Lattice-based (Post Quantum) Cryptography

Divesh Aggarwal

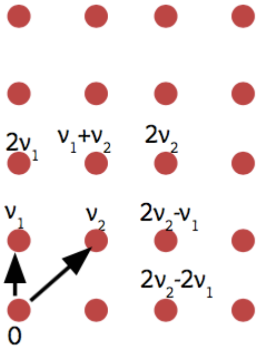
Center of Quantum Technologies, Singapore

February 8, 2018

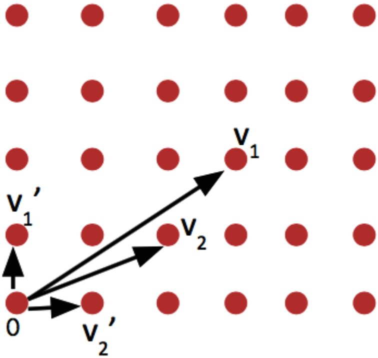
Lattices

- A lattice is a set of points
- $L = \{a_1 v_1 + \cdots + a_n v_n \mid a_i \text{ integers}\}$.

  for some linearly independent vectors $v_1, \ldots, v_n \in \mathbb{R}^n$.
- We call $v_1, \ldots, v_n$ a basis of L, and n the dimension of the lattice.



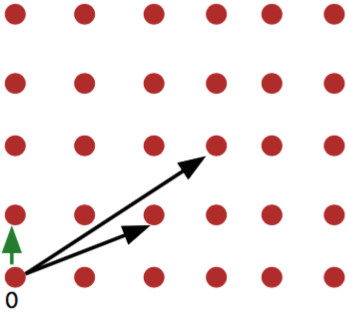
Basis is Not Unique



Good Basis: $v_1', \ v_2'$

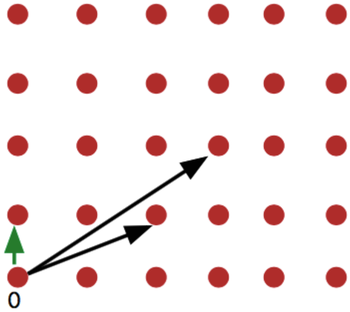
Bad Basis: $v_1, \ v_2$

Hard Problems



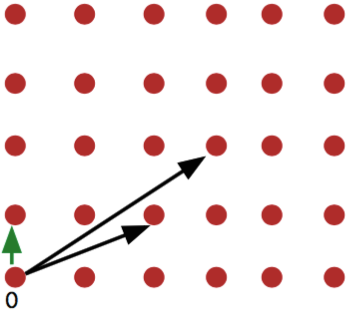
- SVP: Given a lattice, find the shortest non-zero vector (of length λ_1).

Hard Problems



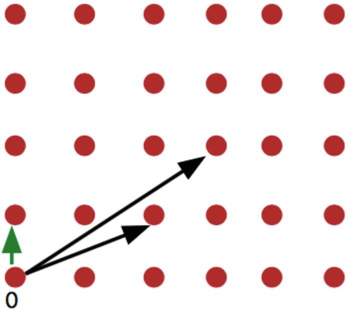
- SVP: Given a lattice, find the shortest non-zero vector (of length λ_1).
- ApproxSVP$_\gamma$: Given a lattice basis, find a vector of length $\gamma \cdot \lambda_1$.

Hard Problems



- SVP: Given a lattice, find the shortest non-zero vector (of length λ_1).
- ApproxSVP$_\gamma$: Given a lattice basis, find a vector of length $\gamma \cdot \lambda_1$.
- GapSVP$_\gamma$: Given a basis and d, decide whether $\lambda_1 \leq d$ or $\lambda_1 > \gamma \cdot d$.

