Public-Key Cryptosystems from the Worst-Case Shortest Vector Problem

Chris Peikert
Georgia Tech

Computer Security & Cryptography Workshop
12 April 2010
Talk Outline

1. State of Lattice-Based Cryptography

2. Main Result: Public-Key Encryption based on GapSVP

3. Future Work
Shortest Vector Problem(s)

A lattice $\mathcal{L} \subset \mathbb{R}^n$ having basis $\mathbf{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$ is:

$$\mathcal{L} = \sum_{i=1}^{n} (\mathbb{Z} \cdot \mathbf{b}_i)$$

**Shortest Vector Problem ($\gamma$-GapSVP)**

- Given $\mathbf{B}$, decide: $\lambda \leq 1$ or $\lambda > \gamma$?

**Unique SVP ($\gamma$-uSVP)**

- Given $\mathbf{B}$ with '$\gamma$-unique' shortest vector, find it.
Shortest Vector Problem(s)

A lattice $\mathcal{L} \subset \mathbb{R}^n$ having basis $\mathbf{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$ is:

$$\mathcal{L} = \sum_{i=1}^{n} (\mathbb{Z} \cdot \mathbf{b}_i)$$

Shortest Vector Problem (γ-GapSVP)

- Given $\mathbf{B}$, decide: $\lambda \leq 1$ or $\lambda > \gamma$?
Shortest Vector Problem(s)

A lattice $\mathcal{L} \subset \mathbb{R}^n$ having basis $\mathbf{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$ is:

$$\mathcal{L} = \sum_{i=1}^{n}(\mathbb{Z} \cdot \mathbf{b}_i)$$

Shortest Vector Problem ($\gamma$-GapSVP)

- Given $\mathbf{B}$, decide: $\lambda \leq 1$ or $\lambda > \gamma$?
Shortest Vector Problem(s)

A lattice $\mathcal{L} \subset \mathbb{R}^n$ having basis $B = \{b_1, \ldots, b_n\}$ is:

$$\mathcal{L} = \sum_{i=1}^{n}(\mathbb{Z} \cdot b_i)$$

Shortest Vector Problem ($\gamma$-GapSVP)

▶ Given $B$, decide: $\lambda \leq 1$ or $\lambda > \gamma$?

Unique SVP ($\gamma$-uSVP)

▶ Given $B$ with ‘$\gamma$-unique’ shortest vector, find it.
Worst-Case Complexity

\[ \gamma = 2^{(\log n)^{1-\epsilon}} \]

\( \sqrt{n} \) \hspace{1cm} \( n \) \hspace{1cm} \( 2^{\sim n} \)

NP-hard* \hspace{1cm} \in \text{coNP} \hspace{1cm} \text{(some) crypto} \hspace{1cm} \in \text{P}

[Ajt98, \ldots, HR07] \hspace{1cm} [GG98, AR05] \hspace{1cm} [Ajt96, \ldots, MR04, Reg05] \hspace{1cm} [LLL82, Sch87]
Worst-Case Complexity

### GapSVP

<table>
<thead>
<tr>
<th>$\gamma = 2^{(\log n)^{1-\epsilon}}$</th>
<th>$\sqrt{n}$</th>
<th>$n$</th>
<th>$2^{\sim n}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>NP-hard*</td>
<td>$\in \text{coNP}$ [Ajt98,…,HR07]</td>
<td>(some) crypto</td>
<td>$\in \text{P}$ [LLL82,Sch87]</td>
</tr>
<tr>
<td>[GG98,AR05]</td>
<td>[Ajt96,…,MR04,Reg05]</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

▶ For $\gamma = \text{poly}(n)$, best algorithm is $2^n$ time & space [AKS01]
Worst-Case Complexity

