Introduction to modern lattice-based cryptography

Damien Stehlé

CNRS/Macquarie University/University of Sydney

Marseille, February 2010
Modern lattice-based cryptography

- **Cryptography**: the study of hiding information.
- “Lattice-based”: the schemes are described with lattices.
- Standard lattice problems provably reduce to attacks against those schemes.
- Modern: we won’t be interested in GGH and NTRU. More recent schemes offer similar asymptotic performance and comparable efficiency.
Modern lattice-based cryptography

- Cryptography: the study of hiding information.
- “Lattice-based”: the schemes are described with lattices.
- Standard lattice problems provably reduce to attacks against those schemes.
- Modern: we won’t be interested in GGH and NTRU. More recent schemes offer similar asymptotic performance and comparable efficiency.
Modern lattice-based cryptography

- Cryptography: the study of hiding information.
- “Lattice-based”: the schemes are described with lattices.
- Standard lattice problems provably reduce to attacks against those schemes.
- Modern: we won’t be interested in GGH and NTRU. More recent schemes offer similar asymptotic performance and comparable efficiency.
Modern lattice-based cryptography

- Cryptography: the study of hiding information.
- “Lattice-based”: the schemes are described with lattices.
- Standard lattice problems provably reduce to attacks against those schemes.
- Modern: we won’t be interested in GGH and NTRU. More recent schemes offer similar asymptotic performance and comparable efficiency.
Why lattice-based cryptography?

(why not factoring or discrete log, as usual?)

- LBC provides unmatched security properties: it relies on worst-case hardness assumptions and seems to resist against quantum computers.
- LBC is asymptotically extremely efficient.
- LBC is simple and flexible: this leads to easier design of complicated cryptographic functions.
- Diversity fosters cross-pollination.
Why lattice-based cryptography?

(why not factoring or discrete log, as usual?)

- LBC provides unmatched security properties: it relies on worst-case hardness assumptions and seems to resist against quantum computers.
- LBC is asymptotically extremely efficient.
- LBC is simple and flexible: this leads to easier design of complicated cryptographic functions.
- Diversity fosters cross-pollination.
Why lattice-based cryptography?

(why not factoring or discrete log, as usual?)

- LBC provides unmatched security properties: it relies on worst-case hardness assumptions and seems to resist against quantum computers.
- LBC is asymptotically extremely efficient.
- LBC is simple and flexible: this leads to easier design of complicated cryptographic functions.
- Diversity fosters cross-pollination.
Why lattice-based cryptography?

(why not factoring or discrete log, as usual?)

- LBC provides unmatched security properties: it relies on worst-case hardness assumptions and seems to resist against quantum computers.
- LBC is asymptotically extremely efficient.
- LBC is simple and flexible: this leads to easier design of complicated cryptographic functions.
- Diversity fosters cross-pollination.
Why lattice-based cryptography?

(why not factoring or discrete log, as usual?)

- LBC provides unmatched security properties: it relies on worst-case hardness assumptions and seems to resist against quantum computers.
- LBC is asymptotically extremely efficient.
- LBC is simple and flexible: this leads to easier design of complicated cryptographic functions.
- Diversity fosters cross-pollination.
Why lattice-based cryptography?

(why not factoring or discrete log, as usual?)

- LBC provides unmatched security properties: it relies on worst-case hardness assumptions and seems to resist against quantum computers.
- LBC is asymptotically extremely efficient.
- LBC is simple and flexible: this leads to easier design of complicated cryptographic functions.
- Diversity fosters cross-pollination.
Goal of this course

To give an overview of recent developments in lattice-based cryptography, and a flavour of the techniques/results.

Disclaimer: This is not a practical crypto course.

Contents: Complexity theory, distributions, quantum computing, cryptography, structured matrices, algebraic number theory, lattices.
Goal of this course

To give an overview of recent developments in lattice-based cryptography, and a flavour of the techniques/results.

Disclaimer: This is not a practical crypto course.

Contents: Complexity theory, distributions, quantum computing, cryptography, structured matrices, algebraic number theory, lattices.
Goal of this course

To give an overview of recent developments in lattice-based cryptography, and a flavour of the techniques/results.

Disclaimer: This is not a practical crypto course.

Contents: Complexity theory, distributions, quantum computing, cryptography, structured matrices, algebraic number theory, lattices.
Plan

1- Background on Euclidean lattices.
2- The SIS problem, or how to hash.
3- The LWE problem, or how to encrypt.
4- Cryptanalysis.
5- More recent developments.
Plan

1- Background on Euclidean lattices.
2- The SIS problem, or how to hash.
3- The LWE problem, or how to encrypt.
4- Cryptanalysis.
5- More recent developments.
Background on Euclidean lattices

a- Arbitrary lattices.
b- Ideal lattices.
c- Lattice Gaussians.
(Arbitrary) lattices

Lattice \equiv \text{ discrete subgroup of } \mathbb{R}^n \\
\equiv \{ \sum x_i b_i : x_i \in \mathbb{Z} \}

If the $b_i$’s are linearly independent, they are called a basis.

Hard pbs: short/close vectors.

Lattice minimum:
$\lambda(L) = \min (\|b\| : b \in L \setminus 0)$.
(Arbitrary) lattices

A lattice is a discrete subgroup of $\mathbb{R}^n$ defined as:

$$\text{Lattice} \equiv \{ \sum x_i b_i : x_i \in \mathbb{Z} \}$$

If the $b_i$'s are linearly independent, they are called a **basis**.

**Hard problems:** short/close vectors.

**Lattice minimum:**

$$\lambda(L) = \min (\|b\| : b \in L \setminus 0).$$
(Arbitrary) lattices

Lattice \equiv \text{discrete subgroup of } \mathbb{R}^n \\
\equiv \{ \sum x_i b_i : x_i \in \mathbb{Z} \}

If the $b_i$’s are linearly independent, they are called a basis.

Hard pbs: short/close vectors.

Lattice minimum:
$$\lambda(L) = \min (\|b\| : b \in L \setminus \{0\}).$$
(Arbitrary) lattices

Lattice $\equiv$ discrete subgroup of $\mathbb{R}^n$
$\equiv \{ \sum x_i b_i : x_i \in \mathbb{Z} \}$

If the $b_i$’s are linearly independent, they are called a basis.

Hard pbs: short/close vectors.

Lattice minimum:
$\lambda(L) = \min (\|b\| : b \in L \setminus 0)$. 
SVP and SIVP

The Shortest Vector Problem: \( \text{SVP}_\gamma \)

Given a basis of \( L \), find \( b \in L \setminus 0 \) such that:

\[
\|b\| \leq \gamma \cdot \min(\|c\| : c \in L \setminus 0).
\]
SVP and SIVP

The Shortest Vector Problem: $\text{SVP}_\gamma$

Given a basis of $L$, find $b \in L \setminus \{0\}$ such that:

$$\|b\| \leq \gamma \cdot \min(\|c\| : c \in L \setminus \{0\}).$$

The Shortest Independent Vectors Problem: $\text{SIVP}_\gamma$

Given a basis of $L$, find $b_1, \ldots, b_n \in L$ lin. indep. such that:

$$\max \|b_i\| \leq \gamma \cdot \min(\max \|c_i\| : c_1, \ldots, c_n \in L \text{ lin. indep.}).$$
The Shortest Vector Problem: $\text{SVP}_\gamma$

Given a basis of $L$, find $\mathbf{b} \in L \setminus \{0\}$ such that:

$$\|\mathbf{b}\| \leq \gamma \cdot \min(\|\mathbf{c}\| : \mathbf{c} \in L \setminus \{0\}).$$

The Shortest Independent Vectors Problem: $\text{SIVP}_\gamma$

Given a basis of $L$, find $\mathbf{b}_1, \ldots, \mathbf{b}_n \in L$ lin. indep. such that:

$$\max \|\mathbf{b}_i\| \leq \gamma \cdot \min(\max \|\mathbf{c}_i\| : \mathbf{c}_1, \ldots, \mathbf{c}_n \in L \text{ lin. indep.}).$$

- NP-hard when $\gamma = O(1)$.
- In lattice-based crypto: $\gamma = \mathcal{P}oly(n)$ (most often).
- Solvable in polynomial time when $\gamma = 2^{\tilde{O}(n)}$. 
Gram-Schmidt Orthogonalisation

- A lattice may have infinitely many bases.
- Quality of a basis: measured by the Gram-Schmidt Orth.
A lattice may have infinitely many bases.

Quality of a basis: measured by the Gram-Schmidt Orth.

\[ b^*_i = \arg\min \|b_i + \sum_{j<i} Rb_j\| \]

Quality measure: \[ \max_i \|b^*_i\| \].
Gram-Schmidt Orthogonalisation

- A lattice may have infinitely many bases.
- Quality of a basis: measured by the Gram-Schmidt Orth.

\[ b_i^* = \arg\min \| b_i + \sum_{j<i} R b_j \| \]

- Quality measure: \( \max_i \| b_i^* \| \).
From short vectors to a short basis

Let \((b_i)_i\) be a basis of a lattice \(L\).
Let \((s_i)_i\) in \(L\) be linearly independent with small GSO.
Can we compute a basis of \(L\) with small GSO?

Write \((s_i)_i = (b_i)_i \cdot T\), with \(T \in \mathbb{Z}^{n \times n}\).
Compute the Hermite Normal Form of \(T\), i.e., \(T = U \cdot T'\) with \(U\) unimodular and \(T' \in \mathbb{Z}^{n \times n}\) upper triangular.
Let \((c_i)_i = (b_i)_i \cdot U\).
\((c_i)_i\) is a basis of \(L\) and \((s_i)_i = (c_i)_i \cdot T'\).
Therefore \(\max ||c_i^*|| \leq \max ||s_i^*||\).
From short vectors to a short basis

- Let \((b_i)_i\) be a basis of a lattice \(L\).
- Let \((s_i)_i\) in \(L\) be linearly independent with small GSO.
- Can we compute a basis of \(L\) with small GSO?

- Write \((s_i)_i = (b_i)_i \cdot T\), with \(T \in \mathbb{Z}^{n \times n}\).
  - Compute the Hermite Normal Form of \(T\), i.e., \(T = U \cdot T'\) with \(U\) unimodular and \(T' \in \mathbb{Z}^{n \times n}\) upper triangular.
  - Let \((c_i)_i = (b_i)_i \cdot U\).
  - \((c_i)_i\) is a basis of \(L\) and \((s_i)_i = (c_i)_i \cdot T'\).
  - Therefore \(\max \|c_i^*\| \leq \max \|s_i^*\|\).
From short vectors to a short basis

- Let \((b_i)_i\) be a basis of a lattice \(L\).
- Let \((s_i)_i\) in \(L\) be linearly independent with small GSO.
- Can we compute a basis of \(L\) with small GSO?

- Write \((s_i)_i = (b_i)_i \cdot T\), with \(T \in \mathbb{Z}^{n \times n}\).
- Compute the Hermite Normal Form of \(T\), i.e., \(T = U \cdot T'\) with \(U\) unimodular and \(T' \in \mathbb{Z}^{n \times n}\) upper triangular.
  Let \((c_i)_i = (b_i)_i \cdot U\).

- \((c_i)_i\) is a basis of \(L\) and \((s_i)_i = (c_i)_i \cdot T'\).
- Therefore \(\max \|c_i^*\| \leq \max \|s_i^*\|\).
From short vectors to a short basis

- Let \((b_i)_i\) be a basis of a lattice \(L\).
- Let \((s_i)_i\) in \(L\) be linearly independent with small GSO.
- Can we compute a basis of \(L\) with small GSO?

- Write \((s_i)_i = (b_i)_i \cdot T\), with \(T \in \mathbb{Z}^{n \times n}\).
- Compute the Hermite Normal Form of \(T\), i.e., \(T = U \cdot T'\) with \(U\) unimodular and \(T' \in \mathbb{Z}^{n \times n}\) upper triangular.
- Let \((c_i)_i = (b_i)_i \cdot U\).
- \((c_i)_i\) is a basis of \(L\) and \((s_i)_i = (c_i)_i \cdot T'\).

Therefore \(\max \|c_i^*\| \leq \max \|s_i^*\|\).
From short vectors to a short basis

- Let \((b_i)_i\) be a basis of a lattice \(L\).
- Let \((s_i)_i\) in \(L\) be linearly independent with small GSO.
- Can we compute a basis of \(L\) with small GSO?

- Write \((s_i)_i = (b_i)_i \cdot T\), with \(T \in \mathbb{Z}^{n \times n}\).
- Compute the Hermite Normal Form of \(T\), i.e., \(T = U \cdot T'\) with \(U\) unimodular and \(T' \in \mathbb{Z}^{n \times n}\) upper triangular.
- Let \((c_i)_i = (b_i)_i \cdot U\).
- \((c_i)_i\) is a basis of \(L\) and \((s_i)_i = (c_i)_i \cdot T'\).
- Therefore \(\max \|c_i^*\| \leq \max \|s_i^*\|\).
From short vectors to a short basis

- Let \((b_i)_i\) be a basis of a lattice \(L\).
- Let \((s_i)_i\) in \(L\) be linearly independent with small GSO.
- Can we compute a basis of \(L\) with small GSO?

- Write \((s_i)_i = (b_i)_i \cdot T\), with \(T \in \mathbb{Z}^{n \times n}\).
- Compute the Hermite Normal Form of \(T\), i.e., \(T = U \cdot T'\) with \(U\) unimodular and \(T' \in \mathbb{Z}^{n \times n}\) upper triangular.
- Let \((c_i)_i = (b_i)_i \cdot U\).
- \((c_i)_i\) is a basis of \(L\) and \((s_i)_i = (c_i)_i \cdot T'\).
- Therefore \(\max \|c_i^*\| \leq \max \|s_i^*\|\).

With a size-reduction, we can get \(\max \|c_i\| \leq \sqrt{n} \cdot \max \|s_i\|\).
Background on lattices

- **a-** Arbitrary lattices.
- **b-** Ideal lattices.
- **c-** Lattice Gaussians.
A lattice $L$ is **ideal** if:

\[
\begin{pmatrix}
 b_0 & b_1 & b_2 & b_3 & \ldots & b_{n-2} & b_{n-1} \\
\end{pmatrix} \in L \\
\Rightarrow \begin{pmatrix}
 -b_{n-1} & b_0 & b_1 & b_2 & \ldots & b_{n-3} & b_{n-2} \\
\end{pmatrix} \in L \\
\Rightarrow \begin{pmatrix}
 -b_{n-2} & -b_{n-1} & b_0 & b_1 & \ldots & b_{n-4} & b_{n-3} \\
\end{pmatrix} \in L \\
\Rightarrow \begin{pmatrix}
 -b_{n-3} & -b_{n-2} & -b_{n-1} & b_0 & \ldots & b_{n-5} & b_{n-4} \\
\end{pmatrix} \in L
\]
A lattice $L$ is **ideal** if:

$$
\begin{pmatrix}
 b_0 & b_1 & b_2 & b_3 & \ldots & b_{n-2} & b_{n-1} \\
 -b_{n-1} & b_0 & b_1 & b_2 & \ldots & b_{n-3} & b_{n-2} \\
 -b_{n-2} & -b_{n-1} & b_0 & b_1 & \ldots & b_{n-4} & b_{n-3} \\
 -b_{n-3} & -b_{n-2} & -b_{n-1} & b_0 & \ldots & b_{n-5} & b_{n-4}
\end{pmatrix}
\in L
$$
A lattice $L$ is **ideal** if:

$$\begin{pmatrix} b_0 & b_1 & b_2 & b_3 & \ldots & b_{n-2} & b_{n-1} \\ -b_{n-1} & b_0 & b_1 & b_2 & \ldots & b_{n-3} & b_{n-2} \\ -b_{n-2} & -b_{n-1} & b_0 & b_1 & \ldots & b_{n-4} & b_{n-3} \\ -b_{n-3} & -b_{n-2} & -b_{n-1} & b_0 & \ldots & b_{n-5} & b_{n-4} \end{pmatrix} \in L$$
A lattice $L$ is **ideal** if:

\[
\begin{pmatrix}
  b_0 & b_1 & b_2 & b_3 & \ldots & b_{n-2} & b_{n-1}
\end{pmatrix} \in L
\]

\[\Rightarrow\]

\[
\begin{pmatrix}
  -b_{n-1} & b_0 & b_1 & b_2 & \ldots & b_{n-3} & b_{n-2}
\end{pmatrix} \in L
\]

\[\Rightarrow\]

\[
\begin{pmatrix}
  -b_{n-2} & -b_{n-1} & b_0 & b_1 & \ldots & b_{n-4} & b_{n-3}
\end{pmatrix} \in L
\]

\[\Rightarrow\]

\[
\begin{pmatrix}
  -b_{n-3} & -b_{n-2} & -b_{n-1} & b_0 & \ldots & b_{n-5} & b_{n-4}
\end{pmatrix} \in L
\]
A lattice $L$ is **ideal** if:

\[
\begin{pmatrix}
  b_0 & b_1 & b_2 & b_3 & \ldots & b_{n-2} & b_{n-1}
\end{pmatrix}
\in L
\]

\[
\Rightarrow
\begin{pmatrix}
  -b_{n-1} & b_0 & b_1 & b_2 & \ldots & b_{n-3} & b_{n-2}
\end{pmatrix}
\in L
\]

\[
\Rightarrow
\begin{pmatrix}
  -b_{n-2} & -b_{n-1} & b_0 & b_1 & \ldots & b_{n-4} & b_{n-3}
\end{pmatrix}
\in L
\]

\[
\Rightarrow
\begin{pmatrix}
  -b_{n-3} & -b_{n-2} & -b_{n-1} & b_0 & \ldots & b_{n-5} & b_{n-4}
\end{pmatrix}
\in L
\]

A lattice $L$ is **ideal** if it is an ideal of $\mathbb{Z}[x]/(x^n + 1)$. 

---

**Ideal lattices**

A lattice $L$ is ideal if

\[
\begin{pmatrix}
  b_0 & b_1 & b_2 & b_3 & \ldots & b_{n-2} & b_{n-1}
\end{pmatrix}
\in L
\]

\[
\Rightarrow
\begin{pmatrix}
  -b_{n-1} & b_0 & b_1 & b_2 & \ldots & b_{n-3} & b_{n-2}
\end{pmatrix}
\in L
\]

\[
\Rightarrow
\begin{pmatrix}
  -b_{n-2} & -b_{n-1} & b_0 & b_1 & \ldots & b_{n-4} & b_{n-3}
\end{pmatrix}
\in L
\]

\[
\Rightarrow
\begin{pmatrix}
  -b_{n-3} & -b_{n-2} & -b_{n-1} & b_0 & \ldots & b_{n-5} & b_{n-4}
\end{pmatrix}
\in L
\]

A lattice $L$ is ideal if it is an ideal of $\mathbb{Z}[x]/(x^n + 1)$. 

