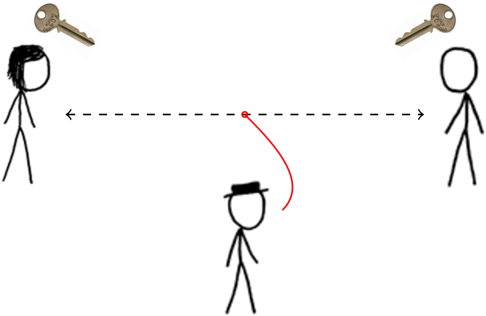
Post-Quantum Cryptography

Chris Peikert
University of Michigan

Tutorial, QIP 2022
6 March

Public-Key Cryptography

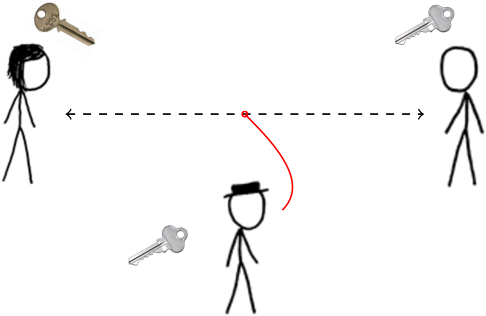
- Cryptography since the ancients: Alice, Bob need the same secret key



(Images courtesy xkcd.org)

Public-Key Cryptography

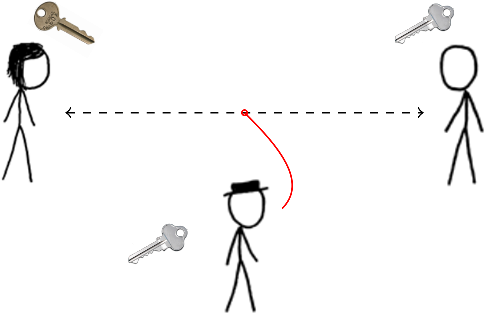
- A paradigm shift [Merkle'74,DH'76,RSA'77]: 'public-key' cryptography



(Images courtesy xkcd.org)

Public-Key Cryptography

- A paradigm shift [Merkle'74,DH'76,RSA'77]: 'public-key' cryptography

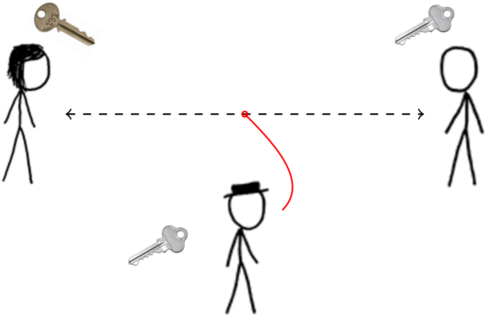


- Alice creates (related) public key  and secret key :

(Images courtesy xkcd.org)

Public-Key Cryptography

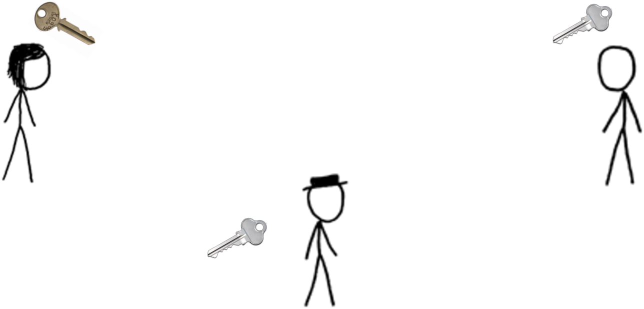
- A paradigm shift [Merkle'74,DH'76,RSA'77]: 'public-key' cryptography



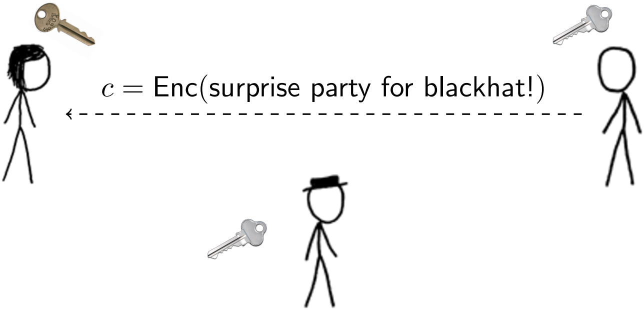
- Alice creates (related) public key 🔑 and secret key 🗝:
  - ★ Anyone can do 'public' ops using 🔑: encrypt, check authenticity
  - ★ Only Alice can do 'privileged' ops using 🗝: decrypt, attest

(Images courtesy xkcd.org)

Bread and Butter of PKC: Encryption



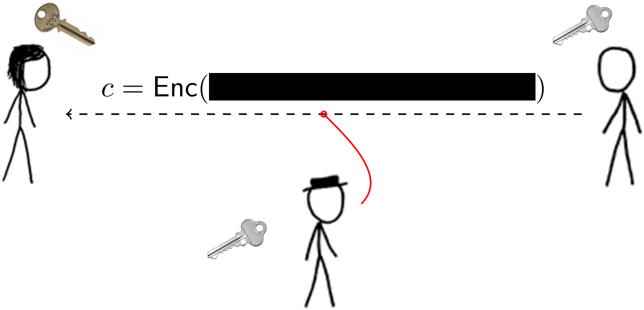
Bread and Butter of PKC: Encryption



$c =$ Enc(surprise party for blackhat!)

- Alice can use the secret key to decrypt the message.

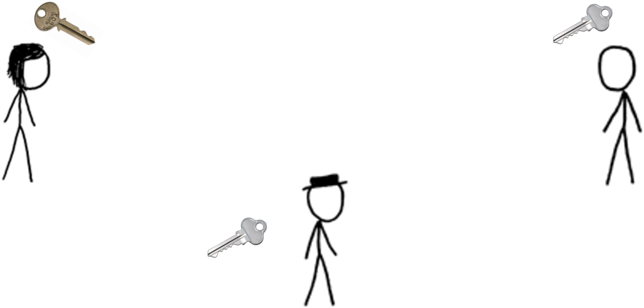
Bread and Butter of PKC: Encryption



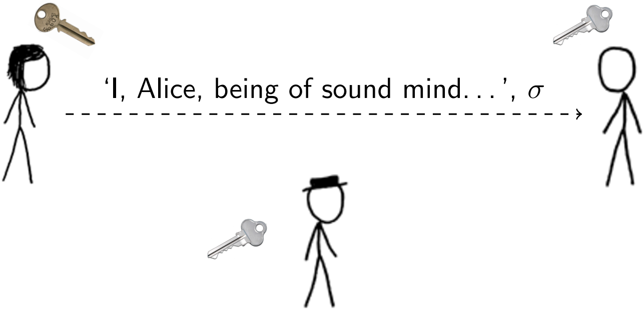
$c = \mathsf{Enc}($[redacted]$)$

- Alice can use the secret key to decrypt the message.
- Eavesdropper who gets the public key and ciphertext learns nothing about the message.

Bread and Butter of PKC: Digital Signatures



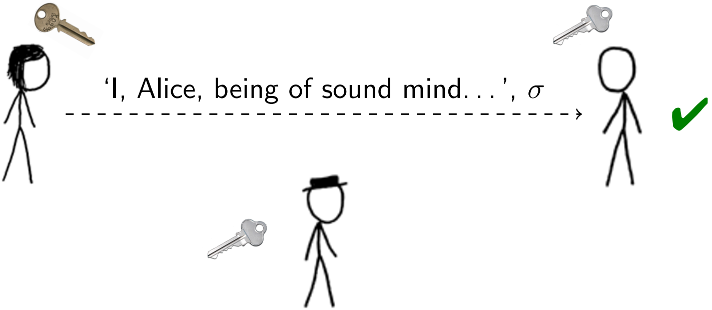
Bread and Butter of PKC: Digital Signatures



'I, Alice, being of sound mind. . . ', σ

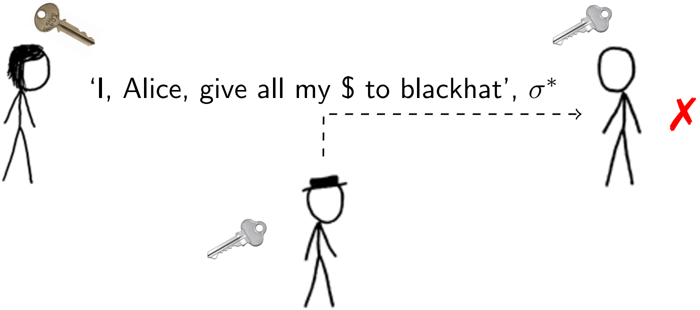
- Alice uses her secret key to create a signature σ for a message.

Bread and Butter of PKC: Digital Signatures



- Alice uses her secret key to create a signature σ for a message.
- Bob can use the public key to verify that the signature is authentic (for this specific message).

Bread and Butter of PKC: Digital Signatures



- Alice uses her secret key to create a signature σ for a message.
- Bob can use the public key to verify that the signature is authentic (for this specific message).
- Attacker can’t forge a valid signature σ^* for an unsigned message.

Hard Problems and PKC

- Public-key crypto inherently requires hard computational problems.

  For one: must be hard to compute the secret key from the public key.

Hard Problems and PKC

- Public-key crypto inherently requires hard computational problems.

  For one: must be hard to compute the secret key from the public key.

- Issue: we don't know whether hard problems exist! (Maybe P=NP.)

Hard Problems and PKC

- Public-key crypto inherently requires hard computational problems.
  For one: must be hard to compute the secret key from the public key.
- Issue: we don't know whether hard problems exist! (Maybe P=NP.)
- 'Solution': conjecture that they do exist—in general, or specifically.

Hard Problems and PKC

- Public-key crypto inherently requires hard computational problems. For one: must be hard to compute the secret key from the public key.
- Issue: we don't know whether hard problems exist! (Maybe P=NP.)
- 'Solution': conjecture that they do exist—in general, or specifically. Then devote scrutiny and algorithmic effort to gain confidence.

Hard Problems and PKC

- Public-key crypto inherently requires hard computational problems.
  For one: must be hard to compute the secret key from the public key.
- Issue: we don't know whether hard problems exist! (Maybe P=NP.)
- 'Solution': conjecture that they do exist—in general, or specifically.
  Then devote scrutiny and algorithmic effort to gain confidence.

> *"Cryptographers seldom sleep well." –Silvio Micali*

Hard Problems and PKC

- Public-key crypto inherently requires hard computational problems. For one: must be hard to compute the secret key from the public key.
- Issue: we don't know whether hard problems exist! (Maybe P=NP.)
- 'Solution': conjecture that they do exist—in general, or specifically. Then devote scrutiny and algorithmic effort to gain confidence.

> *"Cryptographers seldom sleep well." –Silvio Micali*

Case study:

1. RSA/DH 'rely on' the hardness of the factoring/dlog problems:

   Breaking RSA is no harder than factoring: RSA $\leq$ factoring. Obvious.

Hard Problems and PKC

- Public-key crypto inherently requires hard computational problems. For one: must be hard to compute the secret key from the public key.
- Issue: we don't know whether hard problems exist! (Maybe P=NP.)
- 'Solution': conjecture that they do exist—in general, or specifically. Then devote scrutiny and algorithmic effort to gain confidence.

> *"Cryptographers seldom sleep well." –Silvio Micali*

Case study:

1. RSA/DH 'rely on' the hardness of the factoring/dlog problems:
   Breaking RSA is no harder than factoring: RSA $\leq$ factoring. Obvious.
2. RSA/DH are 'based on' the hardness of factoring/dlog variants:
   Breaking RSA is not (much) easier than the 'RSA problem.' Trickier!

How Hard, and Hard How?

- We need crypto problems to be infeasible for any attacker to solve.

How Hard, and Hard How?

- We need crypto problems to be infeasible for any attacker to solve.
- Traditionally, ‘attacker’ $=$ classical algorithm.

How Hard, and Hard How?

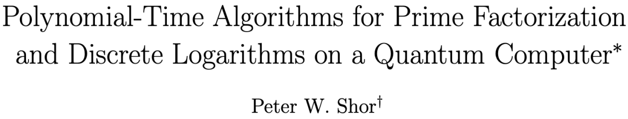
- We need crypto problems to be infeasible for any attacker to solve.
- Traditionally, ‘attacker’ = classical algorithm.
- But for quantum algorithms, ‘feasible’ appears broader:

  [Feynman’82,Deutch’85,BV’93,Simon’94]

How Hard, and Hard How?

- We need crypto problems to be infeasible for any attacker to solve.
- Traditionally, ‘attacker’ = classical algorithm.
- But for quantum algorithms, ‘feasible’ appears broader:

[Feynman’82,Deutch’85,BV’93,Simon’94]

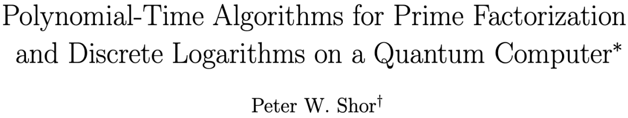


Polynomial-Time Algorithms for Prime Factorization
and Discrete Logarithms on a Quantum Computer*

Peter W. Shor†

How Hard, and Hard How?

- We need crypto problems to be infeasible for any attacker to solve.
- Traditionally, ‘attacker’ = classical algorithm.
- But for quantum algorithms, ‘feasible’ appears broader:
  [Feynman’82,Deutch’85,BV’93,Simon’94]



Polynomial-Time Algorithms for Prime Factorization
and Discrete Logarithms on a Quantum Computer*

Peter W. Shor†

- With a large-scale QC, [Shor’94] totally breaks DH, RSA, and all other widely used public-key crypto!

Post-Quantum Cryptography

Question: Did Shor show that secure PKC is impossible against quantum computers?

Post-Quantum Cryptography

Question: Did Shor show that secure PKC is impossible against quantum computers?

Answer: No! Only that all the PKC we've been widely using is quantumly broken. (What rotten luck. . .)

Post-Quantum Cryptography

Question: Did Shor show that secure PKC is impossible against quantum computers?

Answer: No! Only that all the PKC we've been widely using is quantumly broken. (What rotten luck. . .)

Post-Quantum Cryptography

Post-Quantum Cryptography

Question: Did Shor show that secure PKC is impossible against quantum computers?

Answer: No! Only that all the PKC we've been widely using is quantumly broken. (What rotten luck. . .)

Post-Quantum Cryptography (a.k.a. 'Quantum Resistant', 'Quantum Safe', . . .)

Post-Quantum Cryptography

Question: Did Shor show that secure PKC is impossible against quantum computers?

Answer: No! Only that all the PKC we've been widely using is quantumly broken. (What rotten luck. . .)

Post-Quantum Cryptography (a.k.a. 'Quantum Resistant', 'Quantum Safe', . . .)

Design cryptosystems that can

run on (today's) classical computers,

while being

secure against quantum attacks.

