CS 594 – Advanced Crypto (Spring 2026)

Alex Block

Lecture 24

April 20, 2026

Post-quantum Hardness Assumptions

Last Time: Quantum vs. Modern Crypto

Last Time: Quantum vs. Modern Crypto

- Last time, we saw 2 quantum algorithms and how they affect the modern landscape of cryptography.

Last Time: Quantum vs. Modern Crypto

- Last time, we saw 2 quantum algorithms and how they affect the modern landscape of cryptography.

- **Grover's Algorithm** (also known as Grover Search)

Last Time: Quantum vs. Modern Crypto

- Last time, we saw 2 quantum algorithms and how they affect the modern landscape of cryptography.

- **Grover's Algorithm** (also known as Grover Search)
  - Offers a *quadratic* speed-up when performing *brute-force* attacks.

Last Time: Quantum vs. Modern Crypto

- Last time, we saw 2 quantum algorithms and how they affect the modern landscape of cryptography.

- **Grover's Algorithm** (also known as Grover Search)
  - Offers a *quadratic* speed-up when performing *brute-force* attacks.
    - E.g., finding a λ-bit AES secret key with only $O(2^{\lambda/2})$ quantum operations.

Last Time: Quantum vs. Modern Crypto

- Last time, we saw 2 quantum algorithms and how they affect the modern landscape of cryptography.

- **Grover's Algorithm** (also known as Grover Search)
  - Offers a *quadratic* speed-up when performing *brute-force* attacks.
    - E.g., finding a λ-bit AES secret key with only $O(2^{\lambda/2})$ quantum operations.
  - Improves Birthday attacks to find collisions in hash functions.

Last Time: Quantum vs. Modern Crypto

- Last time, we saw 2 quantum algorithms and how they affect the modern landscape of cryptography.

- **Grover's Algorithm** (also known as Grover Search)
  - Offers a *quadratic* speed-up when performing *brute-force* attacks.
    - E.g., finding a λ-bit AES secret key with only $O(2^{\lambda/2})$ quantum operations.
  - Improves Birthday attacks to find collisions in hash functions.
    - Use $O(2^{\lambda/3})$ quantum operations versus $O(2^{\lambda/2})$ classically (for a hash function with λ-bit outputs).

Last Time: Quantum vs. Modern Crypto

- Last time, we saw 2 quantum algorithms and how they affect the modern landscape of cryptography.

- **Grover's Algorithm** (also known as Grover Search)
  - Offers a *quadratic* speed-up when performing *brute-force* attacks.
    - E.g., finding a λ-bit AES secret key with only $O(2^{\lambda/2})$ quantum operations.
  - Improves Birthday attacks to find collisions in hash functions.
    - Use $O(2^{\lambda/3})$ quantum operations versus $O(2^{\lambda/2})$ classically (for a hash function with λ-bit outputs).

- **Shor's Algorithm**

Last Time: Quantum vs. Modern Crypto

- Last time, we saw 2 quantum algorithms and how they affect the modern landscape of cryptography.

- **Grover's Algorithm** (also known as Grover Search)
  - Offers a *quadratic* speed-up when performing *brute-force* attacks.
    - E.g., finding a λ-bit AES secret key with only $O(2^{\lambda/2})$ quantum operations.
  - Improves Birthday attacks to find collisions in hash functions.
    - Use $O(2^{\lambda/3})$ quantum operations versus $O(2^{\lambda/2})$ classically (for a hash function with λ-bit outputs).

- **Shor's Algorithm**
  - Allows us to factor λ-bit composite numbers or find solve the discrete-log problem in λ-bit groups in only poly(λ) quantum operations.

Moving Towards Post-quantum Crypto

Moving Towards Post-quantum Crypto

- Two main thrusts have been in the works for years to move cryptosystems towards post-quantum security.

Moving Towards Post-quantum Crypto

- Two main thrusts have been in the works for years to move cryptosystems towards post-quantum security.

- **Asymmetric Cryptography: New Hardness Assumptions**

Moving Towards Post-quantum Crypto

- Two main thrusts have been in the works for years to move cryptosystems towards post-quantum security.

- **Asymmetric Cryptography: New Hardness Assumptions**
  - Designing, understanding, and analyzing *new* hardness assumptions which we do not believe are broken by quantum computers.

Moving Towards Post-quantum Crypto

- Two main thrusts have been in the works for years to move cryptosystems towards post-quantum security.

- **Asymmetric Cryptography: New Hardness Assumptions**
  - Designing, understanding, and analyzing *new* hardness assumptions which we do not believe are broken by quantum computers.
  - Build new cryptosystems which are secure versus quantum adversaries under these new hardness assumptions.

Moving Towards Post-quantum Crypto

- Two main thrusts have been in the works for years to move cryptosystems towards post-quantum security.

- **Asymmetric Cryptography: New Hardness Assumptions**
  - Designing, understanding, and analyzing *new* hardness assumptions which we do not believe are broken by quantum computers.
  - Build new cryptosystems which are secure versus quantum adversaries under these new hardness assumptions.
  - NIST has been one main driver of this push; 2016 was the initial call for new, post-quantum secure cryptosystems.