Hard Problems



- SVP: Given a lattice, find the shortest non-zero vector (of length λ_1).
- ApproxSVP$_\gamma$: Given a lattice basis, find a vector of length $\gamma \cdot \lambda_1$.
- GapSVP$_\gamma$: Given a basis and d, decide whether $\lambda_1 \leq d$ or $\lambda_1 > \gamma \cdot d$.
- There are also other hard problems like CVP, SIVP

GapSVP$_\gamma$ - Algorithms and Complexity

γ	$2^{(\log n)^{1-\epsilon}}$	$\sqrt{n}$	$n^{O(1)}$	$(1+\epsilon)^n$
	NP Hard Problem [Ajt98,..., HR07]	$\in$ co-NP [GG98, AR05]	Cryptography [Ajtai96,...]	$\in$ P [LLL82, Sch87]

- Fastest known algorithms for $\gamma = r^{n/r}$ run in $2^{O(r)}\text{poly}(n)$ time, i.e.,

  For $\gamma = \text{poly}(n)$, the running time is exponential in n .

- In particular, fastest known algorithm for γ close to 1 runs in time $2^{n+o(n)}$ [ADRS15,AS18]

- Under the Gap exponential time hypothesis, no algorithm can solve GapSVP$_\gamma$ for a constant $\gamma \approx 1$ in time better than 2^{cn} for some constant c [AS17].

Lattices and Cryptography

- The first applications included attacking knapsack-based cryptosystems [LagOdl85] and variants of RSA [Has85,Cop01].
- Lattices began to be used to create cryptography starting with a breakthrough work of Ajtai[Ajt96].
- Cryptography based on lattices has many advantages compared with ‘traditional’ cryptography like RSA:
  - It has strong, mathematically proven, security.
  - It is believed to be resistant to quantum computers.
  - In some cases, it is much faster.
  - It can do more, e.g., fully homomorphic encryption, which is one of the most important cryptographic primitives.

Lattice-based Crypto

- Public-key Encryption [Reg05,KTX07,PKW08]
- CCA-Secure PKE [PW08,Pei09].
- Identity-based Encryption [GPV08]
- Oblivious Transfer [PVW08]
- Circular Secure Encryption [ACPS09]
- Hierarchical Identity-based Encryption [Gen09,CHKP09,ABB09].
- Fully Homomorphic Encryption [Gen09,BV11,Bra12].
- And more...

Talk Outline

- GGH Public Key Encryption Scheme
- LWE Problem and Applications
- Efficiency from Rings
- Recent Implementations.

GGH Public Key Encryption Scheme

Public key encryption

Suppose Alice wants to send a message m privately to Bob over a public channel.

Key Generation: Bob generates a pair (pk, sk).

Encryption: Alice sends $c = \text{Enc}(pk, m)$ to Bob.

Decryption: Bob obtains the message $m = Dec(sk, m)$.

Public key encryption

Suppose Alice wants to send a message m privately to Bob over a public channel.

Key Generation: Bob generates a pair (pk, sk).

Encryption: Alice sends $c = \text{Enc}(pk, m)$ to Bob.

Decryption: Bob obtains the message $m = Dec(sk, m)$.

Security: An eavesdropper shouldn't learn anything about the message given the public-key and the ciphertext.

Digital Signatures

Suppose you wish to digitally sign the message m.

Key Generation: The algorithm generates a pair (pk, sk).

Sign: A tag for the message is computed using the signing algorithm

$$t = \text{Sign}(sk, m) .$$

Verify: The signature can be verified using the public key

$$\text{Verify}(pk, t', m') \in \{\text{True}, \text{False}\} .$$

Digital Signatures

Suppose you wish to digitally sign the message m.

Key Generation: The algorithm generates a pair (pk, sk).