What’s the Rush?

- Big QCs probably won’t exist for many years, if ever—can’t we wait until they’re more imminent?

What’s the Rush?

- Big QCs probably won’t exist for many years, if ever—can’t we wait until they’re more imminent? No!

What's the Rush?

- Big QCs probably won't exist for many years, if ever—can't we wait until they're more imminent? No!

1. Harvesting attacks: store today's keys/ciphertexts to break later.

What's the Rush?

- Big QCs probably won't exist for many years, if ever—can't we wait until they're more imminent? No!

1. Harvesting attacks: store today's keys/ciphertexts to break later.
2. Rewrite history: forge signatures for old keys (e.g., in blockchains).

What's the Rush?

- Big QCs probably won't exist for many years, if ever—can't we wait until they're more imminent? No!

1. Harvesting attacks: store today's keys/ciphertexts to break later.
2. Rewrite history: forge signatures for old keys (e.g., in blockchains).

> *"Who controls history controls the future."*
>
> *–George Orwell, 1984*

What's the Rush?

- Big QCs probably won't exist for many years, if ever—can't we wait until they're more imminent? No!

① Harvesting attacks: store today's keys/ciphertexts to break later.

② Rewrite history: forge signatures for old keys (e.g., in blockchains).

> *"Who controls history controls the future."*
>
> *–George Orwell, 1984*

–BTTF (1985)

What's the Rush?

- Big QCs probably won't exist for many years, if ever—can't we wait until they're more imminent? No!

❶ Harvesting attacks: store today's keys/ciphertexts to break later.

❷ Rewrite history: forge signatures for old keys (e.g., in blockchains).

❸ Deploying new cryptography at scale takes a long time: 10+ years.

What's the Rush?

- Big QCs probably won't exist for many years, if ever—can't we wait until they're more imminent? No!

1. Harvesting attacks: store today's keys/ciphertexts to break later.
2. Rewrite history: forge signatures for old keys (e.g., in blockchains).
3. Deploying new cryptography at scale takes a long time: 10+ years.

> *"IAD will initiate a transition to quantum resistant algorithms in the not too distant future. . . Our ultimate goal is to provide cost effective security against a potential quantum computer."*
>
> *–NSA, 2015*

What's the Rush?

- Big QCs probably won't exist for many years, if ever—can't we wait until they're more imminent? No!

1. Harvesting attacks: store today's keys/ciphertexts to break later.
2. Rewrite history: forge signatures for old keys (e.g., in blockchains).
3. Deploying new cryptography at scale takes a long time: 10+ years.

> *"IAD will initiate a transition to quantum resistant algorithms in the not too distant future. . . Our ultimate goal is to provide cost effective security against a potential quantum computer."*
>
> *–NSA, 2015*

- NIST PQC standardization process (2016–):
  3rd round, finalists and alternates chosen, selections imminent

Tutorial Agenda

1. A highly selective tour of the PQC landscape:
   concepts, key techniques, theory and practice
2. A lot/some/very little of what I know a lot/some/very little about:
   lattices / isogenies / MQ and codes
3. Important problems that need more scrutiny from quantum experts!

Lattices

Lattice-Based Cryptography

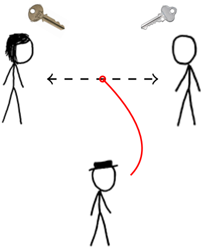
$y = g^x \bmod \mathrm{p}$

$N = p \cdot q$

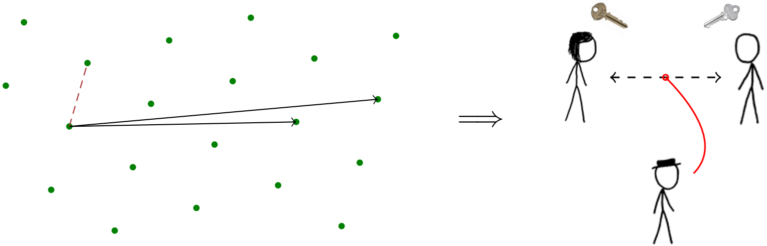
$m^e \bmod N$

$e(g^a, g^b)$

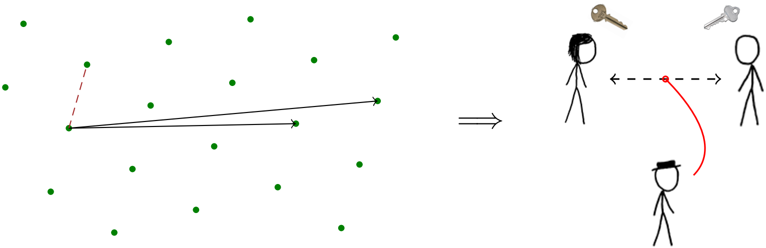
$\Longrightarrow$



Lattice-Based Cryptography



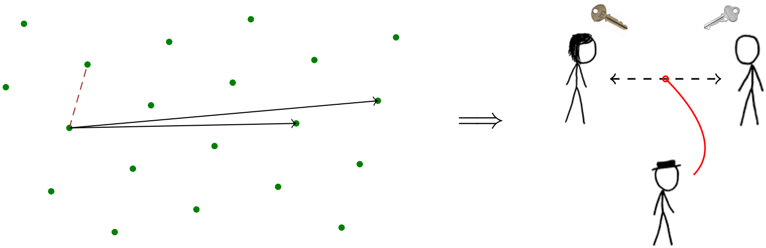
Lattice-Based Cryptography



Why?

- **Efficient**: linear, embarrassingly parallel operations

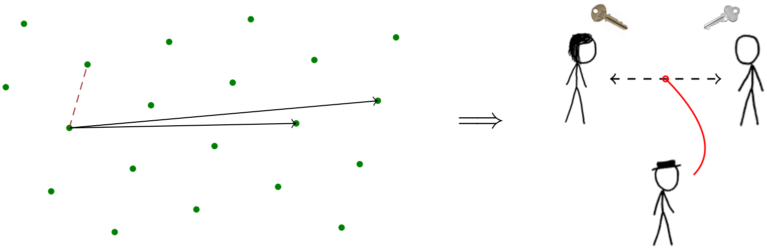
Lattice-Based Cryptography



Why?

- Efficient: linear, embarrassingly parallel operations
- Resists quantum attacks (so far)

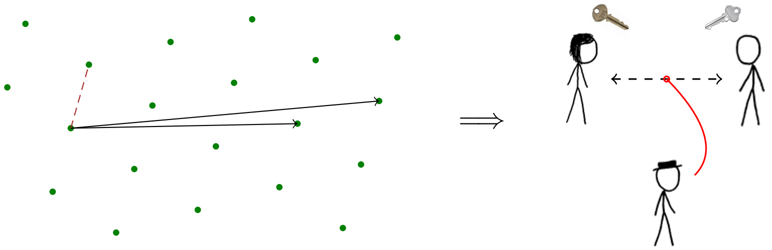
Lattice-Based Cryptography



Why?

- **Efficient**: linear, embarrassingly parallel operations
- Resists **quantum** attacks (so far)
- Security from mild **worst-case** assumptions

Lattice-Based Cryptography

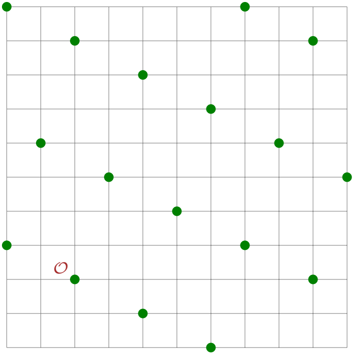


Why?

- Efficient: linear, embarrassingly parallel operations
- Resists quantum attacks (so far)
- Security from mild worst-case assumptions
- Solutions to ‘holy grail’ problems in crypto: FHE and related

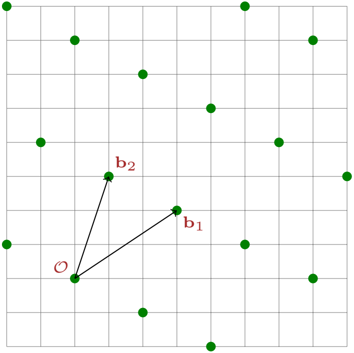
What's a Lattice?

- A periodic 'grid' in $\mathbb{Z}^m$. (Formally: full-rank additive subgroup.)



What's a Lattice?

- A periodic 'grid' in $\mathbb{Z}^m$. (Formally: full-rank additive subgroup.)



- Basis $\mathbf{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_m\}$:

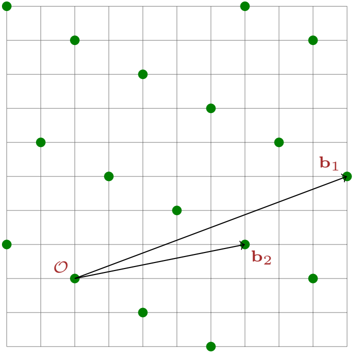
$$\mathcal{L} \quad = \quad \sum_{i=1}^{m} (\mathbb{Z} \cdot \mathbf{b}_i)$$

What's a Lattice?

- A periodic 'grid' in $\mathbb{Z}^m$. (Formally: full-rank additive subgroup.)

- Basis $\mathbf{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_m\}$:

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



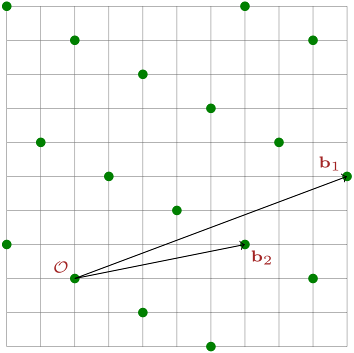
What's a Lattice?

- A periodic 'grid' in $\mathbb{Z}^m$. (Formally: full-rank additive subgroup.)

- Basis $\mathbf{B} = \{\mathbf{b}_1, \dots, \mathbf{b}_m\}$:

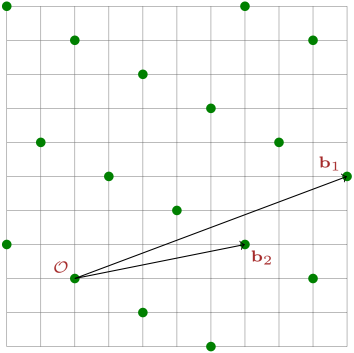
$$\mathcal{L} \quad = \quad \sum_{i=1}^{m} (\mathbb{Z} \cdot \mathbf{b}_i)$$

(Other representations too . . .)



What's a Lattice?

- A periodic 'grid' in $\mathbb{Z}^m$. (Formally: full-rank additive subgroup.)



- Basis $\mathbf{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_m\}$:

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

(Other representations too ...)

Hard Lattice Problems

- Find/detect 'short' nonzero lattice vectors: (Gap)SVP$_\gamma$, SIVP$_\gamma$
- For $\gamma = \text{poly}(m)$, appears to require $2^{\Omega(m)}$ time and space, even quantumly. [LLL'82,Schnorr'87,...,AKS'01,...]

Lattices

Foundations, Digital Signatures

A Hard Problem: Short Integer Solution [Ajtai'96]

- $\mathbb{Z}_q^n = n$-dimensional integer vectors modulo q

A Hard Problem: Short Integer Solution [Ajtai'96]

- $\mathbb{Z}_q^n = n$-dimensional integer vectors modulo q

$$\begin{pmatrix} | \\ \mathbf{a}_1 \\ | \end{pmatrix} \qquad \begin{pmatrix} | \\ \mathbf{a}_2 \\ | \end{pmatrix} \qquad \cdots \qquad \begin{pmatrix} | \\ \mathbf{a}_m \\ | \end{pmatrix} \qquad \in \mathbb{Z}_q^n$$

A Hard Problem: Short Integer Solution [Ajtai'96]

- $\mathbb{Z}_q^n = n$-dimensional integer vectors modulo q
- Goal: find nontrivial $z_1, \ldots, z_m \in \{0, \pm 1\}$ such that:

$$z_1 \cdot \begin{pmatrix} | \\ \mathbf{a}_1 \\ | \end{pmatrix} + z_2 \cdot \begin{pmatrix} | \\ \mathbf{a}_2 \\ | \end{pmatrix} + \cdots + z_m \cdot \begin{pmatrix} | \\ \mathbf{a}_m \\ | \end{pmatrix} = \begin{pmatrix} | \\ 0 \\ | \end{pmatrix} \in \mathbb{Z}_q^n$$

A Hard Problem: Short Integer Solution [Ajtai'96]

- $\mathbb{Z}_q^n = n$-dimensional integer vectors modulo q
- Goal: find nontrivial 'short' $\mathbf{z} \in \mathbb{Z}^m$, $\|\mathbf{z}\| \leq \beta \ll q$ such that:

$$\underbrace{\begin{pmatrix} \cdots\cdot & \mathbf{A} & \cdots\cdot \end{pmatrix}}_{m} \begin{pmatrix} \\ \mathbf{z} \\ \\ \end{pmatrix} = \mathbf{0} \in \mathbb{Z}_q^n$$

A Hard Problem: Short Integer Solution [Ajtai'96]

- $\mathbb{Z}_q^n = n$-dimensional integer vectors modulo q
- Goal: find nontrivial 'short' $\mathbf{z} \in \mathbb{Z}^m$, $\|\mathbf{z}\| \leq \beta \ll q$ such that:

$$\underbrace{\begin{pmatrix} \cdots\cdot & \mathbf{A} & \cdots\cdot \\ & & \end{pmatrix}}_{m} \begin{pmatrix} \\ \mathbf{z} \\ \\ \end{pmatrix} = \mathbf{0} \in \mathbb{Z}_q^n$$

Collision-Resistant Hash Function

- Set $m > n \log_2 q$. Define 'compressing' $f_{\mathbf{A}} : \{0,1\}^m \to \mathbb{Z}_q^n$

$$f_{\mathbf{A}}(\mathbf{x}) = \mathbf{A}\mathbf{x}$$

A Hard Problem: Short Integer Solution [Ajtai'96]

- $\mathbb{Z}_q^n = n$-dimensional integer vectors modulo q
- Goal: find nontrivial 'short' $\mathbf{z} \in \mathbb{Z}^m$, $\|\mathbf{z}\| \leq \beta \ll q$ such that:

$$\underbrace{\begin{pmatrix} \cdots\cdots & \mathbf{A} & \cdots\cdots \\ & & \end{pmatrix}}_{m} \begin{pmatrix} \\ \mathbf{z} \\ \\ \end{pmatrix} = \mathbf{0} \in \mathbb{Z}_q^n$$

Collision-Resistant Hash Function

- Set $m > n \log_2 q$. Define 'compressing' $f_{\mathbf{A}} : \{0,1\}^m \to \mathbb{Z}_q^n$

$$f_{\mathbf{A}}(\mathbf{x}) = \mathbf{A}\mathbf{x}$$

- Collision $\mathbf{x}, \mathbf{x}' \in \{0,1\}^m$ where $\mathbf{A}\mathbf{x} = \mathbf{A}\mathbf{x}' \ldots$

A Hard Problem: Short Integer Solution [Ajtai'96]

- $\mathbb{Z}_q^n = n$-dimensional integer vectors modulo q
- Goal: find nontrivial 'short' $\mathbf{z} \in \mathbb{Z}^m$, $\|\mathbf{z}\| \leq \beta \ll q$ such that:

$$\underbrace{\begin{pmatrix} \cdots\cdot & \mathbf{A} & \cdots\cdot \end{pmatrix}}_{m} \begin{pmatrix} \\ \mathbf{z} \\ \\ \end{pmatrix} = \mathbf{0} \in \mathbb{Z}_q^n$$

Collision-Resistant Hash Function

- Set $m > n \log_2 q$. Define 'compressing' $f_{\mathbf{A}} \colon \{0,1\}^m \to \mathbb{Z}_q^n$

$$f_{\mathbf{A}}(\mathbf{x}) = \mathbf{A}\mathbf{x}$$

- Collision $\mathbf{x}, \mathbf{x}' \in \{0,1\}^m$ where $\mathbf{A}\mathbf{x} = \mathbf{A}\mathbf{x}'$ …

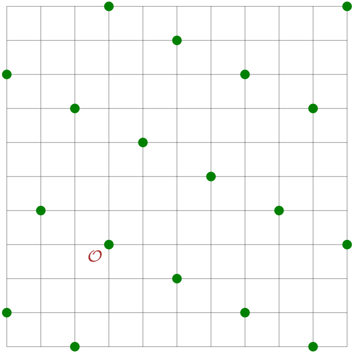
… yields short solution $\mathbf{z} = \mathbf{x} - \mathbf{x}' \in \{0, \pm 1\}^m$.