Moving Towards Post-quantum Crypto

- Two main thrusts have been in the works for years to move cryptosystems towards post-quantum security.

- **Asymmetric Cryptography: New Hardness Assumptions**
  - Designing, understanding, and analyzing *new* hardness assumptions which we do not believe are broken by quantum computers.
  - Build new cryptosystems which are secure versus quantum adversaries under these new hardness assumptions.
  - NIST has been one main driver of this push; 2016 was the initial call for new, post-quantum secure cryptosystems.
    - https://csrc.nist.gov/csrc/media/projects/post-quantum-cryptography/documents/call-for-proposals-final-dec-2016.pdf

Moving Towards Post-quantum Crypto

- Two main thrusts have been in the works for years to move cryptosystems towards post-quantum security.

- **Asymmetric Cryptography: New Hardness Assumptions**
  - Designing, understanding, and analyzing *new* hardness assumptions which we do not believe are broken by quantum computers.
  - Build new cryptosystems which are secure versus quantum adversaries under these new hardness assumptions.
  - NIST has been one main driver of this push; 2016 was the initial call for new, post-quantum secure cryptosystems.
    - https://csrc.nist.gov/csrc/media/projects/post-quantum-cryptography/documents/call-for-proposals-final-dec-2016.pdf
  - New security proofs for quantum adversaries.

Moving Towards Post-quantum Crypto

- Two main thrusts have been in the works for years to move cryptosystems towards post-quantum security.

- **Asymmetric Cryptography: New Hardness Assumptions**
  - Designing, understanding, and analyzing *new* hardness assumptions which we do not believe are broken by quantum computers.
  - Build new cryptosystems which are secure versus quantum adversaries under these new hardness assumptions.
  - NIST has been one main driver of this push; 2016 was the initial call for new, post-quantum secure cryptosystems.
    - https://csrc.nist.gov/csrc/media/projects/post-quantum-cryptography/documents/call-for-proposals-final-dec-2016.pdf
  - New security proofs for quantum adversaries.
    - Not enough to just replace, e.g., discrete-log with a different PQ secure assumption, to get security!

Moving Towards Post-quantum Crypto

- Two main thrusts have been in the works for years to move cryptosystems towards post-quantum security.

Moving Towards Post-quantum Crypto

- Two main thrusts have been in the works for years to move cryptosystems towards post-quantum security.

- **Symmetric Cryptography: New Security Proofs**

Moving Towards Post-quantum Crypto

- Two main thrusts have been in the works for years to move cryptosystems towards post-quantum security.

- **Symmetric Cryptography: New Security Proofs**
  - Motivated by the following question:

Moving Towards Post-quantum Crypto

- Two main thrusts have been in the works for years to move cryptosystems towards post-quantum security.

- **Symmetric Cryptography: New Security Proofs**
  - Motivated by the following question:

Do classical symmetric key primitives remain secure versus quantum adversaries?

Moving Towards Post-quantum Crypto

- Two main thrusts have been in the works for years to move cryptosystems towards post-quantum security.

- **Symmetric Cryptography: New Security Proofs**
  - Motivated by the following question:

Do classical symmetric key primitives remain secure versus quantum adversaries?

  - More nuanced than just running Grover Search.

Moving Towards Post-quantum Crypto

- Two main thrusts have been in the works for years to move cryptosystems towards post-quantum security.

- **Symmetric Cryptography: New Security Proofs**
  - Motivated by the following question:

Do classical symmetric key primitives remain secure versus quantum adversaries?

  - More nuanced than just running Grover Search.
  - Do quantum adversaries get *classical* or *quantum* access to the underlying functions?

Moving Towards Post-quantum Crypto

- Two main thrusts have been in the works for years to move cryptosystems towards post-quantum security.

- **Symmetric Cryptography: New Security Proofs**
  - Motivated by the following question:

Do classical symmetric key primitives remain secure versus quantum adversaries?

  - More nuanced than just running Grover Search.
  - Do quantum adversaries get *classical* or *quantum* access to the underlying functions?
  - Some constructions are secure vs. quantum adversaries only given *classical* access, then completely broken when the adversary has *quantum* access.

Roadmap

Roadmap

- Today

Roadmap

- Today
  - Discuss the new post-quantum hardness assumptions.

Roadmap

- Today
  - Discuss the new post-quantum hardness assumptions.
  - Describe some of the current NIST constructions for these assumptions.

Roadmap

- Today
  - Discuss the new post-quantum hardness assumptions.
  - Describe some of the current NIST constructions for these assumptions.
  - Discuss the most commonly used assumption as of now: lattice-based cryptography.

Roadmap

- Today
  - Discuss the new post-quantum hardness assumptions.
  - Describe some of the current NIST constructions for these assumptions.
  - Discuss the most commonly used assumption as of now: lattice-based cryptography.
- Wednesday

Roadmap

- Today
  - Discuss the new post-quantum hardness assumptions.
  - Describe some of the current NIST constructions for these assumptions.
  - Discuss the most commonly used assumption as of now: lattice-based cryptography.
- Wednesday
  - The Learning-with-Errors assumption.

