How to Use a Short Basis: Trapdoors for Hard Lattices and New Cryptographic Constructions

Chris Peikert
SRI

Work with Craig Gentry and Vinod Vaikuntanathan
Digital Signatures
Digital Signatures

(secret)

(public)

(secret)
Digital Signatures

(secret)

(public)

“I love you” ✅
Digital Signatures

It’s over

(secret)

(public)

“It’s over” X
Trapdoor Permutations [DiffieHellman76]

- Public function $f$, secret “trapdoor” $f^{-1}$
Trapdoor Permutations  [DiffieHellman76]

- Public function $f$, secret “trapdoor” $f^{-1}$

![Diagram](image)

$x$ \[\xrightarrow{f} \] $y$

Dom \[\xrightarrow{f} \] Dom

Candidates: [RSA78, Rabin79, Paillier99]

"General assumption"

Applications: digital signatures, OT, NIZK, . . .

All rely on hardness of factoring

Complex: 2048-bit exponentiation

Lack of diversity

Broken by quantum algorithms [Shor]
Trapdoor Permutations  [DiffieHellman76]

- Public function $f$, secret “trapdoor” $f^{-1}$
**Trapdoor Permutations**  
[DiffieHellman76]

- Public function $f$, secret “trapdoor” $f^{-1}$

```
Dom         x
            |   f^{-1}    |
            v
Dom         y
```
Trapdoor Permutations  [DiffieHellman76]

- Public function $f$, secret “trapdoor” $f^{-1}$

- Candidates: [RSA78, Rabin79, Paillier99]
  - "General assumption"
  - Applications: digital signatures, OT, NIZK, …
**Trapdoor Permutations**  [DiffieHellman76]

- Public function \( f \), secret “trapdoor” \( f^{-1} \)

- **Candidates:** [RSA78,Rabin79,Paillier99]
  - ✔️ “General assumption”
  - ✔️ Applications: digital signatures, OT, NIZK, …

- **All rely on hardness of factoring**
  - ✗ Complex: 2048-bit exponentiation
  - ✗ Lack of diversity
  - ✗ Broken by quantum algorithms [Shor]
Lattice-Based Cryptography

What’s To Like

- **Simple & efficient**: linear ops, small integers
- Resist **subexp & quantum** attacks (so far)
- Security from **worst-case** hardness [Ajtai, ...]
Lattice-Based Cryptography

What’s To Like

- Simple & efficient: linear ops, small integers
- Resist subexp & quantum attacks (so far)
- Security from worst-case hardness [Ajtai,...]

What’s Known

1. One-way & collision-resistant functions [Ajtai,...,MicciancioRegev]
2. Public-key encryption [AjtaiDwork,Regev]
3. Recent developments [LyubMicc,PeikWat,...]
# Lattice-Based Cryptography

## What’s To Like

- Simple & efficient: linear ops, small integers
- Resist subexp & quantum attacks (so far)
- Security from worst-case hardness [Ajtai, ...]

## What’s Known

1. One-way & collision-resistant functions [Ajtai, ..., MicciancioRegev]
2. Public-key encryption [AjtaiDwork, Regev]
3. Recent developments [LyubMicc, PeikWat, ...]

## What’s Missing

- Everything else!
  Practical signatures, protocols, “advanced” crypto, ...
Results: New Lattice-Based Crypto

1 Preimage sampleable trapdoor functions
Results: New Lattice-Based Crypto

1. Preimage sampleable trapdoor functions

\[ D \xrightarrow{f} R \]

- "As good as" trapdoor permutations in many applications
- "Hash and sign" signatures: FDH etc.
- Identity-based encryption, OT [PVW], NCE [CDMW], NISZK [PV], . . .
Results: New Lattice-Based Crypto

1. Preimage sampleable trapdoor functions

\[ \begin{align*}
D & \xrightarrow{f} R \\
x & \mapsto y
\end{align*} \]

"As good as" trapdoor permutations in many applications

- Hash and sign: FDH etc.
- Identity-based encryption, OT [PVW], NCE [CDMW], NISZK [PV], ...
Results: New Lattice-Based Crypto

1 Preimage sampleable trapdoor functions

\[ f^{-1} \]

\[ D \quad x \quad \leftarrow \quad y \quad R \]
Results: New Lattice-Based Crypto