Cool! (But what does this have to do with lattices?)

Cool! (But what does this have to do with lattices?)

- $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ defines a 'q-ary' lattice:

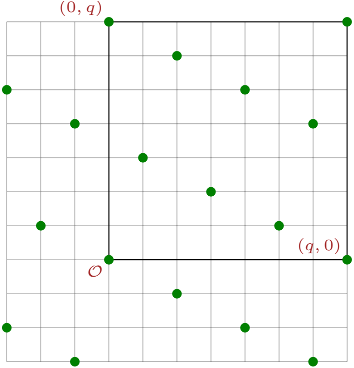
$$\mathcal{L}^{\perp}(\mathbf{A}) := \{\mathbf{z} \in \mathbb{Z}^m \ : \ \mathbf{A}\mathbf{z} = \mathbf{0}\}$$



Cool! (But what does this have to do with lattices?)

- $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ defines a 'q-ary' lattice:

  $$\mathcal{L}^{\perp}(\mathbf{A}) := \{\mathbf{z} \in \mathbb{Z}^m \ : \ \mathbf{Az} = \mathbf{0}\} \supseteq q\mathbb{Z}^m$$

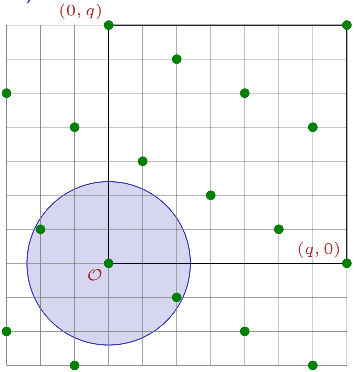


Cool! (But what does this have to do with lattices?)

- $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ defines a 'q-ary' lattice:

  $$\mathcal{L}^{\perp}(\mathbf{A}) := \{\mathbf{z} \in \mathbb{Z}^m \ : \ \mathbf{A}\mathbf{z} = \mathbf{0}\} \supseteq q\mathbb{Z}^m$$

- 'Short' solutions $\mathbf{z}$ lie in ○

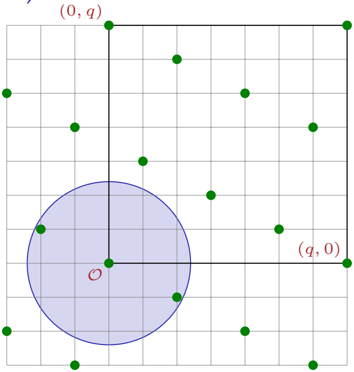


Cool! (But what does this have to do with lattices?)

- $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ defines a 'q-ary' lattice:

$$\mathcal{L}^{\perp}(\mathbf{A}) := \{\mathbf{z} \in \mathbb{Z}^m \ : \ \mathbf{A}\mathbf{z} = \mathbf{0}\} \supseteq q\mathbb{Z}^m$$

- 'Short' solutions $\mathbf{z}$ lie in ◯



Worst-Case to Average-Case Reduction [Ajtai'96,...]

Finding 'short' ($\|\mathbf{z}\| \leq \beta \ll q$) nonzero $\mathbf{z} \in \mathcal{L}^{\perp}(\mathbf{A})$

(for uniformly random $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$)

$\Downarrow$

solving GapSVP$_{\beta\sqrt{n}}$ and SIVP$_{\beta\sqrt{n}}$ on any n-dim lattice

Application: Digital Signatures [GentryPeikertVaikuntanathan'08]

- Generate uniform $vk = \mathbf{A}$ with secret 'trapdoor' $sk = \mathbf{T}$.

Application: Digital Signatures [GentryPeikertVaikuntanathan'08]

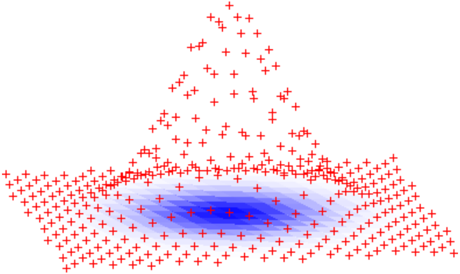
- Generate uniform $vk = \mathbf{A}$ with secret 'trapdoor' $sk = \mathbf{T}$.
- Sign$(\mathbf{T}, \mu)$: use $\mathbf{T}$ to sample a short $\mathbf{z} \in \mathbb{Z}^m$ s.t. $\mathbf{A}\mathbf{z} = H(\mu) \in \mathbb{Z}_q^n$.

Application: Digital Signatures [GentryPeikertVaikuntanathan'08]

- Generate uniform $vk = \mathbf{A}$ with secret 'trapdoor' $sk = \mathbf{T}$.
- Sign$(\mathbf{T}, \mu)$: use $\mathbf{T}$ to sample a short $\mathbf{z} \in \mathbb{Z}^m$ s.t. $\mathbf{A}\mathbf{z} = H(\mu) \in \mathbb{Z}_q^n$.

  Draw $\mathbf{z}$ from a distribution that reveals nothing about the secret key:

  (avoids 'learning' attacks [GS'02,NR'06,DN'12])

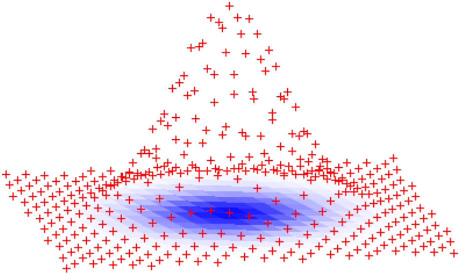


Application: Digital Signatures [GentryPeikertVaikuntanathan'08]

- Generate uniform $vk = \mathbf{A}$ with secret 'trapdoor' $sk = \mathbf{T}$.
- Sign$(\mathbf{T}, \mu)$: use $\mathbf{T}$ to sample a short $\mathbf{z} \in \mathbb{Z}^m$ s.t. $\mathbf{A}\mathbf{z} = H(\mu) \in \mathbb{Z}_q^n$.

  Draw $\mathbf{z}$ from a distribution that reveals nothing about the secret key:

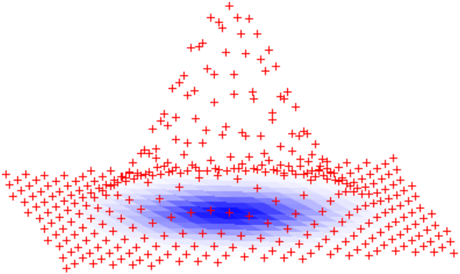
  (avoids 'learning' attacks [GS'02,NR'06,DN'12])



- Verify$(\mathbf{A}, \mu, \mathbf{z})$: check that $\mathbf{A}\mathbf{z} = H(\mu)$ and $\mathbf{z}$ is sufficiently short.

Application: Digital Signatures [GentryPeikertVaikuntanathan'08]

- Generate uniform $vk = \mathbf{A}$ with secret 'trapdoor' $sk = \mathbf{T}$.
- Sign$(\mathbf{T}, \mu)$: use $\mathbf{T}$ to sample a short $\mathbf{z} \in \mathbb{Z}^m$ s.t. $\mathbf{A}\mathbf{z} = H(\mu) \in \mathbb{Z}_q^n$.
  Draw $\mathbf{z}$ from a distribution that reveals nothing about the secret key:
  (avoids 'learning' attacks [GS'02,NR'06,DN'12])



- Verify$(\mathbf{A}, \mu, \mathbf{z})$: check that $\mathbf{A}\mathbf{z} = H(\mu)$ and $\mathbf{z}$ is sufficiently short.
- Security: forging a signature for a new message μ^* requires finding short $\mathbf{z}^*$ s.t. $\mathbf{A}\mathbf{z}^* = H(\mu^*)$. This is SIS: hard!

Signatures In Practice: Falcon [FHK+'17], Dilithium [DKL+'17]

Refinements to the two components of the [GPV'08] framework:

Signatures In Practice: Falcon [FHK+'17], Dilithium [DKL+'17]

Refinements to the two components of the [GPV'08] framework:

1. Generating a 'hard' lattice/trapdoor pair:
   [GGH'97,A'99,HHPSW'01,AP'09,SS'11,MP'12,PP'19, . . .]

Signatures In Practice: Falcon [FHK+'17], Dilithium [DKL+'17]

Refinements to the two components of the [GPV'08] framework:

1. Generating a 'hard' lattice/trapdoor pair:
   [GGH'97,A'99,HHPSW'01,AP'09,SS'11,MP'12,PP'19, . . .]

   Question: is it hard to decode w/in threshold distance on lattices produced by the above (non-blue) methods?

Signatures In Practice: Falcon [FHK+'17], Dilithium [DKL+'17]

Refinements to the two components of the [GPV'08] framework:

1. Generating a 'hard' lattice/trapdoor pair:
   [GGH'97,A'99,HHPSW'01,AP'09,SS'11,MP'12,PP'19, . . .]

   Question: is it hard to decode w/in threshold distance on lattices produced by the above (non-blue) methods?

2. Gaussian lattice sampling: [P'10,MP'12,DP'16,. . .]

Signatures In Practice: Falcon [FHK+'17], Dilithium [DKL+'17]

Refinements to the two components of the [GPV'08] framework:

1. Generating a 'hard' lattice/trapdoor pair:
   [GGH'97,A'99,HHPSW'01,AP'09,SS'11,MP'12,PP'19, . . .]

   Question: is it hard to decode w/in threshold distance on lattices produced by the above (non-blue) methods?

2. Gaussian lattice sampling: [P'10,MP'12,DP'16,. . .]

NIST PQC finalist Falcon uses these to get smallest vk + sig size, also with very fast verification.

Signatures In Practice: Falcon [FHK+'17], Dilithium [DKL+'17]

Refinements to the two components of the [GPV'08] framework:

1. Generating a 'hard' lattice/trapdoor pair:
   [GGH'97,A'99,HHPSW'01,AP'09,SS'11,MP'12,PP'19, . . .]

   Question: is it hard to decode w/in threshold distance on lattices produced by the above (non-blue) methods?

2. Gaussian lattice sampling: [P'10,MP'12,DP'16,. . .]

NIST PQC finalist Falcon uses these to get smallest vk + sig size, also with very fast verification.

Finalist Dilithium uses 'Fiat-Shamir with aborts' [Lyubashevsky'09,'12]: very simple signing algorithm! (No Gaussian sampling needed.)

Signatures In Practice: Falcon [FHK+'17], Dilithium [DKL+'17]

Refinements to the two components of the [GPV'08] framework:

1. Generating a 'hard' lattice/trapdoor pair: [GGH'97,A'99,HHPSW'01,AP'09,SS'11,MP'12,PP'19, . . .]

   Question: is it hard to decode w/in threshold distance on lattices produced by the above (non-blue) methods?

2. Gaussian lattice sampling: [P'10,MP'12,DP'16,. . .]

NIST PQC finalist Falcon uses these to get smallest vk + sig size, also with very fast verification.

Finalist Dilithium uses 'Fiat-Shamir with aborts' [Lyubashevsky'09,'12]: very simple signing algorithm! (No Gaussian sampling needed.)

Questions

1. Is SIS (quantumly) hard for solution norm $\lesssim q$ in ℓ_∞ norm?

Signatures In Practice: Falcon [FHK+'17], Dilithium [DKL+'17]

Refinements to the two components of the [GPV'08] framework:

1. Generating a 'hard' lattice/trapdoor pair:
   [GGH'97,A'99,HHPSW'01,AP'09,SS'11,MP'12,PP'19, . . .]

   Question: is it hard to decode w/in threshold distance on lattices produced by the above (non-blue) methods?

2. Gaussian lattice sampling: [P'10,MP'12,DP'16,. . .]

NIST PQC finalist Falcon uses these to get smallest vk + sig size, also with very fast verification.

Finalist Dilithium uses 'Fiat-Shamir with aborts' [Lyubashevsky'09,'12]: very simple signing algorithm! (No Gaussian sampling needed.)

Questions

1. Is SIS (quantumly) hard for solution norm $\lesssim q$ in ℓ_∞ norm?
2. Tighter security reduction in QROM, or exploit looseness?

See [BDF+'12,KLS'18,DFMS'19,LZ'19].