Roadmap

- Today
  - Discuss the new post-quantum hardness assumptions.
  - Describe some of the current NIST constructions for these assumptions.
  - Discuss the most commonly used assumption as of now: lattice-based cryptography.
- Wednesday
  - The Learning-with-Errors assumption.
  - Key-exchange from LWE.

Roadmap

- Today
  - Discuss the new post-quantum hardness assumptions.
  - Describe some of the current NIST constructions for these assumptions.
  - Discuss the most commonly used assumption as of now: lattice-based cryptography.

- Wednesday
  - The Learning-with-Errors assumption.
  - Key-exchange from LWE.
  - Security considerations for symmetric-key crypto.

Roadmap

- Today
  - Discuss the new post-quantum hardness assumptions.
  - Describe some of the current NIST constructions for these assumptions.
  - Discuss the most commonly used assumption as of now: lattice-based cryptography.

- Wednesday
  - The Learning-with-Errors assumption.
  - Key-exchange from LWE.
  - Security considerations for symmetric-key crypto.
  - Final in-class presentation.

Roadmap

- Today
  - Discuss the new post-quantum hardness assumptions.
  - Describe some of the current NIST constructions for these assumptions.
  - Discuss the most commonly used assumption as of now: lattice-based cryptography.
- Wednesday
  - The Learning-with-Errors assumption.
  - Key-exchange from LWE.
  - Security considerations for symmetric-key crypto.
  - Final in-class presentation.
- Next Week

Roadmap

- Today
  - Discuss the new post-quantum hardness assumptions.
  - Describe some of the current NIST constructions for these assumptions.
  - Discuss the most commonly used assumption as of now: lattice-based cryptography.
- Wednesday
  - The Learning-with-Errors assumption.
  - Key-exchange from LWE.
  - Security considerations for symmetric-key crypto.
  - Final in-class presentation.
- Next Week
  - Final Project Presentations!

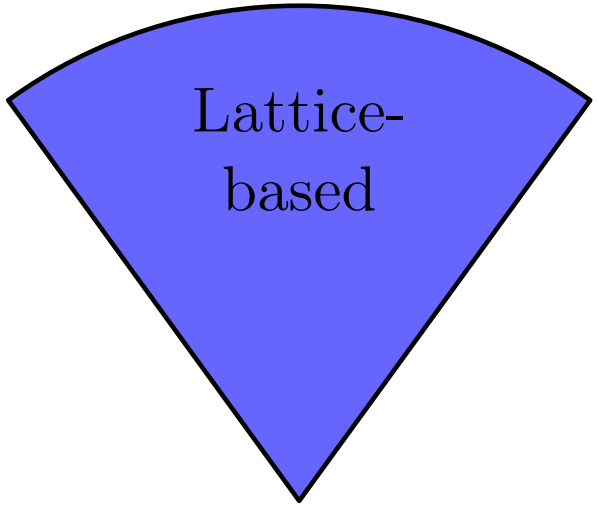
Roadmap

- Today
  - Discuss the new post-quantum hardness assumptions.
  - Describe some of the current NIST constructions for these assumptions.
  - Discuss the most commonly used assumption as of now: lattice-based cryptography.
- Wednesday
  - The Learning-with-Errors assumption.
  - Key-exchange from LWE.
  - Security considerations for symmetric-key crypto.
  - Final in-class presentation.
- Next Week
  - Final Project Presentations!
  - Your presentation days and order will be announced today by 3:00pm.