1. Preimage sampleable trapdoor functions

- Generate \((x, y)\) in two equivalent ways:

\[
\begin{array}{c}
D \xrightarrow{f} x \xrightarrow{f^{-1}} y \\
x \xleftarrow{f^{-1}} D \xrightarrow{f} y \xleftarrow{R}
\end{array}
\]
Results: New Lattice-Based Crypto

1 Preimage sampleable trapdoor functions

- Generate \((x, y)\) in two equivalent ways:

\[
\begin{align*}
D & \longrightarrow x \quad f \quad y \\
\quad x & \quad f^{-1} \quad y \quad R
\end{align*}
\]

- “As good as” trapdoor permutations in many applications
Results: New Lattice-Based Crypto

1. Preimage sampleable trapdoor functions

- Generate \((x, y)\) in two equivalent ways:

\[
\begin{align*}
D &\xrightarrow{f} x \quad \xleftarrow{f^{-1}} y \\
\quad \quad \quad &\xleftarrow{f^{-1}} x \quad \xleftarrow{R} y
\end{align*}
\]

- “As good as” trapdoor permutations in many applications

2. “Hash and sign” signatures: FDH etc.
Results: New Lattice-Based Crypto

1. Preimage sampleable trapdoor functions

   ![Diagram](image)

   - Generate \((x, y)\) in two equivalent ways:
     
     \[
     \begin{align*}
     D &\rightarrow x & f &\rightarrow y \\
     f^{-1} &\leftarrow \rightarrow & x &\leftarrow \rightarrow y &\leftarrow \rightarrow R
     \end{align*}
     \]

   - “As good as” trapdoor permutations in many applications

2. “Hash and sign” signatures: FDH etc.

3. Identity-based encryption, OT \([PVW]\), NCE \([CDMW]\), NISZK \([PV]\), . . .
Results: New Lattice-Based Crypto

1. Preimage sampleable trapdoor functions

   \[ D \xrightarrow{f} x \xrightarrow{f^{-1}} y \]

   - Generate \((x, y)\) in two equivalent ways:

   \[ D \xrightarrow{f} x \xrightarrow{f^{-1}} y \]

   - “As good as” trapdoor permutations in many applications

2. “Hash and sign” signatures: FDH etc.

3. Identity-based encryption, OT [PVW], NCE [CDMW], NISZK [PV], . . .

New Algorithmic Tool

- “Oblivious decoder” on lattices
A lattice $\mathcal{L} \subset \mathbb{R}^n$ having basis $B = \{b_1, \ldots, b_n\}$ is:

$$\mathcal{L} = \sum_{i=1}^{n} (\mathbb{Z} \cdot b_i)$$
A lattice $\mathcal{L} \subset \mathbb{R}^n$ having basis $\mathbf{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$ is:

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

- Given $\mathbf{B}$, find (nonzero) $\mathbf{v} \in \mathcal{L}$ within $\gamma$ factor of shortest.
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 (SVP$_{\gamma}$)**

- Given $\mathbf{B}$, find (nonzero) $\mathbf{v} \in \mathcal{L}$ within $\gamma$ factor of shortest.

**Absolute Distance Decoding (ADD$_{\beta}$)**

- Given $\mathbf{B}$ and target $\mathbf{t} \in \mathbb{R}^n$, find some $\mathbf{v} \in \mathcal{L}$ within distance $\beta$. 
Complexity of Lattice Problems

**SVP\(\gamma\) in the Worst Case**

\[ \gamma = O(1) \quad \text{poly}(n) \quad 2^n \]

- **NP-hard**
  - [Ajt, Mic, Kho]

- **2\(^n\) time**
  - [AKS]

- **poly\((n)\) time**
  - [LLL, Sch]

**Bottom Line**
- On random lattices, SVP\(\gamma\) and ADD\(\beta\) seem exponentially hard.

**ADD\(\beta\)**
- random lattice as hard as SVP\(\beta\)·\(n\)
every lattice
- Decoding hard on average, too
### Complexity of Lattice Problems

#### SVP_γ in the Worst Case

<table>
<thead>
<tr>
<th>γ =</th>
<th>( O(1) )</th>
<th>( \text{poly}(n) )</th>
<th>( 2^n )</th>
</tr>
</thead>
<tbody>
<tr>
<td>NP-hard</td>
<td>[Ajt,Mic,Kho]</td>
<td>( 2^n ) time</td>
<td>[AKS]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>( \text{poly}(n) ) time</td>
<td>[LLL,Sch]</td>
</tr>
</tbody>
</table>