---

**Ideal lattices**

A lattice $L$ is **ideal** if:

\[
\begin{pmatrix}
  b_0 & b_1 & b_2 & b_3 & \ldots & b_{n-2} & b_{n-1}
\end{pmatrix}
\in L
\]

\[
\Rightarrow
\begin{pmatrix}
  -b_{n-1} & b_0 & b_1 & b_2 & \ldots & b_{n-3} & b_{n-2}
\end{pmatrix}
\in L
\]

\[
\Rightarrow
\begin{pmatrix}
  -b_{n-2} & -b_{n-1} & b_0 & b_1 & \ldots & b_{n-4} & b_{n-3}
\end{pmatrix}
\in L
\]

\[
\Rightarrow
\begin{pmatrix}
  -b_{n-3} & -b_{n-2} & -b_{n-1} & b_0 & \ldots & b_{n-5} & b_{n-4}
\end{pmatrix}
\in L
\]

A lattice $L$ is **ideal** if it is an ideal of $\mathbb{Z}[x]/(x^n + 1)$. 

---

**Ideal lattices**

A lattice $L$ is ideal if:

\[
\begin{pmatrix}
  b_0 & b_1 & b_2 & b_3 & \ldots & b_{n-2} & b_{n-1}
\end{pmatrix}
\in L
\]

\[
\Rightarrow
\begin{pmatrix}
  -b_{n-1} & b_0 & b_1 & b_2 & \ldots & b_{n-3} & b_{n-2}
\end{pmatrix}
\in L
\]

\[
\Rightarrow
\begin{pmatrix}
  -b_{n-2} & -b_{n-1} & b_0 & b_1 & \ldots & b_{n-4} & b_{n-3}
\end{pmatrix}
\in L
\]

\[
\Rightarrow
\begin{pmatrix}
  -b_{n-3} & -b_{n-2} & -b_{n-1} & b_0 & \ldots & b_{n-5} & b_{n-4}
\end{pmatrix}
\in L
\]

A lattice $L$ is **ideal** if it is an ideal of $\mathbb{Z}[x]/(x^n + 1)$. 

---

**Ideal lattices**

A lattice $L$ is ideal if:

\[
\begin{pmatrix}
  b_0 & b_1 & b_2 & b_3 & \ldots & b_{n-2} & b_{n-1}
\end{pmatrix}
\in L
\]

\[
\Rightarrow
\begin{pmatrix}
  -b_{n-1} & b_0 & b_1 & b_2 & \ldots & b_{n-3} & b_{n-2}
\end{pmatrix}
\in L
\]

\[
\Rightarrow
\begin{pmatrix}
  -b_{n-2} & -b_{n-1} & b_0 & b_1 & \ldots & b_{n-4} & b_{n-3}
\end{pmatrix}
\in L
\]

\[
\Rightarrow
\begin{pmatrix}
  -b_{n-3} & -b_{n-2} & -b_{n-1} & b_0 & \ldots & b_{n-5} & b_{n-4}
\end{pmatrix}
\in L
\]

A lattice $L$ is **ideal** if it is an ideal of $\mathbb{Z}[x]/(x^n + 1)$. 

---

**Ideal lattices**

A lattice $L$ is ideal if:

\[
\begin{pmatrix}
  b_0 & b_1 & b_2 & b_3 & \ldots & b_{n-2} & b_{n-1}
\end{pmatrix}
\in L
\]

\[
\Rightarrow
\begin{pmatrix}
  -b_{n-1} & b_0 & b_1 & b_2 & \ldots & b_{n-3} & b_{n-2}
\end{pmatrix}
\in L
\]

\[
\Rightarrow
\begin{pmatrix}
  -b_{n-2} & -b_{n-1} & b_0 & b_1 & \ldots & b_{n-4} & b_{n-3}
\end{pmatrix}
\in L
\]

\[
\Rightarrow
\begin{pmatrix}
  -b_{n-3} & -b_{n-2} & -b_{n-1} & b_0 & \ldots & b_{n-5} & b_{n-4}
\end{pmatrix}
\in L
\]

A lattice $L$ is **ideal** if it is an ideal of $\mathbb{Z}[x]/(x^n + 1)$. 

---

**Ideal lattices**

A lattice $L$ is ideal if:

\[
\begin{pmatrix}
  b_0 & b_1 & b_2 & b_3 & \ldots & b_{n-2} & b_{n-1}
\end{pmatrix}
\in L
\]

\[
\Rightarrow
\begin{pmatrix}
  -b_{n-1} & b_0 & b_1 & b_2 & \ldots & b_{n-3} & b_{n-2}
\end{pmatrix}
\in L
\]

\[
\Rightarrow
\begin{pmatrix}
  -b_{n-2} & -b_{n-1} & b_0 & b_1 & \ldots & b_{n-4} & b_{n-3}
\end{pmatrix}
\in L
\]

\[
\Rightarrow
\begin{pmatrix}
  -b_{n-3} & -b_{n-2} & -b_{n-1} & b_0 & \ldots & b_{n-5} & b_{n-4}
\end{pmatrix}
\in L
\]

A lattice $L$ is **ideal** if it is an ideal of $\mathbb{Z}[x]/(x^n + 1)$.
A lattice $L$ is **ideal** if:

$\begin{pmatrix}
b_0 & b_1 & b_2 & b_3 & \ldots & b_{n-2} & b_{n-1}
\end{pmatrix} \in L$

$\Rightarrow$

$\begin{pmatrix}
-b_{n-1} & b_0 & b_1 & b_2 & \ldots & b_{n-3} & b_{n-2}
\end{pmatrix} \in L$

$\Rightarrow$

$\begin{pmatrix}
-b_{n-2} & -b_{n-1} & b_0 & b_1 & \ldots & b_{n-4} & b_{n-3}
\end{pmatrix} \in L$

$\Rightarrow$

$\begin{pmatrix}
-b_{n-3} & -b_{n-2} & -b_{n-1} & b_0 & \ldots & b_{n-5} & b_{n-4}
\end{pmatrix} \in L$

A lattice $L$ is **ideal** if it is an ideal of $\mathbb{Z}[x]/(x^n + 1)$.

Easy property: all minima of an ideal lattice are equal.

$$\lambda_k(L) = \min(r : \dim \text{span}(L \cap B(r)) \geq k).$$
How special are ideal lattices?

Advantages

- The negacyclic structure allows one to save space.
  Warning: an ideal lattice may have no negacyclic basis.
- We can multiply vectors together.
- Fast polynomial arithmetic.
How special are ideal lattices?

**Advantages**
- The negacyclic structure allows one to save space.  
  Warning: an ideal lattice may have no negacyclic basis.
- We can multiply vectors together.
- Fast polynomial arithmetic.

**Drawbacks**
- NP-hardness results not valid anymore.
- Decisional SVP becomes easier (algebraic number theory).
How special are ideal lattices?

**Advantages**

- The negacyclic structure allows one to save space.
  Warning: an ideal lattice may have no negacyclic basis.
- We can multiply vectors together.
- Fast polynomial arithmetic.

**Drawbacks**

- NP-hardness results not valid anymore.
- Decisional SVP becomes easier (algebraic number theory).

But no known computational advantage for Id-SVP/Id-SIVP.
Ideal lattices and algebraic number theory

Let $\zeta$ be a primitive $(2n)$-th root of unity and $K = \mathbb{Q}(\zeta) \approx \mathbb{Q}[x]/(x^n + 1)$.

- $K$ is a cyclotomic number field with $n$ canonical embeddings $\sigma_i : K \to \mathbb{C}$.
- For $x \in K$: $T_2(x)^2 := \sum |\sigma_i(x)|^2$ and $N x := \prod |\sigma_i(x)|$.
- The ring of integers $\mathcal{O}_K$ of $K$ is the set of algebraic integers belonging to $K$. Here, it is $\mathbb{Z}[x]/(x^n + 1)$.
- An ideal lattice $L$ is an ideal of $\mathbb{Z}[x]/(x^n + 1)$ and thus an integral ideal of $K$, i.e., an ideal of $K$ contained in $\mathcal{O}_K$. We define $N(L) = [L : \mathcal{O}_K] = \det(L)$. 

Damien Stehlé
Ideal lattices and algebraic number theory

- Let $\zeta$ be a primitive $(2n)$-th root of unity and $K = \mathbb{Q}(\zeta) \cong \mathbb{Q}[x]/(x^n + 1)$.
- $K$ is a cyclotomic number field with $n$ canonical embeddings $\sigma_i : K \rightarrow \mathbb{C}$.
- For $x \in K$: $T_2(x)^2 := \sum |\sigma_i(x)|^2$ and $N x := \prod |\sigma_i(x)|$.
- The ring of integers $\mathcal{O}_K$ of $K$ is the set of algebraic integers belonging to $K$. Here, it is $\mathbb{Z}[x]/(x^n + 1)$.
- An ideal lattice $L$ is an ideal of $\mathbb{Z}[x]/(x^n + 1)$ and thus an integral ideal of $K$, i.e., an ideal of $K$ contained in $\mathcal{O}_K$. We define $N(L) = [L : \mathcal{O}_K] = \det(L)$. 

Damien Stehlé

Introduction to modern lattice-based cryptography
Let $\zeta$ be a primitive $(2n)$-th root of unity and $K = \mathbb{Q}(\zeta) \cong \mathbb{Q}[x]/(x^n + 1)$.

$K$ is a cyclo
tomic number field with $n$ canonical embeddings $\sigma_i : K \to \mathbb{C}$.

For $x \in K$: $T_2(x)^2 := \sum |\sigma_i(x)|^2$ and $N x := \prod |\sigma_i(x)|$.

The ring of integers $\mathcal{O}_K$ of $K$ is the set of algebraic integers belonging to $K$. Here, it is $\mathbb{Z}[x]/(x^n + 1)$.

An ideal lattice $L$ is an ideal of $\mathbb{Z}[x]/(x^n + 1)$ and thus an integral ideal of $K$, i.e., an ideal of $K$ contained in $\mathcal{O}_K$. We define $N(L) = [L : \mathcal{O}_K] = \det(L)$. 

Ideal lattices and algebraic number theory
Ideal lattices and algebraic number theory

- Let $\zeta$ be a primitive $(2n)$-th root of unity and $K = \mathbb{Q}(\zeta) \approx \mathbb{Q}[x]/(x^n + 1)$.
- $K$ is a **cyclo
tom
c field** with $n$ canonical embeddings $\sigma_i : K \to \mathbb{C}$.
- For $x \in K$: $T_2(x)^2 := \sum |\sigma_i(x)|^2$ and $N x := \prod |\sigma_i(x)|$.
- The **ring of integers** $\mathcal{O}_K$ of $K$ is the set of algebraic integers belonging to $K$. Here, it is $\mathbb{Z}[x]/(x^n + 1)$.
- An ideal lattice $L$ is an ideal of $\mathbb{Z}[x]/(x^n + 1)$ and thus an integral ideal of $K$, i.e., an ideal of $K$ contained in $\mathcal{O}_K$.
- We define $N(L) = [L : \mathcal{O}_K] = \det(L)$.
Let \( \zeta \) be a primitive \((2n)\)-th root of unity and \( K = \mathbb{Q}(\zeta) \cong \mathbb{Q}[x]/(x^n + 1) \).

\( K \) is a cyclotomic number field with \( n \) canonical embeddings \( \sigma_i : K \rightarrow \mathbb{C} \).

For \( x \in K \): \( T_2(x)^2 := \sum |\sigma_i(x)|^2 \) and \( \mathcal{N} x := \prod |\sigma_i(x)| \).

The ring of integers \( \mathcal{O}_K \) of \( K \) is the set of algebraic integers belonging to \( K \). Here, it is \( \mathbb{Z}[x]/(x^n + 1) \).

An ideal lattice \( L \) is an ideal of \( \mathbb{Z}[x]/(x^n + 1) \) and thus an integral ideal of \( K \), i.e., an ideal of \( K \) contained in \( \mathcal{O}_K \). We define \( \mathcal{N}(L) = [L : \mathcal{O}_K] = \det(L) \).
Approximating $\text{Id-SVP}$ is easy

- The coefficient norm (in $\mathbb{Z}[x]/(x^n + 1)$) is a scaling of factor $\sqrt{n}$ of the $T_2$-norm.
- For any $x \in K$, we have $(\mathcal{N} x)^{2/n} \leq T_2(x)^{2/n}$.
- For any $x \in L$, we have $(x) \subseteq L$, and thus $\mathcal{N} x = \mathcal{N}(x) \geq \mathcal{N}(L)$. This gives:
  \[
  \sqrt{n} \cdot \mathcal{N}(L)^{1/n} \leq \sqrt{n} \cdot \mathcal{N}(x)^{1/n} \leq T_2(x).
  \]
- Let $s \in L$ reaching $\lambda(L)$. Minkowski’s theorem gives:
  \[
  T_2(s) \leq \sqrt{n} \cdot \text{vol}\,T_2(L)^{1/n} \leq n \cdot \mathcal{N}(L)^{1/n}.
  \]
- Overall, we know $\lambda(L)$ up to a factor $\sqrt{n}$. 

\[\text{ Damien Stehlé  
 Introduction to modern lattice-based cryptography  
 01/02/2010  
 16/81}\]
Approximating \text{Id-SVP} is easy

- The coefficient norm (in $\mathbb{Z}[x]/(x^n + 1)$) is a scaling of factor $\sqrt{n}$ of the $T_2$-norm.

- For any $x \in K$, we have $(N(x))^{2/n} \leq T_2(x)^{2/n}$.

  For any $x \in L$, we have $(x) \subseteq L$, and thus $N(x) = (x) \geq N(L)$. This gives:

  \[ \sqrt{n} \cdot N(L)^{1/n} \leq \sqrt{n} \cdot (x)^{1/n} \leq T_2(x). \]

- Let $s \in L$ reaching $\lambda(L)$. Minkowski’s theorem gives:

  \[ T_2(s) \leq \sqrt{n} \cdot \text{vol} \cdot T_2(L)^{1/n} \leq n \cdot N(L)^{1/n}. \]

- Overall, we know $\lambda(L)$ up to a factor $\sqrt{n}$. 
Approximating \textit{Id-SVP} is easy

- The coefficient norm (in $\mathbb{Z}[x]/(x^n + 1)$) is a scaling of factor $\sqrt{n}$ of the $T_2$-norm.
- For any $x \in K$, we have $(N x)^{2/n} \leq T_2(x)^{2/n}$.
- For any $x \in L$, we have $(x) \subseteq L$, and thus $N x = N(x) \geq N(L)$. This gives:

\[
\sqrt{n} \cdot N(L)^{1/n} \leq N(x)^{1/n} \leq T_2(x).
\]

- Let $s \in L$ reaching $\lambda(L)$. Minkowski’s theorem gives:

\[
T_2(s) \leq \sqrt{n} \cdot \vol_{T_2}(L)^{1/n} \leq n \cdot N(L)^{1/n}.
\]

- Overall, we know $\lambda(L)$ up to a factor $\sqrt{n}$. 
Approximating \text{Id-SVP} is easy

- The coefficient norm (in \( \mathbb{Z}[x]/(x^n + 1) \)) is a scaling of factor \( \sqrt{n} \) of the \( T_2 \)-norm.
- For any \( x \in K \), we have \( (N x)^{2/n} \leq T_2(x)^{2/n} \).
- For any \( x \in L \), we have \( (x) \subseteq L \), and thus \( \sqrt{n} \leq \sqrt{n} \leq \sqrt{n} \leq \sqrt{n} \).
- Let \( s \in L \) reaching \( \lambda(L) \). Minkowski’s theorem gives:
  \[
  T_2(s) \leq \sqrt{n} \cdot vol_{T_2}(L)^{1/n} \leq n \cdot N(L)^{1/n}.
  \]
- Overall, we know \( \lambda(L) \) up to a factor \( \sqrt{n} \).
Approximating $\text{Id-SVP}$ is easy

- The coefficient norm (in $\mathbb{Z}[x]/(x^n + 1)$) is a scaling of factor $\sqrt{n}$ of the $T_2$-norm.
- For any $x \in K$, we have $(N(x)^{2/n} \leq T_2(x)^{2/n}.$
- For any $x \in L$, we have $(x) \subseteq L$, and thus $N(x) = N(x) \geq N(L).$ This gives:

$$\sqrt{n} \cdot N(L)^{1/n} \leq \sqrt{n} \cdot N(x)^{1/n} \leq T_2(x).$$

- Let $s \in L$ reaching $\lambda(L).$ Minkowski’s theorem gives:

$$T_2(s) \leq \sqrt{n} \cdot \text{vol}_{T_2}(L)^{1/n} \leq n \cdot N(L)^{1/n}.$$

- Overall, we know $\lambda(L)$ up to a factor $\sqrt{n}.$
Background on lattices

a- Arbitrary lattices.
b- Ideal lattices.
c- Lattice Gaussians.
A handy distribution: the discrete Gaussian
A handy distribution: the discrete Gaussian

For $\mathbf{b} \in \mathbb{R}^n$ and $\mathbf{c} \in \mathbb{R}^n$:

$$\rho_{\sigma,\mathbf{c}}(\mathbf{b}) := e^{-\pi \frac{\|\mathbf{b} - \mathbf{c}\|^2}{\sigma^2}}.$$  

$\sigma$ is the standard deviation.
A handy distribution: the discrete Gaussian

For $\mathbf{b} \in \mathbb{R}^n$ and $\mathbf{c} \in \mathbb{R}^n$:

$$\rho_{\sigma, \mathbf{c}}(\mathbf{b}) := e^{-\pi \frac{\|\mathbf{b} - \mathbf{c}\|^2}{\sigma^2}}.$$

$\sigma$ is the standard deviation.

For $L \subseteq \mathbb{R}^n$ and $\mathbf{c} \in \mathbb{R}^n$: $\rho_{\sigma, \mathbf{c}}(L) = \sum_{\mathbf{b} \in L} \rho_{\sigma, \mathbf{c}}(\mathbf{b})$ is finite.