### GapSVP

$$\gamma = 2^{(\log n)^{1-\epsilon}}$$

- $\sqrt{n}$
- $n$
- $2^{\sim n}$

- **NP-hard**\footnote{[Ajt98,\ldots,HR07]}
- **$\in$ coNP**\footnote{[GG98,AR05]}
- (some) crypto\footnote{[Ajt96,\ldots,MR04,Reg05],[LLL82,Sch87]}
- **$\in$ P**

For $\gamma = \text{poly}(n)$, best algorithm is $2^n$ time & space \footnote{[AKS01]}

### uSVP

$$\gamma = ??$$

- $\sqrt[4]{n}$
- $n^{1.5}$

- **NP-hard**\footnote{[Cai98]}
- **$\in$ coAM**
- crypto\footnote{[AD97/07,Reg03]}
Taxonomy of Lattice-Based Crypto

‘minicrypt’

OWF [Ajt96, …]

ID schemes [MV03, Lyu08]

Sigs [LM08, GPV08, CHK10]
Taxonomy of Lattice-Based Crypto

‘minicrypt’

OWF [Ajt96,…]

Sigs [LM08,GPV08,CHKP10]

ID schemes [MV03,Lyu08]

GapSVP etc. hard
Taxonomy of Lattice-Based Crypto

‘minicrypt’

- OWF [Ajt96,…]
- Sigs [LM08,GPV08,CHKP10]
- ID schemes [MV03,Lyu08]

‘CRYPTOMANIA’

- PKE [AD97,Reg03,Reg05]
- CCA [PW08]
- ID-based [GPV08]

GapSVP etc. hard
Taxonomy of Lattice-Based Crypto

‘minicrypt’

OWF [Ajt96,...]

Sigs [LM08,GPV08,CHKP10]

ID schemes [MV03,Lyu08]

GapSVP etc. hard

‘CRYPTOMANIA’

PKE [AD97,Reg03,Reg05]

CCA [PW08]

ID-based [GPV08]

(Obl. tran. [PVW08], leakage [AGV09],
homom [G09,GHV10], KDM [ACPS09],
HIBE [CHKP10], Deniable [OP10], ...)}
Taxonomy of Lattice-Based Crypto

‘minicrypt’

OWF [Ajt96,…]

Sigs [LM08,GPV08,CHKP10]

ID schemes [MV03,Lyu08]

GapSVP etc. hard

‘CRYPTOMANIA’

PKE [AD97,Reg03,Reg05]

CCA [PW08]

ID-based [GPV08]

(Obl. tran. [PVW08], leakage [AGV09],
homom [G09, GHV10], KDM [ACPS09],
HIBE [CHKP10], Deniable [OP10], …)

uSVP hard

GapSVP etc. quantum-hard
Learning With Errors

- Generalizes ‘learning parity with noise’: dim $n$, modulus $q \geq 2$
Learning With Errors

- Generalizes ‘learning parity with noise’: dim $n$, modulus $q \geq 2$

- **Search:** find $s \in \mathbb{Z}_q^n$ given ‘noisy random inner products’

  \[
  a_1, b_1 \approx \langle a_1, s \rangle \mod q
  \]

  \[
  a_2, b_2 \approx \langle a_2, s \rangle \mod q
  \]

  \[\vdots\]
Learning With Errors

- Generalizes ‘learning parity with noise’: dim $n$, modulus $q \geq 2$
- **Search**: find $s \in \mathbb{Z}_q^n$ given ‘noisy random inner products’
  
  \[
  a_1, \quad b_1 = \langle a_1, s \rangle + x_1 \mod q \\
  a_2, \quad b_2 = \langle a_2, s \rangle + x_2 \mod q \\
  \vdots
  \]

  Uniform $a_i \in \mathbb{Z}_q^n$, Gaussian errors $x_i$

  $\alpha \cdot q \geq \sqrt{n}$
Learning With Errors

- Generalizes ‘learning parity with noise’: dim $n$, modulus $q \geq 2$

- **Search:** find $s \in \mathbb{Z}_q^n$ given ‘noisy random inner products’
  - $a_1, b_1 = \langle a_1, s \rangle + x_1 \mod q$
  - $a_2, b_2 = \langle a_2, s \rangle + x_2 \mod q$
  - ...