Lattices

Public-Key Encryption

Another Hard Problem: Learning With Errors [Regev'05]

- Parameters: dimension n, modulus q, error distribution

Another Hard Problem: Learning With Errors [Regev'05]

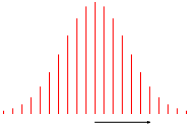
- Parameters: dimension n, modulus q, error distribution
- **Search:** find secret $\mathbf{s} \in \mathbb{Z}_q^n$ given many 'noisy inner products'

$$\begin{array}{ll} \mathbf{a}_1 \leftarrow \mathbb{Z}_q^n \quad , & b_1 \approx \langle \mathbf{s} \, , \, \mathbf{a}_1 \rangle \bmod q \\ \mathbf{a}_2 \leftarrow \mathbb{Z}_q^n \quad , & b_2 \approx \langle \mathbf{s} \, , \, \mathbf{a}_2 \rangle \bmod q \\ & \vdots \end{array}$$

Another Hard Problem: Learning With Errors [Regev'05]

- Parameters: dimension n, modulus q, error distribution
- **Search:** find secret $\mathbf{s} \in \mathbb{Z}_q^n$ given many 'noisy inner products'

$$\begin{array}{lcl} \mathbf{a}_1 \leftarrow \mathbb{Z}_q^n & , & b_1 = \langle \mathbf{s} \, , \, \mathbf{a}_1 \rangle + e_1 \in \mathbb{Z}_q \\ \mathbf{a}_2 \leftarrow \mathbb{Z}_q^n & , & b_2 = \langle \mathbf{s} \, , \, \mathbf{a}_2 \rangle + e_2 \in \mathbb{Z}_q \\ & \vdots & \end{array}$$

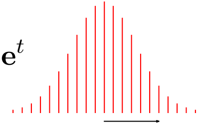


$\sqrt{n} \leq$ error $\ll q$, 'rate' α

Another Hard Problem: Learning With Errors [Regev'05]

- Parameters: dimension n, modulus q, error distribution
- **Search:** find secret $\mathbf{s} \in \mathbb{Z}_q^n$ given many 'noisy inner products'

$$\begin{pmatrix} \cdots & \mathbf{A} & \cdots \\ & & \end{pmatrix} \quad , \quad \begin{pmatrix} \cdots & \mathbf{b}^t & \cdots \end{pmatrix} = \mathbf{s}^t\mathbf{A} + \mathbf{e}^t$$

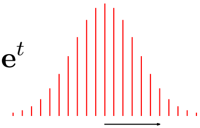


$\sqrt{n} \leq$ error $\ll q$, 'rate' α

Another Hard Problem: Learning With Errors [Regev'05]

- Parameters: dimension n, modulus q, error distribution
- **Search:** find secret $\mathbf{s} \in \mathbb{Z}_q^n$ given many 'noisy inner products'

$$\left(\cdots \quad \mathbf{A} \quad \cdots \right) \quad , \quad \left(\cdots \quad \mathbf{b}^t \quad \cdots \right) = \mathbf{s}^t \mathbf{A} + \mathbf{e}^t$$



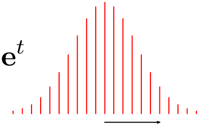
$\sqrt{n} \leq$ error $\ll q$, 'rate' α

- **Decision:** distinguish $(\mathbf{A}\ ,\ \mathbf{b})$ from uniform $(\mathbf{A}\ ,\ \mathbf{b})$

Another Hard Problem: Learning With Errors [Regev'05]

- Parameters: dimension n, modulus q, error distribution
- **Search:** find secret $\mathbf{s} \in \mathbb{Z}_q^n$ given many 'noisy inner products'

$$\left(\cdots \quad \mathbf{A} \quad \cdots \right) \quad , \quad \left(\cdots \quad \mathbf{b}^t \quad \cdots \right) = \mathbf{s}^t \mathbf{A} + \mathbf{e}^t$$



$\sqrt{n} \leq$ error $\ll q$, 'rate' α

- **Decision:** distinguish $(\mathbf{A}\ ,\ \mathbf{b})$ from uniform $(\mathbf{A}\ ,\ \mathbf{b})$

LWE is Hard

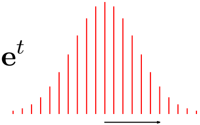
(n/α)-approx *worst case* lattice problems $\underset{\uparrow}{\leq}$ search-LWE $\underset{\uparrow}{\leq}$ decision-LWE $\leq$ crypto

(quantum [R'05]) [BFKL'93,R'05,. . .]

Another Hard Problem: Learning With Errors [Regev'05]

- Parameters: dimension n, modulus q, error distribution
- **Search:** find secret $\mathbf{s} \in \mathbb{Z}_q^n$ given many 'noisy inner products'

$$\left(\cdots \quad \mathbf{A} \quad \cdots \right) \quad , \quad \left(\cdots \quad \mathbf{b}^t \quad \cdots \right) = \mathbf{s}^t \mathbf{A} + \mathbf{e}^t$$



$\sqrt{n} \leq$ error $\ll q$, 'rate' α

- **Decision:** distinguish $(\mathbf{A}\ ,\ \mathbf{b})$ from uniform $(\mathbf{A}\ ,\ \mathbf{b})$

LWE is Hard

(n/α)-approx *worst case* lattice problems $\underset{\substack{\uparrow \\ \text{(quantum [R'05])}}}{\leq}$ search-LWE $\underset{\substack{\uparrow \\ \text{[BFKL'93,R'05,\ldots]}}}{\leq}$ decision-LWE $\leq$ crypto

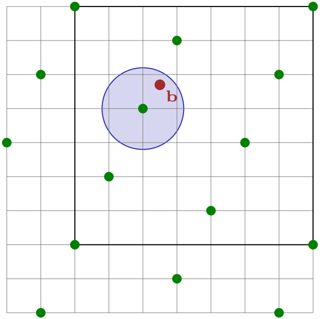
- Also fully *classical* reductions, for worse params [Peikert'09,BLPRS'13]

LWE is a Lattice Problem

- LWE is ‘dual’ to SIS. Let

$$\mathcal{L}(\mathbf{A}) = \{\mathbf{z}^t \equiv \mathbf{s}^t\mathbf{A} \bmod q\}\ .$$

Given $\mathbf{A}$ and $\mathbf{b} \approx \mathbf{s}\mathbf{A}$, find $\mathbf{s}$.



LWE is a Lattice Problem

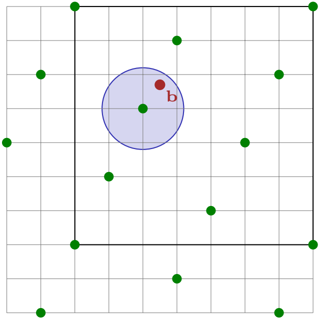
- LWE is ‘dual’ to SIS. Let

$$\mathcal{L}(\mathbf{A}) = \{\mathbf{z}^t \equiv \mathbf{s}^t\mathbf{A} \bmod q\} \ .$$

Given $\mathbf{A}$ and $\mathbf{b} \approx \mathbf{s}\mathbf{A}$, find $\mathbf{s}$.

Bounded-Distance Decoding (BDD$_\alpha$)

- Given a target that's ‘α-far’ from a lattice point, find that point.

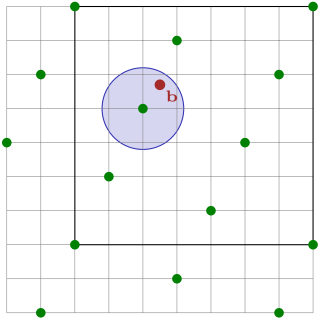


LWE is a Lattice Problem

- LWE is 'dual' to SIS. Let

$$\mathcal{L}(\mathbf{A}) = \{\mathbf{z}^t \equiv \mathbf{s}^t \mathbf{A} \bmod q\} \ .$$

Given $\mathbf{A}$ and $\mathbf{b} \approx \mathbf{s}\mathbf{A}$, find $\mathbf{s}$.



Bounded-Distance Decoding (BDD$_\alpha$)

- Given a target that's 'α-far' from a lattice point, find that point.

Theorem [Regev'05]

solving BDD$_\alpha$ on lattice $\mathcal{L}$

(quantum)

$$\Downarrow$$

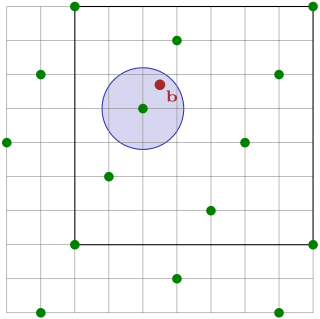
Gaussian sampling '$(1/\alpha)$-short' points from dual lattice $\mathcal{L}^*$

LWE is a Lattice Problem

- LWE is ‘dual’ to SIS. Let

$$\mathcal{L}(\mathbf{A}) = \{\mathbf{z}^t \equiv \mathbf{s}^t\mathbf{A} \bmod q\} \ .$$

  Given $\mathbf{A}$ and $\mathbf{b} \approx \mathbf{s}\mathbf{A}$, find $\mathbf{s}$.



Bounded-Distance Decoding (BDD_α)

- Given a target that's ‘α-far’ from a lattice point, find that point.

Theorem [Regev’05]

solving BDD_α on lattice $\mathcal{L}$

(quantum)

$$\Downarrow$$

Gaussian sampling ‘$(1/\alpha)$-short’ points from dual lattice $\mathcal{L}^*$

- Key Open Problem: ‘dequantize’ this theorem!

LWE is Versatile

Cryptography we can build from LWE:

LWE is Versatile

Cryptography we can build from LWE:

- ✓ Key Exchange and Public Key Encryption
- ✓ Oblivious Transfer
- ✓ Actively Secure Encryption (w/o random oracles)
- ✓ Low-Depth Pseudorandom Functions

LWE is Versatile

Cryptography we can build from LWE:

✔ Key Exchange and Public Key Encryption

✔ Oblivious Transfer

✔ Actively Secure Encryption (w/o random oracles)

✔ Low-Depth Pseudorandom Functions

✔✔ Identity-Based Encryption (w/o RO)

✔✔ Hierarchical ID-Based Encryption (w/o RO)

✔✔ Noninteractive Zero Knowledge for NP

LWE is Versatile

Cryptography we can build from LWE:

- ✓ Key Exchange and Public Key Encryption
- ✓ Oblivious Transfer
- ✓ Actively Secure Encryption (w/o random oracles)
- ✓ Low-Depth Pseudorandom Functions

- - - - - - - - -

- ✓✓ Identity-Based Encryption (w/o RO)
- ✓✓ Hierarchical ID-Based Encryption (w/o RO)
- ✓✓ Noninteractive Zero Knowledge for NP

- - - - - - - - -

- !!! Fully Homomorphic Encryption
- !!! Attribute-Based Encryption for arbitrary access policies

and much, much more. . .

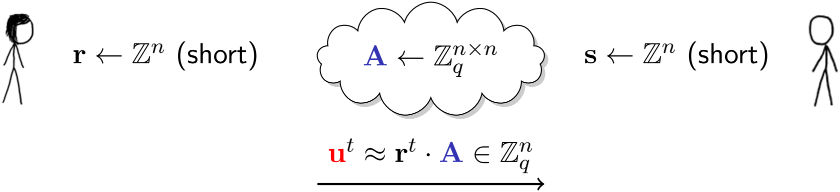
Key Exchange/Encryption from LWE [Regev'05,LPS'10,LP'11]

$\mathbf{r} \leftarrow \mathbb{Z}^n$ (short)

$\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times n}$

$\mathbf{s} \leftarrow \mathbb{Z}^n$ (short)

Key Exchange/Encryption from LWE [Regev'05,LPS'10,LP'11]



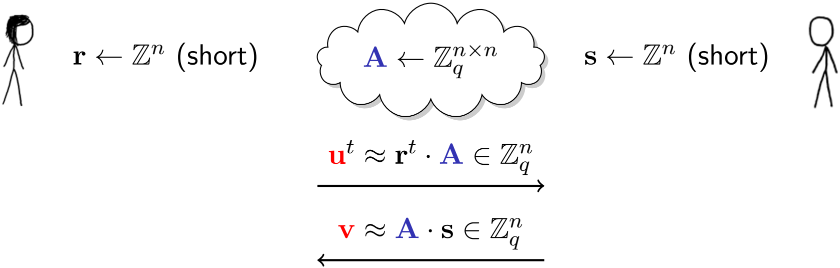
$\mathbf{r} \leftarrow \mathbb{Z}^n$ (short)

$\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times n}$

$\mathbf{s} \leftarrow \mathbb{Z}^n$ (short)

$$\xrightarrow{\mathbf{u}^t \approx \mathbf{r}^t \cdot \mathbf{A} \in \mathbb{Z}_q^n}$$

Key Exchange/Encryption from LWE [Regev'05,LPS'10,LP'11]



$\mathbf{r} \leftarrow \mathbb{Z}^n$ (short)

$\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times n}$

$\mathbf{s} \leftarrow \mathbb{Z}^n$ (short)

$$\xrightarrow{\mathbf{u}^t \approx \mathbf{r}^t \cdot \mathbf{A} \in \mathbb{Z}_q^n}$$

$$\xleftarrow{\mathbf{v} \approx \mathbf{A} \cdot \mathbf{s} \in \mathbb{Z}_q^n}$$

Key Exchange/Encryption from LWE [Regev'05,LPS'10,LP'11]

$\mathbf{r} \leftarrow \mathbb{Z}^n$ (short)

$\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times n}$

$\mathbf{s} \leftarrow \mathbb{Z}^n$ (short)

$$\xrightarrow{\mathbf{u}^t \approx \mathbf{r}^t \cdot \mathbf{A} \in \mathbb{Z}_q^n}$$

$$\xleftarrow{\mathbf{v} \approx \mathbf{A} \cdot \mathbf{s} \in \mathbb{Z}_q^n}$$

$\mathbf{r}^t \cdot \mathbf{v} \approx \mathbf{r}^t \mathbf{A} \mathbf{s} \approx k$

$k \approx \mathbf{u}^t \cdot \mathbf{s} \approx \mathbf{r}^t \mathbf{A} \mathbf{s} \in \mathbb{Z}_q$

Key Exchange/Encryption from LWE [Regev'05,LPS'10,LP'11]

$\mathbf{r} \leftarrow \mathbb{Z}^n$ (short)

$\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times n}$

$\mathbf{s} \leftarrow \mathbb{Z}^n$ (short)

$$\xrightarrow{\mathbf{u}^t \approx \mathbf{r}^t \cdot \mathbf{A} \in \mathbb{Z}_q^n}$$

$$\xleftarrow{\mathbf{v} \approx \mathbf{A} \cdot \mathbf{s} \in \mathbb{Z}_q^n}$$

$$\mathbf{r}^t \cdot \mathbf{v} \approx \mathbf{r}^t \mathbf{A} \mathbf{s} \approx k$$

$$k \approx \mathbf{u}^t \cdot \mathbf{s} \approx \mathbf{r}^t \mathbf{A} \mathbf{s} \in \mathbb{Z}_q$$

$$\xleftarrow{c = k + \text{bit} \cdot \frac{q}{2} \in \mathbb{Z}_q}$$

Key Exchange/Encryption from LWE [Regev'05,LPS'10,LP'11]

$\mathbf{r} \leftarrow \mathbb{Z}^n$ (short)

$\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times n}$

$\mathbf{s} \leftarrow \mathbb{Z}^n$ (short)

$$\xrightarrow{\mathbf{u}^t \approx \mathbf{r}^t \cdot \mathbf{A} \in \mathbb{Z}_q^n}$$

$$\xleftarrow{\mathbf{v} \approx \mathbf{A} \cdot \mathbf{s} \in \mathbb{Z}_q^n}$$