Discrete $n$-dimensional Gaussian:

$$\forall \mathbf{b} \in L : D_{L, \sigma, \mathbf{c}}(\mathbf{b}) = \frac{\rho_{\sigma, \mathbf{c}}(\mathbf{b})}{\rho_{\sigma, \mathbf{c}}(L)}.$$
The Poisson Summation Formula (PSF)

Dual lattice:

- If $L \subseteq \mathbb{R}^n$ is full rank, its **dual** is

$$\hat{L} = \left\{ \hat{b} : \forall b \in L, \langle \hat{b}, b \rangle \in \mathbb{Z} \right\}.$$

- If $B \in \mathbb{R}^{n \times n}$ is a basis of $L$, then $B^{-T}$ is a basis of $\hat{L}$.
The Poisson Summation Formula (PSF)

Dual lattice:

- If $L \subseteq \mathbb{R}^n$ is full rank, its dual is

$$\hat{L} = \left\{ \hat{b} : \forall b \in L, \langle \hat{b}, b \rangle \in \mathbb{Z} \right\}.$$ 

- If $B \in \mathbb{R}^{n \times n}$ is a basis of $L$, then $B^{-T}$ is a basis of $\hat{L}$.

Poisson summation formula for $n$-dimensional Gaussians (derived from Fourier analysis):

$$\rho_{\sigma, c}(L) = \det(\hat{L}) \cdot \sigma^n \cdot \sum_{\hat{b} \in \hat{L}} \left[ \rho_{1/\sigma}(\hat{b}) \cdot \exp(-2\pi i \langle \hat{b}, c \rangle) \right].$$
The Poisson Summation Formula (PSF)

**Dual lattice:**
- If \( L \subseteq \mathbb{R}^n \) is full rank, its dual is
  \[
  \hat{L} = \left\{ \hat{b} : \forall b \in L, \langle \hat{b}, b \rangle \in \mathbb{Z} \right\}.
  \]
- If \( B \in \mathbb{R}^{n \times n} \) is a basis of \( L \), then \( B^{-T} \) is a basis of \( \hat{L} \).

**Poisson summation formula** for \( n \)-dimensional Gaussians (derived from Fourier analysis):

\[
\rho_{\sigma, c}(L) = \det(\hat{L}) \cdot \sigma^n \cdot \sum_{\hat{b} \in \hat{L}} \left[ \rho_{1/\sigma}(\hat{b}) \cdot \exp(-2\pi i \langle \hat{b}, c \rangle) \right].
\]

Consequence: \( \forall \sigma \geq 1 : \rho_\sigma(L \setminus B(0, \sigma \sqrt{n})) \leq 2^{-n+1} \rho_\sigma(L) \).
The smoothing parameter

- Define $\eta_\varepsilon(L)$ as the smallest $\sigma$ such that $\rho_{1/\sigma}(\hat{L} \setminus \mathbf{0}) \leq \varepsilon$.
- If $\sigma \geq \eta_\varepsilon(L)$, then $\rho_{\sigma,c}(L)$ is quasi-constant.

\[
\rho_{\sigma,c}(L) \in \left[(1 - \varepsilon) \cdot \det(\hat{L}) \cdot \sigma^n, (1 + \varepsilon) \cdot \det(\hat{L}) \cdot \sigma^n\right].
\]

- If $(\mathbf{b}_i)_i$ is a basis of $L$, we have:

\[
\eta_\varepsilon(L) \leq \max \|\mathbf{b}_i^*\| \cdot \sqrt{\log(3n/\varepsilon)}.
\]
The smoothing parameter

- Define $\eta_\varepsilon(L)$ as the smallest $\sigma$ such that $\rho_{1/\sigma}(\hat{L} \setminus \mathbf{0}) \leq \varepsilon$.
- If $\sigma \geq \eta_\varepsilon(L)$, then $\rho_{\sigma,c}(L)$ is quasi-constant.

$$\rho_{\sigma,c}(L) \in \left[ (1 - \varepsilon) \cdot \det(\hat{L}) \cdot \sigma^n, (1 + \varepsilon) \cdot \det(\hat{L}) \cdot \sigma^n \right].$$

- If $(b_i)_i$ is a basis of $L$, we have:

$$\eta_\varepsilon(L) \leq \max \|b_i^*\| \cdot \sqrt{\log(3n/\varepsilon)}.$$
The smoothing parameter

- Define $\eta_\varepsilon(L)$ as the smallest $\sigma$ such that $\rho_{1/\sigma}(\hat{L} \setminus \mathbf{0}) \leq \varepsilon$.
- If $\sigma \geq \eta_\varepsilon(L)$, then $\rho_{\sigma,c}(L)$ is quasi-constant.

$$\rho_{\sigma,c}(L) \in \left[(1 - \varepsilon) \cdot \det(\hat{L}) \cdot \sigma^n, (1 + \varepsilon) \cdot \det(\hat{L}) \cdot \sigma^n\right].$$

- If $(\mathbf{b}_i)_i$ is a basis of $L$, we have:

$$\eta_\varepsilon(L) \leq \max \|\mathbf{b}_i^*\| \cdot \sqrt{\log(3n/\varepsilon)}.$$
The smoothing parameter

- Define $\eta_\varepsilon(L)$ as the smallest $\sigma$ such that $\rho_{1/\sigma}(\hat{L} \setminus \mathbf{0}) \leq \varepsilon$.
- If $\sigma \geq \eta_\varepsilon(L)$, then $\rho_{\sigma,c}(L)$ is quasi-constant.

$$\rho_{\sigma,c}(L) \in \left[(1 - \varepsilon) \cdot \det(\hat{L}) \cdot \sigma^n, (1 + \varepsilon) \cdot \det(\hat{L}) \cdot \sigma^n\right].$$

- If $(\mathbf{b}_i)_i$ is a basis of $L$, we have:

$$\eta_\varepsilon(L) \leq \max \|\mathbf{b}_i^*\| \cdot \sqrt{\log(3n/\varepsilon)}.$$ 

Typically, we will use $\varepsilon = 2^{-n}$. 

$$\rho_{\sigma,c}(L) \in \left[(1 - \varepsilon) \cdot \det(\hat{L}) \cdot \sigma^n, (1 + \varepsilon) \cdot \det(\hat{L}) \cdot \sigma^n\right].$$
Proof that $\eta_{2^{-n+2}}(L) \leq \sqrt{n} \cdot \max \| b^*_i \|

- First: $\eta_{2^{-n+2}}(L) \leq \sqrt{n} \lambda(\hat{L})$.

$$\rho_{1/\sigma}(\hat{L} \setminus 0) = \rho(\sigma \hat{L} \setminus B(0, \sqrt{n})) \leq 2^{-n+1} \rho(\sigma \hat{L}) \leq 2^{-n+2}.$$ 

- Second: $1/\lambda(\hat{L}) \leq \max \| b^*_i \|$. 

Recall that $B' = B^{\perp}$ is a basis of $\hat{L}$. We have:

$$\lambda(\hat{L}) = \min \| b' \| \text{ and } \frac{1}{\min \| b' \|} = \max \| b^*_i \|. $$
Proof that $\eta_{2^{-n+2}}(L) \leq \sqrt{n} \cdot \max \|b_i^*\|

- First: $\eta_{2^{-n+2}}(L) \leq \sqrt{n}/\lambda(\hat{L})$.
  
  $$\rho_{1/\sigma}(\hat{L} \setminus \mathbf{0}) = \rho(\sigma \hat{L} \setminus B(\mathbf{0}, \sqrt{n})) \leq 2^{n+1} \rho(\sigma \hat{L}) \leq 2^{-n+2}.$$ 

- Second: $1/\lambda(\hat{L}) \leq \max \|b_i^*\|$.
  
  Recall that $B' = B^{-T}$ is a basis of $\hat{L}$. We have:
  
  $$\lambda(\hat{L}) \geq \min \|b_{i*}''\| \quad \text{and} \quad \frac{1}{\min \|b_{i*}''\|} = \max \|b_{i*}\|.$$
Proof that $\eta_{2^{-n+2}}(L) \leq \sqrt{n} \cdot \max \|b_i^*\|$ 

- First: $\eta_{2^{-n+2}}(L) \leq \sqrt{n}/\lambda(\hat{L})$.

  \[
  \rho_{1/\sigma}(\hat{L} \setminus \mathbf{0}) = \rho(\sigma\hat{L} \setminus \mathcal{B}(\mathbf{0}, \sqrt{n})) \leq 2^{-n+1} \rho(\sigma\hat{L}) \leq 2^{-n+2}.
  \]

- Second: $1/\lambda(\hat{L}) \leq \max \|b_i^*\|$.

  Recall that $B' = B^{-T}$ is a basis of $\hat{L}$. We have:

  \[
  \lambda(\hat{L}) \geq \min \|b'_i^*\| \text{ and } \frac{1}{\min \|b'_i^*\|} = \max \|b_i^*\|.
  \]
Proof that $\eta_{2^{-n+2}}(L) \leq \sqrt{n} \cdot \max \|b_i^*\|

- First: $\eta_{2^{-n+2}}(L) \leq \sqrt{n}/\lambda(\hat{L})$.
  
  \[
  \rho_{1/\sigma}(\hat{L} \setminus 0) = \rho(\sigma \hat{L} \setminus B(0, \sqrt{n})) \leq 2^{-n+1} \rho(\sigma \hat{L}) \leq 2^{-n+2}.
  \]

- Second: $1/\lambda(\hat{L}) \leq \max \|b_i^*\|$. Recall that $B' = B^{-T}$ is a basis of $\hat{L}$. We have:
  
  \[
  \lambda(\hat{L}) \geq \min \|b'_i^*\| \quad \text{and} \quad \frac{1}{\min \|b'_i^*\|} = \max \|b_i^*\|.
  \]
Sampling according to $D_{L,\sigma}$

**Input:** A basis $(b_i)_i$ of $L$, $\sigma$.

**Output:** $b \in L$.

1. $b := 0$. For $i$ from $n$ to 1, do
2. $\sigma_i := \sigma/\|b_i^*\|$, $c_i := -\langle b, b_i^* \rangle / \|b_i^*\|^2$;
3. Sample $z_i$ from $D_{\mathbb{Z},\sigma_i,c_i}$;
4. $b := b + z_i b_i$.
5. Return $b$. 
Sampling according to $D_{L,\sigma}$

**Input:** A basis $(b_i)_i$ of $L$, $\sigma$.

**Output:** $b \in L$.

1. $b := 0$. For $i$ from $n$ to 1, do
2. $\sigma_i := \sigma/\|b_i^*\|$, $c_i := -\langle b, b_i^* \rangle/\|b_i^*\|^2$;
3. Sample $z_i$ from $D_{\mathbb{Z},\sigma_i,c_i}$;
4. $b := b + z_i b_i$.
5. Return $b$.

This is a randomized version of Babai/size-reduction. The 1-dim discrete Gaussian sample can be obtained by rejection from a continuous Gaussian. It can be easily modified to sample according to $D_{L,\sigma,c}$. 

Sampling according to $D_{L,\sigma}$

Using the GSO, we have that the probability of returning $b = \sum (-c_i + z_i) b^*_i$ is:

$$\prod D_{\mathbb{Z},\sigma_i,c_i}(z_i) = \prod \frac{\rho_{\sigma_i,c_i}(z_i)}{\rho_{\sigma_i,c_i}(\mathbb{Z})} = \rho_{\sigma}(b) \cdot \prod \rho_{\sigma_i,c_i}^{-1}(\mathbb{Z}).$$
Sampling according to $D_{L,\sigma}$

Using the GSO, we have that the probability of returning $b = \sum (-c_i + z_i) b_i^*$ is:

$$
\prod D_{\mathbb{Z},\sigma_i,c_i}(z_i) = \prod \frac{\rho_{\sigma_i,c_i}(z_i)}{\rho_{\sigma_i,c_i}(\mathbb{Z})} = \rho_\sigma(b) \cdot \prod \rho_{\sigma_i,c_i}^{-1}(\mathbb{Z}).
$$

If $\sigma \geq \sqrt{n} \cdot \max \|b_i^*\|$, each $\sigma_i$ is $\geq \eta_\varepsilon(\mathbb{Z})$. Thus:

$$
Pr[b] \in \left( \rho_\sigma(b) \cdot \prod \rho_{\sigma_i}^{-1}(\mathbb{Z}) \right) \cdot \left[ \frac{1}{(1 + \varepsilon)^n}, \frac{1}{(1 - \varepsilon)^n} \right].
$$
Sampling according to $D_{L,\sigma}$

Using the GSO, we have that the probability of returning $b = \sum(-c_i + z_i)b_i^*$ is:

$$\prod D_{\mathbb{Z},\sigma_i,c_i}(z_i) = \prod \frac{\rho_{\sigma_i,c_i}(z_i)}{\rho_{\sigma_i,c_i}(\mathbb{Z})} = \rho_{\sigma}(b) \cdot \prod \rho_{\sigma_i,c_i}^{-1}(\mathbb{Z}).$$

If $\sigma \geq \sqrt{n} \cdot \max \|b_i^*\|$, each $\sigma_i$ is $\geq \eta_\varepsilon(\mathbb{Z})$. Thus:

$$\Pr[b] \in \left( \rho_{\sigma}(b) \cdot \prod_i \rho_{\sigma_i}^{-1}(\mathbb{Z}) \right) \cdot \left[ \frac{1}{(1+\varepsilon)^n}, \frac{1}{(1-\varepsilon)^n} \right].$$

The statistical distance between $D_{L,\sigma}$ and the output distribution is exponentially small:

$$\sum_{b \in L} |\Pr[b] - D_{L,\sigma}(b)| = 2^{-\Omega(n)}.$$
Plan

1- Background on Euclidean lattices.
2- The SIS problem, or how to hash.
3- The LWE problem, or how to encrypt.
4- Cryptanalysis.
5- More recent developments.
The SIS problem

a- Non structured SIS.
b- Structured SIS.
c- A trapdoor for SIS.
The Small Integer Solution Problem

Given a uniform $A \in \mathbb{Z}_q^{mn \times n}$, find $s \in \mathbb{Z}^{mn} \setminus \mathbf{0}$ such that:
\[\|s\| \leq \beta \quad \text{and} \quad sA = \mathbf{0} \mod q.\]
SIS_{\beta,q,m} \ [\text{Ajtai’96}]

The Small Integer Solution Problem

Given a uniform $A \in \mathbb{Z}_q^{mn \times n}$, find $s \in \mathbb{Z}^{mn} \setminus 0$ such that:

$$||s|| \leq \beta \quad \text{and} \quad sA = 0 \mod q.$$
The Small Integer Solution Problem

Given a uniform $A \in \mathbb{Z}_{q}^{mn \times n}$, find $s \in \mathbb{Z}^{mn} \setminus 0$ such that:

$\|s\| \leq \beta$ and $sA = 0 \mod q$.

Many interpretations:

- Small codeword problem.
- Short lattice vector problem:

$$A^\perp = \{s \in \mathbb{Z}^{mn} : sA = 0 \ [q]\}.$$
Cryptographic application of SIS

- **Hash**: an efficiently computable function $H : \mathcal{D} \mapsto \mathcal{R}$ with $|\mathcal{R}| \ll |\mathcal{D}|$ is collision resistant if finding $x \neq x'$ in $\mathcal{D}$ such that $H(x) = H(x')$ is computationally hard.

- **Applications**: message integrity, password verification, file identification, digital signature, etc.

- **SIS-based hash**: $s \in \{0,1\}^{mn} \mapsto sA [q]$.

- By linearity, SIS reduces to finding a collision.

- **Compression ratio**: $\frac{mn}{n \log q} = \frac{m}{\log q}$. 

Cryptographic application of SIS

- Hash: an efficiently computable function $H : \mathcal{D} \mapsto \mathcal{R}$ with $|\mathcal{R}| \ll |\mathcal{D}|$ is collision resistant if finding $x \neq x'$ in $\mathcal{D}$ such that $H(x) = H(x')$ is computationally hard.

- Applications: message integrity, password verification, file identification, digital signature, etc.

- SIS-based hash: $s \in \{0, 1\}^{mn} \mapsto sA \ [q]$.

- By linearity, SIS reduces to finding a collision.

- Compression ratio: $\frac{mn}{n \log q} = \frac{m}{\log q}$.
How hard is SIS? A unique level of security.

Worst-case to average-case reduction

Any efficient SIS algorithm succeeding with non-negligible probability leads to an efficient SIVP algorithm.
How hard is SIS? A unique level of security.

Worst-case to average-case reduction \((\gamma \approx n\beta)\)

Any efficient SIS algorithm succeeding with non-negligible probability leads to an efficient SIVP algorithm.
How hard is SIS? A unique level of security.

Worst-case to average-case reduction \((\gamma \approx n\beta)\)

Any efficient SIS algorithm succeeding with non-negligible probability leads to an efficient SIVP algorithm.

Intuition:

- Start with a **short** basis of the lattice \(L \subseteq \mathbb{Z}^n\).
- Sample \(mn\) **short** random lattice points.
- Look at their coordinates wrt the basis, modulo \(q\).
- A SIS solution provides a **shorter** vector of \(L\).
- Repeat to get a basis **shorter** than the initial one.
- Repeat to get **shorter and shorter** bases of \(L\).
How hard is SIS? A unique level of security.

Worst-case to average-case reduction \((\gamma \approx n\beta)\)

Any efficient SIS algorithm succeeding with non-negligible probability leads to an efficient SIVP algorithm.

Intuition:

- Start with a short basis of the lattice \(L \subseteq \mathbb{Z}^n\).
- Sample \(mn\) short random lattice points.
- Look at their coordinates wrt the basis, modulo \(q\).
- A SIS solution provides a shorter vector of \(L\).
- Repeat to get a basis shorter than the initial one.
- Repeat to get shorter and shorter bases of \(L\).
How hard is SIS? A unique level of security.

Worst-case to average-case reduction \((\gamma \approx n^\beta)\)

Any efficient SIS algorithm succeeding with non-negligible probability leads to an efficient SIVP algorithm.

Intuition:

- Start with a short basis of the lattice \(L \subseteq \mathbb{Z}^n\).
- Sample \(mn\) short random lattice points.
- Look at their coordinates wrt the basis, modulo \(q\).
- A SIS solution provides a shorter vector of \(L\).
- Repeat to get a basis shorter than the initial one.
- Repeat to get shorter and shorter bases of \(L\).
How hard is SIS? A unique level of security.