Sign: A tag for the message is computed using the signing algorithm

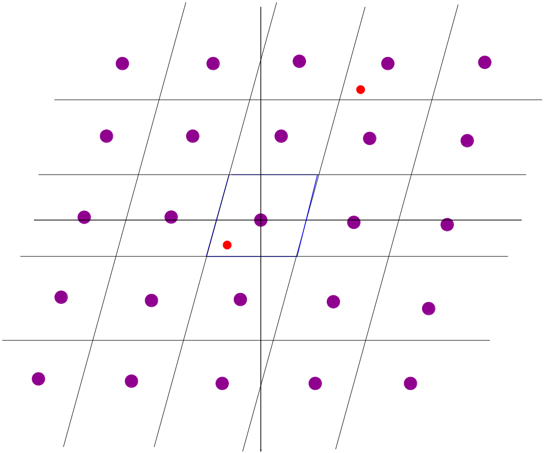
$$t = \text{Sign}(sk, m) .$$

Verify: The signature can be verified using the public key

$$\text{Verify}(pk, t', m') \in \{\text{True}, \text{False}\} .$$

Security: An eavesdropper shouldn't be able to forge a signature given a valid signature and the public key.

Reducing a vector modulo a basis

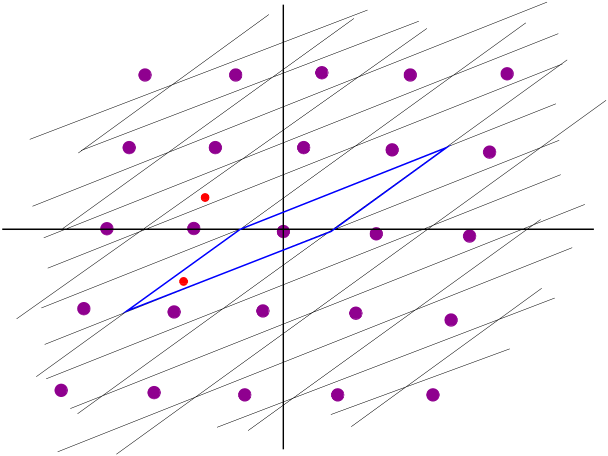


$u \mod B = u - B\lceil B^{-1}t\rfloor.$

If $u = \alpha_1 \cdot u_1 + \cdots + \alpha_n \cdot u_n,$ then

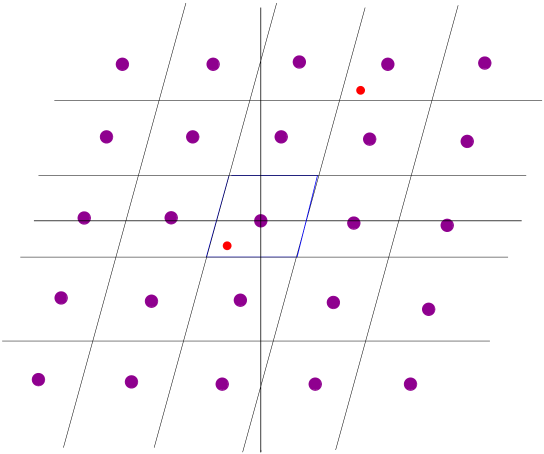
$$u \mod B = (\alpha_1 - \lfloor\alpha_1\rceil) \cdot u_1 + \cdots + (\alpha_n - \lfloor\alpha_n\rceil) \cdot u_n \ .$$

Bad Basis



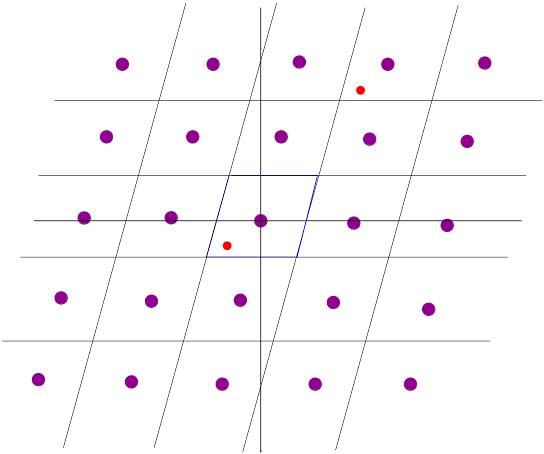
$u \mod B$ is likely a long vector.

Good Basis



$u \mod B$ is a short vector.