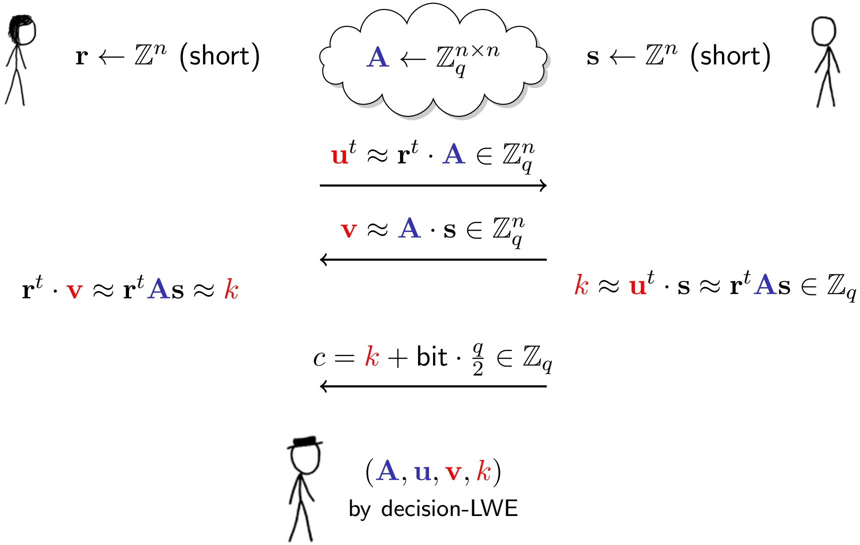
$\mathbf{r}^t \cdot \mathbf{v} \approx \mathbf{r}^t \mathbf{A} \mathbf{s} \approx k$

$k \approx \mathbf{u}^t \cdot \mathbf{s} \approx \mathbf{r}^t \mathbf{A} \mathbf{s} \in \mathbb{Z}_q$

$$\xleftarrow{c = k + \text{bit} \cdot \frac{q}{2} \in \mathbb{Z}_q}$$

$(\mathbf{A}, \mathbf{u}, \mathbf{v}, k)$

Key Exchange/Encryption from LWE [Regev'05,LPS'10,LP'11]



$\mathbf{r} \leftarrow \mathbb{Z}^n$ (short)

$\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times n}$

$\mathbf{s} \leftarrow \mathbb{Z}^n$ (short)

$$\xrightarrow{\mathbf{u}^t \approx \mathbf{r}^t \cdot \mathbf{A} \in \mathbb{Z}_q^n}$$

$$\xleftarrow{\mathbf{v} \approx \mathbf{A} \cdot \mathbf{s} \in \mathbb{Z}_q^n}$$

$\mathbf{r}^t \cdot \mathbf{v} \approx \mathbf{r}^t \mathbf{A} \mathbf{s} \approx k$

$k \approx \mathbf{u}^t \cdot \mathbf{s} \approx \mathbf{r}^t \mathbf{A} \mathbf{s} \in \mathbb{Z}_q$

$$\xleftarrow{c = k + \text{bit} \cdot \frac{q}{2} \in \mathbb{Z}_q}$$

$(\mathbf{A}, \mathbf{u}, \mathbf{v}, k)$

by decision-LWE

Key Exchange/Encryption from LWE [Regev'05,LPS'10,LP'11]

$\mathbf{r} \leftarrow \mathbb{Z}^n$ (short)

$\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times n}$

$\mathbf{s} \leftarrow \mathbb{Z}^n$ (short)

$$\xrightarrow{\mathbf{u}^t \approx \mathbf{r}^t \cdot \mathbf{A} \in \mathbb{Z}_q^n}$$

$$\xleftarrow{\mathbf{v} \approx \mathbf{A} \cdot \mathbf{s} \in \mathbb{Z}_q^n}$$

$\mathbf{r}^t \cdot \mathbf{v} \approx \mathbf{r}^t \mathbf{A} \mathbf{s} \approx k$

$k \approx \mathbf{u}^t \cdot \mathbf{s} \approx \mathbf{r}^t \mathbf{A} \mathbf{s} \in \mathbb{Z}_q$

$$\xleftarrow{c = k + \text{bit} \cdot \frac{q}{2} \in \mathbb{Z}_q}$$

$(\mathbf{A}, \mathbf{u}, \mathbf{v}, k)$

by decision-LWE

LWE In Practice: Frodo(KEM) [BCD+'16,ABD+'17]

NIST PQC alternate FrodoKEM: $640 \leq n \leq 1344$ and $q \in \{2^{15}, 2^{16}\}$.

- Uses Gaussian error of std dev $1.4 \leq \sigma \leq 2.8 \ll \sqrt{n}$.

LWE In Practice: Frodo(KEM) [BCD+'16,ABD+'17]

NIST PQC alternate FrodoKEM: $640 \leq n \leq 1344$ and $q \in \{2^{15}, 2^{16}\}$.

- Uses Gaussian error of std dev $1.4 \leq \sigma \leq 2.8 \ll \sqrt{n}$.
- These params seem hard, according to cryptanalysis. Any theory?

LWE In Practice: Frodo(KEM) [BCD+'16,ABD+'17]

NIST PQC alternate FrodoKEM: $640 \leq n \leq 1344$ and $q \in \{2^{15}, 2^{16}\}$.

- Uses Gaussian error of std dev $1.4 \leq \sigma \leq 2.8 \ll \sqrt{n}$.
- These params seem hard, according to cryptanalysis. Any theory?
- Regev's full quantum reduction doesn't apply for such error, but a component (classical) reduction 'BDD with DGS $\leq$ LWE' does.

LWE In Practice: Frodo(KEM) [BCD+'16,ABD+'17]

NIST PQC alternate FrodoKEM: $640 \le n \le 1344$ and $q \in \{2^{15}, 2^{16}\}$.

- Uses Gaussian error of std dev $1.4 \le \sigma \le 2.8 \ll \sqrt{n}$.
- These params seem hard, according to cryptanalysis. Any theory?
- Regev's full quantum reduction doesn't apply for such error, but a component (classical) reduction 'BDD with DGS $\le$ LWE' does.

BDD with DGS (implicit in [AR'04,R'05,LLM'06,DRS'14])

- Solve BDD to distance d, given N Gaussian samples of width (say) $\ge 2\sqrt{\log N}/d$ over the dual lattice.

LWE In Practice: Frodo(KEM) [BCD+'16,ABD+'17]

NIST PQC alternate FrodoKEM: $640 \leq n \leq 1344$ and $q \in \{2^{15}, 2^{16}\}$.

- Uses Gaussian error of std dev $1.4 \leq \sigma \leq 2.8 \ll \sqrt{n}$.
- These params seem hard, according to cryptanalysis. Any theory?
- Regev's full quantum reduction doesn't apply for such error, but a component (classical) reduction 'BDD with DGS $\leq$ LWE' does.

BDD with DGS (implicit in [AR'04,R'05,LLM'06,DRS'14])

- Solve BDD to distance d, given N Gaussian samples of width (say) $\geq 2\sqrt{\log N}/d$ over the dual lattice.
- Known algorithms can exploit narrower samples, but not these (?).

LWE In Practice: Frodo(KEM) [BCD+'16,ABD+'17]

NIST PQC alternate FrodoKEM: $640 \leq n \leq 1344$ and $q \in \{2^{15}, 2^{16}\}$.

- Uses Gaussian error of std dev $1.4 \leq \sigma \leq 2.8 \ll \sqrt{n}$.
- These params seem hard, according to cryptanalysis. Any theory?
- Regev's full quantum reduction doesn't apply for such error, but a component (classical) reduction 'BDD with DGS $\leq$ LWE' does.

BDD with DGS (implicit in [AR'04,R'05,LLM'06,DRS'14])

- Solve BDD to distance d, given N Gaussian samples of width (say) $\geq 2\sqrt{\log N}/d$ over the dual lattice.
- Known algorithms can exploit narrower samples, but not these (?).

Questions

1. Is BDD w/DGS actually hard? What effect does N have?

LWE In Practice: Frodo(KEM) [BCD+'16,ABD+'17]

NIST PQC alternate FrodoKEM: $640 \leq n \leq 1344$ and $q \in \{2^{15}, 2^{16}\}$.

- Uses Gaussian error of std dev $1.4 \leq \sigma \leq 2.8 \ll \sqrt{n}$.
- These params seem hard, according to cryptanalysis. Any theory?
- Regev's full quantum reduction doesn't apply for such error, but a component (classical) reduction 'BDD with DGS $\leq$ LWE' does.

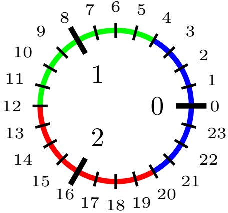
BDD with DGS (implicit in [AR'04,R'05,LLM'06,DRS'14])

- Solve BDD to distance d, given N Gaussian samples of width (say) $\geq 2\sqrt{\log N}/d$ over the dual lattice.
- Known algorithms can exploit narrower samples, but not these (?).

Questions

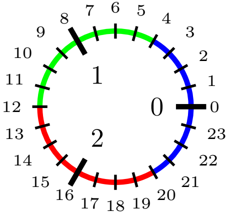
1. Is BDD w/DGS actually hard? What effect does N have?
2. Tightness of the BDD w/DGS $\leq$ LWE reduction in N, σ.

Learning With **Rounding** (LWR) [BanerjeePeikertRosen'12]



Learning With **Rounding** (LWR) [BanerjeePeikertRosen'12]

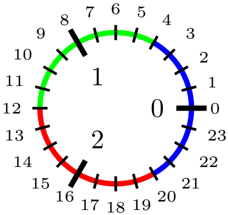
- KEY IDEA: generate error deterministically, by rounding $\mathbb{Z}_q$ to a 'sparser' subset.



Learning With **Rounding** (LWR) [BanerjeePeikertRosen'12]

- KEY IDEA: generate error deterministically, by rounding $\mathbb{Z}_q$ to a 'sparser' subset.

  Let $p < q$ and define $\lfloor x \rceil_p := \lfloor x \cdot p/q \rceil \bmod p$.

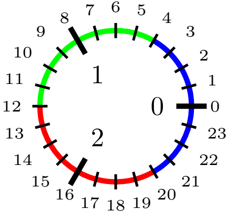


Learning With **Rounding** (LWR) [BanerjeePeikertRosen'12]

- KEY IDEA: generate error deterministically, by rounding $\mathbb{Z}_q$ to a 'sparser' subset.

  Let $p < q$ and define $\lfloor x \rceil_p := \lfloor x \cdot p/q \rceil \bmod p$.

  (LWE decryption uses this to remove error!)

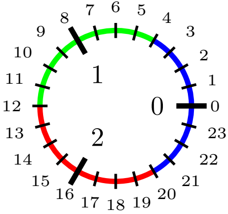


Learning With **Rounding** (LWR) [BanerjeePeikertRosen'12]

- KEY IDEA: generate error deterministically, by rounding $\mathbb{Z}_q$ to a 'sparser' subset.

  Let $p < q$ and define $\lfloor x \rceil_p := \lfloor x \cdot p/q \rceil \bmod p$.

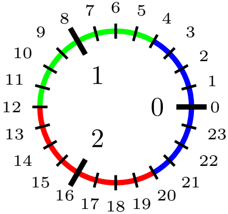
  (LWE decryption uses this to remove error!)



- LWR problem: find $\mathbf{s}$ (or distinguish from random), given pairs

$$\left(\mathbf{a}_i \ , \ \lfloor \langle \mathbf{s}, \mathbf{a}_i \rangle \rceil_p \right) \in \mathbb{Z}_q \times \mathbb{Z}_p \ .$$

Learning With **Rounding** (LWR) [BanerjeePeikertRosen'12]



- KEY IDEA: generate error deterministically, by rounding $\mathbb{Z}_q$ to a 'sparser' subset.

  Let $p < q$ and define $\lfloor x \rceil_p := \lfloor x \cdot p/q \rceil \bmod p$.

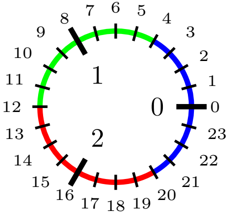
  (LWE decryption uses this to remove error!)

- LWR problem: find $\mathbf{s}$ (or distinguish from random), given pairs

$$\left(\mathbf{a}_i \ , \ \lfloor \langle \mathbf{s}, \mathbf{a}_i \rangle \rceil_p \right) \in \mathbb{Z}_q \times \mathbb{Z}_p \ .$$

LWE conceals low bits with random error; LWR just discards them.

Learning With **Rounding** (LWR) [BanerjeePeikertRosen'12]



- KEY IDEA: generate error deterministically, by rounding $\mathbb{Z}_q$ to a 'sparser' subset.

  Let $p < q$ and define $\lfloor x \rceil_p := \lfloor x \cdot p/q \rceil \bmod p$.

  (LWE decryption uses this to remove error!)

- LWR problem: find $\mathbf{s}$ (or distinguish from random), given pairs

$$\left(\mathbf{a}_i \ , \ \lfloor \langle \mathbf{s}, \mathbf{a}_i \rangle \rceil_p \right) \in \mathbb{Z}_q \times \mathbb{Z}_p \ .$$

LWE conceals low bits with random error; LWR just discards them.

Theorem [BPR'12,. . .]

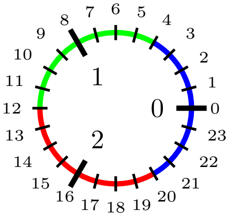
- For $q \geq p \cdot E \cdot 2^{\lambda}$, LWR is no easier than LWE with error size E, for security parameter $\approx \lambda$.

Learning With **Rounding** (LWR) [BanerjeePeikertRosen'12]

- KEY IDEA: generate error deterministically, by rounding $\mathbb{Z}_q$ to a 'sparser' subset.

  Let $p < q$ and define $\lfloor x \rceil_p := \lfloor x \cdot p/q \rceil \bmod p$.

  (LWE decryption uses this to remove error!)



- LWR problem: find $\mathbf{s}$ (or distinguish from random), given pairs

$$\left(\mathbf{a}_i \ , \ \lfloor \langle \mathbf{s}, \mathbf{a}_i \rangle \rceil_p\right) \in \mathbb{Z}_q \times \mathbb{Z}_p \ .$$

  LWE conceals low bits with random error; LWR just discards them.

Theorem [BPR'12,...]

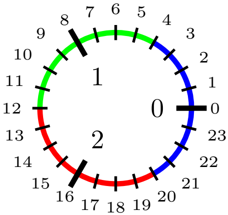
- For $q \geq p \cdot E \cdot 2^\lambda$, LWR is no easier than LWE with error size E, for security parameter $\approx \lambda$. (Error width $q/p > 2^\lambda$, rate $\alpha = E/q < 2^{-\lambda}$.)

Learning With **Rounding** (LWR) [BanerjeePeikertRosen'12]

- KEY IDEA: generate error deterministically, by rounding $\mathbb{Z}_q$ to a 'sparser' subset.