  Uniform $a_i \in \mathbb{Z}_q^n$, Gaussian errors $x_i$

- **Decision:** distinguish from uniform $(a_i, b_i)$
Learning With Errors

- Generalizes ‘learning parity with noise’: dim $n$, modulus $q \geq 2$

- **Search:** find $s \in \mathbb{Z}_q^n$ given ‘noisy random inner products’

  $a_1 , b_1 = \langle a_1 , s \rangle + x_1 \mod q$

  $a_2 , b_2 = \langle a_2 , s \rangle + x_2 \mod q$

  \vdots

  Uniform $a_i \in \mathbb{Z}_q^n$, Gaussian errors $x_i$

- **Decision:** distinguish from uniform $(a_i , b_i)$

**State of the Art**

$(n/\alpha)$-GapSVP etc. $\leq$ search-LWE $\leq$ decision-LWE $\leq$ crypto

quantum [Reg05]  
prime $q = \text{poly}(n)$ [BFKL94,R05]  
[R05,PW08,GPV08, PVW08,AGV09,ACPS09,…]
Our Results

First public-key encryption based on classical GapSVP hardness
Our Results

First public-key encryption based on classical GapSVP hardness

1. **Classical reduction:** GapSVP \(\leq\) Learning With Errors
Our Results

First public-key encryption based on classical GapSVP hardness

1. Classical reduction: $\text{GapSVP} \leq \text{Learning With Errors}$
   - Standard $(n/\alpha)$-GapSVP: large LWE modulus $q \geq 2^n$
Our Results

First public-key encryption based on classical GapSVP hardness

1. Classical reduction: GapSVP \(\leq\) Learning With Errors
   - Standard \((n/\alpha)\)-GapSVP: large LWE modulus \(q \geq 2^n\)
   - New \(\zeta\)-to-\((n/\alpha)\)’-GapSVP: \(q \approx \zeta\) \[\approx poly(n)\]
Our Results

First public-key encryption based on classical GapSVP hardness

1. Classical reduction: GapSVP \( \leq \) Learning With Errors
   - Standard \((n/\alpha)\)-GapSVP: large LWE modulus \( q \geq 2^n \)
   - New ‘\(\zeta\)-to-(\(n/\alpha\))’-GapSVP: \( q \approx \zeta \) \( [= \text{poly}(n)] \)

2. LWE search \( \leq \) decision for large \( q \) \( [\gg \text{poly}(n)] \)
   \( \Rightarrow \) GapSVP-hardness of prior LWE-based crypto \[\text{[Reg05,\ldots]}\]
Our Results

First public-key encryption based on classical GapSVP hardness

1. Classical reduction: GapSVP $\leq$ Learning With Errors
   - Standard $(n/\alpha)$-GapSVP: large LWE modulus $q \geq 2^n$
   - New $\zeta$-to-$(n/\alpha)$'-GapSVP: $q \approx \zeta$ [ = poly$(n)$ ]

2. LWE search $\leq$ decision for large $q$ [ $\gg$ poly$(n)$ ]
   $\Rightarrow$ GapSVP-hardness of prior LWE-based crypto [Reg05, …]

3. New LWE-based chosen ciphertext-secure encryption
   - Simpler construction, milder assumption than prior CCA [PW08]
Hardness of LWE

BDD on $\mathcal{L}$:
$d \ll \lambda/2$

$LWE$
BDD on $\mathcal{L}$:
\[ d \ll \lambda/2 \]
[Regev05] Hardness of LWE

\[ \text{BDD on } \mathcal{L}: \quad d \ll \lambda/2 \]

\[ \text{LWE} \]
[Regev05] Hardness of LWE

BDD on $\mathcal{L}$:
$d \ll \lambda/2$

$\mathcal{L}^*$

$\text{BDD}$

$\text{LWE}$

$\text{BDD}$

$\text{LWE}$

GapSVP

SIVP
Why Quantum?