Worst-case to average-case reduction ($\gamma \approx n\beta$)

Any efficient SIS algorithm succeeding with non-negligible probability leads to an efficient SIVP algorithm.

Intuition:

- Start with a **short** basis of the lattice $L \subseteq \mathbb{Z}^n$.
- Sample $mn$ **short** random lattice points.
- Look at their coordinates wrt the basis, modulo $q$.
- A SIS solution provides a **shorter** vector of $L$.
- Repeat to get a basis **shorter** than the initial one.
- Repeat to get **shorter and shorter** bases of $L$. 
How hard is SIS? A unique level of security.

Worst-case to average-case reduction \((\gamma \approx n\beta)\)

Any efficient SIS algorithm succeeding with non-negligible probability leads to an efficient SIVP algorithm.

Intuition:
- Start with a short basis of the lattice \(L \subseteq \mathbb{Z}^n\).
- Sample \(mn\) short random lattice points.
- Look at their coordinates wrt the basis, modulo \(q\).
- A SIS solution provides a shorter vector of \(L\).
- Repeat to get a basis shorter than the initial one.
- Repeat to get shorter and shorter bases of \(L\).
How hard is SIS? A unique level of security.

Worst-case to average-case reduction \((\gamma \approx n\beta)\)

Any efficient SIS algorithm succeeding with non-negligible probability leads to an efficient SIVP algorithm.

Intuition:

- Start with a **short** basis of the lattice \(L \subseteq \mathbb{Z}^n\).
- Sample \(mn\) **short** random lattice points.
- Look at their coordinates wrt the basis, modulo \(q\).
- A SIS solution provides a **shorter** vector of \(L\).
- Repeat to get a basis **shorter** than the initial one.
- Repeat to get **shorter and shorter** bases of \(L\).
How hard is SIS? A unique level of security.

Worst-case to average-case reduction \((\gamma \approx n\beta)\)

Any efficient SIS algorithm succeeding with non-negligible probability leads to an efficient SIVP algorithm.

Intuition:

- Start with a **short** basis of the lattice \(L \subseteq \mathbb{Z}^n\).
- Sample \(mn\) **short** random lattice points.
- Look at their coordinates wrt the basis, modulo \(q\).
- A SIS solution provides a **shorter** vector of \(L\).
- Repeat to get a basis **shorter** than the initial one.
- Repeat to get **shorter and shorter** bases of \(L\).
The $D_{L,\sigma}$ sampler provides valid SIS inputs

- Suppose we start with a basis $(b_i)$ such that $\max \|b_i\| = B$.
- Use the $D_{L,\sigma}$ sampler with $\sigma = \sqrt{nB}$. The output is exponentially close to $D_{L,\sigma}$. Let $(c_i)$ be the samples.
- With high probability: $\forall i : \|c_i\| \leq \sqrt{n\sigma} = nB$.
- Are their coordinates wrt the $b_i$'s uniform mod $q$?
- Yes, because $D_{L,\sigma}$ mod $qL$ is (quasi)-uniform.
- $D_{qL,\sigma,c}$ is (quasi)-independent of $c \in L$ (PSF), when $\sigma \geq \eta_{\varepsilon}(qL) = q \cdot \eta_{\varepsilon}(L)$. 
The $D_{L,\sigma}$ sampler provides valid SIS inputs

- Suppose we start with a basis $(b_i)$ such that $\max \|b_i\| = B$.

- Use the $D_{L,\sigma}$ sampler with $\sigma = \sqrt{nB}$. The output is exponentially close to $D_{L,\sigma}$. Let $(c_i)$ be the samples.

  - With high probability: $\forall i : \|c_i\| \leq \sqrt{n\sigma} = nB$.
  - Are their coordinates wrt the $b_i$’s uniform mod $q$?
    - Yes, because $D_{L,\sigma} \mod qL$ is (quasi)-uniform.
    - $D_{qL,\sigma,c}$ is (quasi)-independent of $c \in L$ (PSF), when $\sigma \geq \eta_\epsilon(qL) = q \cdot \eta_\epsilon(L)$. 

The $D_{L,\sigma}$ sampler provides valid SIS inputs

- Suppose we start with a basis $(b_i)$ such that $\max \|b_i\| = B$.
- Use the $D_{L,\sigma}$ sampler with $\sigma = \sqrt{nB}$. The output is exponentially close to $D_{L,\sigma}$. Let $(c_i)$ be the samples.
- With high probability: $\forall i : \|c_i\| \leq \sqrt{n\sigma} = nB$.
- Are their coordinates wrt the $b_i$'s uniform mod $q$?
  - Yes, because $D_{L,\sigma}$ mod $qL$ is (quasi)-uniform.
- $D_{qL,\sigma,c}$ is (quasi)-independent of $c \in L$ (PSF), when $\sigma \geq \eta(ql) = q \cdot \eta(L)$. 
The $D_{L,\sigma}$ sampler provides valid SIS inputs

- Suppose we start with a basis $(b_i)$ such that $\max \|b_i\| = B$.
- Use the $D_{L,\sigma}$ sampler with $\sigma = \sqrt{nB}$. The output is exponentially close to $D_{L,\sigma}$. Let $(c_i)$ be the samples.
- With high probability: $\forall i : \|c_i\| \leq \sqrt{n\sigma} = nB$.
- Are their coordinates wrt the $b_i$’s uniform mod $q$?
  - Yes, because $D_{L,\sigma}$ mod $qL$ is (quasi)-uniform.
  - $D_{qL,\sigma,c}$ is (quasi)-independent of $c \in L$ (PSF), when $\sigma \geq \eta_\varepsilon(qL) = q \cdot \eta_\varepsilon(L)$. 
The $D_{L,\sigma}$ sampler provides valid SIS inputs

- Suppose we start with a basis $(b_i)$ such that $\max \|b_i\| = B$.
- Use the $D_{L,\sigma}$ sampler with $\sigma = \sqrt{nB}$. The output is exponentially close to $D_{L,\sigma}$. Let $(c_i)$ be the samples.
- With high probability: $\forall i : \|c_i\| \leq \sqrt{n\sigma} = nB$.
- Are their coordinates wrt the $b_i$’s uniform mod $q$?
- Yes, because $D_{L,\sigma}$ mod $qL$ is (quasi)-uniform.
- $D_{qL,\sigma,c}$ is (quasi)-independent of $c \in L$ (PSF), when $\sigma \geq \eta_\varepsilon(qL) = q \cdot \eta_\varepsilon(L)$. 
The $D_{L,\sigma}$ sampler provides valid SIS inputs

- Suppose we start with a basis $(b_i)$ such that $\max \|b_i\| = B$.
- Use the $D_{L,\sigma}$ sampler with $\sigma = \sqrt{nB}$. The output is exponentially close to $D_{L,\sigma}$. Let $(c_i)$ be the samples.
- With high probability: $\forall i : \|c_i\| \leq \sqrt{n\sigma} = nB$.
- Are their coordinates wrt the $b_i$'s uniform mod $q$?
- Yes, because $D_{L,\sigma}$ mod $qL$ is (quasi)-uniform.
- $D_{qL,\sigma,c}$ is (quasi)-independent of $c \in L$ (PSF), when $\sigma \geq \eta_\varepsilon(qL) = q \cdot \eta_\varepsilon(L)$.
Shortness of the output vectors

- We start with a basis \((b_i)\) with \(\max \|b_i\| = B\).
- The \(c_i\)'s satisfy: \(\forall i : \|c_i\| \leq nB\). Let \(x_i\) be their coordinates vectors, reduced mod \(q\).
- The oracle finds \(s \in \mathbb{Z}^{mn}\) with \(\sum s_i x_i = 0 \pmod{q}\) and \(0 < \|s\| \leq \beta\).
- Consider \(c = \frac{1}{q} \sum s_i c_i\): \(c \in L\) and \(\|c\| \leq \frac{\beta n^2 B}{q}\).
- If \(q\) is large enough, we obtain a shorter lattice vector.
- By analyzing the lattice Gaussian further, one can prove that by iterating, with high probability we can find a full rank set of short lattice vectors.
- We can convert the latter into a short basis.
Shortness of the output vectors

- We start with a basis \(( \mathbf{b}_i )\) with \(\max \| \mathbf{b}_i \| = B\).
- The \(\mathbf{c}_i\)'s satisfy: \(\forall i : \| \mathbf{c}_i \| \leq nB\). Let \(\mathbf{x}_i\) be their coordinates vectors, reduced mod \(q\).
- The oracle finds \(\mathbf{s} \in \mathbb{Z}^{mn}\) with \(\sum s_i \mathbf{x}_i = 0 \ [q]\) and \(0 < \| \mathbf{s} \| \leq \beta\).
- Consider \(\mathbf{c} = \frac{1}{q} \sum s_i \mathbf{c}_i\): \(\mathbf{c} \in L\) and \(\| \mathbf{c} \| \leq \frac{\beta n^2 B}{q}\).
- If \(q\) is large enough, we obtain a shorter lattice vector.
- By analyzing the lattice Gaussian further, one can prove that by iterating, with high probability we can find a full rank set of short lattice vectors.
- We can convert the latter into a short basis.
Shortness of the output vectors

- We start with a basis \((b_i)\) with \(\max \|b_i\| = B\).
- The \(c_i\)'s satisfy: \(\forall i : \|c_i\| \leq nB\). Let \(x_i\) be their coordinates vectors, reduced mod \(q\).
- The oracle finds \(s \in \mathbb{Z}^{mn}\) with \(\sum s_i x_i = 0 [q]\) and \(0 < \|s\| \leq \beta\).
- Consider \(c = \frac{1}{q} \sum s_i c_i\): \(c \in L\) and \(\|c\| \leq \frac{\beta n^2 B}{q}\).
- If \(q\) is large enough, we obtain a shorter lattice vector.
- By analyzing the lattice Gaussian further, one can prove that by iterating, with high probability we can find a full rank set of short lattice vectors.
- We can convert the latter into a short basis.
Shortness of the output vectors

- We start with a basis \((b_i)\) with \(\max \|b_i\| = B\).
- The \(c_i\)'s satisfy: \(\forall i : \|c_i\| \leq nB\). Let \(x_i\) be their coordinates vectors, reduced mod \(q\).
- The oracle finds \(s \in \mathbb{Z}^{mn}\) with \(\sum s_i x_i = 0 \ [q]\) and \(0 < \|s\| \leq \beta\).
- Consider \(c = \frac{1}{q} \sum s_i c_i\): \(c \in L\) and \(\|c\| \leq \frac{\beta n^2 B}{q}\).
- If \(q\) is large enough, we obtain a shorter lattice vector.
- By analyzing the lattice Gaussian further, one can prove that by iterating, with high probability we can find a full rank set of short lattice vectors.
- We can convert the latter into a short basis.
We start with a basis \((b_i)\) with \(\max \|b_i\| = B\).

The \(c_i\)'s satisfy: \(\forall i : \|c_i\| \leq nB\). Let \(x_i\) be their coordinates vectors, reduced mod \(q\).

The oracle finds \(s \in \mathbb{Z}^{mn}\) with \(\sum s_i x_i = 0 [q]\) and \(0 < \|s\| \leq \beta\).

Consider \(c = \frac{1}{q} \sum s_i c_i\): \(c \in L\) and \(\|c\| \leq \frac{\beta n^2 B}{q}\).

If \(q\) is large enough, we obtain a shorter lattice vector.

By analyzing the lattice Gaussian further, one can prove that by iterating, with high probability we can find a full rank set of short lattice vectors.

We can convert the latter into a short basis.
Shortness of the output vectors

- We start with a basis \((\mathbf{b}_i)\) with \(\max \|\mathbf{b}_i\| = B\).
- The \(\mathbf{c}_i\)'s satisfy: \(\forall i : \|\mathbf{c}_i\| \leq nB\). Let \(\mathbf{x}_i\) be their coordinates vectors, reduced mod \(q\).
- The oracle finds \(\mathbf{s} \in \mathbb{Z}^{mn}\) with \(\sum s_i \mathbf{x}_i = 0 \ [q]\) and \(0 < \|\mathbf{s}\| \leq \beta\).
- Consider \(\mathbf{c} = \frac{1}{q} \sum s_i \mathbf{c}_i\): \(\mathbf{c} \in L\) and \(\|\mathbf{c}\| \leq \frac{\beta n^2 B}{q}\).
- If \(q\) is large enough, we obtain a shorter lattice vector.
- By analyzing the lattice Gaussian further, one can prove that by iterating, with high probability we can find a full rank set of short lattice vectors.
- We can convert the latter into a short basis.
The SIS problem

- Non structured SIS.
- **Structured SIS.**
- A trapdoor for SIS.
Id-SIS, graphically

- Each block is negacyclic.
- The $i$th row is: $x^i \cdot a(x) \mod x^n + 1$.
- Structured matrices $\equiv$ polynomials $\equiv$ fast algorithms.
Ideal SIS, algebraically

**SIS**

Given a uniform \( A \in \mathbb{Z}_{q}^{mn \times n} \), find \( s \in \mathbb{Z}^{mn} \setminus 0 \) such that:

\[
\|s\| \leq \beta \quad \text{and} \quad sA = 0 \pmod{q}.
\]

**Id-SIS**

Given uniform \( a_1, \ldots, a_m \in \mathbb{Z}_q[x]/(x^n + 1) \), find \( s_1, \ldots, s_m \in \mathbb{Z}[x]/(x^n + 1) \) not all 0 such that:

\[
\|s\| \leq \beta \quad \text{and} \quad \sum s_i a_i = 0 \pmod{(q, x^n + 1)}.
\]
Ideal SIS, algebraically

**Id-SIS**

Given uniform $a_1, \ldots, a_m \in \mathbb{Z}_q[x]/(x^n + 1)$, find $s_1, \ldots, s_m \in \mathbb{Z}[x]/(x^n + 1)$ not all 0 such that:

$$\|s\| \leq \beta \quad \text{and} \quad \sum s_i a_i = 0 \mod (q, x^n + 1).$$

**Worst-case to average-case reduction**

Any efficient **Id-SIS** algorithm succeeding with non-negligible probability leads to an efficient **Id-SIVP** algorithm.
Efficient hashing

- **SIS hash**: \( s \in \{0, 1\}^{mn} \mapsto sA \ [q] \).
- **Id-SIS hash**: \( s_1, \ldots, s_m \in \{0, 1\}[x] \) of degrees \(< n\) are mapped to \( \sum s_i(x)a_i(x) \ [q, x^n + 1] \).
- If \( 2n|q - 1 \), then \( x^n + 1 \) splits completely mod \( q \).
  \( \Rightarrow \) Fast Discrete Fourier Transform mod \( q \).
- **Storage**: \( \tilde{O}(n^2) \rightarrow \tilde{O}(n) \); complexity: \( \tilde{O}(n^2) \rightarrow \tilde{O}(n) \).
Efficient hashing

- **SIS hash**: $s \in \{0, 1\}^{mn} \mapsto sA \ [q]$.

- **Id-SIS hash**: $s_1, \ldots, s_m \in \{0, 1\}[x]$ of degrees $< n$ are mapped to $\sum s_i(x)a_i(x) \ [q, x^n + 1]$.

- If $2n|q - 1$, then $x^n + 1$ splits completely mod $q$.
  \[\Rightarrow\] Fast Discrete Fourier Transform mod $q$.

- Storage: $\tilde{O}(n^2) \rightarrow \tilde{O}(n)$; complexity: $\tilde{O}(n^2) \rightarrow \tilde{O}(n)$.  


Efficient hashing

- SIS hash: \( s \in \{0, 1\}^{mn} \mapsto sA \mod q \).
- Id-SIS hash: \( s_1, \ldots, s_m \in \{0, 1\}[x] \) of degrees \( < n \) are mapped to \( \sum s_i(x)a_i(x) \mod q, x^n + 1 \).
- If \( 2n|q - 1 \), then \( x^n + 1 \) splits completely mod \( q \).
  \( \Rightarrow \) Fast Discrete Fourier Transform mod \( q \).
- Storage: \( \tilde{O}(n^2) \rightarrow \tilde{O}(n) \); complexity: \( \tilde{O}(n^2) \rightarrow \tilde{O}(n) \).
Efficient hashing

- SIS hash: \( s \in \{0, 1\}^{mn} \mapsto sA \mod q \).
- Id-SIS hash: \( s_1, \ldots, s_m \in \{0, 1\}[x] \) of degrees \(< n\) are mapped to \( \sum s_i(x)a_i(x) \mod q, x^n + 1 \).
- If \( 2n \mid q - 1 \), then \( x^n + 1 \) splits completely mod \( q \).
  \( \Rightarrow \) Fast Discrete Fourier Transform mod \( q \).

- Storage: \( \tilde{O}(n^2) \rightarrow \tilde{O}(n) \); complexity: \( \tilde{O}(n^2) \rightarrow \tilde{O}(n) \).

This is SWIFFT and it was proposed to the SHA-3 contest.
With \( n = 2^6 \), \( m = 2^4 \), \( q \approx 2^8 \): \( \approx 2^{13}\) bits to store \( A \).
The SIS problem

a- Non structured SIS.

b- Structured SIS.

c- A trapdoor for SIS.
A uniform $A$ with a good basis for $A^\perp$

If $m = \Omega(\log q)$ then we can efficiently sample $A \in \mathbb{Z}_{q}^{mn \times n}$ and $T_A$ such that

- The statistical distance from $A$ to uniform is $2^{-\Omega(n)}$.
- The rows of $T_A$ are small: $\max \| t_i^* \| = O(\sqrt{n \log q})$.
- $T_A \in \mathbb{Z}^{mn \times mn}$ is a basis of $A^\perp$. 

A uniform \( A \) with a good basis for \( A^\perp \)

If \( m = \Omega(\log q) \) then we can efficiently sample \( A \in \mathbb{Z}_q^{mn \times n} \) and \( T_A \) such that

- The statistical distance from \( A \) to uniform is \( 2^{-\Omega(n)} \).
- The rows of \( T_A \) are small: \( \max \|t_i^*\| = O(\sqrt{n \log q}) \).
- \( T_A \in \mathbb{Z}^{mn \times mn} \) is a basis of \( A^\perp \).
A uniform $A$ with a good basis for $A^\perp$

If $m = \Omega(\log q)$ then we can efficiently sample $A \in \mathbb{Z}_q^{mn \times n}$ and $T_A$ such that

- The statistical distance from $A$ to uniform is $2^{-\Omega(n)}$.
- The rows of $T_A$ are small: $\max \|t_i^*\| = O(\sqrt{n \log q})$.
- $T_A \in \mathbb{Z}^{mn \times mn}$ is a basis of $A^\perp$.

