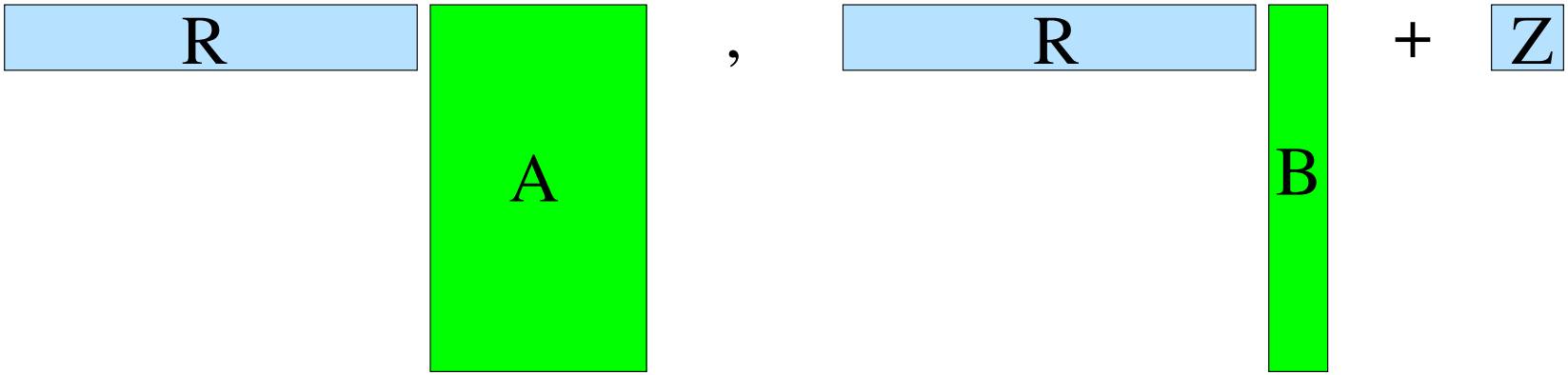
- **Private key**: secret vector $\mathbf{s} \in \mathbb{Z}_q^n$ chosen uniform random
- **Public key**: m samples from LWE distribution with secret $\mathbf{s}$, given as $m \times n$ matrix A and $m \times 1$ matrix B
- **Encryption**: for each bit b of message do
  - choose random vector $\mathbf{r} \in \mathbb{Z}_q^m$ with small coefficients
  - ciphertext = $(\mathbf{c}, d) = (\mathbf{r}^t \cdot A, \mathbf{r}^t \cdot B + b \cdot \lfloor \frac{q}{2} \rfloor) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$
- **Decryption**: given ciphertext $(\mathbf{c}, d)$
  - if $d - \langle \mathbf{c}, \mathbf{s} \rangle$ closer to 0 than to $\lfloor \frac{q}{2} \rfloor$ modulo q, then message is 0 else it is 1

Public Key Encryption based on LWE

Private/public key setup:



Encryption:



Ring-LWE

- Main problem with LWE: requires n elements in $\mathbb{Z}_q$ to generate only one extra random looking element in $\mathbb{Z}_q$

$$\mathbf{a}, b := < \mathbf{a}, \mathbf{s} > + e$$

Ring-LWE

- Main problem with LWE: requires n elements in $\mathbb{Z}_q$ to generate only one extra random looking element in $\mathbb{Z}_q$

$$\mathbf{a}, b := < \mathbf{a}, \mathbf{s} > + e$$

- Instead of inner product, try to use another type of product such that result is again in $\mathbb{Z}_q^n$ and not just $\mathbb{Z}_q$
- First idea: coordinate wise multiplication
  - Not secure since each coordinate is independent
  - Easy search to find each coordinate of $\mathbf{s}$

Ring-LWE

- Better idea: use multiplication in polynomial ring
- Consider $R := \mathbb{Z}_q[x]/(x^n + 1)$ with $n = 2^k$
- Then can identify $\mathbb{Z}_q^n$ with R by

$$[a_0, a_1, \ldots, a_{n-1}] \mapsto \sum_{i=0}^{n-1} a_i x^i$$

Ring-LWE

- Better idea: use multiplication in polynomial ring
- Consider $R := \mathbb{Z}_q[x]/(x^n + 1)$ with $n = 2^k$
- Then can identify $\mathbb{Z}_q^n$ with R by

$$[a_0, a_1, \ldots, a_{n-1}] \mapsto \sum_{i=0}^{n-1} a_i x^i$$

- **Addition** is simply coordinate wise addition
- **Multiplication** is polynomial multiplication followed by reduction modulo $x^n + 1$

Ring-LWE

- Ring-LWE:
  - secret element $\mathbf{s} \in R$
  - elements $\mathbf{a}_i$ chosen randomly in R
  - coefficients noise polynomial $\mathbf{e}_i$ small independent normal variables

Ring-LWE

- Ring-LWE:
  - secret element $\mathbf{s} \in R$
  - elements $\mathbf{a}_i$ chosen randomly in R
  - coefficients noise polynomial $\mathbf{e}_i$ small independent normal variables
- **Search**: given many tuples $(\mathbf{a}_i, \mathbf{a}_i * \mathbf{s} + \mathbf{e}_i)$ recover $\mathbf{s}$

Ring-LWE

- Ring-LWE:
  - secret element $\mathbf{s} \in R$
  - elements $\mathbf{a}_i$ chosen randomly in R
  - coefficients noise polynomial $\mathbf{e}_i$ small independent normal variables
- **Search**: given many tuples $(\mathbf{a}_i, \mathbf{a}_i * \mathbf{s} + \mathbf{e}_i)$ recover $\mathbf{s}$
- **Decision**: given many tuples $(\mathbf{a}_i, \mathbf{b}_i) \in R^2$, decide whether there exists an $\mathbf{s} \in R$ and small $\mathbf{e}_i \in R$ such that

$$\mathbf{b}_i = \mathbf{a}_i * \mathbf{s} + \mathbf{e}_i$$

Conclusion

- LWE is building block for post-quantum cryptography
- Much more versatile than direct application of hard lattice problems
- Downside LWE: public keys are much larger than RSA/ECC
- Ring-LWE: more efficient but also more structure...
- Exact security level is still very much open ...