Good Basis

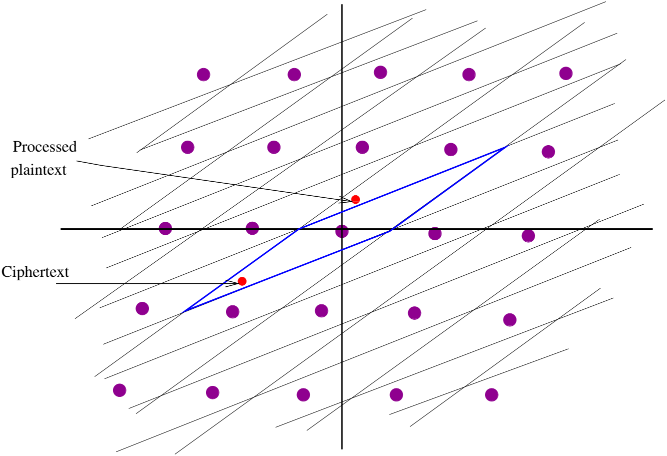


$u \mod B$ is a short vector.

Note that $u - u \mod B$ is a lattice vector close to u.

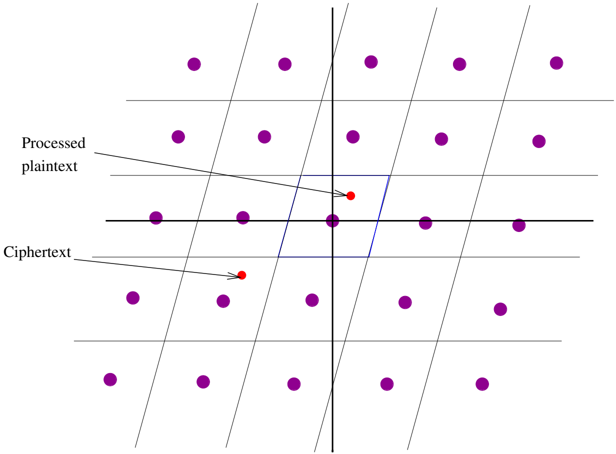
The encryption scheme

- Map the message m to a vector close to the origin.
- Encryption: $c = m \mod B_{pk}$ (public basis)



The encryption scheme

- Map the message m to a vector close to the origin.
- Encryption: $c = m \mod B_{pk}$ (public basis)
- Decryption: $m = c \mod B_{sk}$ (secret basis)



More on the GGH Scheme

- There is a dual digital signature scheme based on the same principle.
  - Map the message m to a random vector in space.
  - Signature: $t = m - m \mod B_{sk}$ (secret(good) basis) resulting in a lattice vector close to m.
  - Verification: Use the public (bad) basis to check whether t is a lattice vector and that $t - m$ is short.

More on the GGH Scheme

- There is a dual digital signature scheme based on the same principle.
  - Map the message m to a random vector in space.
  - Signature: $t = m - m \mod B_{sk}$ (secret(good) basis) resulting in a lattice vector close to m.
  - Verification: Use the public (bad) basis to check whether t is a lattice vector and that $t - m$ is short.

- Nguyen in 1999 pointed out a flaw in the GGH scheme. He showed that every ciphertext reveals information about the plaintext.

More on the GGH Scheme

- There is a dual digital signature scheme based on the same principle.
  - Map the message m to a random vector in space.
  - Signature: $t = m - m \mod B_{sk}$ (secret(good) basis) resulting in a lattice vector close to m.
  - Verification: Use the public (bad) basis to check whether t is a lattice vector and that $t - m$ is short.
- Nguyen in 1999 pointed out a flaw in the GGH scheme. He showed that every ciphertext reveals information about the plaintext.
- The principle of GGH, however, has been used in several follow-up schemes.