$T_A$

(small)

| A | 0 |

Principle:

- Assume $(a_i)_{i \leq k}$ are iid uniform.
- Take $(x_i)_i$ iid uniform in $\{-1, 0, 1\}$.
- Then $a_{k+1} = \sum_{i \leq k} x_i a_i$ is close to uniform.
A trapdoor for SIS

Suppose we know \( u \in \mathbb{Z}_q^n \), \( A \) and \( T_A \). How do we find a small \( s \in \mathbb{Z}^{mn} \) such that \( sA = u \ [q] \)?

- With linear algebra, find \( c \in \mathbb{Z}^{mn} \) such that \( cA = u \ [q] \).
- It suffices to find a vector \( b \) of \( A \perp \) that is close to \( c \): \( \|c - b\| \) is small and \( (c - b)A = u \ [q] \).
- Use the sampler from \( D_{L,\sigma,c} \) with:

\[
\sigma = \sqrt{n} \cdot \max \|t^*_i\| = O(n \sqrt{\log q}).
\]

- We have \( \|c - b\| \leq \sigma \sqrt{n} = O(n^{1.5} \sqrt{\log q}) \) with probability \( \geq 1 - 2^{-\Omega(n)} \).
- And we do not leak any information about the trapdoor.
A trapdoor for SIS

- Suppose we know $\mathbf{u} \in \mathbb{Z}_q^n$, $A$ and $T_A$. How do we find a small $\mathbf{s} \in \mathbb{Z}^{mn}$ such that $sA = u \ [q]$?
- With linear algebra, find $\mathbf{c} \in \mathbb{Z}^{mn}$ such that $cA = u \ [q]$.
- It suffices to find a vector $\mathbf{b}$ of $A^\perp$ that is close to $\mathbf{c}$: $\|\mathbf{c} - \mathbf{b}\|$ is small and $(\mathbf{c} - \mathbf{b})A = u \ [q]$.
- Use the sampler from $D_{L,\sigma,c}$ with:

$$\sigma = \sqrt{n} \cdot \max \|t_i^*\| = O(n \sqrt{\log q}).$$

- We have $\|\mathbf{c} - \mathbf{b}\| \leq \sigma \sqrt{n} = O(n^{1.5} \sqrt{\log q})$ with probability $\geq 1 - 2^{-\Omega(n)}$.
- And we do not leak any information about the trapdoor.
A trapdoor for SIS

- Suppose we know \( \mathbf{u} \in \mathbb{Z}_q^n \), \( A \) and \( T_A \). How do we find a small \( \mathbf{s} \in \mathbb{Z}^{mn} \) such that \( \mathbf{s} A = \mathbf{u} [q] \)?
- With linear algebra, find \( \mathbf{c} \in \mathbb{Z}^{mn} \) such that \( \mathbf{c} A = \mathbf{u} [q] \).
- It suffices to find a vector \( \mathbf{b} \) of \( A^\perp \) that is close to \( \mathbf{c} \): \( \| \mathbf{c} - \mathbf{b} \| \) is small and \( (\mathbf{c} - \mathbf{b}) A = \mathbf{u} [q] \).
- Use the sampler from \( D_{L,\sigma,c} \) with:

\[
\sigma = \sqrt{n} \cdot \max \| \mathbf{t}_i^* \| = O(n^{1.5} \sqrt{\log q}).
\]

- We have \( \| \mathbf{c} - \mathbf{b} \| \leq \sigma \sqrt{n} = O(n^{1.5} \sqrt{\log q}) \) with probability \( \geq 1 - 2^{-\Omega(n)} \).
- And we do not leak any information about the trapdoor.
A trapdoor for SIS

Suppose we know $u \in \mathbb{Z}_q^n$, $A$ and $T_A$. How do we find a small $s \in \mathbb{Z}^{mn}$ such that $sA = u \mod q$?

With linear algebra, find $c \in \mathbb{Z}^{mn}$ such that $cA = u \mod q$.

It suffices to find a vector $b$ of $A^\perp$ that is close to $c$: $\|c - b\|$ is small and $(c - b)A = u \mod q$.

Use the sampler from $D_{L,\sigma,c}$ with:

$$\sigma = \sqrt{n} \cdot \max \|t_i^*\| = O(n \sqrt{\log q}).$$

We have $\|c - b\| \leq \sigma \sqrt{n} = O(n^{1.5} \sqrt{\log q})$ with probability $\geq 1 - 2^{-\Omega(n)}$.

And we do not leak any information about the trapdoor.
A trapdoor for SIS

- Suppose we know $u \in \mathbb{Z}_q^n$, $A$ and $T_A$. How do we find a small $s \in \mathbb{Z}^{mn}$ such that $sA = u \mod q$?
- With linear algebra, find $c \in \mathbb{Z}^{mn}$ such that $cA = u \mod q$.
- It suffices to find a vector $b$ of $A^\perp$ that is close to $c$: $\|c - b\|$ is small and $(c - b)A = u \mod q$.
- Use the sampler from $D_{L,\sigma,c}$ with:

$$\sigma = \sqrt{n} \cdot \max \|t_i^*\| = O(n \sqrt{\log q}).$$

- We have $\|c - b\| \leq \sigma \sqrt{n} = O(n^{1.5} \sqrt{\log q})$ with probability $\geq 1 - 2^{-\Omega(n)}$.
- And we do not leak any information about the trapdoor.
Cryptographic application: hash-and-sign

- **Signature:** to ensure the authenticity of a document.
  - Signer’s public key: $A$; private key: $T_A$.
  - To sign $M$, use the trapdoor to find $s$ short with $sA = \mathcal{H}(M)$, where $\mathcal{H}$ is a public random oracle.
  - To verify $(M, s)$, see whether $sA = \mathcal{H}(M)$ and $\|s\|$ small.
  - Can be made at least as hard to break as to solve SIS, in the random oracle model.
Cryptographic application: hash-and-sign

- Signature: to ensure the authenticity of a document.
- Signer’s public key: $A$; private key: $T_A$.
- To sign $M$, use the trapdoor to find $s$ short with $sA = \mathcal{H}(M)$, where $\mathcal{H}$ is a public random oracle.
- To verify $(M, s)$, see whether $sA = \mathcal{H}(M)$ and $\|s\|$ small.
- Can be made at least as hard to break as to solve SIS, in the random oracle model.
Cryptographic application: hash-and-sign

- Signature: to ensure the authenticity of a document.
- Signer’s public key: $A$; private key: $T_A$.
- To sign $M$, use the trapdoor to find $s$ short with $sA = \mathcal{H}(M)$, where $\mathcal{H}$ is a public random oracle.
- To verify $(M, s)$, see whether $sA = \mathcal{H}(M)$ and $\|s\|$ small.
- Can be made at least as hard to break as to solve SIS, in the random oracle model.
A trapdoor for Id-SIS

If $m = \Omega(n \log q)$ and $x^n + 1$ has $O(1)$ factors mod $q$, then we can efficiently sample $a_1, \ldots, a_m \in \mathbb{Z}_q[x]/(x^n + 1)$ and $T_A \in (\mathbb{Z}[x]/(x^n + 1))^{m \times m}$ such that

- The statistical distance from $a$ to uniform is $2^{-\Omega(n)}$.
- The rows of $\text{rot}(T_A)$ are small: $\max \|t_i^*\| = O(\sqrt{n \log q})$.
- $T_A \in \mathbb{Z}^{mn \times mn}$ is a basis of a full-rank sublattice of $A^\perp$. 

Let $\text{rot}(T_A)$ be the Moore-Penrose pseudo-inverse of $T_A$. 

$\text{rot}(T_A)$ can be obtained by singular value decomposition (SVD) of $T_A$. 

The SIS problem 

The LWE problem 

Cryptanalysis 

Recent developments 

Conclusion
A trapdoor for Id-SIS

If $m = \Omega(n \log q)$ and $x^n + 1$ has $O(1)$ factors mod $q$, then we can efficiently sample $a_1, \ldots, a_m \in \mathbb{Z}_q[x]/(x^n + 1)$ and $T_A \in (\mathbb{Z}[x]/(x^n + 1))^{m \times m}$ such that

- The statistical distance from $a$ to uniform is $2^{-\Omega(n)}$.
- The rows of $\text{rot}(T_A)$ are small: $\max \|t_i^*\| = O(\sqrt{n \log q})$.
- $T_A \in \mathbb{Z}^{mn \times mn}$ is a basis of a full-rank sublattice of $A^\perp$. 

```
\begin{array}{c|c|c}
| & & |
\hline
T_A & A & 0 \\
\hline
\end{array}
```

$T_A$ (small)
Comparison with SIS’ trapdoor

Drawbacks (wrt SIS):

- There are non-trivial ideals in $\mathbb{Z}_q[x]/(x^n + 1)$.
- $A^\perp$ has a structure of $\mathcal{O}_K$-module: a full pseudo-basis of $A^\perp$ could be obtained from $T_A$ using the Cohen-Bosma-Pohst HNF for Dedekind domains.
Comparison with SIS’ trapdoor

Drawbacks (wrt SIS):

- There are non-trivial ideals in $\mathbb{Z}_q[x]/(x^n + 1)$.
- $A^\perp$ has a structure of $\mathcal{O}_K$-module: a full pseudo-basis of $A^\perp$ could be obtained from $T_A$ using the Cohen-Bosma-Pohst HNF for Dedekind domains.

Advantages:

- Compact trapdoor: $(mn)^2 \log q$ bits $\rightarrow m^2 n \log q$ bits.
- Verifying the signature is faster.
- But there exists a more efficient Id-SIS-based signature anyway [Lyubashevsky'09].
Comparison with SIS’ trapdoor

Drawbacks (wrt SIS):

- There are non-trivial ideals in $\mathbb{Z}_q[x]/(x^n + 1)$.
- $A^\perp$ has a structure of $\mathcal{O}_K$-module: a full pseudo-basis of $A^\perp$ could be obtained from $T_A$ using the Cohen-Bosma-Pohst HNF for Dedekind domains.

Advantages:

- Compact trapdoor: $(mn)^2 \log q$ bits $\rightarrow m^2 n \log q$ bits.
- Verifying the signature is faster.
- But there exists a more efficient Id-SIS-based signature anyway [Lyubashevsky’09].
Plan

1- Background on Euclidean lattices.
2- The SIS problem, or how to hash.
3- The LWE problem, or how to encrypt.
4- Cryptanalysis.
5- More recent developments.
The LWE problem

a- Non structured LWE.
b- Structured LWE.
c- Encrypting with LWE.
**LWE_{α,q,m} [Regev’05]**

**The Learning With Errors Problem**

Take $A$ uniform in $\mathbb{Z}_q^{mn \times n}$, $s$ uniform in $\mathbb{Z}_q^n$ and $e$ sampled from $\mathcal{N}_{αq}^{mn}$. Given $A$ and $As + e$ [q], find $s$. 
The Learning With Errors Problem

Take $A$ uniform in $\mathbb{Z}_{q}^{mn \times n}$, $s$ uniform in $\mathbb{Z}_{q}^{n}$ and $e$ sampled from $\mathcal{N}_{\alpha q}^{mn}$. Given $A$ and $As + e \pmod{q}$, find $s$. 

\[ \begin{array}{c|c|c}
A & s & e \\
\text{uniform} & \text{uniform} & \text{small} \\
\end{array} \]
The Learning With Errors Problem

Take $A$ uniform in $\mathbb{Z}_q^{mn \times n}$, $s$ uniform in $\mathbb{Z}_q^n$ and $e$ sampled from $\mathcal{N}_{\alpha q}^{mn}$. Given $A$ and $As + e [q]$, find $s$.

Many interpretations:
- Given many $\langle a_i, s \rangle + e_i$, find $s$.
- Resembles LPN (over $\mathbb{Z}_2$).
- Resembles Subset-Sum [LPS’09].
- Closest codeword problem.
- Lattice problem . . .
LWE as a lattice problem

The Learning With Errors Problem

Take $A$ uniform in $\mathbb{Z}_q^{mn \times n}$, $s$ uniform in $\mathbb{Z}_q^n$ and $e$ sampled from $\mathcal{N}_{\alpha q}^{mn}$. Given $A$ and $As + e [q]$, find $s$. 
LWE as a lattice problem

The Learning With Errors Problem

Take $A$ uniform in $\mathbb{Z}^{mn \times n}_q$, $s$ uniform in $\mathbb{Z}_q^n$ and $e$ sampled from $\mathcal{N}_{\alpha q}$. Given $A$ and $As + e \pmod{q}$, find $s$.

Let $L_A = \{b \in \mathbb{Z}^{mn} : \exists x \in \mathbb{Z}_q^n, b = Ax \pmod{q}\}$.
LWE as a lattice problem

The Learning With Errors Problem

Take $A$ uniform in $\mathbb{Z}_{q}^{mn \times n}$, $s$ uniform in $\mathbb{Z}_{q}^{n}$ and $e$ sampled from $\mathcal{N}_{\alpha q}^{mn}$. Given $A$ and $As + e \mod q$, find $s$.

Let $L_A = \{b \in \mathbb{Z}^{mn} : \exists x \in \mathbb{Z}_{q}^{n}, b = Ax \mod q\}$.

- $L_A$ is an $(mn)$-dimensional lattice and $\hat{L_A} = \frac{1}{q}A^\perp$.
- BDD_{\alpha,q} (bounded distance decoding): Take $A$ uniform in $\mathbb{Z}_{q}^{mn \times n}$, take $b \in L_A$ arbitrary and $e$ sampled from $\mathcal{N}_{\alpha q}^{mn}$; given $b + e$, find $b$.
- If we can solve LWE then we can solve BDD.
LWE as a lattice problem

The Learning With Errors Problem

Take $A$ uniform in $\mathbb{Z}_{q}^{mn \times n}$, $s$ uniform in $\mathbb{Z}_{q}^{n}$ and $e$ sampled from $\mathcal{N}_{\alpha q}^{mn}$. Given $A$ and $As + e \pmod{q}$, find $s$.

Let $L_A = \{b \in \mathbb{Z}^{mn} : \exists x \in \mathbb{Z}_{q}^{n}, b = Ax \pmod{q}\}$.

- $L_A$ is an $(mn)$-dimensional lattice and $\widehat{L_A} = \frac{1}{q} A^\perp$.
- BDD$_{\alpha,q}$ (bounded distance decoding): Take $A$ uniform in $\mathbb{Z}_{q}^{mn \times n}$, take $b \in L_A$ arbitrary and $e$ sampled from $\mathcal{N}_{\alpha q}^{mn}$; given $b + e$, find $b$.
- If we can solve LWE then we can solve BDD.
The Learning With Errors Problem

Take $A$ uniform in $\mathbb{Z}_q^{mn \times n}$, $s$ uniform in $\mathbb{Z}_q^n$ and $e$ sampled from $\mathcal{N}_{\alpha q}^{mn}$. Given $A$ and $As + e [q]$, find $s$.

Let $L_A = \{ b \in \mathbb{Z}^{mn} : \exists x \in \mathbb{Z}_q^n, b = Ax [q] \}$.

- $L_A$ is an $(mn)$-dimensional lattice and $\widehat{L_A} = \frac{1}{q} A^\perp$.
- BDD$_{\alpha, q}$ (bounded distance decoding): Take $A$ uniform in $\mathbb{Z}_q^{mn \times n}$, take $b \in L_A$ arbitrary and $e$ sampled from $\mathcal{N}_{\alpha q}^{mn}$; given $b + e$, find $b$.
- If we can solve LWE then we can solve BDD.
LWE as a one-way function

- **OWF**: easy to evaluate and hard to invert.
- **LWE’s one-way function**: \( s \in \mathbb{Z}_q^n \mapsto As + e\, [q] \).
- **Expansion**: \( n \log q \) bits \( \mapsto \) \( mn \log q \) bits.
LWE as a one-way function

- OWF: easy to evaluate and hard to invert.
- LWE’s one-way function: $\mathbf{s} \in \mathbb{Z}_q^n \mapsto A\mathbf{s} + \mathbf{e} [q]$.
- Expansion: $n \log q$ bits $\mapsto mn \log q$ bits.

A one-way function with trapdoor.

- Generate $A$ together with $T_A$.
  - $T_A \cdot (A\mathbf{s} + \mathbf{e}) = T_A\mathbf{e} [q]$.
  - $T_A$ and $\mathbf{e}$ are small: we have $T_A\mathbf{e}$ over $\mathbb{Z}$.
    We recover $\mathbf{e}$ and then $\mathbf{s}$ by linear algebra.
- Sufficient condition:
  \[
  \frac{q}{2} > \sqrt{n\alpha q} \cdot \max \|t_i\| \iff n^{1.5}\alpha \sqrt{\log q} = o(1).
  \]
LWE as a one-way function

- OWF: easy to evaluate and hard to invert.
- LWE’s one-way function: \( \mathbf{s} \in \mathbb{Z}_q^n \mapsto A\mathbf{s} + \mathbf{e} \ [q] \).
- Expansion: \( n \log q \) bits \( \mapsto m n \log q \) bits.

A one-way function with trapdoor.

- Generate \( A \) together with \( T_A \).
- \( T_A \cdot (A\mathbf{s} + \mathbf{e}) = T_A\mathbf{e} \ [q] \).
- \( T_A \) and \( \mathbf{e} \) are small: we have \( T_A\mathbf{e} \) over \( \mathbb{Z} \).
- We recover \( \mathbf{e} \) and then \( \mathbf{s} \) by linear algebra.
- Sufficient condition:

\[
\frac{q}{2} > \sqrt{n} \alpha q \cdot \max \|t_i\| \iff n^{1.5} \alpha \sqrt{\log q} = o(1).
\]
LWE as a one-way function

- OWF: easy to evaluate and hard to invert.
- LWE’s one-way function: $s \in \mathbb{Z}_q^n \mapsto As + e [q]$.
- Expansion: $n \log q$ bits $\mapsto mn \log q$ bits.

A one-way function with trapdoor.