#### Average-Case

- [Ajtai96, . . . , MicciancioRegev04]:
  - SVP_γ random lattice as hard as SVP_γ.n every lattice
# Complexity of Lattice Problems

## SVP\(\gamma\) in the Worst Case

<table>
<thead>
<tr>
<th>(\gamma = O(1))</th>
<th>(\text{poly}(n))</th>
<th>(2^n)</th>
</tr>
</thead>
<tbody>
<tr>
<td>NP-hard</td>
<td>2(^n) time</td>
<td>poly((n)) time</td>
</tr>
<tr>
<td>[Ajt,Mic,Kho]</td>
<td>[AKS]</td>
<td>[LLL,Sch]</td>
</tr>
</tbody>
</table>

## Average-Case

- [Ajtai96,...,MicciancioRegev04]:
  - \(\text{ADD}_\beta\) random lattice as hard as \(\text{SVP}_{\beta \cdot n}\) every lattice
- Decoding hard on average, too
Complexity of Lattice Problems

**SVP$_\gamma$ in the Worst Case**

$\gamma = O(1)$ \quad poly(n)$ \quad 2^n$

- NP-hard \quad 2^n \quad poly(n) \space time
- [Ajt,Mic,Kho] \quad [AKS] \quad [LLL,Sch]

**Average-Case**

- [Ajtai96,...,MicciancioRegev04]:
  - ADD$_\beta$ random lattice \quad as hard as \quad SVP$_{\beta\cdot n}$ every lattice
- Decoding hard on average, too

**Bottom Line**

- On random lattices, SVP$_\gamma$ and ADD$_\beta$ seem exponentially hard
GGH Signatures  [GoldreichGoldwasserHalevi96]

- “Hard” (public) verification basis $B$, short (secret) signing basis $S$
GGH Signatures  [GoldreichGoldwasserHalevi96]

- “Hard” (public) verification basis $B$, short (secret) signing basis $S$
- Sign with “nearest-plane” algorithm [Babai]
GGH Signatures  [GoldreichGoldwasserHalevi96]

- "Hard" (public) verification basis $B$, short (secret) signing basis $S$
- Sign with “nearest-plane” algorithm [Babai]
GGH Signatures [GoldreichGoldwasserHalevi96]

- “Hard” (public) verification basis $\mathbf{B}$, short (secret) signing basis $\mathbf{S}$
- Sign with “nearest-plane” algorithm [Babai]
GGH Signatures [GoldreichGoldwasserHalevi96]

- “Hard” (public) verification basis $B$, short (secret) signing basis $S$
- Sign with “nearest-plane” algorithm [Babai]
GGH Signatures [GoldreichGoldwasserHalevi96]

- “Hard” (public) verification basis \( B \), short (secret) signing basis \( S \)
- Sign with “nearest-plane” algorithm [Babai]
GGH Signatures \cite{GoldreichGoldwasserHalevi96}

- "Hard" (public) verification basis $B$, short (secret) signing basis $S$
- Sign with "nearest-plane" algorithm \cite{Babai}

Issues

1. Generating short & hard bases together
   - Ad-hoc, no worst-case hardness

2. Secret key leakage
   - Total break after several signatures \cite{NguyenRegev}
GGH Signatures [GoldreichGoldwasserHalevi96]

- “Hard” (public) verification basis $B$, short (secret) signing basis $S$
- Sign with “nearest-plane” algorithm [Babai]
GGH Signatures [GoldreichGoldwasserHalevi96]

- “Hard” (public) verification basis $B$, short (secret) signing basis $S$
- Sign with “nearest-plane” algorithm [Babai]

Issues

1. Generating short & hard bases together
   - Ad-hoc, no worst-case hardness

2. Secret key leakage
   - Total break after several signatures [NguyenRegev]
GGH Signatures [GoldreichGoldwasserHalevi96]

- “Hard” (public) verification basis $B$, short (secret) signing basis $S$
- Sign with “nearest-plane” algorithm [Babai]

Issues

1. Generating short & hard bases together
   - Ad-hoc, no worst-case hardness
GGH Signatures [GoldreichGoldwasserHalevi96]

- “Hard” (public) verification basis $B$, short (secret) signing basis $S$
- Sign with “nearest-plane” algorithm [Babai]