More on the GGH Scheme

- There is a dual digital signature scheme based on the same principle.
  - Map the message m to a random vector in space.
  - Signature: $t = m - m \mod B_{sk}$ (secret(good) basis) resulting in a lattice vector close to m.
  - Verification: Use the public (bad) basis to check whether t is a lattice vector and that $t - m$ is short.
- Nguyen in 1999 pointed out a flaw in the GGH scheme. He showed that every ciphertext reveals information about the plaintext.
- The principle of GGH, however, has been used in several follow-up schemes.
- In fact, the first fully homomorphic encryption scheme candidate by Gentry [2009] was based on the same principle.

Talk Outline

- GGH Public Key Encryption Scheme
- LWE Problem and Applications
- Efficiency from Rings
- Recent Implementations.

LWE Problem and Applications

Learning with Errors Problem [Regev05]

- Parameters: Dimension n, modulus $q = \text{poly}(n)$, error distribution
- Search: Find uniformly random secret $\mathbf{s} \in \mathbb{Z}_q^n$ given

$$\mathbf{A} = \begin{bmatrix} A_{11} & A_{12} & A_{13} & \dots & A_{1n} \\ A_{21} & A_{22} & A_{23} & \dots & A_{2n} \\ \dots & \dots & \dots & \dots & \dots \\ A_{m1} & A_{m2} & A_{m3} & \dots & A_{mn} \end{bmatrix} \quad \text{and} \quad \mathbf{b} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e} \,,$$

where $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$ is chosen uniformly at random and $\mathbf{e} \in \mathbb{Z}_q^m$ is some 'short noise distribution'.

Learning with Errors Problem [Regev05]

- Parameters: Dimension n, modulus $q = \text{poly}(n)$, error distribution
- Search: Find uniformly random secret $\mathbf{s} \in \mathbb{Z}_q^n$ given

$$\mathbf{A} = \begin{bmatrix} A_{11} & A_{12} & A_{13} & \dots & A_{1n} \\ A_{21} & A_{22} & A_{23} & \dots & A_{2n} \\ \dots & \dots & \dots & \dots & \dots \\ A_{m1} & A_{m2} & A_{m3} & \dots & A_{mn} \end{bmatrix} \quad \text{and} \quad \mathbf{b} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e} \,,$$

where $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$ is chosen uniformly at random and $\mathbf{e} \in \mathbb{Z}_q^m$ is some 'short noise distribution'.

- Decision: Distinguish $(\mathbf{A}, \mathbf{b})$ from $(\mathbf{A}, \mathbf{u})$ where $\mathbf{u}$ is uniform in $\mathbb{Z}_q^m$.

Learning with Errors Problem [Regev05]

- Parameters: Dimension n, modulus $q = \text{poly}(n)$, error distribution
- Search: Find uniformly random secret $\mathbf{s} \in \mathbb{Z}_q^n$ given

$$\mathbf{A} = \begin{bmatrix} A_{11} & A_{12} & A_{13} & \ldots & A_{1n} \\ A_{21} & A_{22} & A_{23} & \ldots & A_{2n} \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ A_{m1} & A_{m2} & A_{m3} & \ldots & A_{mn} \end{bmatrix} \quad \text{and} \quad \mathbf{b} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e}\,,$$

  where $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$ is chosen uniformly at random and $\mathbf{e} \in \mathbb{Z}_q^m$ is some 'short noise distribution'.
- Decision: Distinguish $(\mathbf{A}, \mathbf{b})$ from $(\mathbf{A}, \mathbf{u})$ where $\mathbf{u}$ is uniform in $\mathbb{Z}_q^m$.
- Worst case to Average Case:

  Break Crypto $\Longrightarrow$ Decision LWE $\Longrightarrow$ Search LWE $\Longrightarrow$ $\text{GapSVP}_{\text{poly}(n)}$

A PKE scheme from decision LWE hardness

Suppose Alice wants to send a bit $\mu \in \{0,1\}$ privately to Bob over a public channel.