- ‘Obvious’ answer: iterative step

- BDD on $\mathcal{L}$
- quantum FT

Choose some $x \in \mathcal{L}$

Perturb to $y \approx x$

Invoke oracle on $y$

Returns $x$ — we already knew that!

Quantum can "uncompute" $x$
Why Quantum?

▸ ‘Obvious’ answer: iterative step

▸ A better answer: to make use of BDD/LWE oracle

1. Choose some \( x \in \mathcal{L} \)
2. Perturb to \( y \approx x \)
3. Invoke oracle on \( y \)
Why Quantum?

▶ ‘Obvious’ answer: iterative step

▶ A better answer: to make use of BDD/LWE oracle

1. Choose some $x \in \mathcal{L}$
2. Perturb to $y \approx x$
3. Invoke oracle on $y$
4. Returns $x$ — we already knew that!
Why Quantum?

▶ ‘Obvious’ answer: iterative step

▶ A better answer: to make use of BDD/LWE oracle

1. Choose some \( x \in \mathcal{L} \)
2. Perturb to \( y \approx x \)
3. Invoke oracle on \( y \)
4. Returns \( x \) — we already knew that!

✔ Quantum can “uncompute” \( x \)
Our Approach

**New way** of solving GapSVP in a reduction
Our Approach

New way of solving GapSVP in a reduction

“The Usual”

\[ y \downarrow \]

\[ \text{BDD (LWE)} \]

\[ \downarrow \]

\[ x \]
Our Approach

New way of solving GapSVP in a reduction

“The Usual”

IMAGINE

Illegal BDD instance
⇓
Incorrect (& unknown!) LWE distribution

SO WHAT!
When $\lambda \ll d$,
o oracle cannot guess
⇓
Distinguishes large $\lambda$ from small
▶ View as [Gold98] AM proof between reduction and oracle
Our Approach

New way of solving GapSVP in a reduction

“The Usual”

IMAGINE

Illegal BDD instance

Incorrect (& unknown!)

LWE distribution

When $\lambda \ll d$,

oracle cannot guess

$\downarrow$

Distinguishes large $\lambda$ from small

$\Rightarrow$

View as $\textit{GoldGold98}$ AM proof between reduction and oracle
Our Approach

New way of solving GapSVP in a reduction

“The Usual”

BDD (LWE)

y ↓

x

y ↓

BDD (LWE)

y ↓

??

Illegal BDD instance

Incorrect (& unknown!) LWE distribution

SO WHAT!

When \( \lambda \ll d \), oracle cannot guess \( x \)

\( \downarrow \)

Distinguishes large \( \lambda \) from small
Our Approach

New way of solving GapSVP in a reduction

“The Usual”

IMAGINE

Illegal BDD instance

\[ \Downarrow \]

Incorrect (& unknown!) LWE distribution

SO WHAT!

When \( \lambda \ll d \), oracle cannot guess \( x \)

\[ \Downarrow \]

Distinguishes large \( \lambda \) from small

View as [GoldGold98] AM proof between reduction and oracle
Technical Obstacles

1. What about in $\text{BDD} \rightarrow \text{LWE}$ reduction?

(No quantum allowed!)

⋆ Use $[\text{GPV08}]$ sampling algorithm with "best available" basis for $L^\ast$.

'ζ-good' basis $\Rightarrow LWE$ modulus $q \approx ζ$.

(LLL-reduced basis is $2^n$-good.)

⋆ 'One shot' (non-iterative) reduction

$LWE$ search / decision equivalence?

(Normally requires prime $q = \text{poly}(n)$...)