  Let $p < q$ and define $\lfloor x \rceil_p := \lfloor x \cdot p/q \rceil \bmod p$.

  (LWE decryption uses this to remove error!)



- LWR problem: find $\mathbf{s}$ (or distinguish from random), given pairs

$$\left(\mathbf{a}_i \ , \ \lfloor \langle \mathbf{s}, \mathbf{a}_i \rangle \rceil_p \right) \in \mathbb{Z}_q \times \mathbb{Z}_p \ .$$

  LWE conceals low bits with random error; LWR just discards them.

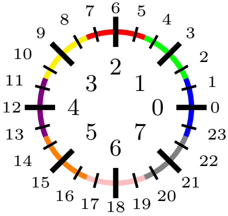
Theorem [BPR'12,...]

- For $q \geq p \cdot E \cdot 2^{\lambda}$, LWR is no easier than LWE with error size E, for security parameter $\approx \lambda$. (Error width $q/p > 2^{\lambda}$, rate $\alpha = E/q < 2^{-\lambda}$.)

Proof idea: w.h.p., $(\, \mathbf{a}_i \, , \, \lfloor \langle \mathbf{s}, \mathbf{a}_i \rangle + e \rceil_p \,) = (\, \mathbf{a}_i \, , \, \lfloor \langle \mathbf{s}, \mathbf{a}_i \rangle \rceil_p \,)$.

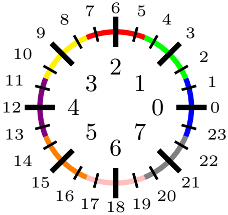
LWR In Practice: SABER [DKRV'17], NTRU Prime [BCLV'17]

- Theorems require large 'rounding width' q/p.



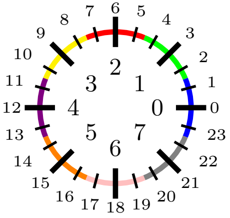
LWR In Practice: SABER [DKRV'17], NTRU Prime [BCLV'17]

- Theorems require large 'rounding width' q/p.
- But systems use small q/p, e.g., $q/p \approx 3$ or 8.



LWR In Practice: SABER [DKRV'17], NTRU Prime [BCLV'17]

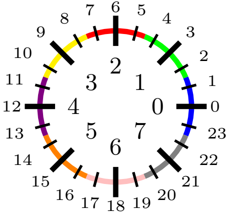
- Theorems require large 'rounding width' q/p.
- But systems use small q/p, e.g., $q/p \approx 3$ or 8.
- Heuristically, seems to resist known attacks.
  But little public scrutiny of such 'small rounding'!



LWR In Practice: SABER [DKRV'17], NTRU Prime [BCLV'17]

- Theorems require large 'rounding width' q/p.
- But systems use small q/p, e.g., $q/p \approx 3$ or 8.
- Heuristically, seems to resist known attacks.

  But little public scrutiny of such 'small rounding'!



Open Questions

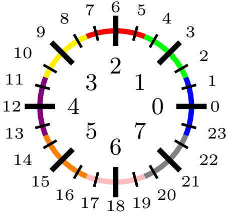
① Any theoretical support for small rounding?

Tighter connection to LWE? 'Native' worst-case hardness?

LWR In Practice: SABER [DKRV'17], NTRU Prime [BCLV'17]

- Theorems require large 'rounding width' q/p.
- But systems use small q/p, e.g., $q/p \approx 3$ or 8.
- Heuristically, seems to resist known attacks.

  But little public scrutiny of such 'small rounding'!



Open Questions

① Any theoretical support for small rounding?

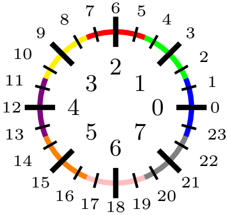
Tighter connection to LWE? 'Native' worst-case hardness?

② (Quantum) attacks that exploit small rounding?

LWR In Practice: SABER [DKRV'17], NTRU Prime [BCLV'17]

- Theorems require large 'rounding width' q/p.
- But systems use small q/p, e.g., $q/p \approx 3$ or 8.
- Heuristically, seems to resist known attacks.

  But little public scrutiny of such 'small rounding'!



Open Questions

① Any theoretical support for small rounding?

Tighter connection to LWE? 'Native' worst-case hardness?

② (Quantum) attacks that exploit small rounding?

Regev'02 uses rounding to quantumly reduce BDD to a 'noisy' cyclic hidden-shift problem, which has a $\exp(\sqrt{\log |G|})$ quantum algorithm.

Could those techniques be useful here?

Lattices

Efficiency from Algebraic Structure

SIS/LWE/LWR are Efficient(-ish)

$$(\cdots\ \mathbf{a}_i\ \cdots)\begin{pmatrix}\vdots\\ \mathbf{s}\\ \vdots\end{pmatrix}+e_i\ =\ b_i\in\mathbb{Z}_q$$

- Getting one random-looking scalar $b_i \in \mathbb{Z}_q$ requires an n-dim mod-q inner product

SIS/LWE/LWR are Efficient(-ish)

$$\begin{pmatrix} \cdots & \mathbf{a}_i & \cdots \end{pmatrix} \begin{pmatrix} \vdots \\ \mathbf{s} \\ \vdots \end{pmatrix} + e_i \;=\; b_i \in \mathbb{Z}_q$$

- Getting one random-looking scalar $b_i \in \mathbb{Z}_q$ requires an n-dim mod-q inner product
- Can amortize each $\mathbf{a}_i$ over many secrets $\mathbf{s}_j$, but still $\tilde{O}(n)$ work per scalar output.

SIS/LWE/LWR are Efficient(-ish)

$$(\cdots \mathbf{a}_i \cdots) \begin{pmatrix} \vdots \\ \mathbf{s} \\ \vdots \end{pmatrix} + e_i \; = \; b_i \in \mathbb{Z}_q$$

- Getting one random-looking scalar $b_i \in \mathbb{Z}_q$ requires an n-dim mod-q inner product
- Can amortize each $\mathbf{a}_i$ over many secrets $\mathbf{s}_j$, but still $\tilde{O}(n)$ work per scalar output.
- Cryptosystems have rather large keys:

$$pk = \underbrace{\begin{pmatrix} & \vdots & \\ & \mathbf{A} & \\ & \vdots & \end{pmatrix}}_{n} \quad , \quad \left.\begin{pmatrix} \vdots \\ \mathbf{b} \\ \vdots \end{pmatrix}\right\} \Omega(n)$$

SIS/LWE/LWR are Efficient(-ish)

$$(\cdots \mathbf{a}_i \cdots) \begin{pmatrix} \vdots \\ \mathbf{s} \\ \vdots \end{pmatrix} + e_i \; = \; b_i \in \mathbb{Z}_q$$

- Getting one random-looking scalar $b_i \in \mathbb{Z}_q$ requires an n-dim mod-q inner product
- Can amortize each $\mathbf{a}_i$ over many secrets $\mathbf{s}_j$, but still $\tilde{O}(n)$ work per scalar output.
- Cryptosystems have rather large keys:

$$pk = \left. \underbrace{\begin{pmatrix} & \vdots & \\ & \mathbf{A} & \\ & \vdots & \end{pmatrix}}_{n} \quad , \quad \begin{pmatrix} \vdots \\ \mathbf{b} \\ \vdots \end{pmatrix} \right\} \Omega(n)$$

- Inherently $\geq n^2$ time to encrypt & decrypt an n-bit message.

Wishful Thinking…

$$\begin{pmatrix} \vdots \\ \mathbf{a} \\ \vdots \end{pmatrix} \star \begin{pmatrix} \vdots \\ \mathbf{s} \\ \vdots \end{pmatrix} + \begin{pmatrix} \vdots \\ \mathbf{e} \\ \vdots \end{pmatrix} = \begin{pmatrix} \vdots \\ \mathbf{b} \\ \vdots \end{pmatrix} \in \mathbb{Z}_q^d$$

- Get d pseudorandom scalars from just one (cheap) product operation?
- Replace $\mathbb{Z}_q^{d \times d}$-chunks by $\mathbb{Z}_q^d$.

Wishful Thinking…

$$\begin{pmatrix} \vdots \\ \mathbf{a} \\ \vdots \end{pmatrix} \star \begin{pmatrix} \vdots \\ \mathbf{s} \\ \vdots \end{pmatrix} + \begin{pmatrix} \vdots \\ \mathbf{e} \\ \vdots \end{pmatrix} = \begin{pmatrix} \vdots \\ \mathbf{b} \\ \vdots \end{pmatrix} \in \mathbb{Z}_q^d$$

- Get d pseudorandom scalars from just one (cheap) product operation?
- Replace $\mathbb{Z}_q^{d \times d}$-chunks by $\mathbb{Z}_q^d$.

Question

- How to define the product '$\star$' so that $(\mathbf{a}, \mathbf{b})$ is pseudorandom?

Wishful Thinking…

$$\begin{pmatrix} \vdots \\ \mathbf{a} \\ \vdots \end{pmatrix} \star \begin{pmatrix} \vdots \\ \mathbf{s} \\ \vdots \end{pmatrix} + \begin{pmatrix} \vdots \\ \mathbf{e} \\ \vdots \end{pmatrix} = \begin{pmatrix} \vdots \\ \mathbf{b} \\ \vdots \end{pmatrix} \in \mathbb{Z}_q^d$$

- Get d pseudorandom scalars from just one (cheap) product operation?
- Replace $\mathbb{Z}_q^{d \times d}$-chunks by $\mathbb{Z}_q^d$.

Question

- How to define the product '$\star$' so that $(\mathbf{a}, \mathbf{b})$ is pseudorandom?
- Careful! With small error, coordinate-wise multiplication is insecure!

Wishful Thinking. . .

$$\begin{pmatrix}\vdots\\ \mathbf{a}\\ \vdots\end{pmatrix} \star \begin{pmatrix}\vdots\\ \mathbf{s}\\ \vdots\end{pmatrix} + \begin{pmatrix}\vdots\\ \mathbf{e}\\ \vdots\end{pmatrix} = \begin{pmatrix}\vdots\\ \mathbf{b}\\ \vdots\end{pmatrix} \in \mathbb{Z}_q^d$$

- Get d pseudorandom scalars from just one (cheap) product operation?
- Replace $\mathbb{Z}_q^{d \times d}$-chunks by $\mathbb{Z}_q^d$.

Question

- How to define the product '$\star$' so that $(\mathbf{a}, \mathbf{b})$ is pseudorandom?
- Careful! With small error, coordinate-wise multiplication is insecure!

Answer

- '$\star$' $=$ multiplication in a polynomial ring: e.g., $\mathbb{Z}_q[X]/(X^d + 1)$.

  Fast and practical with FFT: $d \log d$ operations mod q.

Wishful Thinking...

$$\begin{pmatrix} \vdots \\ \mathbf{a} \\ \vdots \end{pmatrix} \star \begin{pmatrix} \vdots \\ \mathbf{s} \\ \vdots \end{pmatrix} + \begin{pmatrix} \vdots \\ \mathbf{e} \\ \vdots \end{pmatrix} = \begin{pmatrix} \vdots \\ \mathbf{b} \\ \vdots \end{pmatrix} \in \mathbb{Z}_q^d$$

- Get d pseudorandom scalars from just one (cheap) product operation?
- Replace $\mathbb{Z}_q^{d \times d}$-chunks by $\mathbb{Z}_q^d$.

Question

- How to define the product '$\star$' so that $(\mathbf{a}, \mathbf{b})$ is pseudorandom?
- Careful! With small error, coordinate-wise multiplication is insecure!

Answer

- '$\star$' $=$ multiplication in a polynomial ring: e.g., $\mathbb{Z}_q[X]/(X^d + 1)$.

  Fast and practical with FFT: $d \log d$ operations mod q.
- Same ring structures used in NTRU cryptosystem [HPS'98], compact one-way / CR hash functions [Mic'02,PR'06,LM'06,...]

Wishful Thinking. . .

$$\begin{pmatrix} \vdots \\ \mathbf{a} \\ \vdots \end{pmatrix} \star \begin{pmatrix} \vdots \\ \mathbf{s} \\ \vdots \end{pmatrix} + \begin{pmatrix} \vdots \\ \mathbf{e} \\ \vdots \end{pmatrix} = \begin{pmatrix} \vdots \\ \mathbf{b} \\ \vdots \end{pmatrix} \in \mathbb{Z}_q^d$$

- Get d pseudorandom scalars from just one (cheap) product operation?
- Replace $\mathbb{Z}_q^{d \times d}$-chunks by $\mathbb{Z}_q^d$.

LWE Over Rings/Modules, Over Simplified [LPR'10,BGV'11,LS'12]

- Let $\boxed{R = \mathbb{Z}[X]/(X^d + 1)}$ for d a power of two, and $\boxed{R_q = R/qR}$

LWE Over Rings/Modules, Over Simplified [LPR'10,BGV'11,LS'12]

- Let $\boxed{R = \mathbb{Z}[X]/(X^d + 1)}$ for d a power of two, and $\boxed{R_q = R/qR}$
  - ⋆ Elements of R_q are degree $< d$ polynomials with mod-q coefficients
  - ⋆ Operations in R_q are very efficient using FFT-like algorithms

LWE Over Rings/Modules, Over Simplified [LPR'10,BGV'11,LS'12]

- Let $\boxed{R = \mathbb{Z}[X]/(X^d + 1)}$ for d a power of two, and $\boxed{R_q = R/qR}$
  - ★ Elements of R_q are degree $< d$ polynomials with mod-q coefficients
  - ★ Operations in R_q are very efficient using FFT-like algorithms
- **Search**: find secret vector of polynomials $\mathbf{s} \in R_q^k$, given:

$$\begin{array}{ccc} \mathbf{a}_1 \leftarrow R_q^k & , & b_1 \approx \langle \mathbf{s}, \mathbf{a}_1 \rangle \in R_q \\ \mathbf{a}_2 \leftarrow R_q^k & , & b_2 \approx \langle \mathbf{s}, \mathbf{a}_2 \rangle \in R_q \\ & \vdots & \end{array}$$

LWE Over Rings/Modules, Over Simplified [LPR'10,BGV'11,LS'12]

- Let $\boxed{R = \mathbb{Z}[X]/(X^d + 1)}$ for d a power of two, and $\boxed{R_q = R/qR}$
  - ⋆ Elements of R_q are degree $< d$ polynomials with mod-q coefficients
  - ⋆ Operations in R_q are very efficient using FFT-like algorithms
- **Search**: find secret vector of polynomials $\mathbf{s} \in R_q^k$, given:

$$\mathbf{a}_1 \leftarrow R_q^k \quad , \quad b_1 \approx \langle \mathbf{s}, \mathbf{a}_1 \rangle \in R_q$$

$$\mathbf{a}_2 \leftarrow R_q^k \quad , \quad b_2 \approx \langle \mathbf{s}, \mathbf{a}_2 \rangle \in R_q$$

$$\vdots$$

- ⋆ Each eq. is d related eq.'s on a secret of dim $n = kd$ over $\mathbb{Z}_q$.