Key Generation: Bob chooses q, m, n and $\mathbf{A} \in \mathbb{Z}_q^{m\times n}$, $\mathbf{s} \in \mathbb{Z}_q^n$ are chosen uniformly at random and $\mathbf{e} \in \mathbb{Z}_q^m$ is chosen from the ‘LWE noise distribution’. Then

$$pk = \mathbf{A},\ \mathbf{b} = \mathbf{A}\cdot\mathbf{s} + \mathbf{e}, \qquad sk = \mathbf{s}\ .$$

Encryption: Alice chooses uniform $r \in \{0,1\}^m$ and sends

$$c\ =\ (r\cdot\mathbf{A},\quad r\cdot\mathbf{b} + \mu\cdot\lfloor\frac{q}{2}\rfloor)\ .$$

Decryption: Bob on receiving the ciphertext (C_1, c_2) checks whether $c_2 - C_1\cdot s$ is closer to 0 or $\lfloor\frac{q}{2}\rfloor$ and hence deciphers the message bit μ.

A PKE scheme from decision LWE hardness

Suppose Alice wants to send a bit $\mu \in \{0, 1\}$ privately to Bob over a public channel.

Key Generation: Bob chooses q, m, n and $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$, $\mathbf{s} \in \mathbb{Z}_q^n$ are chosen uniformly at random and $\mathbf{e} \in \mathbb{Z}_q^m$ is chosen from the 'LWE noise distribution'. Then

$$pk = \mathbf{A},\ \mathbf{b} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e}, \qquad sk = \mathbf{s}\ .$$

Encryption: Alice chooses uniform $r \in \{0, 1\}^m$ and sends

$$c\ =\ (r \cdot \mathbf{A},\quad r \cdot \mathbf{b} + \mu \cdot \lfloor \frac{q}{2} \rfloor)\ .$$

Decryption: Bob on receiving the ciphertext (C_1, c_2) checks whether $c_2 - C_1 \cdot s$ is closer to 0 or $\lfloor \frac{q}{2} \rfloor$ and hence deciphers the message bit μ.

Security is almost immediate from the Decision LWE Hardness assumption. The ciphertext looks random given the public key.

Talk Outline

- GGH Public Key Encryption Scheme
- LWE Problem and Applications
- Efficiency from Rings
- Recent Implementations.

Efficiency from Rings.

How efficient is LWE?

- Getting one pseudorandom $b_i \in \mathbb{Z}_q$ requires an n-dimensional inner product modulo q.
- Cryptosystems have larger keys of size larger than $n^2 \log q$.
- Wishful thinking:

$$(a_1, a_2, \ldots, a_n) \star (s_1, s_2, \ldots, s_n) + (e_1, e_2, \ldots, e_n) = (b_1, b_2, \ldots, b_n) .$$

  - Get n pseudorandom elements in $\mathbb{Z}_q$ from one inner product.
  - Replace every n^2 length key by a key of length n.

How efficient is LWE?

- Getting one pseudorandom $b_i \in \mathbb{Z}_q$ requires an n-dimensional inner product modulo q.
- Cryptosystems have larger keys of size larger than $n^2 \log q$.
- Wishful thinking:

$$(a_1, a_2, \ldots, a_n) \star (s_1, s_2, \ldots, s_n) + (e_1, e_2, \ldots, e_n) = (b_1, b_2, \ldots, b_n) .$$

  - Get n pseudorandom elements in $\mathbb{Z}_q$ from one inner product.
  - Replace every n^2 length key by a key of length n.

- Question: How to define the $\star$ operation?

How efficient is LWE?

- Getting one pseudorandom $b_i \in \mathbb{Z}_q$ requires an n-dimensional inner product modulo q.
- Cryptosystems have larger keys of size larger than $n^2 \log q$.
- Wishful thinking:

$$(a_1, a_2, \ldots, a_n) \star (s_1, s_2, \ldots, s_n) + (e_1, e_2, \ldots, e_n) = (b_1, b_2, \ldots, b_n) .$$

  - Get n pseudorandom elements in $\mathbb{Z}_q$ from one inner product.
  - Replace every n^2 length key by a key of length n.
- Question: How to define the $\star$ operation?
  - With small error, co-ordinate wise multiplication is insecure.