Option 1: crypto directly based on search-LWE

Option 2: search = decision for 'smooth' $q$ and Gaussian error
1. What about in $BDD \rightarrow LWE$ reduction? (No quantum allowed!)

- Use [GPV08] sampling algorithm with ‘best available’ basis for $\mathcal{L}^*$.
Technical Obstacles

1. What about in $BDD \rightarrow LWE$ reduction? (No quantum allowed!)

   - Use [GPV08] sampling algorithm with ‘best available’ basis for $L^*$. ‘$\zeta$-good’ basis $\Rightarrow$ LWE modulus $q \approx \zeta$.

   (LLL-reduced basis is $2''$-good.)
Technical Obstacles

1. What about in \( BDD \rightarrow LWE \) reduction?

- Use \([GPV08]\) sampling algorithm with ‘best available’ basis for \( \mathcal{L}^* \).

  ‘\( \zeta \)-good’ basis \( \Rightarrow \) LWE modulus \( q \approx \zeta \).

  (LLL-reduced basis is \( 2^n \)-good.)

- ‘One shot’ (non-iterative) reduction

(No quantum allowed!)
Technical Obstacles

1. What about in \( BDD \rightarrow LWE \) reduction? (No quantum allowed!)

   - Use [GPV08] sampling algorithm with ‘best available’ basis for \( \mathcal{L}^* \).
   - ‘\( \zeta \)-good’ basis \( \Rightarrow \) LWE modulus \( q \approx \zeta \).
     (LLL-reduced basis is 2\( ^n \)-good.)
   - ‘One shot’ (non-iterative) reduction

2. LWE search / decision equivalence?

   (Normally requires prime \( q = \text{poly}(n) \ldots \) )
Technical Obstacles

1. What about in $\text{BDD} \rightarrow \text{LWE}$ reduction? (No quantum allowed!)
   - Use [GPV08] sampling algorithm with ‘best available’ basis for $\mathcal{L}^*$.
     - ‘$\zeta$-good’ basis $\Rightarrow$ LWE modulus $q \approx \zeta$.
     (LLL-reduced basis is $2^n$-good.)
   - ‘One shot’ (non-iterative) reduction

2. LWE search / decision equivalence? (Normally requires prime $q = \text{poly}(n)$...)
   - Option 1: crypto directly based on search-LWE
   - Option 2: search = decision for ‘smooth’ $q$ and Gaussian error
Reducing Search to Decision

- Suppose $D$ distinguishes $(a \in \mathbb{Z}_q^n, \ b \approx \langle \mathbf{a}, \mathbf{s} \rangle) \leftarrow A_{s,\alpha}$ from uniform.
Reducing Search to Decision

- Suppose $D$ distinguishes $(a \in \mathbb{Z}_q^n, b \approx \langle a, s \rangle) \leftarrow A_{s,\alpha}$ from uniform.

- Let $q = q_1 \cdots q_t [\gg \text{poly}(n)]$ for distinct $(1/\alpha) \leq q_i \leq \text{poly}(n)$. 
Reducing Search to Decision

- Suppose $D$ distinguishes $(\mathbf{a} \in \mathbb{Z}_q^n, \ b \approx \langle \mathbf{a}, \mathbf{s} \rangle) \leftarrow A_{s,\alpha}$ from uniform.
- Let $q = q_1 \cdot \cdots \cdot q_t \ [\gg \text{poly}(n)]$ for distinct $(1/\alpha) \leq q_i \leq \text{poly}(n)$.

Find $s$: Chinese remaindering & “smoothing”

- To test if $s_1 = 0 \mod q_i$:

$$(\mathbf{a}, \ b) \mapsto (\mathbf{a} + r \cdot \mathbf{e}_1, \ b) \quad \text{for} \quad r \leftarrow (q/q_i) \cdot \mathbb{Z}_{q_i}$$
Reducing Search to Decision

- Suppose $D$ distinguishes $(a \in \mathbb{Z}_q^n, b \approx \langle a, s \rangle) \leftarrow A_{s,\alpha}$ from uniform.