LWE Over Rings/Modules, Over Simplified [LPR'10,BGV'11,LS'12]

- Let $\boxed{R = \mathbb{Z}[X]/(X^d + 1)}$ for d a power of two, and $\boxed{R_q = R/qR}$
  - ⋆ Elements of R_q are degree $< d$ polynomials with mod-q coefficients
  - ⋆ Operations in R_q are very efficient using FFT-like algorithms
- **Search**: find secret vector of polynomials $\mathbf{s} \in R_q^k$, given:

$$\begin{array}{lcl} \mathbf{a}_1 \leftarrow R_q^k & , & b_1 \approx \langle \mathbf{s}, \mathbf{a}_1 \rangle \in R_q \\ \mathbf{a}_2 \leftarrow R_q^k & , & b_2 \approx \langle \mathbf{s}, \mathbf{a}_2 \rangle \in R_q \\ & \vdots & \end{array}$$

⋆ Each eq. is d related eq.'s on a secret of dim $n = kd$ over $\mathbb{Z}_q$.

⋆ LWE: $d = 1, k = n$.
Ring-LWE: $d = n, k = 1$.
Module-LWE: interpolate.

LWE Over Rings/Modules, Over Simplified [LPR'10,BGV'11,LS'12]

- Let $\boxed{R = \mathbb{Z}[X]/(X^d + 1)}$ for d a power of two, and $\boxed{R_q = R/qR}$
  - ⋆ Elements of R_q are degree $< d$ polynomials with mod-q coefficients
  - ⋆ Operations in R_q are very efficient using FFT-like algorithms
- **Search**: find secret vector of polynomials $\mathbf{s} \in R_q^k$, given:

$$\begin{array}{ll} \mathbf{a}_1 \leftarrow R_q^k \quad , & b_1 \approx \langle \mathbf{s}, \mathbf{a}_1 \rangle \in R_q \\ \mathbf{a}_2 \leftarrow R_q^k \quad , & b_2 \approx \langle \mathbf{s}, \mathbf{a}_2 \rangle \in R_q \\ \vdots & \end{array}$$

  - ⋆ Each eq. is d related eq.'s on a secret of dim $n = kd$ over $\mathbb{Z}_q$.
  - ⋆ LWE: $d = 1, k = n$.
    Ring-LWE: $d = n, k = 1$.
    Module-LWE: interpolate.
- **Decision**: distinguish $(\mathbf{a}_i \ , \ b_i)$ from uniform $(\mathbf{a}_i \ , \ b_i) \in R_q^k \times R_q$

Hardness of Ring/Module-LWE

Theorems [. . . ,SSTX'09,LPR'10,LS'12,PRS'17,RSW'18,. . .]

worst-case approx-SVP on rank-k *module* lattices over R $\leq$ search R^k-LWE $\leq$ decision R^k-LWE

(quantum, any $R = \mathcal{O}_K$)

(classical, any $R = \mathcal{O}_K$)

Hardness of Ring/Module-LWE

Theorems […,SSTX'09,LPR'10,LS'12,PRS'17,RSW'18,…]

worst-case approx-SVP on rank-k *module* lattices over R $\underset{\substack{\text{(quantum,}\\ \text{any } R = \mathcal{O}_K)}}{\leq}$ search R^k-LWE $\underset{\substack{\text{(classical,}\\ \text{any } R = \mathcal{O}_K)}}{\leq}$ decision R^k-LWE

Open Questions

1. Can we 'de-quantize' the worst-case/average-case reduction?

Hardness of Ring/Module-LWE

Theorems [. . . ,SSTX'09,LPR'10,LS'12,PRS'17,RSW'18,. . .]

worst-case approx-SVP on rank-k *module* lattices over R $\leq$ search R^k-LWE $\leq$ decision R^k-LWE

(quantum, any $R = \mathcal{O}_K$)

(classical, any $R = \mathcal{O}_K$)

Open Questions

1. Can we 'de-quantize' the worst-case/average-case reduction?

   The classical GapSVP $\leq$ LWE reduction is of limited use: for the relevant factors, GapSVP for ideals ($k = 1$) is easy.

Hardness of Ring/Module-LWE

Theorems [. . . ,SSTX'09,LPR'10,LS'12,PRS'17,RSW'18,. . .]

worst-case approx-SVP on rank-k *module* lattices over R $\leq$ search R^k-LWE $\leq$ decision R^k-LWE

(quantum, any $R = \mathcal{O}_K$)

(classical, any $R = \mathcal{O}_K$)

Open Questions

1. Can we 'de-quantize' the worst-case/average-case reduction?

   The classical GapSVP $\leq$ LWE reduction is of limited use: for the relevant factors, GapSVP for ideals ($k = 1$) is easy.

2. How hard (or not) is approx-SVP on ideal/module lattices?

Hardness of Ring/Module-LWE

Theorems [. . . ,SSTX'09,LPR'10,LS'12,PRS'17,RSW'18,. . .]

worst-case approx-SVP on rank-k *module* lattices over R $\underset{\text{(quantum, any } R = \mathcal{O}_K\text{)}}{\leq}$ search R^k-LWE $\underset{\text{(classical, any } R = \mathcal{O}_K\text{)}}{\leq}$ decision R^k-LWE

Open Questions

1. Can we 'de-quantize' the worst-case/average-case reduction?

   The classical GapSVP $\leq$ LWE reduction is of limited use: for the relevant factors, GapSVP for ideals ($k = 1$) is easy.

2. How hard (or not) is approx-SVP on ideal/module lattices?

   For poly-approx, no significant improvements vs. general lattices, even for ideals
   For subexp-approx, we have better quantum algs for ideals, but not for $k > 1$:
   [CGS'15,CDPR'16,CDW'17,PHS'19,. . .]

Hardness of Ring/Module-LWE

Theorems [. . . ,SSTX'09,LPR'10,LS'12,PRS'17,RSW'18,. . .]

worst-case approx-SVP on rank-k *module* lattices over R $\underset{\text{(quantum, any } R = \mathcal{O}_K\text{)}}{\leq}$ search R^k-LWE $\underset{\text{(classical, any } R = \mathcal{O}_K\text{)}}{\leq}$ decision R^k-LWE

Open Questions

1. Can we 'de-quantize' the worst-case/average-case reduction?

   The classical GapSVP $\leq$ LWE reduction is of limited use: for the relevant factors, GapSVP for ideals $(k = 1)$ is easy.

2. How hard (or not) is approx-SVP on ideal/module lattices?

   For poly-approx, no significant improvements vs. general lattices, even for ideals
   For subexp-approx, we have better quantum algs for ideals, but not for $k > 1$:
   [CGS'15,CDPR'16,CDW'17,PHS'19,. . .]

3. Are there reverse reductions? (Seems not, without increasing k. . .)

Ring/Module-LWE In Practice: Kyber, SABER, NTRU(')

- NTRU(') use fixed rank $k = 1$ over rings of increasing degree d.
  Kyber, SABER use increasing rank k over a ring of fixed degree d.

Ring/Module-LWE In Practice: Kyber, SABER, NTRU(')

- NTRU(') use fixed rank $k = 1$ over rings of increasing degree d.
  Kyber, SABER use increasing rank k over a ring of fixed degree d.

  Cryptanalysis suggests that $n = kd$ mainly controls hardness, even though increasing k yields 'less structure'. Any distinction?

Ring/Module-LWE In Practice: Kyber, SABER, NTRU(')

- NTRU(') use fixed rank $k = 1$ over rings of increasing degree d.

  Kyber, SABER use increasing rank k over a ring of fixed degree d.

  Cryptanalysis suggests that $n = kd$ mainly controls hardness, even though increasing k yields 'less structure'. Any distinction?

- Theorems require moderate error sizes $\gg \sqrt{n}$ in each coefficient.

  Systems use small error sizes $\in [1, 7]$.

  Seems hard according to cryptanalysis. Theory? (Quantum) attacks?

Lattices: Closing Thoughts

1. Lattices are a source of many seemingly quantum-hard problems, and offer an amazing platform for cryptography.

Lattices: Closing Thoughts

1. Lattices are a source of many seemingly quantum-hard problems, and offer an amazing platform for cryptography.

2. There are (moderate to huge) gaps between theorems and practical parameters. Narrow them, exploit them—or both!

Lattices: Closing Thoughts

1. Lattices are a source of many seemingly quantum-hard problems, and offer an amazing platform for cryptography.

2. There are (moderate to huge) gaps between theorems and practical parameters. Narrow them, exploit them—or both!

3. Many important questions need attention from quantum experts. The future of our digital security may depend on it!

Bonus: Isogenies

Elliptic Curves and Isogenies

- An elliptic curve E over a field $\mathbb{F}$ is the set of solutions $(x, y) \in \mathbb{F}^2$ to

  $$y^2 = x^3 + ax + b$$

  for suitable fixed $a, b \in \mathbb{F}$, plus a 'point at infinity' $\mathcal{O}$.

Elliptic Curves and Isogenies

- An elliptic curve E over a field $\mathbb{F}$ is the set of solutions $(x, y) \in \mathbb{F}^2$ to

  $$y^2 = x^3 + ax + b$$

  for suitable fixed $a, b \in \mathbb{F}$, plus a 'point at infinity' $\mathcal{O}$.
- With suitable 'point addition,' E is a group with identity $\mathcal{O}$.

Elliptic Curves and Isogenies

- An elliptic curve E over a field $\mathbb{F}$ is the set of solutions $(x, y) \in \mathbb{F}^2$ to

  $$y^2 = x^3 + ax + b$$

  for suitable fixed $a, b \in \mathbb{F}$, plus a 'point at infinity' $\mathcal{O}$.
- With suitable 'point addition,' E is a group with identity $\mathcal{O}$.
- Since 1980s, cryptography has used dlog problem on ECs over finite $\mathbb{F}$.

Elliptic Curves and Isogenies

- An elliptic curve E over a field $\mathbb{F}$ is the set of solutions $(x, y) \in \mathbb{F}^2$ to

  $$y^2 = x^3 + ax + b$$

  for suitable fixed $a, b \in \mathbb{F}$, plus a 'point at infinity' $\mathcal{O}$.
- With suitable 'point addition,' E is a group with identity $\mathcal{O}$.
- Since 1980s, cryptography has used dlog problem on ECs over finite $\mathbb{F}$. But this is quantumly broken by Shor's algorithm.

Elliptic Curves and Isogenies

- An elliptic curve E over a field $\mathbb{F}$ is the set of solutions $(x, y) \in \mathbb{F}^2$ to

  $$y^2 = x^3 + ax + b$$

  for suitable fixed $a, b \in \mathbb{F}$, plus a ‘point at infinity’ $\mathcal{O}$.
- With suitable ‘point addition,’ E is a group with identity $\mathcal{O}$.
- Since 1980s, cryptography has used dlog problem on ECs over finite $\mathbb{F}$.

  But this is quantumly broken by Shor’s algorithm.

  So are ECs hopeless for crypto? Maybe not!

Elliptic Curves and Isogenies

- An elliptic curve E over a field $\mathbb{F}$ is the set of solutions $(x, y) \in \mathbb{F}^2$ to

  $$y^2 = x^3 + ax + b$$

  for suitable fixed $a, b \in \mathbb{F}$, plus a 'point at infinity' $\mathcal{O}$.
- With suitable 'point addition,' E is a group with identity $\mathcal{O}$.
- Since 1980s, cryptography has used dlog problem on ECs over finite $\mathbb{F}$.

  But this is quantumly broken by Shor's algorithm.

  So are ECs hopeless for crypto? Maybe not!
- An isogeny is a map from one elliptic curve $E/\mathbb{F}$ to another $E'/\mathbb{F}$ satisfying certain algebraic conditions. (Not necessarily an isomorphism.)

Elliptic Curves and Isogenies

- An elliptic curve E over a field $\mathbb{F}$ is the set of solutions $(x, y) \in \mathbb{F}^2$ to

  $$y^2 = x^3 + ax + b$$

  for suitable fixed $a, b \in \mathbb{F}$, plus a 'point at infinity' $\mathcal{O}$.
- With suitable 'point addition,' E is a group with identity $\mathcal{O}$.
- Since 1980s, cryptography has used dlog problem on ECs over finite $\mathbb{F}$.

  But this is quantumly broken by Shor's algorithm.

  So are ECs hopeless for crypto? Maybe not!
- An isogeny is a map from one elliptic curve $E/\mathbb{F}$ to another $E'/\mathbb{F}$ satisfying certain algebraic conditions. (Not necessarily an isomorphism.)
- There are proposals to use conjectured-hard problems related to finding isogenies between isogenous curves.

Cryptography from Isogenies

Two main kinds of constructions using isogenies:

1. Use isogenies for commutative group action to get DH-style key agreement [Couveignes'97,RostovtsevStolbunov'04].

Cryptography from Isogenies

Two main kinds of constructions using isogenies:

1. Use isogenies for commutative group action to get DH-style key agreement [Couveignes'97,RostovtsevStolbunov'04].

   Real instantiation: CSIDH [CLMPR'18]. Very small (?), not so fast.

Cryptography from Isogenies

Two main kinds of constructions using isogenies:

1. Use isogenies for commutative group action to get DH-style key agreement [Couveignes'97,RostovtsevStolbunov'04].

   Real instantiation: CSIDH [CLMPR'18]. Very small (?), not so fast.

   Much less quantum security than initially conjectured [BS'18/'20,P'20].

Cryptography from Isogenies

Two main kinds of constructions using isogenies:

1. Use isogenies for commutative group action to get DH-style key agreement [Couveignes'97,RostovtsevStolbunov'04].

   Real instantiation: CSIDH [CLMPR'18]. Very small (?), not so fast.

   Much less quantum security than initially conjectured [BS'18/'20,P'20].

2. Use isogeny graph on 'supersingular' curves: SIDH [DeFeoJaoPlut'11].

Cryptography from Isogenies

Two main kinds of constructions using isogenies:

1. Use isogenies for commutative group action to get DH-style key agreement [Couveignes'97,RostovtsevStolbunov'04].

   Real instantiation: CSIDH [CLMPR'18]. Very small (?), not so fast.

   Much less quantum security than initially conjectured [BS'18/'20,P'20].

2. Use isogeny graph on 'supersingular' curves: SIDH [DeFeoJaoPlut'11].

   Real instantiation: NIST alternate SIKE [JAC+'17]. Small, not so fast.

Cryptography from Isogenies

Two main kinds of constructions using isogenies:

1. Use isogenies for commutative group action to get DH-style key agreement [Couveignes'97,RostovtsevStolbunov'04].

   Real instantiation: CSIDH [CLMPR'18]. Very small (?), not so fast.