LWE over Rings

- Let $R = \mathbb{Z}[X]/(X^n + 1)$, and $R_q = R/qR$.
  - Operations in R_q are very efficient using algorithms similar to FFT.
- Search: Find secret $s \in R_q$ given

$$a_1 \leftarrow R_q\ ,\quad b_1 = a_1 \cdot s + e_1$$

$$a_2 \leftarrow R_q\ ,\quad b_2 = a_2 \cdot s + e_2$$

$$\cdots$$

$$\cdots$$

- Decision: Distinguish (a_i, b_i) from (a_i, u_i) where u_i is uniform in R_q.

LWE over Rings

- Let $R = \mathbb{Z}[X]/(X^n + 1)$, and $R_q = R/qR$.
  - Operations in R_q are very efficient using algorithms similar to FFT.
- Search: Find secret $s \in R_q$ given

$$a_1 \leftarrow R_q\ ,\quad b_1 = a_1 \cdot s + e_1$$
$$a_2 \leftarrow R_q\ ,\quad b_2 = a_2 \cdot s + e_2$$
$$\cdots$$
$$\cdots$$

- Decision: Distinguish (a_i, b_i) from (a_i, u_i) where u_i is uniform in R_q.
- Worst case to Average Case [LPR10, PRS17]

Decision R-LWE $\Longrightarrow$ Search R-LWE $\Longrightarrow$ ApproxSVP$_{\mathrm{poly}(n)}$ on ideal lattices

Complexity of SVP on Ideal Lattices

- We know that if we can solve R-LWE, then we can also solve SVP on ideal lattices.
- The other direction is unknown.
- There has been some recent progress[CDPR16, BS16, K16, CDW16] giving an efficient quantum algorithm for $2^{O(\sqrt{n \log n})}$ approximation of SVP on Ideal Lattices.
- This does not say anything about the hardness of Ring-LWE.
- There is more algebraic structure in Ring-LWE that can possibly lead to quantum attacks, but so far there has been little success.

Complexity of SVP on Ideal Lattices

- We know that if we can solve R-LWE, then we can also solve SVP on ideal lattices.
- The other direction is unknown.
- There has been some recent progress[CDPR16, BS16, K16, CDW16] giving an efficient quantum algorithm for $2^{O(\sqrt{n \log n})}$ approximation of SVP on Ideal Lattices.
- This does not say anything about the hardness of Ring-LWE.
- There is more algebraic structure in Ring-LWE that can possibly lead to quantum attacks, but so far there has been little success.
- Ring-LWE based crypto is more efficient, but *perhaps* less secure.

Talk Outline

- GGH Public Key Encryption Scheme
- LWE Problem and Applications
- Efficiency from Rings
- Recent Implementations.

Recent Implementations.

Key Exchange

- [BCNS15] Ring-LWE based key exchange.
- NewHope [ADPS'15]: Optimized Ring-LWE key exchange with $\lambda = 200$ bit quantum security.
  - Comparable to or even faster than current ECDH with 128-bit classical security.
  - Google has experimentally deployed NewHope + ECDH.
- Frodo [BCDMNNRS'16]: Plain-LWE key exchange with some optimizations. Conjectures 128-bit quantum security.
  - About 10 times slower than NewHope, but almost as fast as ECDH (and much faster than RSA).
- NTRU EES743EP1 [WEJ13].

Other Implementations

BLISS: [DDLL'13] An efficient digital signature scheme based on [Lyu09,Lyu12]

DILITHIUM: [DLLSSS17] Another efficient digital signature based on [GLP12] that eliminates some of the vulnerabilities of BLISS.

HELib: [HaleviShoup] Implementation of Fully Homomorphic Encryption.

$\Lambda \circ \lambda$: [CroPei'16] A high level framework aimed at advanced lattice cryptosystems.

Lots of ongoing work including many proposals of new post-quantum crypto schemes submitted to NIST.

Questions?