- Let $q = q_1 \cdots q_t [\gg \text{poly}(n)]$ for distinct $(1/\alpha) \leq q_i \leq \text{poly}(n)$. 

Find $s$: Chinese remaindering & “smoothing”

- To test if $s_1 = 0 \mod q_i$:

$$ (a, b) \mapsto (a + r \cdot e_1, b) \quad \text{for} \quad r \leftarrow (q/q_i) \cdot \mathbb{Z}_{q_i} $$

- If yes, maps $A_{s,\alpha}$ to itself. If not, maps $A_{s,\alpha}$ to uniform?
Reducing Search to Decision

- Suppose $D$ distinguishes $(a \in \mathbb{Z}_q^n, b \approx \langle a, s \rangle) \leftarrow A_{s, \alpha}$ from uniform.
- Let $q = q_1 \cdots q_t \gg \text{poly}(n)$ for distinct $(1/\alpha) \leq q_i \leq \text{poly}(n)$.

Find $s$: Chinese remaindering & “smoothing”

- To test if $s_1 = 0 \mod q_i$:
  
  $$(a, b) \mapsto (a + r \cdot e_1, b) \quad \text{for} \quad r \leftarrow (q/q_i) \cdot \mathbb{Z}_{q_i}$$

- If yes, maps $A_{s, \alpha}$ to itself. If not, maps $A_{s, \alpha}$ to uniform!

Gaussians of width $\alpha q \geq (q/q_i)$ separated by $(q/q_i)$

$\Rightarrow \text{uniform}^*$ by smoothing bounds [MicReg04]
Reducing Search to Decision

- Suppose $D$ distinguishes $(a \in \mathbb{Z}_q^n, b \approx \langle a, s \rangle) \leftarrow A_{s,\alpha}$ from uniform.

- Let $q = q_1 \cdots q_t$ [$\gg \text{poly}(n)$] for distinct $(1/\alpha) \leq q_i \leq \text{poly}(n)$.

Find $s$: Chinese remaindering & “smoothing”

- To test if $s_1 = 0 \mod q_i$:
  $$(a, b) \mapsto (a + r \cdot e_1, b) \text{ for } r \leftarrow (q/q_i) \cdot \mathbb{Z}_{q_i}$$

- If yes, maps $A_{s,\alpha}$ to itself. If not, maps $A_{s,\alpha}$ to uniform!

  Gaussians of width $\alpha q \geq (q/q_i)$ separated by $(q/q_i)$

  $\Rightarrow$ uniform* by smoothing bounds [MicReg04]

- (NB: for general error dists, hybrid argument over $q_i$’s fails.)
Chosen-Ciphertext Security

Intuitive Definition [RS91,DDN91,NY95]
- Encryption conceals message, even given decryption oracle

Elementary Construction
- Follows "witness-recovering decryption" approach [PW08].
- Define $g_A(s,x) = A^t s + x$.
  - Can generate $A$ with "trapdoor" for $g^{-1}_A$ [GGH97,Ajt99,AP09].
- Distinguish $g_A^1(s,x_1),...,g_A^k(s,x_k)$ [same $s$!]$\iff$ solve LWE
  - So $g_A^1,...,g_A^k$ pseudorandom under 'correlated inputs' [RS09].
- Correlation-secure injective TDF $\Rightarrow$ CCA-secure encryption
  - But care needed to make $g_A$ "chosen-output secure."
Chosen-Ciphertext Security

Intuitive Definition [RS91,DDN91,NY95]
▶ Encryption conceals message, even given decryption oracle

Elementary Construction
▶ Follows “witness-recovering decryption” approach [PW08].
Chosen-Ciphertext Security