Issues

1. Generating short & hard bases together
   - Ad-hoc, no worst-case hardness

2. Secret key leakage
   - Total break after several signatures [NguyenRegev]
Gaussians and Lattices

“Uniform” in $\mathbb{R}^n$ when std dev $\geq$ shortest basis

[Regev, MicciancioRegev]
Gaussians and Lattices

"Uniform" in $\mathbb{R}^n$ when std dev $\geq$ shortest basis

[Regev, MicciancioRegev]
Gaussians and Lattices

"Uniform" in $\mathbb{R}^n$ when std dev $\geq$ shortest basis

[Regev, MicciancioRegev]
“Uniform” in $\mathbb{R}^n$ when $\text{std dev} \geq \text{shortest basis}$

[Regev, MicciancioRegev]
Our Trapdoor Function

- “Hard” public basis \( B \), short secret basis \( S \)  
  [Ajtai99, AP08]
Our Trapdoor Function

- “Hard” public basis $B$
  short secret basis $S$
  [Ajtai99,AP08]

- Input $v \in L$, error $e$
Our Trapdoor Function

- "Hard" public basis $B$, short secret basis $S$ 
  [Ajtai99,AP08]

- Input $v \in \mathcal{L}$, error $e$
Our Trapdoor Function

- “Hard” public basis $B$, short secret basis $S$ [Ajtai99, AP08]
- Input $v \in \mathcal{L}$, error $e$
- Uniform output $t$
Our Trapdoor Function

- “Hard” public basis $B$, short secret basis $S$ [Ajtai99,AP08]
- Input $v \in \mathcal{L}$, error $e$
- Uniform output $t$
Our Trapdoor Function

- “Hard” public basis $B$, short secret basis $S$ ([Ajtai99,AP08])
- Input $v \in \mathcal{L}$, error $e$
- Uniform output $t$
- Conditional distribution is “discrete Gaussian” $D_{\mathcal{L},t}$

Analysis tool in [Ban,AR,Reg,MR,Pei,...]
Inverting: Gaussian Sampler / Decoder

- Given basis $S$, samples $D_{\mathcal{L}, t}$ for any std dev $\geq \max ||s_i||$
  -Leaks nothing about $S$!
Inverting: Gaussian Sampler / Decoder

- Given basis $S$, samples $D_{\mathcal{L},t}$ for any std dev $\geq \max \|s_i\|$
  -Leaks nothing about $S$!

- Randomized nearest-plane [Babai,Klein]
Inverting: Gaussian Sampler / Decoder

- Given basis $S$, samples $D_{\mathcal{L},t}$ for any std dev $\geq \max \|s_i\|$
  -Leaks nothing about $S$!

- Randomized nearest-plane [Babai,Klein]
Inverting: Gaussian Sampler / Decoder

- Given basis $S$, samples $D_{L,t}$ for any std dev $\geq \max ||s_i||$
  - Leaks nothing about $S$!

- Randomized nearest-plane [Babai,Klein]
Inverting: Gaussian Sampler / Decoder

- Given basis $S$, samples $D_{L,t}$ for any std dev $\geq \max \|s_i\|
  -Leaks nothing about $S$!

- Randomized nearest-plane [Babai,Klein]
Inverting: Gaussian Sampler / Decoder

- Given basis $S$, samples $D_{\mathcal{L},t}$ for any std dev $\geq \max ||s_i||$
  - Leaks nothing about $S$!

- Randomized nearest-plane [Babai,Klein]

**[Klein]:** std dev $\leq \min ||\tilde{s}_i|| \Rightarrow$ solves CVP variant
Inverting: Gaussian Sampler / Decoder

- Given basis $S$, samples $D_{L,t}$ for any std dev $\geq \max ||s_i||$
  -Leaks nothing about $S$!

- Randomized nearest-plane [Babai,Klein]

  ![Diagram of nearest-plane algorithm]

  

  [Klein]: std dev $\leq \min ||\tilde{s}_i||$ $\Rightarrow$ solves CVP variant
  [This work]: std dev $\geq \max ||\tilde{s}_i||$ $\Rightarrow$ samples $D_{L,t}$ exactly*
Identity-Based Encryption

- Proposed by [Shamir84]:

- Master keys $mpk$, $msk$
- With $mpk$: encrypt to ID "Alice" or "Bob" or . . .
- With $msk$: extract $sk_{Alice}$ or $sk_{Bob}$ or . . .