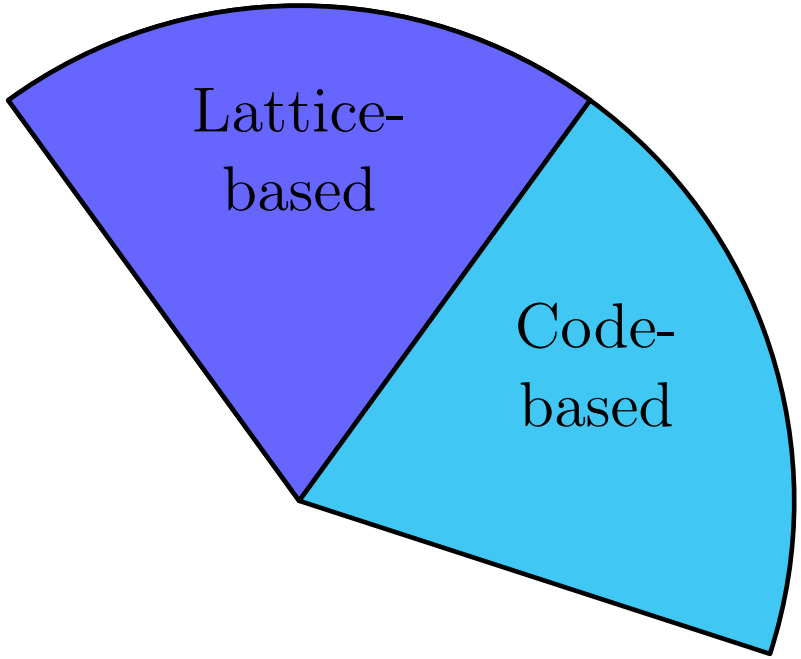
Post-quantum Hardness Assumptions

The 5 Assumptions

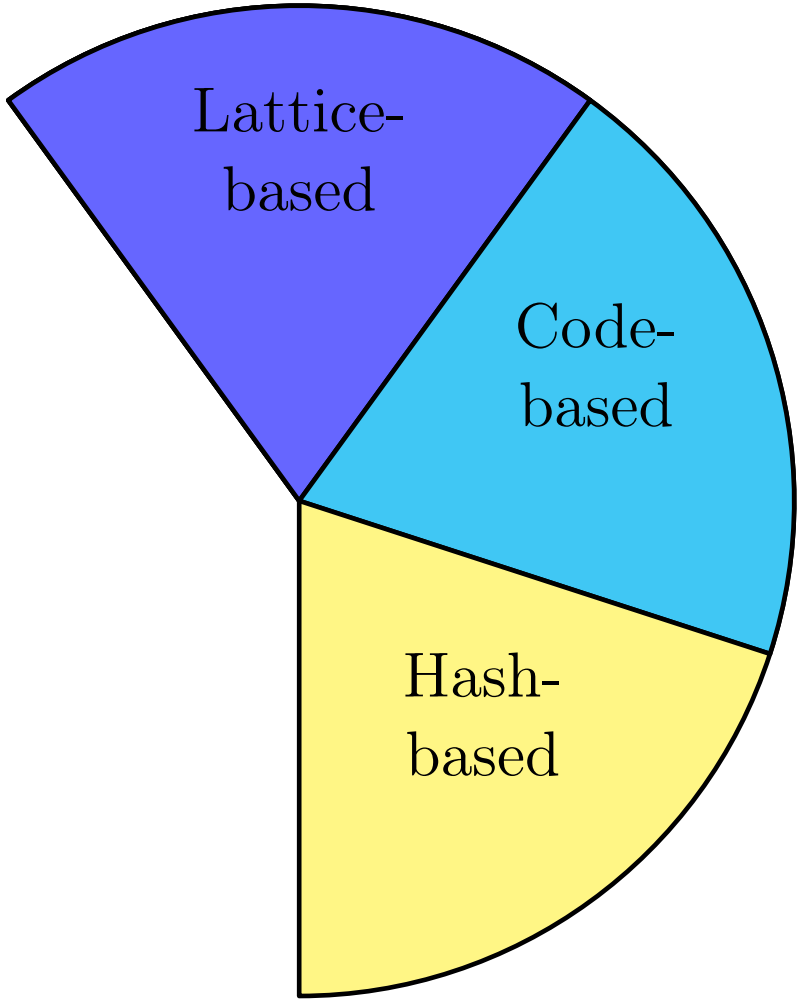
The 5 Assumptions



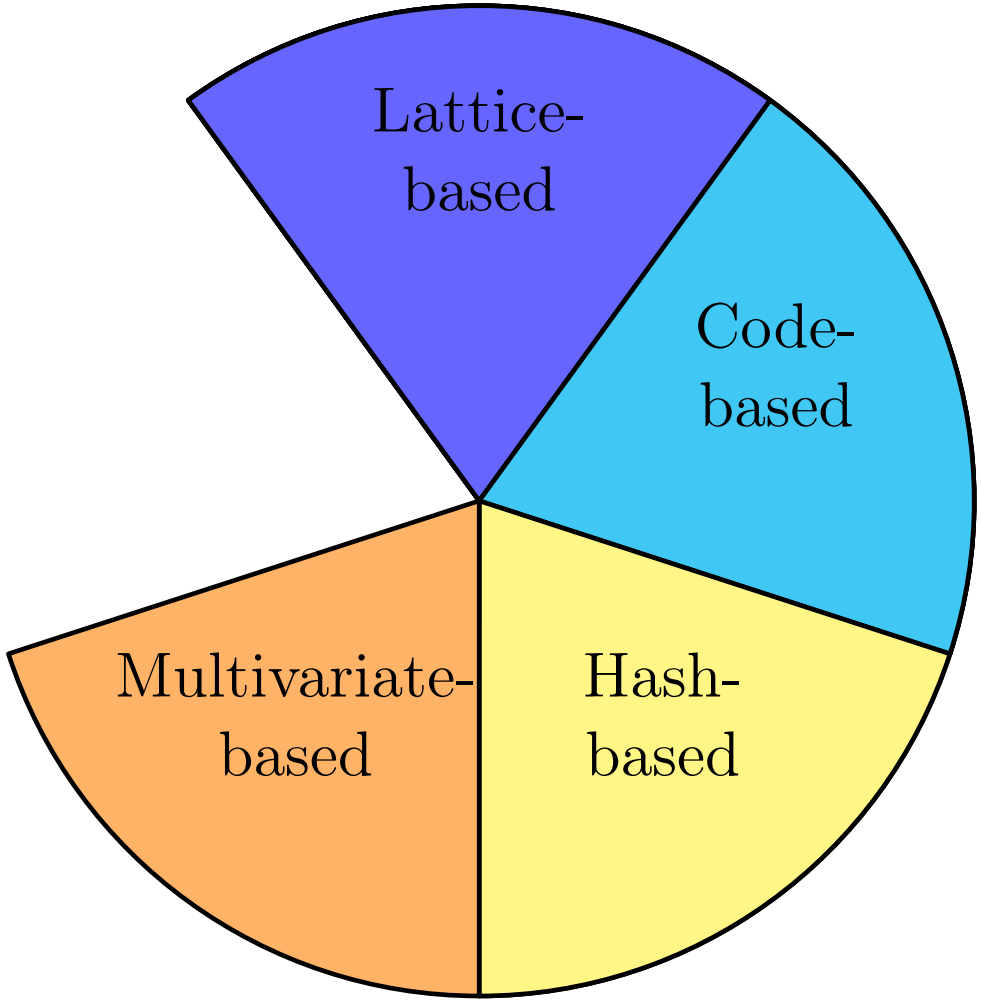
The 5 Assumptions



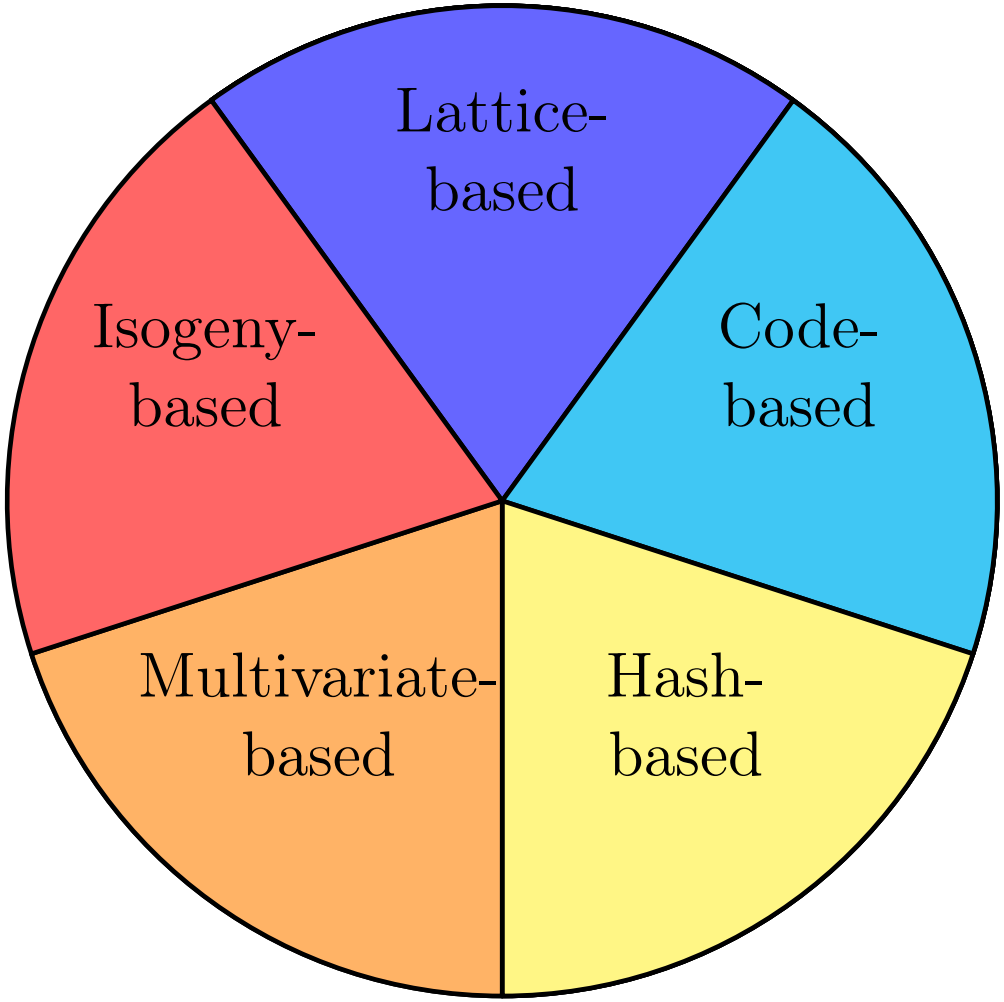
The 5 Assumptions



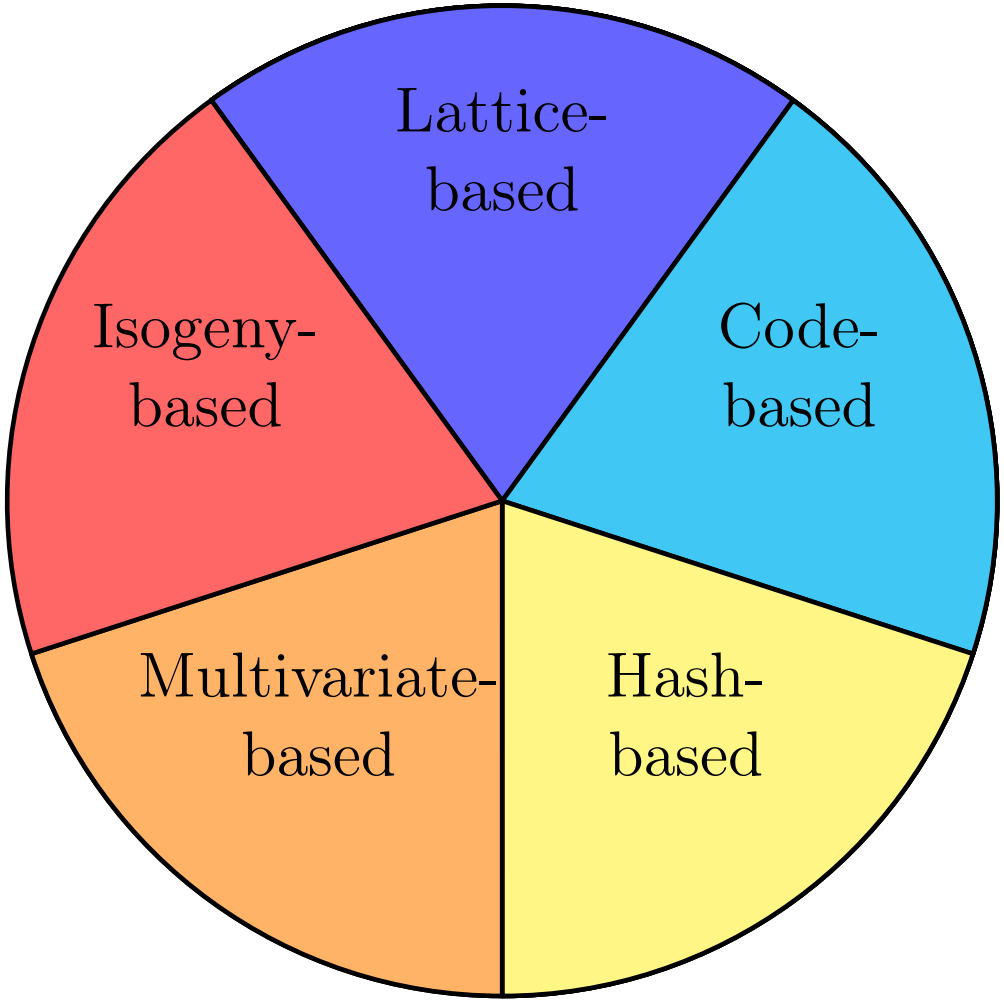
The 5 Assumptions



The 5 Assumptions

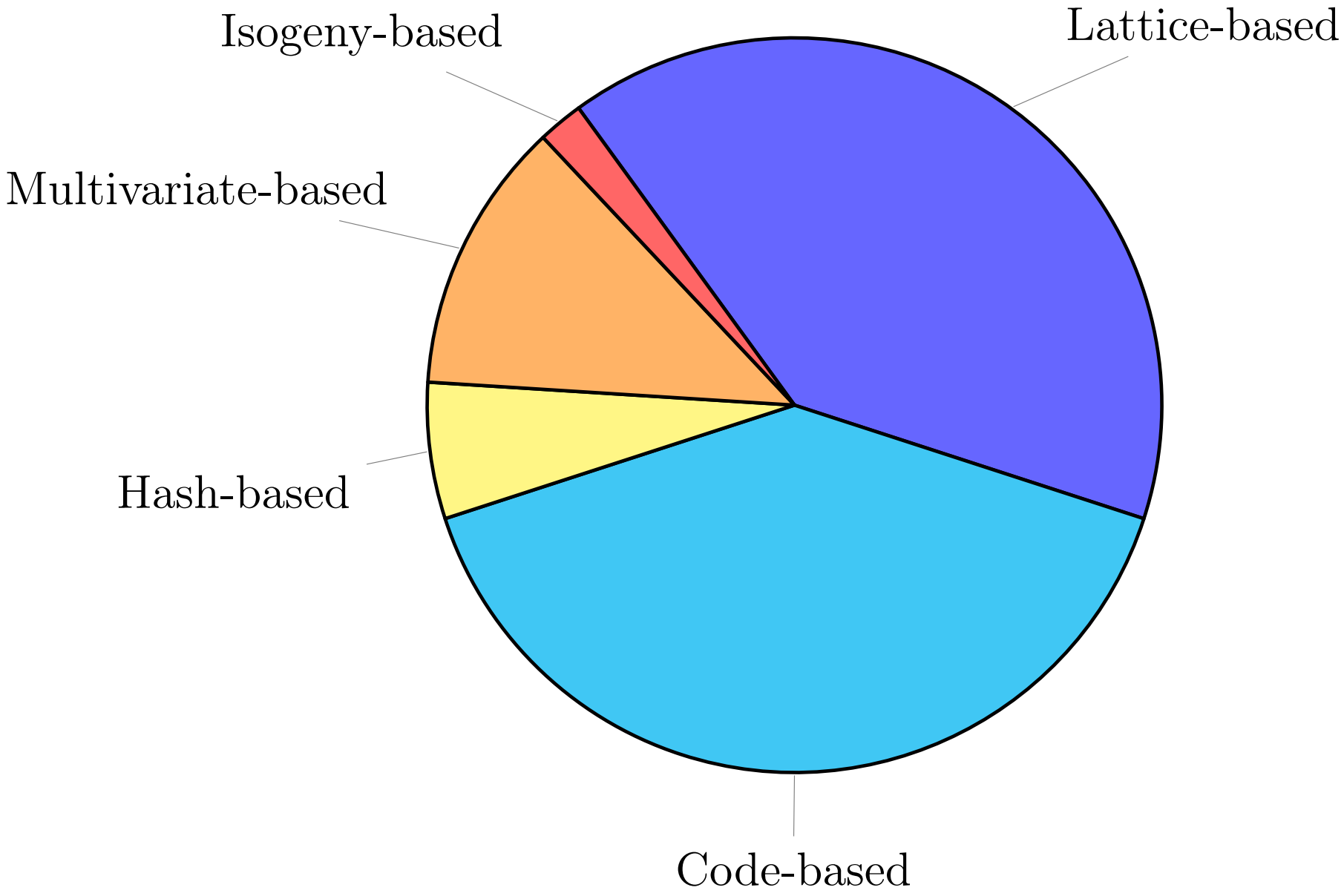


The 5 Assumptions



Not all assumptions are equal!

The 5 Assumptions (Scaled)



Isogeny-based PQ Cryptography

Isogeny-based PQ Cryptography

- Formally: *Supersingular Elliptic-curve Isogeny Cryptography.*

Isogeny-based PQ Cryptography

- Formally: *Supersingular Elliptic-curve Isogeny Cryptography*.
- Introduced in 2006 by Rostovtsev and Stolbunov.

Isogeny-based PQ Cryptography

- Formally: *Supersingular Elliptic-curve Isogeny Cryptography.*
- Introduced in 2006 by Rostovtsev and Stolbunov.
  - Built off of the rich area of elliptic-curve cryptography from Miller (1985) and Koblitz (1987).