- Generate $A$ together with $T_A$.
- $T_A \cdot (As + e) = T_A e [q]$.
- $T_A$ and $e$ are small: we have $T_A e$ over $\mathbb{Z}$.
  We recover $e$ and then $s$ by linear algebra.

Sufficient condition:

$$\frac{q}{2} > \sqrt{n} \alpha q \cdot \max \|t_i\| \iff n^{1.5} \alpha \sqrt{\log q} = o(1).$$
LWE as a one-way function

- OWF: easy to evaluate and hard to invert.
- LWE’s one-way function: \( s \in \mathbb{Z}_q^n \mapsto As + e \ [q] \).
- Expansion: \( n \log q \) bits \( \mapsto \) \( mn \log q \) bits.

A one-way function with trapdoor.

- Generate \( A \) together with \( T_A \).
- \( T_A \cdot (As + e) = T_Ae \ [q] \).
- \( T_A \) and \( e \) are small: we have \( T_Ae \) over \( \mathbb{Z} \).
  - We recover \( e \) and then \( s \) by linear algebra.
- Sufficient condition:

\[
\frac{q}{2} > \sqrt{n \alpha q \cdot \max \|t_i\|} \iff n^{1.5} \alpha \sqrt{\log q} = o(1).
\]
How hard is LWE?

Quantum worst-case to average-case reduction

Any efficient LWE algorithm succeeding with non-negligible probability leads to an efficient *quantum* SIVP algorithm.
How hard is LWE?

Quantum worst-case to average-case reduction \((\gamma \approx n/\alpha)\)

Any efficient LWE algorithm succeeding with non-negligible probability leads to an efficient quantum SIVP algorithm.
How hard is LWE?

Quantum worst-case to average-case reduction \( (\gamma \approx n/\alpha) \)
Any efficient LWE algorithm succeeding with non-negligible probability leads to an efficient \textbf{quantum} SIVP algorithm.

- Efficient quantum computers make LWE more secure!
- [Peikert’09] de-quantumized the reduction, with larger \( q \) or unusual variant of SIVP.
- [SSTX’09]: simpler (but weaker) quantum reduction.
How hard is $\text{BDD}_{\alpha,q}$? Rough intuition.

$L \rightarrow \hat{L}$

Fourier transform
How hard is BDD_{\alpha, q}? Rough intuition.

- The Fourier transform of the distribution is implemented with the quantum Fourier transform.
- The input quantum state is built with the LWE oracle.
- The measurement gives a small SIS solution.
More formally

- If $D$ is a distribution over a finite domain $\mathcal{D}$ that can be sampled efficiently (classically), then the quantum state $\sum_{x \in \mathcal{D}} \sqrt{D(x)} |x\rangle$ can be built efficiently.
- When a state $\sum_{x \in \mathcal{D}} \sqrt{D(x)} |x\rangle$ is measured, then $x_0 \in \mathcal{D}$ is returned with probability $D(x)$.
- Apart from measurements, only invertible (unitary) operations can be applied to states.
- We want to build the state

$$\sum_{e \in \mathbb{R}^n, b \in L} \rho_{\alpha q}(e) |b + e\rangle.$$
More formally

- If $D$ is a distribution over a finite domain $\mathcal{D}$ that can be sampled efficiently (classically), then the quantum state $\sum_{x \in \mathcal{D}} \sqrt{D(x)} |x\rangle$ can be built efficiently.

- When a state $\sum_{x \in \mathcal{D}} \sqrt{D(x)} |x\rangle$ is measured, then $x_0 \in \mathcal{D}$ is returned with probability $D(x)$.

- Apart from measurements, only invertible (unitary) operations can be applied to states.

- We want to build the state

$$\sum_{e \in \mathbb{R}^n, b \in L} \rho_{\alpha q}(e) |b + e\rangle.$$
More details, but still informal

- $L$ is infinite $\Rightarrow$ we work modulo $L$.
- $\mathbb{R}^n$ is infinite $\Rightarrow$ we work in a very fine grid $L/R$.
- Gaussians vanish quickly $\Rightarrow$ we neglect their tails.
More details, but still informal

- $L$ is infinite $\Rightarrow$ we work modulo $L$.
- $\mathbb{R}^n$ is infinite $\Rightarrow$ we work in a very fine grid $L/R$.
- Gaussians vanish quickly $\Rightarrow$ we neglect their tails.
More details, but still informal

- $L$ is infinite $\Rightarrow$ we work modulo $L$.
- $\mathbb{R}^n$ is infinite $\Rightarrow$ we work in a very fine grid $L/R$.
- Gaussians vanish quickly $\Rightarrow$ we neglect their tails.
More details, but still informal

- $L$ is infinite $\Rightarrow$ we work modulo $L$.
- $\mathbb{R}^n$ is infinite $\Rightarrow$ we work in a very fine grid $L/R$.
- Gaussians vanish quickly $\Rightarrow$ we neglect their tails.

1. We build $\sum_{e \in L/R, \|e\| \leq \alpha q \sqrt{n}} \rho_{\alpha q}(e) \ket{e} \ket{e}$.
2. We reduce mod $L$: $\sum \rho_{\alpha q}(e) \ket{e \mod L} \ket{e}$.
3. We use the BDD oracle: $\sum \rho_{\alpha q}(e) \ket{e \mod L} \ket{0}$.
4. Applying the quantum Fourier transform and measuring provides a sample from $\widetilde{D}_{L,1/(\alpha q)}$, i.e., $D_{A^\perp,1/\alpha}$.
More details, but still informal

- $L$ is infinite $\Rightarrow$ we work modulo $L$.
- $\mathbb{R}^n$ is infinite $\Rightarrow$ we work in a very fine grid $L/R$.
- Gaussians vanish quickly $\Rightarrow$ we neglect their tails.

1. We build $\sum_{e \in L/R, \|e\| \leq \alpha q \sqrt{n}} \rho_{\alpha q}(e) |e\rangle |e\rangle$.
2. We reduce mod $L$: $\sum \rho_{\alpha q}(e) |e \mod L\rangle |e\rangle$.
3. We use the BDD oracle: $\sum \rho_{\alpha q}(e) |e \mod L\rangle |0\rangle$.
4. Applying the quantum Fourier transform and measuring provides a sample from $D_{\hat{L_A},1/(\alpha q)}$, i.e., $D_{A\perp,1/\alpha}$.
More details, but still informal

- $L$ is infinite $\Rightarrow$ we work modulo $L$.
- $\mathbb{R}^n$ is infinite $\Rightarrow$ we work in a very fine grid $L/R$.
- Gaussians vanish quickly $\Rightarrow$ we neglect their tails.

1. We build $\sum_{e \in L/R, \|e\| \leq \alpha q \sqrt{n}} \rho_{\alpha q}(e) |e\rangle |e\rangle$.
2. We reduce mod $L$: $\sum \rho_{\alpha q}(e) |e \mod L\rangle |e\rangle$.
3. We use the BDD oracle: $\sum \rho_{\alpha q}(e) |e \mod L\rangle |0\rangle$.
4. Applying the quantum Fourier transform and measuring provides a sample from $D_{\hat{L}^{\perp},1/(\alpha q)}$, i.e., $D_{A^{\perp},1/\alpha}$.

Additional difficulty: the oracle may solve LWE with probability $\ll 1$. Use the trace distance.
The LWE problem

- **Non structured LWE.**
- **Structured LWE.**
- **Encrypting with LWE.**
Ideal LWE [SSTX’09]

Id-LWE: Take a block negacylic LWE matrix (as for Id-SIS).

Any efficient **Id-LWE** algo. succeeding with non-negligible probability leads to an efficient quantum **Id-SIVP** algo.
Id-LWE: Take a block negacylic LWE matrix (as for Id-SIS).

Any efficient **Id-LWE** algo. succeeding with non-negligible probability leads to an efficient quantum **Id-SIVP** algo.

Polynomial interpretation:
Let $a_1, \ldots, a_m \in \mathbb{Z}_q[x]/(x^n + 1)$ be the polynomials corresponding to the block matrix. Then $A \cdot s$ corresponds to:

$$(\bar{a}_i(x) \cdot s(x) \mod (q, x^n + 1))_{i \leq m},$$

where $\bar{a}(x) = a_0 - \sum_{1 \leq k < n} a_{n-k}x^k$. 
A faster trapdoor one-way function

- Evaluation cost: $\tilde{O}(n^2) \Rightarrow \tilde{O}(n)$ bit operations.
- For the inversion, use the structured $T_A$ from Id-SIS.
- $T_A \cdot (As + e) = T_A e$ over the integers. Multiply by $T_A^{-1}$ to recover $e$, and then $s$.
- Evaluation/inversion cost: $\tilde{O}(n^2) \Rightarrow \tilde{O}(n)$ bit operations.
- Less practical than Id-SIS hash, because we cannot take $q$ such that $x^n + 1$ splits completely mod $q$. 

A faster trapdoor one-way function

- Evaluation cost: $\tilde{O}(n^2) \Rightarrow \tilde{O}(n)$ bit operations.
- For the inversion, use the structured $T_A$ from Id-SIS.
- $T_A \cdot (As + e) = T_Ae$ over the integers. Multiply by $T_A^{-1}$ to recover $e$, and then $s$.
- Evaluation/inversion cost: $\tilde{O}(n^2) \Rightarrow \tilde{O}(n)$ bit operations.
- Less practical than Id-SIS hash, because we cannot take $q$ such that $x^n + 1$ splits completely mod $q$. 
A faster trapdoor one-way function

- Evaluation cost: $\tilde{O}(n^2) \Rightarrow \tilde{O}(n)$ bit operations.
- For the inversion, use the structured $T_A$ from Id-SIS.
  \[ T_A \cdot (A \mathbf{s} + \mathbf{e}) = T_A \mathbf{e} \] over the integers. Multiply by $T_A^{-1}$ to recover $\mathbf{e}$, and then $\mathbf{s}$.
- Evaluation/inversion cost: $\tilde{O}(n^2) \Rightarrow \tilde{O}(n)$ bit operations.
- Less practical than Id-SIS hash, because we cannot take $q$ such that $x^n + 1$ splits completely mod $q$. 
A faster trapdoor one-way function

- Evaluation cost: $\tilde{O}(n^2) \Rightarrow \tilde{O}(n)$ bit operations.
- For the inversion, use the structured $T_A$ from Id-SIS.
  
  $$T_A \cdot (As + e) = T_A e \text{ over the integers. Multiply by } T_A^{-1}$$
  
  to recover $e$, and then $s$.
- Evaluation/inversion cost: $\tilde{O}(n^2) \Rightarrow \tilde{O}(n)$ bit operations.
- Less practical than Id-SIS hash, because we cannot take $q$ such that $x^n + 1$ splits completely mod $q$. 

The LWE problem

- Non structured LWE.
- Structured LWE.
- Encrypting with LWE.
Decisional-LWE

Computational-LWE

Take $A$ uniform in $\mathbb{Z}_{q}^{mn \times n}$, $s$ uniform in $\mathbb{Z}_{q}^{n}$ and $e$ sampled from $\mathcal{N}_{\alpha q}^{mn}$. Given $A$ and $As + e \ [q]$, find $s$.

Decisional-LWE

Take $A$ uniform in $\mathbb{Z}_{q}^{mn \times n}$, $s$ uniform in $\mathbb{Z}_{q}^{n}$ and $e$ sampled from $\mathcal{N}_{\alpha q}^{mn}$. Distinguish between the distributions $(A, As + e \ [q])$ and uniform over $\mathbb{Z}_{q}^{mn \times n} \times \mathbb{Z}_{q}^{mn}$.
**Decisional-LWE**

**Computational-LWE**

Take $A$ uniform in $\mathbb{Z}_q^{mn \times n}$, $s$ uniform in $\mathbb{Z}_q^n$ and $e$ sampled from $\mathcal{N}_{\alpha q}^{mn}$. Given $A$ and $As + e [q]$, find $s$.

**Decisional-LWE**

Take $A$ uniform in $\mathbb{Z}_q^{mn \times n}$, $s$ uniform in $\mathbb{Z}_q^n$ and $e$ sampled from $\mathcal{N}_{\alpha q}^{mn}$. Distinguish between the distributions $(A, As + e [q])$ and uniform over $\mathbb{Z}_q^{mn \times n} \times \mathbb{Z}_q^{mn}$.

Regev proved that Dec-LWE is at least as hard as Comp-LWE.
**Decisional-LWE**

**Computational-LWE**

Take $A$ uniform in $\mathbb{Z}_{q}^{mn \times n}$, $s$ uniform in $\mathbb{Z}_{q}^{n}$ and $e$ sampled from $\mathcal{N}_{\alpha q}^{mn}$. Given $A$ and $As + e [q]$, find $s$.

**Decisional-LWE**

Take $A$ uniform in $\mathbb{Z}_{q}^{mn \times n}$, $s$ uniform in $\mathbb{Z}_{q}^{n}$ and $e$ sampled from $\mathcal{N}_{\alpha q}^{mn}$. Distinguish between the distributions $(A, As + e [q])$ and uniform over $\mathbb{Z}_{q}^{mn \times n} \times \mathbb{Z}_{q}^{mn}$.

Regev proved that Dec-LWE is at least as hard as Comp-LWE. The adaptation to Id-LWE is not known to hold.
Encrypting with LWE

\[ A \quad s + \quad e \]

\[ A' \quad e' + \left\lfloor \frac{q}{2} \right\rfloor \cdot M \]
Encrypting with LWE

- Public key: \( A \in \mathbb{Z}_q^{mn \times n}, A' \in \mathbb{Z}_q^{n \times n}; \) private key: \( T_A \).
- Encrypting \( M \in \{0, 1\}^n \): generate \( s \in \mathbb{Z}_q^n, e \in \mathbb{Z}_q^{mn} \) and \( e' \in \mathbb{Z}_q^n \); compute \([As + e; A's + e' + \lfloor \frac{q}{2} \rfloor \cdot M]\).
- Decryption: recover \( s \) from the first part of the ciphertext, using \( T_A \); compute \( A's \) to obtain \( e' + \lfloor \frac{q}{2} \rfloor M \); round to the closest multiple of \( \lfloor \frac{q}{2} \rfloor \) to recover \( M \).
Encrypting with LWE

- Public key: $A \in \mathbb{Z}_q^{mn \times n}$, $A' \in \mathbb{Z}_q^{n \times n}$; private key: $T_A$.
- Encrypting $M \in \{0, 1\}^n$: generate $s \in \mathbb{Z}_q^n$, $e \in \mathbb{Z}_q^{mn}$ and $e' \in \mathbb{Z}_q^n$; compute $[As + e; A's + e' + \left\lfloor \frac{q}{2} \right\rfloor \cdot M]$.
- Decryption: recover $s$ from the first part of the ciphertext, using $T_A$; compute $A's$ to obtain $e' + \left\lfloor \frac{q}{2} \right\rfloor M$; round to the closest multiple of $\left\lfloor \frac{q}{2} \right\rfloor$ to recover $M$.

A CPA attack would lead to an algorithm for Decisional-LWE.
Encrypting with Id-LWE

- We cannot use the decisional variant of Id-LWE.
- But we have a trapdoor one-way function which is at least as hard to invert as solving Computational-Id-LWE.
Encrypting with Id-LWE

- We cannot use the decisional variant of Id-LWE.
- But we have a trapdoor one-way function which is at least as hard to invert as solving Computational-Id-LWE.
- There is a generic transformation from trapdoor OWF to CPA-secure encryption scheme (Goldreich-Levin).
  - Encryption: evaluate the OWF with a random $s$; let $\rho$ be the used random bits, seen as a vector in $\mathbb{Z}_2^\ell$; multiply $\rho$ with a random public Toeplitz matrix over $\mathbb{Z}_2$; use the output vector to mask the message $M$.
  - Decryption: use the trapdoor to recover $s$; apply the Toeplitz matrix to recover $M$. 
Encrypting with Id-LWE

- There is a generic transformation from trapdoor OWF to CPA-secure encryption scheme (Goldreich-Levin).

- Encryption: evaluate the OWF with a random $s$; let $\rho$ be the used random bits, seen as a vector in $\mathbb{Z}_2^\ell$; multiply $\rho$ with a random public Toeplitz matrix over $\mathbb{Z}_2$; use the output vector to mask the message $M$.

- Decryption: use the trapdoor to recover $s$; apply the Toeplitz matrix to recover $M$. 
Encrypting with Id-LWE

- There is a generic transformation from trapdoor OWF to CPA-secure encryption scheme (Goldreich-Levin).
- Encryption: evaluate the OWF with a random $s$; let $\rho$ be the used random bits, seen as a vector in $\mathbb{Z}_2^\ell$; multiply $\rho$ with a random public Toeplitz matrix over $\mathbb{Z}_2$; use the output vector to mask the message $M$.
- Decryption: use the trapdoor to recover $s$; apply the Toeplitz matrix to recover $M$.
- Encryption/decryption of $\tilde{\Omega}(n)$ bits in time $\tilde{O}(n)$.
- CPA-secure if $\text{Id-SVP} \tilde{O}(n^2)$ is hard for sub-exponential quantum algorithms.
- But impractical because of the generic transformation.
Plan

1- Background on Euclidean lattices.
2- The SIS problem, or how to hash.
3- The LWE problem, or how to encrypt.
4- Cryptanalysis.
5- More recent developments.
Attacking SIS/Id-SIS/LWE/Id-LWE

- The only known attack consists in finding a small vector/basis of the lattice $A^\perp = \{ s \in \mathbb{Z}^{mn} : sA = 0 \ [q] \}$.

- Generalized birthday attack: may be feasible if $m$ is large. Its cost is easily determined [MR’09].

- Lattice reduction: may be applied to a subset of the rows (trade-off between approximation factor and existence of short vectors).
Attacking SIS/Id-SIS/LWE/Id-LWE

- The only known attack consists in finding a small vector/basis of the lattice $A^\perp = \{s \in \mathbb{Z}^{mn} : sA = 0 \ [q]\}$.
- Generalized birthday attack: may be feasible if $m$ is large. Its cost is easily determined [MR’09].
- Lattice reduction: may be applied to a subset of the rows (trade-off between approximation factor and existence of short vectors).
Attacking SIS/Id-SIS/LWE/Id-LWE

- The only known attack consists in finding a small vector/basis of the lattice $A^\perp = \{ s \in \mathbb{Z}^{mn} : sA = 0 \ [q] \}$.  
- Generalized birthday attack: may be feasible if $m$ is large. Its cost is easily determined [MR’09].  
- Lattice reduction: may be applied to a subset of the rows (trade-off between approximation factor and existence of short vectors).
Attacking SIS/Id-SIS/LWE/Id-LWE

- The only known attack consists in finding a small vector/basis of the lattice $A^\perp = \{ s \in \mathbb{Z}^{mn} : sA = 0 \ [q] \}$.
- Generalized birthday attack: may be feasible if $m$ is large. Its cost is easily determined [MR’09].
- Lattice reduction: may be applied to a subset of the rows (trade-off between approximation factor and existence of short vectors).