**Intuitive Definition** [RS91,DDN91,NY95]
- Encryption conceals message, even given decryption oracle

**Elementary Construction**
- Follows “witness-recovering decryption” approach [PW08].
- Define $g_A(s, x) = A^t s + x$.
  Can generate $A$ with “trapdoor” for $g_A^{-1}$ [GGH97,Ajt99,AP09]
Chosen-Ciphertext Security

Intuitive Definition [RS91,DDN91,NY95]
▶ Encryption conceals message, even given decryption oracle

Elementary Construction
▶ Follows “witness-recovering decryption” approach [PW08].
▶ Define $g_A(s, x) = A^t s + x$.
Can generate $A$ with “trapdoor” for $g_A^{-1}$ [GGH97,Ajt99,AP09]
▶ Distinguish $g_{A_1}(s, x_1), \ldots, g_{A_k}(s, x_k)$ [same $s$!] $\iff$ solve LWE
So $g_{A_1}, \ldots, g_{A_k}$ pseudorandom under ‘correlated inputs’ [RS09]
Chosen-Ciphertext Security

Intuitive Definition [RS91,DDN91,NY95]
▶ Encryption conceals message, even given decryption oracle

Elementary Construction
▶ Follows “witness-recovering decryption” approach [PW08].

▶ Define \( g_A(s, x) = A^t s + x \).
  Can generate \( A \) with “trapdoor” for \( g_A^{-1} \) [GGH97,Ajt99,AP09]

▶ Distinguish \( g_{A_1}(s, x_1), \ldots, g_{A_k}(s, x_k) \) [same \( s \!\!\!\!\!\!\!\!] \iff \) solve LWE
  So \( g_{A_1}, \ldots, g_{A_k} \) pseudorandom under ‘correlated inputs’ [RS09]

▶ Correlation-secure injective TDF \( \Rightarrow \) CCA-secure encryption
  But care needed to make \( g_A \) “chosen-output secure.”
Building off of our approach, [LyuMic09] showed

\[(\gamma \sqrt{n})\text{-GapSVP} \leq \gamma\text{-uSVP} \leq \text{crypto} \]  

[AjtDwo97,Reg03]

Open: classical, iterative reduction from lattices to LWE

I'd expect it to work for GapSVP, SIVP, etc. with \( q = \text{poly}(n) \)

Open: complexity of new ‘\( \zeta \)-to-\( \gamma \)’-GapSVP problem?

NP-hard for nontrivial \( \zeta \)? Better algorithms?
Building off of our approach, [LyuMic09] showed

\[(\gamma \sqrt{n})\text{-GapSVP} \leq \gamma\text{-uSVP} \leq \text{crypto} \ [\text{AjtDwo97,Reg03}]\]

“Unifies” two styles of cryptosystems [AD97,Reg03] and [Reg05,...] under (almost) same assumption.
Building off of our approach, [LyuMic09] showed

\[(\gamma\sqrt{n})\text{-GapSVP} \leq \gamma\text{-uSVP} \leq \text{crypto} \ [\text{AjtDwo97,Reg03}]\]

“Unifies” two styles of cryptosystems [AD97,Reg03] and [Reg05,…] under (almost) same assumption.

Open: classical, iterative reduction from lattices to LWE

I’d expect it to work for GapSVP, SIVP, etc. with \(q = \text{poly}(n)\)
Building off of our approach, [LyuMic09] showed

\[(\gamma \sqrt{n})\text{-GapSVP} \leq \gamma\text{-uSVP} \leq \text{crypto} \ [\text{AjtDwo97,Reg03}]\]

“Unifies” two styles of cryptosystems [AD97,Reg03] and [Reg05,…] under (almost) same assumption.

1. **Open**: classical, iterative reduction from lattices to LWE

   I’d expect it to work for GapSVP, SIVP, etc. with \( q = \text{poly}(n) \)

2. **Open**: complexity of new ‘\( \zeta \)-to-\( \gamma \)’-GapSVP problem?

   NP-hard for nontrivial \( \zeta \)? Better algorithms?