Isogeny-based PQ Cryptography

- Formally: *Supersingular Elliptic-curve Isogeny Cryptography*.
- Introduced in 2006 by Rostovtsev and Stolbunov.
  - Built off of the rich area of elliptic-curve cryptography from Miller (1985) and Koblitz (1987).
  - Standard elliptic-curve crypto is widely used *but* is not post-quantum secure!

Isogeny-based PQ Cryptography

- Formally: *Supersingular Elliptic-curve Isogeny Cryptography.*
- Introduced in 2006 by Rostovtsev and Stolbunov.
  - Built off of the rich area of elliptic-curve cryptography from Miller (1985) and Koblitz (1987).
  - Standard elliptic-curve crypto is widely used *but* is not post-quantum secure!

- **Idea:** use *isogenies* (specific types of morphisms) between two elliptic curves to construct cryptographic protocols.

Isogeny-based PQ Cryptography

- Formally: *Supersingular Elliptic-curve Isogeny Cryptography.*
- Introduced in 2006 by Rostovtsev and Stolbunov.
  - Built off of the rich area of elliptic-curve cryptography from Miller (1985) and Koblitz (1987).
  - Standard elliptic-curve crypto is widely used *but* is not post-quantum secure!

- **Idea:** use *isogenies* (specific types of morphisms) between two elliptic curves to construct cryptographic protocols.
  - However, in 2010, Childs, Jao, and Soukharven found a subexponential-time quantum algorithm to break the construction of Rostovtsev and Stolbunov.

Isogeny-based PQ Cryptography

- Formally: *Supersingular Elliptic-curve Isogeny Cryptography.*
- Introduced in 2006 by Rostovtsev and Stolbunov.
  - Built off of the rich area of elliptic-curve cryptography from Miller (1985) and Koblitz (1987).
  - Standard elliptic-curve crypto is widely used *but* is not post-quantum secure!

- **Idea:** use *isogenies* (specific types of morphisms) between two elliptic curves to construct cryptographic protocols.
  - However, in 2010, Childs, Jao, and Soukharven found a subexponential-time quantum algorithm to break the construction of Rostovtsev and Stolbunov.

- Jao and De Feo in 2011 avoided this attack by using isogenies over *supersingular* elliptic curves.

Isogeny-based PQ Cryptography

- Formally: *Supersingular Elliptic-curve Isogeny Cryptography.*
- Introduced in 2006 by Rostovtsev and Stolbunov.
  - Built off of the rich area of elliptic-curve cryptography from Miller (1985) and Koblitz (1987).
  - Standard elliptic-curve crypto is widely used *but* is not post-quantum secure!

- **Idea:** use *isogenies* (specific types of morphisms) between two elliptic curves to construct cryptographic protocols.
  - However, in 2010, Childs, Jao, and Soukharven found a subexponential-time quantum algorithm to break the construction of Rostovtsev and Stolbunov.

- Jao and De Feo in 2011 avoided this attack by using isogenies over *supersingular* elliptic curves.
  - The supersingular structure avoids the attack of Childs, Jao, and Soukharven.