But... although quite old (Lagrange, Gauss, Hermite, Minkowski, etc)... lattice reduction is not so well understood.
Lattice reduction

- Principle: start from an arbitrary basis of the lattice, and progressively improve it.
- Quality of a basis: measured by the Gram-Schmidt Orth.
Lattice reduction

- **Principle:** start from an arbitrary basis of the lattice, and progressively improve it.

- **Quality of a basis:** measured by the Gram-Schmidt Orth.

\[ b_i^* = \text{argmin} \| b_i + \sum_{j<i} \mathbb{R} b_j \| \]

- Quality measure: \( (\| b_i^* \|)_{i=1..n} \).

**Why?**

- The slower the \( \| b_i^* \| \)'s decrease, the more orthogonal.

- Their product is constant.

- If they decrease slowly, then \( b_1 \) must be small.
Lattice reduction

- Principle: start from an arbitrary basis of the lattice, and progressively improve it.
- Quality of a basis: measured by the Gram-Schmidt Orth.

\[ b_i^* = \text{argmin} \| b_i + \sum_{j<i} \mathbb{R} b_j \| \]

- Quality measure: \( (\| b_i^* \|)_{i=1..n} \).

Why?

- The slower the \( \| b_i^* \| \)'s decrease, the more orthogonal.
- Their product is constant.
- If they decrease slowly, then \( b_1 \) must be small.
Lattice reduction

- Principle: start from an arbitrary basis of the lattice, and progressively improve it.
- Quality of a basis: measured by the Gram-Schmidt Orth.
  \[
  \mathbf{b}_i^* = \arg\min \| \mathbf{b}_i + \sum_{j<i} \mathbb{R} \mathbf{b}_j \|
  \]
- Quality measure: \((\| \mathbf{b}_i^* \|)_{i=1..n}\).

Why?

- The slower the \(\| \mathbf{b}_i^* \|\)’s decrease, the more orthogonal.
- Their product is constant.
- If they decrease slowly, then \(\mathbf{b}_1\) must be small.
BKZ: a trade-off between LLL and HKZ

\[ \log \| b_i^* \| \]

LLL

\[ \log \| b_i^* \| \]

HKZ

\[ i \]
BKZ: a trade-off between LLL and HKZ

\[ \log \|b_i^*\| \]

LLL too weak

HKZ too costly

\[ \log \|b_i\| \]
BKZ: a trade-off between LLL and HKZ

- [Schnorr’87]: use HKZ within smaller-dimensional blocks.
- BKZ is the best practical variant [SE’94].
- Best theoretical variant: [GN’08].
BKZ: a trade-off between LLL and HKZ

- [Schnorr’87]: use HKZ within smaller-dimensional blocks.
- BKZ is the best practical variant [SE’94].
- Best theoretical variant: [GN’08].
Schnorr’s hierarchy

- Theoretical rule of the thumb for block-size $k$:  
  \[ \text{Cost} \ Poly(n) \cdot 2^k \text{ and } \text{SVP approximation factor } n^{n/k}. \]
  
- Seems satisfied by BKZ for small block-sizes.
  
- But the cost unexpectedly blows up with block-size $\approx 30$. 
Schnorr’s hierarchy

- Theoretical rule of the thumb for block-size $k$:
  Cost $\mathcal{P}oly(n) \cdot 2^k$ and SVP approximation factor $n^{n/k}$.
- Seems satisfied by BKZ for small block-sizes.
- But the cost unexpectedly blows up with block-size $\approx 30$.

**Warnings**

- The runtime of BKZ is not polynomial in the block-size.
- BKZ is the only implemented/available variant of Schnorr’s hierarchy.
Solving SVP: see workshop session

It is not known yet how far we can solve SVP.
Solving SVP: see workshop session

It is not known yet how far we can solve SVP.

- [KFP’83] is the best deterministic algorithm.
- Cost: time $n^{n/(2e)}$, space $\mathcal{P}oly(n)$ [HS’07].
- Tree pruning, parallelisation, hardware implementation (in progress).
Solving SVP: see workshop session

It is not known yet how far we can solve SVP.

- [KFP’83] is the best deterministic algorithm.
  Cost: time $n^{n/(2e)}$, space $\mathcal{P}oly(n)$ [HS’07].
- Tree pruning, parallelisation, hardware implementation (in progress).

- [AKS’01] is the best probabilistic algorithm.
  Cost: time $2^{3.2n}$, space $2^{1.3n}$ [MV’09].
- Fresh new result: time $2^{2.5n}$ and space $2^{1.2n}$ [PS’09].
- Heuristically: time $2^{0.4n}$ and space $2^{0.2n}$ [MV’09].
Plan

1- Background on Euclidean lattices.
2- The SIS problem, or how to hash.
3- The LWE problem, or how to encrypt.
4- Cryptanalysis.
5- More recent developments.
More recent developments

a- **Identity-based encryption.**

b- **Fully homomorphic encryption.**
(H)-IBE

- Identity-based encryption: encryption scheme for which the public key of a user is uniquely determined by its identity; the user’s private key is computed by a trusted authority, using a master private key. No need for a public key distribution infrastructure.
- Given as an open problem in 1984, by Shamir.
- First realization by Boneh and Franklin in 2001, using bilinear pairings on elliptic curves.
Identity-based encryption: encryption scheme for which the public key of a user is uniquely determined by its identity; the user’s private key is computed by a trusted authority, using a master private key. No need for a public key distribution infrastructure.

Given as an open problem in 1984, by Shamir.

First realization by Boneh and Franklin in 2001, using bilinear pairings on elliptic curves.

Hierarchical identity-based encryption: same as IBE, but each entity in level $k$ of a hierarchy can generate the private keys of all entities of lower levels in the hierarchy.
HIBE using LWE

- Encode an identity $id$ as a string of bits of length $\leq k$.
- An identity $id$ is higher in the hierarchy than an identity $id'$ if $id$ is a prefix of $id'$.
- The master has identity $\{}$. 
HIBE using LWE

- Encode an identity $id$ as a string of bits of length $\leq k$.
- An identity $id$ is higher in the hierarchy than an identity $id'$ if $id$ is a prefix of $id'$.
- The master has identity $\{\}$.
- Sample $A \in \mathbb{Z}_q^{mn \times n}$ together with a trapdoor $T_A$ (a short basis for $A^\perp$). These are the master’s keys.
- Generate $(A^0_1, A^1_1), \ldots, (A^0_k, A^1_k)$ iid uniformly in $\mathbb{Z}_q^{mn \times n}$.
- User $id = i_1 \ldots i_\ell$ has public key $A_{id}$, the vertical concatenation of $A, A^i_1, \ldots, A^i_\ell$.
- The private key of user $id$ is a short basis of $A_{id}^\perp$.
- The encryption scheme is the LWE encryption scheme.
Private key extraction

- Suppose $id$ is a prefix of $id'$. How does user $id$ extract a private key for user $id'$ from his/her own private key?
- How to obtain a $T_{A_{id}}$ from a $T_{A_{id'}}$?
- Writing the new rows as combinations of the previous ones suffices to obtain a basis of $A_{id'}^\perp$ with small GSO.
Private key extraction

- Suppose \( id \) is a prefix of \( id' \). How does user \( id \) extract a private key for user \( id' \) from his/her own private key?
- How to obtain a \( T_{A_{id}} \) from a \( T_{A_{id'}} \)?
- Writing the new rows as combinations of the previous ones suffices to obtain a basis of \( A_{id'}^\perp \) with small GSO.
Private key extraction

- Suppose \( id \) is a prefix of \( id' \). How does user \( id \) extract a private key for user \( id' \) from his/her own private key?
- How to obtain a \( T_{A_{id}} \) from a \( T_{A_{id'}} \)?
- Writing the new rows as combinations of the previous ones suffices to obtain a basis of \( A_{id'}^\perp \) with small GSO.

\[
T_A \\
A = 0 \\
A' = UA \\
\Rightarrow \\
T_A \\
0 \\
A = 0 \\
\begin{array}{|c|c|c|}
- \quad U & Id & A' \\
\end{array}
\]
Private key randomization

- But now $id'$ knows the private key of $id$!
- $id$ has to randomize the private key of $id'$ to hide its own.
  - Use the previous basis of $A_{id'}^\perp$ with small GSO to sample from $D_{A_{id'}^\perp,\sigma}$ for a small $\sigma$.
  - With sufficiently many samples, we obtain a full rank set of short vectors in $A_{id'}^\perp$.
  - Convert it into a short basis.
  - The output distribution is independent of the initial basis.
Private key randomization

- But now $id'$ knows the private key of $id$!
- $id$ has to randomize the private key of $id'$ to hide its own.
- Use the previous basis of $A_{id'}^\perp$ with small GSO to sample from $D_{A_{id'}^\perp,\sigma}$ for a small $\sigma$.
- With sufficiently many samples, we obtain a full rank set of short vectors in $A_{id'}^\perp$.
- Convert it into a short basis.
- The output distribution is independent of the initial basis.
Private key randomization

- But now $id'$ knows the private key of $id$!
- $id$ has to randomize the private key of $id'$ to hide its own.
- Use the previous basis of $A_{id'}$ with small GSO to sample from $D_{A_{id'},\sigma}$ for a small $\sigma$.
- With sufficiently many samples, we obtain a full rank set of short vectors in $A_{id'}$.
- Convert it into a short basis.
- The output distribution is independent of the initial basis.
Private key randomization

- But now $id'$ knows the private key of $id$!
- $id$ has to randomize the private key of $id'$ to hide its own.
- Use the previous basis of $A_{id'}^\perp$ with small GSO to sample from $D_{A_{id'}^\perp, \sigma}$ for a small $\sigma$.
- With sufficiently many samples, we obtain a full rank set of short vectors in $A_{id'}^\perp$.
- Convert it into a short basis.
- The output distribution is independent of the initial basis.
Concluding on IBE

- We have an HIBE that is secure under worst-case lattice assumptions, for selective identity CPA attacks..
- This leads to adaptative identity CPA secure HIBE, CCA2-secure encryption, etc.
- Similar techniques lead to signatures that are secure in the standard model (without the random oracle).
Concluding on IBE

- We have an HIBE that is secure under worst-case lattice assumptions, for selective identity CPA attacks.
- This leads to adaptative identity CPA secure HIBE, CCA2-secure encryption, etc.
- Similar techniques lead to signatures that are secure in the standard model (without the random oracle).

Open problems

- Improving the efficiency.
- The SVP approximation factor increases quickly with the number of levels in the hierarchy: $\gamma = n^{O(k)}$. Can we avoid this?
Recent developments

- Identity-based encryption.
- Fully homomorphic encryption.
Homomorphic encryption

Given $C_1 = \mathcal{E}(M_1)$ and $C_2 = \mathcal{E}(M_2)$, can we compute $\mathcal{E}(f(M_1, M_2))$ for some $f$, without decrypting?

E.g., for ElGamal: $g^{m_1} \cdot g^{m_2} = g^{m_1 + m_2}$.

An encryption scheme is fully homomorphic if any function (given as a circuit) of any number of $M_i$’s can be evaluated in the ciphertext domain:

$$\forall k, \forall f, \exists g: D[g(E(M_1), \ldots, E(M_k))] = f(M_1, \ldots, M_k).$$

The bit-size of the output of $g$ must be independent of the circuit size of $f$. 
Homomorphic encryption

- Given $C_1 = \mathcal{E}(M_1)$ and $C_2 = \mathcal{E}(M_2)$, can we compute $\mathcal{E}(f(M_1, M_2))$ for some $f$, without decrypting?
- E.g., for ElGamal: $g^{m_1} \cdot g^{m_2} = g^{m_1+m_2}$.
- An encryption scheme is fully homomorphic if any function (given as a circuit) of any number of $M_i$'s can be evaluated in the ciphertext domain:

$$\forall k, \forall f, \exists g : D[g(\mathcal{E}(M_1), \ldots, \mathcal{E}(M_k))] = f(M_1, \ldots, M_k).$$

- The bit-size of the output of $g$ must be independent of the circuit size of $f$. 
Homomorphic encryption

- Given $C_1 = \mathcal{E}(M_1)$ and $C_2 = \mathcal{E}(M_2)$, can we compute $\mathcal{E}(f(M_1, M_2))$ for some $f$, without decrypting?
- E.g., for ElGamal: $g^{m_1} \cdot g^{m_2} = g^{m_1+m_2}$.
- An encryption scheme is fully homomorphic if any function (given as a circuit) of any number of $M_i$’s can be evaluated in the ciphertext domain:

$$\forall k, \forall f, \exists g : \mathcal{D}[g(\mathcal{E}(M_1), \ldots, \mathcal{E}(M_k))] = f(M_1, \ldots, M_k).$$

- The bit-size of the output of $g$ must be independent of the circuit size of $f$.
Homomorphic encryption

- Given $C_1 = \mathcal{E}(M_1)$ and $C_2 = \mathcal{E}(M_2)$, can we compute $\mathcal{E}(f(M_1, M_2))$ for some $f$, without decrypting?
- E.g., for ElGamal: $g^{m_1} \cdot g^{m_2} = g^{m_1+m_2}$.
- An encryption scheme is fully homomorphic if any function (given as a circuit) of any number of $M_i$’s can be evaluated in the ciphertext domain:

$$\forall k, \forall f, \exists g : \mathcal{D}[g(\mathcal{E}(M_1), \ldots, \mathcal{E}(M_k))] = f(M_1, \ldots, M_k).$$

- The bit-size of the output of $g$ must be independent of the circuit size of $f$. 
Homomorphic encryption

- Given $C_1 = \mathcal{E}(M_1)$ and $C_2 = \mathcal{E}(M_2)$, can we compute $\mathcal{E}(f(M_1, M_2))$ for some $f$, without decrypting?
- E.g., for ElGamal: $g^{m_1} \cdot g^{m_2} = g^{m_1+m_2}$.
- An encryption scheme is fully homomorphic if any function (given as a circuit) of any number of $M_i$’s can be evaluated in the ciphertext domain:

$$\forall k, \forall f, \exists g : \mathcal{D}[g(\mathcal{E}(M_1), \ldots, \mathcal{E}(M_k))] = f(M_1, \ldots, M_k).$$

- The bit-size of the output of $g$ must be independent of the circuit size of $f$. 
Homomorphic encryption

**Given** $C_1 = \mathcal{E}(M_1)$ and $C_2 = \mathcal{E}(M_2)$, can we compute $\mathcal{E}(f(M_1, M_2))$ for some $f$, without decrypting?

E.g., for ElGamal: $g^{m_1} \cdot g^{m_2} = g^{m_1 + m_2}$.

An encryption scheme is fully homomorphic if any function (given as a circuit) of any number of $M_i$’s can be evaluated in the ciphertext domain:

$$\forall k, \forall f, \exists g : D[g(\mathcal{E}(M_1), \ldots, \mathcal{E}(M_k))] = f(M_1, \ldots, M_k).$$

The bit-size of the output of $g$ must be independent of the circuit size of $f$.

Many applications:
- Use untrusted parties to run programs (cloud computing).
- Search over private data (PIR), etc.
The 'holy grail' of cryptography

- The question was first asked by Rivest, Adleman and Dertouzous in 1978.
- Solved by Craig Gentry in 2009, using ideal lattices.
The 'holy grail' of cryptography

- The question was first asked by Rivest, Adleman and Dertouzous in 1978.
- Solved by Craig Gentry in 2009, using ideal lattices.
The 'holy grail' of cryptography

- The question was first asked by Rivest, Adleman and Dertouzous in 1978.
- Solved by Craig Gentry in 2009, using ideal lattices.

IBM announcement (25/06/09): An IBM Researcher has solved a thorny mathematical problem that has confounded scientists since the invention of public-key encryption several decades ago. The breakthrough, called "privacy homomorphism," or "fully homomorphic encryption," makes possible the deep and unlimited analysis of encrypted information [...] without sacrificing confidentiality.
A somewhat homomorphic scheme

- Sample a good basis $B_{j}^{sk}$ of an ideal lattice $J$ of “large” determinant, i.e., large minimum, large successive minima, large covering radius, etc.
- Let $B_{j}^{pk}$ be the HNF of $B_{j}^{sk}$.
- To encrypt $M \in \{0, 1\}[x]$, take a small random $r \in \mathbb{Z}[x]/(x^n + 1)$ and output $C = M + 2r \mod B_{j}^{pk}$.
- To decrypt: if $C$ is within distance $\ll \lambda(J)$ of $J$, then Babai’s rounding-off algorithm finds $M + 2r$:
  $$C - B_{j}^{sk} [(B_{j}^{sk})^{-1} C] \Rightarrow M + 2r.$$
A somewhat homomorphic scheme

- Sample a good basis $B_{sk}^j$ of an ideal lattice $J$ of “large” determinant, i.e., large minimum, large successive minima, large covering radius, etc.
- Let $B_{pk}^j$ be the HNF of $B_{sk}^j$.
- To encrypt $M \in \{0, 1\}[x]$, take a small random $r \in \mathbb{Z}[x]/(x^n + 1)$ and output $C = M + 2r \mod B_{pk}^j$.
- To decrypt: if $C$ is within distance $\ll \lambda(J)$ of $J$, then Babai’s rounding-off algorithm finds $M + 2r$:

$$C - B_{sk}^j [(B_{sk}^j)^{-1} C] \Rightarrow M + 2r.$$
A somewhat homomorphic scheme

- Sample a good basis $B_{sk}^J$ of an ideal lattice $J$ of “large” determinant, i.e., large minimum, large successive minima, large covering radius, etc.
- Let $B_{pk}^J$ be the HNF of $B_{sk}^J$.
- To encrypt $M \in \{0, 1\}[x]$, take a small random $r \in \mathbb{Z}[x]/(x^n + 1)$ and output $C = M + 2r \mod B_{pk}^J$.
- To decrypt: if $C$ is within distance $\ll \lambda(J)$ of $J$, then Babai’s rounding-off algorithm finds $M + 2r$:

$$C - B_{sk}^J \lfloor (B_{sk}^J)^{-1} C \rfloor \Rightarrow M + 2r.$$
A somewhat homomorphic scheme

- Sample a good basis $B_{sk}^J$ of an ideal lattice $J$ of “large” determinant, i.e., large minimum, large successive minima, large covering radius, etc.
- Let $B_{pk}^J$ be the HNF of $B_{sk}^J$.
- To encrypt $M \in \{0, 1\}[x]$, take a small random $r \in \mathbb{Z}[x]/(x^n + 1)$ and output $C = M + 2r \mod B_{pk}^J$.
- To decrypt: if $C$ is within distance $\ll \lambda(J)$ of $J$, then Babai’s rounding-off algorithm finds $M + 2r$:

$$C - B_{sk}^J \left[ (B_{sk}^J)^{-1} C \right] \Rightarrow M + 2r.$$
Correctness and security

- **Correctness.** We must have

\[ r_{Enc} := \max_{M,r} \| M + 2r \| < r_{Babai,B_{sk}^J}(J) =: r_{Dec}(J). \]

- **Security:** BDD must be hard to solve without \( B_{sk}^J \).
- With lattice reduction, in time \( \approx 2^k \) we can solve this BDD if \( r_{Enc} \leq 2^{n/k} \cdot r_{Dec} \). Gentry takes \( r_{Dec} \approx 2^{\sqrt{n}} \cdot r_{Enc} \).
Correctness and security

- **Correctness.** We must have

\[ r_{Enc} := \max_{\mathbf{M}, r} \| \mathbf{M} + 2r \| < r_{Babai,B_j}^{sk}(J) =: r_{Dec}(J). \]

- **Security:** BDD must be hard to solve without \( B_j^{sk} \).