- [BonehFranklin01]: bilinear pairings
- [Cocks01]: quadratic residuosity ($mod N = pq$)

- Lattice-based
- QR-based ($[Cocks,BGH]$)
Identity-Based Encryption

- Proposed by [Shamir84]:
  - Master keys $mpk$, $msk$
Identity-Based Encryption

- Proposed by [Shamir84]:
  - Master keys $mpk, msk$
  - With $mpk$: encrypt to ID “Alice” or “Bob” or …

- [BonehFranklin01]: bilinear pairings
- [Cocks01]: quadratic residuosity (mod $N = pq$)

Lattice-based
- QR-based [Cocks, BGH]

- $mpk$: random lattice
- $msk$: trapdoor basis
- $Hash(ID)$: uniform $y \in R^n$
- $y \in QR_N$
- $sk_{ID}$: random $\in f^{-1}(y)$
- $\sqrt{y_{12}}$
Identity-Based Encryption

- Proposed by [Shamir84]:
  - Master keys $mpk$, $msk$
  - With $mpk$: encrypt to ID “Alice” or “Bob” or...
  - With $msk$: extract $sk_{Alice}$ or $sk_{Bob}$ or...

- [BonehFranklin01]: bilinear pairings
- [Cocks01]: quadratic residuosity (mod $N = pq$)

- Lattice-based QR-based [Cocks,BGH]
  - $mpk$: random lattice
  - $msk$: trapdoor basis
  - $y \in R_n$ uniform
  - $y \in QR_N$ uniform
  - $sk_{ID}$ random $\in f^{-1}(y)$ random $\sqrt{y}$
Identity-Based Encryption

- Proposed by [Shamir84]:
  - Master keys $mpk$, $msk$
  - With $mpk$: encrypt to ID “Alice” or “Bob” or ...
  - With $msk$: extract $sk_{Alice}$ or $sk_{Bob}$ or ...

- [BonehFranklin01]: bilinear pairings
Identity-Based Encryption

- Proposed by [Shamir84]:
  - Master keys $mpk$, $msk$
  - With $mpk$: encrypt to ID “Alice” or “Bob” or …
  - With $msk$: extract $sk_{Alice}$ or $sk_{Bob}$ or …

- [BonehFranklin01]: bilinear pairings
- [Cocks01]: quadratic residuosity (mod $N = pq$)
Identity-Based Encryption

- Proposed by [Shamir84]:
  - Master keys \( mpk, msk \)
  - With \( mpk \): encrypt to ID “Alice” or “Bob” or …
  - With \( msk \): extract \( sk_{Alice} \) or \( sk_{Bob} \) or …

- [BonehFranklin01]: bilinear pairings
- [Cocks01]: quadratic residuosity (mod \( N = pq \))

<table>
<thead>
<tr>
<th></th>
<th>Lattice-based</th>
<th>QR-based [Cocks,BGH]</th>
</tr>
</thead>
<tbody>
<tr>
<td>( mpk )</td>
<td>random lattice</td>
<td>random ( N = p \cdot q )</td>
</tr>
</tbody>
</table>
Identity-Based Encryption

- Proposed by [Shamir84]:
  - Master keys $mpk, msk$
  - With $mpk$: encrypt to ID “Alice” or “Bob” or …
  - With $msk$: extract $sk_{Alice}$ or $sk_{Bob}$ or …

- [BonehFranklin01]: bilinear pairings
- [Cocks01]: quadratic residuosity (mod $N = pq$)

<table>
<thead>
<tr>
<th></th>
<th>Lattice-based</th>
<th>QR-based [Cocks,BGH]</th>
</tr>
</thead>
<tbody>
<tr>
<td>$mpk$</td>
<td>random lattice</td>
<td>random $N = p \cdot q$</td>
</tr>
<tr>
<td>$msk$</td>
<td>trapdoor basis</td>
<td>trapdoor $p, q$</td>
</tr>
</tbody>
</table>
## Identity-Based Encryption

- Proposed by [Shamir84]:
  - Master keys \( mpk, msk \)
  - With \( mpk \): encrypt to ID “Alice” or “Bob” or . . .
  - With \( msk \): extract \( sk_{Alice} \) or \( sk_{Bob} \) or . . .