   Much less quantum security than initially conjectured [BS'18/'20,P'20].

2. Use isogeny graph on 'supersingular' curves: SIDH [DeFeoJaoPlut'11].

   Real instantiation: NIST alternate SIKE [JAC+'17]. Small, not so fast.

   No (quantum) cryptanalytic improvements since original proposal. Opportunity?

CSIDH (‘sea-side’) [CastryckLangeMartindalePannyRenes’18]

- Isogeny-based ‘post-quantum commutative group action’ following [Couveignes’97,RostovtsevStolbunov’04]: abelian group G, set Z, action

$$\star\colon G \times Z \to Z \ .$$

CSIDH ('sea-side') [CastryckLangeMartindalePannyRenes'18]

- Isogeny-based 'post-quantum commutative group action' following [Couveignes'97,RostovtsevStolbunov'04]: abelian group G, set Z, action

$$\star\colon G \times Z \to Z \ .$$

DiffieHellman-style noninteractive key exchange with public param $z \in Z$:

Alice: secret $a \in G$, public $p_A = a \star z \in Z$

Bob: secret $b \in G$, public $p_B = b \star z \in Z$

Shared key: $a \star p_B = b \star p_A = (a + b) \star z$, by commutativity

CSIDH ('sea-side') [CastryckLangeMartindalePannyRenes'18]

- Isogeny-based 'post-quantum commutative group action' following [Couveignes'97,RostovtsevStolbunov'04]: abelian group G, set Z, action

$$\star\colon G \times Z \to Z \ .$$

DiffieHellman-style noninteractive key exchange with public param $z \in Z$:

Alice: secret $a \in G$, public $p_A = a \star z \in Z$

Bob: secret $b \in G$, public $p_B = b \star z \in Z$

Shared key: $a \star p_B = b \star p_A = (a + b) \star z$, by commutativity

- Efficient! 64-byte keys, 80ms key exchange for claimed NIST level 1 quantum security: as hard as AES-128 key search

CSIDH ('sea-side') [CastryckLangeMartindalePannyRenes'18]

- Isogeny-based 'post-quantum commutative group action' following [Couveignes'97,RostovtsevStolbunov'04]: abelian group G, set Z, action

$$\star\colon G \times Z \to Z\ .$$

DiffieHellman-style noninteractive key exchange with public param $z \in Z$:

Alice: secret $a \in G$, public $p_A = a \star z \in Z$

Bob: secret $b \in G$, public $p_B = b \star z \in Z$

Shared key: $a \star p_B = b \star p_A = (a+b) \star z$, by commutativity

- Efficient! 64-byte keys, 80ms key exchange for claimed NIST level 1 quantum security: as hard as AES-128 key search
- Signatures [Stolbunov'12,DeFeoGalbraith'19,BeullensKleinjungVercauteren'19]: pk + sig = 1468 bytes at same claimed security level

Attacking the CSIDH, Quantumly

- Secret-key recovery: given $z, a \star z \in Z$, find $a \in G$ (or equivalent).

Attacking the CSIDH, Quantumly

- Secret-key recovery: given $z, a \star z \in Z$, find $a \in G$ (or equivalent). Reduces to Hidden-Shift Problem (HShP) on G [ChildsJaoSoukharev'10].

Attacking the CSIDH, Quantumly

- Secret-key recovery: given $z, a \star z \in Z$, find $a \in G$ (or equivalent). Reduces to Hidden-Shift Problem (HShP) on G [ChildsJaoSoukharev'10].

Quantum HShP Algorithm Ingredients [Kuperberg'03,. . .]

1. Oracle outputs random 'labeled' quantum states, by evaluating $\star$ on a uniform superposition over G.

Attacking the CSIDH, Quantumly

- Secret-key recovery: given $z, a \star z \in Z$, find $a \in G$ (or equivalent). Reduces to Hidden-Shift Problem (HShP) on G [ChildsJaoSoukharev'10].

Quantum HShP Algorithm Ingredients [Kuperberg'03,...]

1. Oracle outputs random 'labeled' quantum states, by evaluating $\star$ on a uniform superposition over G.
2. Sieve combines labeled states to generate 'more favorable' ones.

Attacking the CSIDH, Quantumly

- Secret-key recovery: given $z, a \star z \in Z$, find $a \in G$ (or equivalent).
  Reduces to Hidden-Shift Problem (HShP) on G [ChildsJaoSoukharev'10].

Quantum HShP Algorithm Ingredients [Kuperberg'03,...]

1. Oracle outputs random 'labeled' quantum states, by evaluating $\star$ on a uniform superposition over G.
2. Sieve combines labeled states to generate 'more favorable' ones.
3. Measurement of 'very favorable' state recovers bit(s) of hidden shift.

Attacking the CSIDH, Quantumly

- Secret-key recovery: given $z, a \star z \in Z$, find $a \in G$ (or equivalent). Reduces to Hidden-Shift Problem (HShP) on G [ChildsJaoSoukharev'10].

Quantum HShP Algorithm Ingredients [Kuperberg'03,...]

① Oracle outputs random 'labeled' quantum states, by evaluating $\star$ on a uniform superposition over G.

② Sieve combines labeled states to generate 'more favorable' ones.

③ Measurement of 'very favorable' state recovers bit(s) of hidden shift.

Sieve Algorithms

[Kuperberg'03] $2^{O(\sqrt{n})}$ oracle queries and qubits $(n = \log|G|)$

Attacking the CSIDH, Quantumly

- Secret-key recovery: given $z, a \star z \in Z$, find $a \in G$ (or equivalent). Reduces to Hidden-Shift Problem (HShP) on G [ChildsJaoSoukharev'10].

Quantum HShP Algorithm Ingredients [Kuperberg'03,. . .]

1. Oracle outputs random 'labeled' quantum states, by evaluating $\star$ on a uniform superposition over G.
2. Sieve combines labeled states to generate 'more favorable' ones.
3. Measurement of 'very favorable' state recovers bit(s) of hidden shift.

Sieve Algorithms

[Kuperberg'03] $2^{O(\sqrt{n})}$ oracle queries and qubits $(n = \log|G|)$

[Regev'04] $2^{O(\sqrt{n \log n})}$ oracle queries, only poly(n) qubits

Attacking the CSIDH, Quantumly

- Secret-key recovery: given $z, a \star z \in Z$, find $a \in G$ (or equivalent). Reduces to Hidden-Shift Problem (HShP) on G [ChildsJaoSoukharev'10].

Quantum HShP Algorithm Ingredients [Kuperberg'03,. . .]

① Oracle outputs random 'labeled' quantum states, by evaluating $\star$ on a uniform superposition over G.

② Sieve combines labeled states to generate 'more favorable' ones.

③ Measurement of 'very favorable' state recovers bit(s) of hidden shift.

Sieve Algorithms

[Kuperberg'03] $2^{O(\sqrt{n})}$ oracle queries and qubits $(n = \log|G|)$

[Regev'04] $2^{O(\sqrt{n \log n})}$ oracle queries, only poly(n) qubits

[Kuperberg'11] $2^{O(\sqrt{n})}$ oracle queries and bits of quantum-accessible RAM.

Attacking the CSIDH, Quantumly

- Secret-key recovery: given $z, a \star z \in Z$, find $a \in G$ (or equivalent). Reduces to Hidden-Shift Problem (HShP) on G [ChildsJaoSoukharev'10].

Quantum HShP Algorithm Ingredients [Kuperberg'03,. . .]

1. Oracle outputs random 'labeled' quantum states, by evaluating $\star$ on a uniform superposition over G.
2. Sieve combines labeled states to generate 'more favorable' ones.
3. Measurement of 'very favorable' state recovers bit(s) of hidden shift.

Sieve Algorithms

[Kuperberg'03] $2^{O(\sqrt{n})}$ oracle queries and qubits $(n = \log|G|)$

[Regev'04] $2^{O(\sqrt{n \log n})}$ oracle queries, only poly(n) qubits

[Kuperberg'11] $2^{O(\sqrt{n})}$ oracle queries and bits of quantum-accessible RAM. 'Collimation sieve' subsumes prior two, offers more tradeoffs.

Attacking the CSIDH, Quantumly

- Secret-key recovery: given $z, a \star z \in Z$, find $a \in G$ (or equivalent).
  Reduces to Hidden-Shift Problem (HShP) on G [ChildsJaoSoukharev'10].

Quantum HShP Algorithm Ingredients [Kuperberg'03,...]

① Oracle outputs random 'labeled' quantum states, by evaluating $\star$ on a uniform superposition over G.

② Sieve combines labeled states to generate 'more favorable' ones.

③ Measurement of 'very favorable' state recovers bit(s) of hidden shift.

Sieve Algorithms

[Kuperberg'03] $2^{O(\sqrt{n})}$ oracle queries and qubits $(n = \log|G|)$

[Regev'04] $2^{O(\sqrt{n \log n})}$ oracle queries, only poly(n) qubits

[Kuperberg'11] $2^{O(\sqrt{n})}$ oracle queries and bits of quantum-accessible RAM.
'Collimation sieve' subsumes prior two, offers more tradeoffs.
E.g., $\log(\text{queries}) \cdot \log(\text{QRACM}) \gtrsim n$.

Prior Security Estimates for CSIDH-512

- Oracle costs $\leq 2^{43.3}$ T-gates ($+$ much cheaper linear gates) for ‘best case,’ somewhat non-uniform superposition [BLMP’19]

Prior Security Estimates for CSIDH-512

- Oracle costs $\leq 2^{43.3}$ T-gates ($+$ much cheaper linear gates) for 'best case,' somewhat non-uniform superposition [BLMP'19]

  Good reason to expect similar cost for uniform superposition [BKV'19]

Prior Security Estimates for CSIDH-512

- Oracle costs $\leq 2^{43.3}$ T-gates (+ much cheaper linear gates) for 'best case,' somewhat non-uniform superposition [BLMP'19]

  Good reason to expect similar cost for uniform superposition [BKV'19]

- Sieve costs:

Work	Algorithm	Oracle queries	Sieve memory
CSIDH paper [CLMPR'18]	[Regev'04]	2^{62}	poly(n)

Prior Security Estimates for CSIDH-512

- Oracle costs $\leq 2^{43.3}$ T-gates ($+$ much cheaper linear gates) for 'best case,' somewhat non-uniform superposition [BLMP'19]

  Good reason to expect similar cost for uniform superposition [BKV'19]

- Sieve costs:

Work	Algorithm	Oracle queries	Sieve memory
CSIDH paper [CLMPR'18]	[Regev'04]	2^{62}	poly(n)
[BonnetainSchrottenloher'18]	[Kuperberg'03]	$2^{32.5}$	2^{31} qubits

Prior Security Estimates for CSIDH-512

- Oracle costs $\leq 2^{43.3}$ T-gates ($+$ much cheaper linear gates) for 'best case,' somewhat non-uniform superposition [BLMP'19]

  Good reason to expect similar cost for uniform superposition [BKV'19]

- Sieve costs:

Work	Algorithm	Oracle queries	Sieve memory
CSIDH paper [CLMPR'18]	[Regev'04]	2^{62}	poly(n)
[BonnetainSchrottenloher'18]	[Kuperberg'03]	$2^{32.5}$	2^{31} qubits
None prior!	[Kuperberg'11]	??	??

C-Sieving on the CSIDH [P'20]

- Improve Kuperberg's c-sieve in 'practice,' and analyze its concrete complexity on proposed CSIDH parameters.

C-Sieving on the CSIDH [P'20]

- Improve Kuperberg's c-sieve in 'practice,' and analyze its concrete complexity on proposed CSIDH parameters.
- Run simulations up to the actual CSIDH-512 order $|G| \approx 2^{257.1}$.

C-Sieving on the CSIDH [P'20]

- Improve Kuperberg's c-sieve in 'practice,' and analyze its concrete complexity on proposed CSIDH parameters.
- Run simulations up to the actual CSIDH-512 order $|G| \approx 2^{257.1}$.

Work	Algorithm	Oracle queries	Sieve memory
[CLMPR'18]	[Regev'04]	2^{62}	poly(n)
[BS'18]	[Kuperberg'03]	$2^{32.5}$	2^{31} qubits
This work	[Kuperberg'11]	$2^{18.7}$	2^{32} bits QRACM
		$2^{15.7}$	2^{40} bits QRACM
		$2^{14.1}$	2^{48} bits QRACM

C-Sieving on the CSIDH [P'20]

- Improve Kuperberg's c-sieve in 'practice,' and analyze its concrete complexity on proposed CSIDH parameters.
- Run simulations up to the actual CSIDH-512 order $|G| \approx 2^{257.1}$.

Work	Algorithm	Oracle queries	Sieve memory
[CLMPR'18]	[Regev'04]	2^{62}	poly(n)
[BS'18]	[Kuperberg'03]	$2^{32.5}$	2^{31} qubits
This work	[Kuperberg'11]	$2^{18.7}$	2^{32} bits QRACM
		$2^{15.7}$	2^{40} bits QRACM
		$2^{14.1}$	2^{48} bits QRACM

- Conclusion: proposed CSIDH parameters have relatively little quantum security beyond the cost of quantum evaluation of $\star$.

C-Sieving on the CSIDH [P'20]

- Improve Kuperberg's c-sieve in 'practice,' and analyze its concrete complexity on proposed CSIDH parameters.
- Run simulations up to the actual CSIDH-512 order $|G| \approx 2^{257.1}$.

Work	Algorithm	Oracle queries	Sieve memory
[CLMPR'18]	[Regev'04]	2^{62}	poly(n)
[BS'18]	[Kuperberg'03]	$2^{32.5}$	2^{31} qubits
This work	[Kuperberg'11]	$2^{18.7}$	2^{32} bits QRACM
		$2^{15.7}$	2^{40} bits QRACM
		$2^{14.1}$	2^{48} bits QRACM

- Conclusion: proposed CSIDH parameters have relatively little quantum security beyond the cost of quantum evaluation of $\star$.

*Independently, [BonnetainSchrottenloher'20] gave a complementary, theoretical c-sieve analysis, arriving at similar conclusions.

Isogenies: Closing Thoughts

1. Isogenies are a relatively new platform for cryptography, yielding small and reasonably performant systems.

Isogenies: Closing Thoughts

1. Isogenies are a relatively new platform for cryptography, yielding small and reasonably performant systems.
2. However, they have received relatively little cryptanalysis so far, with mixed results.

Isogenies: Closing Thoughts

1. Isogenies are a relatively new platform for cryptography, yielding small and reasonably performant systems.

2. However, they have received relatively little cryptanalysis so far, with mixed results.

3. Fundamental questions need attention from quantum experts!

Isogenies: Closing Thoughts

1. Isogenies are a relatively new platform for cryptography, yielding small and reasonably performant systems.

2. However, they have received relatively little cryptanalysis so far, with mixed results.

3. Fundamental questions need attention from quantum experts!

Thanks!