  With lattice reduction, in time \( \approx 2^k \) we can solve this BDD if \( r_{Enc} \leq 2^{n/k} \cdot r_{Dec} \). Gentry takes \( r_{Dec} \approx 2^{\sqrt{n}} \cdot r_{Enc} \).
Correctness and security

- Correctness. We must have

\[ r_{Enc} := \max_{M, r} \| M + 2r \| < r_{Babai,B_j^{sk}}(J) =: r_{Dec}(J). \]

- Security: BDD must be hard to solve without \( B_j^{sk} \).
- With lattice reduction, in time \( \approx 2^k \) we can solve this BDD if \( r_{Enc} \leq 2^{n/k} \cdot r_{Dec} \). Gentry takes \( r_{Dec} \approx 2^{\sqrt{n}} \cdot r_{Enc} \).
Correctness and security

- Correctness. We must have

\[ r_{Enc} := \max_{M,r} \| M + 2r \| < r_{Babai,B_j^{sk}}(J) =: r_{Dec}(J). \]

- Security: BDD must be hard to solve without \( B_j^{sk} \).

- With lattice reduction, in time \( \approx 2^k \) we can solve this BDD if \( r_{Enc} \leq 2^{n/k} \cdot r_{Dec} \). Gentry takes \( r_{Dec} \approx 2^{\sqrt{n}} \cdot r_{Enc} \).

If \( J \) and \( B_j^{sk} \) are chosen according to a well specified efficiently samplable distribution, if \( M \in \{0,1\} \) and if \( r \) is sampled from some discrete Gaussian, then the latter scheme can be made CPA-secure under the assumption that \( \text{Id-SVP}_\gamma \) is hard to solve for quantum polynomial-time algorithms, for some \( \gamma \) that grows faster than any polynomial in \( n \).
Correctness and security

- Security: BDD must be hard to solve without $B_{sk}^J$.
- With lattice reduction, in time $\approx 2^k$ we can solve this BDD if $r_{Enc} \leq 2^{n/k} \cdot r_{Dec}$. Gentry takes $r_{Dec} \approx 2^{\sqrt{n}} \cdot r_{Enc}$.

If $J$ and $B_{sk}^J$ are chosen according to a well specified efficiently samplable distribution, if $M \in \{0, 1\}$ and if $r$ is sampled from some discrete Gaussian, then the latter scheme can be made CPA-secure under the assumption that $\text{Id-SVP}_\gamma$ is hard to solve for quantum polynomial-time algorithms, for some $\gamma$ that grows faster than any polynomial in $n$.

This is a dimension-preserving worst-case to average-case reduction, but much weaker than the Id-SIS/Id-LWE ones.
Why is it homomorphic?

- To encrypt $M \in \{0, 1\}[x]$, take a small random $r \in \mathbb{Z}[x]/(x^n + 1)$ and output $C = M + 2r \mod B_j^{pk}$.

- Addition: $C_i = M_i + 2r_i \mod B_j^{pk}$ implies

  $$C_1 + C_2 = (M_1 + M_2) + 2(r_1 + r_2) \mod B_j^{pk}.$$ 

- Multiplication (we have polynomials):

  $$C_1 \times C_2 = (M_1 \times M_2) + 2(r_1 \times M_2 + r_2 \times M_1 + 2r_1 \times r_2) \mod B_j^{pk}.$$
Why is it homomorphic?

- To encrypt $M \in \{0, 1\}[x]$, take a small random $r \in \mathbb{Z}[x]/(x^n + 1)$ and output $C = M + 2r \mod B_{pk}^n$.

- Addition: $C_i = M_i + 2r_i \mod B_{pk}^n$ implies

  $$C_1 + C_2 = (M_1 + M_2) + 2(r_1 + r_2) \mod B_{pk}^n.$$ 

- Multiplication (we have polynomials):

  $$C_1 \times C_2 = (M_1 \times M_2) + 2(r_1 \times M_2 + r_2 \times M_1 + 2r_1 \times r_2) \mod B_{pk}^n.$$
Why is it homomorphic?

- To encrypt $M \in \{0, 1\}[x]$, take a small random $r \in \mathbb{Z}[x]/(x^n + 1)$ and output $C = M + 2r \mod B_{j}^{pk}$.

- Addition: $C_i = M_i + 2r_i \mod B_{j}^{pk}$ implies

$$C_1 + C_2 = (M_1 + M_2) + 2(r_1 + r_2) \mod B_{j}^{pk}.$$

- Multiplication (we have polynomials):

$$C_1 \times C_2 = (M_1 \times M_2) + 2(r_1 \times M_2 + r_2 \times M_1 + 2r_1 \times r_2) \mod B_{j}^{pk}.$$
Why is it only “somewhat” homomorphic?
Why is it only “somewhat” homomorphic?

The more operations are applied the further away from $J$. 
Why is it only “somewhat” homomorphic?

The more operations are applied the further away from $J$.

- $\text{dist}(\mathbf{C}_1 + \mathbf{C}_2, J) \leq \text{dist}(\mathbf{C}_1, J) + \text{dist}(\mathbf{C}_2, J)$.
- $\text{dist}(\mathbf{C}_1 \times \mathbf{C}_2, J) \leq K \cdot \text{dist}(\mathbf{C}_1, J) \cdot \text{dist}(\mathbf{C}_2, J)$. 
Why is it only “somewhat” homomorphic?

The more operations are applied the further away from $J$.

- $\text{dist}(\mathbf{C}_1 + \mathbf{C}_2, J) \leq \text{dist}(\mathbf{C}_1, J) + \text{dist}(\mathbf{C}_2, J)$.
- $\text{dist}(\mathbf{C}_1 \times \mathbf{C}_2, J) \leq K \cdot \text{dist}(\mathbf{C}_1, J) \cdot \text{dist}(\mathbf{C}_2, J)$.

E.g.: If we have $t$ ciphertexts to multiply, then $K^t \cdot r_{Enc}^t$ may become larger than $r_{Dec}$.
Making the scheme fully homomorphic

- If many operations have been applied, we try to “refresh” the ciphertext.
- We cannot decrypt using the private key.
- Trick: encode $C = E(M, J_{pk}^1)$ further using a second public key, and decode homomorphically using the encryption of the first private key.

$$D(E(C, J_{pk}^2), E(J_{sk}^1, J_{pk}^2)) = E(D(C, J_{sk}^1), J_{pk}^2)$$

- Refreshing as many times as required, we can apply any circuit privately.
Making the scheme fully homomorphic

- If many operations have been applied, we try to “refresh” the ciphertext.
- We cannot decrypt using the private key.
- Trick: encode $C = \mathcal{E}(M, J_{1}^{pk})$ further using a second public key, and decode homomorphically using the encryption of the first private key.

$$D \left( \mathcal{E}(C, J_{1}^{pk}), \mathcal{E}(J_{1}^{pk}, J_{2}^{pk}) \right) = \mathcal{E} \left( D \left( C, J_{1}^{sk} \right), J_{2}^{sk} \right).$$

- Refreshing as many times as required, we can apply any circuit privately.
Making the scheme fully homomorphic

- If many operations have been applied, we try to “refresh” the ciphertext.
- We cannot decrypt using the private key.
- Trick: encode $C = \mathcal{E}(M, J_{1}^{pk})$ further using a second public key, and decode homomorphically using the encryption of the first private key.

\[
\mathcal{D} \left( \mathcal{E}(C, J_{2}^{pk}), \mathcal{E}(J_{1}^{sk}, J_{2}^{pk}) \right) = \mathcal{E}(\mathcal{D}(C, J_{1}^{sk}), J_{2}^{pk}).
\]

- Refreshing as many times as required, we can apply any circuit privately.
Making the scheme fully homomorphic

- If many operations have been applied, we try to “refresh” the ciphertext.
- We cannot decrypt using the private key.
- Trick: encode $C = \mathcal{E}(M, J_{1}^{pk})$ further using a second public key, and decode homomorphically using the encryption of the first private key.

\[ D \left( \mathcal{E}(C, J_{2}^{pk}), \mathcal{E}(J_{1}^{sk}, J_{2}^{pk}) \right) = \mathcal{E}(D \left( C, J_{1}^{sk} \right), J_{2}^{pk}) \].

- Refreshing as many times as required, we can apply any circuit privately.
Making the scheme fully homomorphic

- If many operations have been applied, we try to “refresh” the ciphertext.
- We cannot decrypt using the private key.
- Trick: encode $C = \mathcal{E}(M, J_{1}^{pk})$ further using a second public key, and decode homomorphically using the encryption of the first private key.

$$\mathcal{D} \left( \mathcal{E}(C, J_{2}^{pk}), \mathcal{E}(J_{1}^{sk}, J_{2}^{pk}) \right) = \mathcal{E}(\mathcal{D}(C, J_{1}^{sk}), J_{2}^{pk})$$

- Refreshing as many times as required, we can apply any circuit privately.
The decryption circuit

- Problem: Is the decryption circuit simple enough so that it can be itself be applied without refreshing?
- Decryption: $C - B^s_k [(B^s_k)^{-1}C]$ provides $M + 2r$.
- $B^s_k [(B^s_k)^{-1}C]$ seems too complicated.
- We need to “squash” the decryption circuit.
Problem: Is the decryption circuit simple enough so that it can be itself be applied without refreshing?
Decryption: \( \mathbf{C} - B_j^{sk} \lfloor (B_j^{sk})^{-1} \mathbf{C} \rfloor \) provides \( \mathbf{M} + 2r \).
\( B_j^{sk} \lfloor (B_j^{sk})^{-1} \mathbf{C} \rfloor \) seems too complicated.
We need to “squash” the decryption circuit.
The decryption circuit

- Problem: Is the decryption circuit simple enough so that it can be itself be applied without refreshing?
- Decryption: $C - B_j^{sk} \lfloor (B_j^{sk})^{-1} C \rfloor$ provides $M + 2r$.
- $B_j^{sk} \lfloor (B_j^{sk})^{-1} C \rfloor$ seems too complicated.
- We need to “squash” the decryption circuit.
The decryption circuit

- Problem: Is the decryption circuit simple enough so that it can be itself be applied without refreshing?
- Decryption: \( C - B^s_j \lfloor (B^s_j)^{-1} C \rfloor \) provides \( M + 2r \).
- \( B^s_j \lfloor (B^s_j)^{-1} C \rfloor \) seems too complicated.
- We need to “squash” the decryption circuit.

Outline of Gentry’s solution:

- There exists \( v^s_j \) with: \( \forall C: B^s_j \lfloor (B^s_j)^{-1} C \rfloor = \lfloor v^s_j C \rfloor \).
- Generate random public \( v_i \)'s with a secret sparse subset \( S \) which sums to \( v^s_j \): \( \sum_{i \in S} v_i = v^s_j \).
- The \( v_i C \)'s can be computed, and then the decryption reduces to summing up the few correct ones.
The decryption circuit

- Problem: Is the decryption circuit simple enough so that it can be itself be applied without refreshing?
- Decryption: \( C - B_{j}^{sk} \lfloor (B_{j}^{sk})^{-1}C \rfloor \) provides \( M + 2r \).
- \( B_{j}^{sk} \lfloor (B_{j}^{sk})^{-1}C \rfloor \) seems too complicated.
- We need to “squash” the decryption circuit.

Outline of Gentry’s solution:

- There exists \( v_{j}^{sk} \) with: \( \forall C : B_{j}^{sk} \lfloor (B_{j}^{sk})^{-1}C \rfloor = \lfloor v_{j}^{sk}C \rfloor \).
- Generate random public \( v_{i} \)'s with a secret sparse subset \( S \) which sums to \( v_{j}^{sk}: \sum_{i \in S} v_{i} = v_{j}^{sk} \).
- The \( v_{i}C \)'s can be computed, and then the decryption reduces to summing up the few correct ones.
The decryption circuit

- Problem: Is the decryption circuit simple enough so that it can be itself be applied without refreshing?
- Decryption: \( C - B_j^{sk} \lfloor (B_j^{sk})^{-1} C \rfloor \) provides \( M + 2r \).
- \( B_j^{sk} \lfloor (B_j^{sk})^{-1} C \rfloor \) seems too complicated.
- We need to “squash” the decryption circuit.

Outline of Gentry’s solution:

- There exists \( v_j^{sk} \) with: \( \forall C : B_j^{sk} \lfloor (B_j^{sk})^{-1} C \rfloor = \lfloor v_j^{sk} C \rfloor \).
- Generate random public \( v_i \)'s with a secret sparse subset \( S \) which sums to \( v_j^{sk} \): \( \sum_{i \in S} v_i = v_j^{sk} \).
- The \( v_i C \)'s can be computed, and then the decryption reduces to summing up the few correct ones.
The decryption circuit

- Problem: Is the decryption circuit simple enough so that it can be itself be applied without refreshing?

- Decryption: \( C - B_{j^k} \lfloor (B_{j^k})^{-1} C \rfloor \) provides \( M + 2r \).

- \( B_{j^k} \lfloor (B_{j^k})^{-1} C \rfloor \) seems too complicated.

- We need to “squash” the decryption circuit.

Outline of Gentry’s solution:

- There exists \( v_{j^k} \) with: \( \forall C : B_{j^k} \lfloor (B_{j^k})^{-1} C \rfloor = \lfloor v_{j^k} C \rfloor \).

- Generate random public \( v_i \)’s with a secret sparse subset \( S \) which sums to \( v_{j^k} \): \( \sum_{i \in S} v_i = v_{j^k} \).

- The \( v_i C \)’s can be computed, and then the decryption reduces to summing up the few correct ones.
Plan

1- Background on Euclidean lattices.
2- The SIS problem, or how to hash.
3- The LWE problem, or how to encrypt.
4- Cryptanalysis.
5- More recent developments.
Conclusion

- The schemes are becoming more and more efficient, in particular thanks to structured matrices / ideal lattices.
- Lattice reduction is improving.

- But still not many schemes are implemented.
- Lattice reduction can probably still be improved much.
- Mainly one library used for cryptanalysis (Shoup’s NTL), and it is known to behave oddly [GN’08].
Conclusion

- The schemes are becoming more and more efficient, in particular thanks to structured matrices / ideal lattices.
- Lattice reduction is improving.
- But still not many schemes are implemented.
- Lattice reduction can probably still be improved much.
- Mainly one library used for crytanalysis (Shoup’s NTL), and it is known to behave oddly [GN’08].
Conclusion

- The schemes are becoming more and more efficient, in particular thanks to structured matrices / ideal lattices.
- Lattice reduction is improving.
- But still not many schemes are implemented.
- Lattice reduction can probably still be improved much.
- Mainly one library used for cryptanalysis (Shoup’s NTL), and it is known to behave oddly [GN’08].
Conclusion

- The schemes are becoming more and more efficient, in particular thanks to structured matrices / ideal lattices.
- Lattice reduction is improving.
- But still not many schemes are implemented.
- Lattice reduction can probably still be improved much.
- Mainly one library used for cryptanalysis (Shoup’s NTL), and it is known to behave oddly [GN’08].
Open problems

- Can we adapt (some of) the techniques to linear codes?
- Can quantum computers improve lattice algorithms?
- Can we use lattice algorithms to factor integers or compute discrete logarithms?
- Are ideal lattices weaker than general lattices?
- Assess the practical limits of lattice reduction.
Open problems

- Can we adapt (some of) the techniques to linear codes?
- Can quantum computers improve lattice algorithms?
- Can we use lattice algorithms to factor integers or compute discrete logarithms?
- Are ideal lattices weaker than general lattices?
- Assess the practical limits of lattice reduction.
Open problems

- Can we adapt (some of) the techniques to linear codes?
- Can quantum computers improve lattice algorithms?
- Can we use lattice algorithms to factor integers or compute discrete logarithms?
- Are ideal lattices weaker than general lattices?
- Assess the practical limits of lattice reduction.
Open problems

- Can we adapt (some of) the techniques to linear codes?
- Can quantum computers improve lattice algorithms?
- Can we use lattice algorithms to factor integers or compute discrete logarithms?
- Are ideal lattices weaker than general lattices?
- Assess the practical limits of lattice reduction.
Open problems

- Can we adapt (some of) the techniques to linear codes?
- Can quantum computers improve lattice algorithms?
- Can we use lattice algorithms to factor integers or compute discrete logarithms?
- Are ideal lattices weaker than general lattices?
- Assess the practical limits of lattice reduction.
Open problems

- Can we adapt (some of) the techniques to linear codes?
- Can quantum computers improve lattice algorithms?
- Can we use lattice algorithms to factor integers or compute discrete logarithms?
- Are ideal lattices weaker than general lattices?
- Assess the practical limits of lattice reduction.