- [BonehFranklin01]: bilinear pairings
- [Cocks01]: quadratic residuosity (mod \( N = pq \))

<table>
<thead>
<tr>
<th></th>
<th>Lattice-based</th>
<th>QR-based [Cocks,BGH]</th>
</tr>
</thead>
<tbody>
<tr>
<td>( mpk )</td>
<td>random lattice</td>
<td>random ( N = p \cdot q )</td>
</tr>
<tr>
<td>( msk )</td>
<td>trapdoor basis</td>
<td>trapdoor ( p, q )</td>
</tr>
<tr>
<td>Hash(( ID ))</td>
<td>uniform ( y \in \mathbb{R}^n )</td>
<td>uniform ( y \in QR_N )</td>
</tr>
</tbody>
</table>
Identity-Based Encryption

- Proposed by [Shamir84]:
  - Master keys $mpk$, $msk$
  - With $mpk$: encrypt to ID “Alice” or “Bob” or . . .
  - With $msk$: extract $sk_{Alice}$ or $sk_{Bob}$ or . . .

- [BonehFranklin01]: bilinear pairings

- [Cocks01]: quadratic residuosity (mod $N = pq$)

<table>
<thead>
<tr>
<th></th>
<th>Lattice-based</th>
<th>QR-based [Cocks,BGH]</th>
</tr>
</thead>
<tbody>
<tr>
<td>$mpk$</td>
<td>random lattice</td>
<td>random $N = p \cdot q$</td>
</tr>
<tr>
<td>$msk$</td>
<td>trapdoor basis</td>
<td>trapdoor $p, q$</td>
</tr>
<tr>
<td>$\text{Hash}(ID)$</td>
<td>uniform $y \in \mathbb{R}^n$</td>
<td>uniform $y \in QR_N$</td>
</tr>
<tr>
<td>$sk_{ID}$</td>
<td>random $\in f^{-1}(y)$</td>
<td>random $\sqrt{y}$</td>
</tr>
</tbody>
</table>
Cryptosystem with Master Trapdoor

Primal $\mathcal{L}$

Dual $\mathcal{L}^*$

For $v \in \mathcal{L}^*$: $\langle v, pk \rangle = \langle v, sk \rangle \mod 1$

For $w \approx v$: $\langle v, pk \rangle \approx \langle w, sk \rangle \mod 1$

"quasi"-agreement

Security: decoding $w$, a.k.a. "learning with errors"

Quantum worst-case connection [Regev]

Now: classical worst-case hardness [P]
For $v \in \mathcal{L}^*$: $\langle v, pk \rangle = \langle v, sk \rangle \mod 1$
Cryptosystem with Master Trapdoor

\[
\begin{align*}
\text{Primal } \mathcal{L} \\
\text{Dual } \mathcal{L}^*
\end{align*}
\]

\[\begin{align*}
\text{For } v \in \mathcal{L}^*: & \quad \langle v, pk \rangle = \langle v, sk \rangle \mod 1 \\
\text{For } w \approx v: & \quad \langle v, pk \rangle \approx \langle w, sk \rangle \mod 1 \quad \text{“quasi”-agreement}
\end{align*}\]
**Cryptosystem with Master Trapdoor**

- **Primal** $\mathcal{L}$
- **Dual** $\mathcal{L}^*$

- For $v \in \mathcal{L}^*$: $\langle v, pk \rangle = \langle v, sk \rangle \mod 1$
- For $w \approx v$: $\langle v, pk \rangle \approx \langle w, sk \rangle \mod 1$ ("quasi"-agreement)
- Security: decoding $w$, a.k.a. "learning with errors"
  - Quantum worst-case connection [Regev]
  - Now: classical worst-case hardness [P]
Open Problems

1. Tighter sampling for random lattices?
Open Problems

1. Tighter sampling for random lattices?
2. Practical “plain model” signatures?
Open Problems

1. Tighter sampling for random lattices?
2. Practical “plain model” signatures?
3. Relate factoring to lattice problems?
Open Problems

1. Tighter sampling for random lattices?
2. Practical “plain model” signatures?
3. Relate factoring to lattice problems?
4. “Essence” of quantum-immune crypto?
Open Problems

1. Tighter sampling for random lattices?
2. Practical “plain model” signatures?
3. Relate factoring to lattice problems?
4. “Essence” of quantum-immune crypto?

Thanks!

(Artwork courtesy of xkcd.org)