Hard Isogeny Problems

Hard Isogeny Problems

1. **The Isogeny Problem:** Given two (random) elliptic curves E_1 and E_2, find an isogeny of prime-power degree ℓ^n between them.

Hard Isogeny Problems

1. **The Isogeny Problem:** Given two (random) elliptic curves E_1 and E_2, find an isogeny of prime-power degree ℓ^n between them.
   - **Fact:** There are $\approx \frac{(p+1)}{12}$ *supersingular* curves E over $\mathbb{F}_{p^2}$ for prime p.

Hard Isogeny Problems

1. **The Isogeny Problem:** Given two (random) elliptic curves E_1 and E_2, find an isogeny of prime-power degree ℓ^n between them.
   - **Fact:** There are $\approx \frac{(p+1)}{12}$ *supersingular* curves E over $\mathbb{F}_{p^2}$ for prime p.
   - **Fact:** we can build a $(\ell+1)$-regular *isogeny graph* $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ as

Hard Isogeny Problems

1. **The Isogeny Problem:** Given two (random) elliptic curves E_1 and E_2, find an isogeny of prime-power degree ℓ^n between them.
   - **Fact:** There are $\approx \frac{(p+1)}{12}$ *supersingular* curves E over $\mathbb{F}_{p^2}$ for prime p.
   - **Fact:** we can build a $(\ell+1)$-regular *isogeny graph* $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ as

$$\mathcal{V} = \{E : \text{ supersingular over } \mathbb{F}_{p^2}\}$$

Hard Isogeny Problems

1. **The Isogeny Problem:** Given two (random) elliptic curves E_1 and E_2, find an isogeny of prime-power degree ℓ^n between them.
   - **Fact:** There are $\approx \frac{(p+1)}{12}$ *supersingular* curves E over $\mathbb{F}_{p^2}$ for prime p.
   - **Fact:** we can build a $(\ell+1)$-regular *isogeny graph* $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ as

$$\mathcal{V} = \{E\colon \text{ supersingular over } \mathbb{F}_{p^2}\}$$

$$\mathcal{E} = \{(E_1, E_2)\colon \phi\colon E_1 \to E_2 \text{ is an isogeny for } E_1, E_2 \in \mathcal{V}\}.$$

deg ℓ

Hard Isogeny Problems

1. **The Isogeny Problem:** Given two (random) elliptic curves E_1 and E_2, find an isogeny of prime-power degree ℓ^n between them.
   - **Fact:** There are $\approx \frac{(p+1)}{12}$ *supersingular* curves E over $\mathbb{F}_{p^2}$ for prime p.
   - **Fact:** we can build a $(\ell+1)$-regular *isogeny graph* $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ as

$$\mathcal{V} = \{E\colon \text{ supersingular over } \mathbb{F}_{p^2}\}$$

$$\mathcal{E} = \{(E_1, E_2)\colon \phi\colon E_1 \to E_2 \text{ is an isogeny for } E_1, E_2 \in \mathcal{V}\}.$$

   - An isogeny of degree ℓ^n corresponds to the composition of n isogenies of degree ℓ.

Hard Isogeny Problems

1. **The Isogeny Problem:** Given two (random) elliptic curves E_1 and E_2, find an isogeny of prime-power degree ℓ^n between them.
   - **Fact:** There are $\approx \frac{(p+1)}{12}$ *supersingular* curves E over $\mathbb{F}_{p^2}$ for prime p.
   - **Fact:** we can build a $(\ell+1)$-regular *isogeny graph* $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ as

$$\mathcal{V} = \{E\colon \text{ supersingular over } \mathbb{F}_{p^2}\}$$

$$\mathcal{E} = \{(E_1, E_2)\colon \phi\colon E_1 \to E_2 \text{ is an isogeny for } E_1, E_2 \in \mathcal{V}\}.$$

   - An isogeny of degree ℓ^n corresponds to the composition of n isogenies of degree ℓ. A *path* in the isogeny graph $\mathcal{G}$ of length n corresponds to an isogeny of degree ℓ^n.

Hard Isogeny Problems

1. **The Isogeny Problem:** Given two (random) elliptic curves E_1 and E_2, find an isogeny of prime-power degree ℓ^n between them.
   - **Fact:** There are $\approx \frac{(p+1)}{12}$ *supersingular* curves E over $\mathbb{F}_{p^2}$ for prime p.
   - **Fact:** we can build a $(\ell+1)$-regular *isogeny graph* $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ as

$$\mathcal{V} = \{E\colon \text{ supersingular over } \mathbb{F}_{p^2}\}$$
$$\mathcal{E} = \{(E_1, E_2)\colon \phi\colon E_1 \to E_2 \text{ is an isogeny for } E_1, E_2 \in \mathcal{V}\}.$$

   - An isogeny of degree ℓ^n corresponds to the composition of n isogenies of degree ℓ. A *path* in the isogeny graph $\mathcal{G}$ of length n corresponds to an isogeny of degree ℓ^n.

2. **The ℓ-IsogenyPath Problem:** Given supersingular E_1 and E_2, find an ℓ-isogeny path from E_1 to E_2.

Hard Isogeny Problems

1. **The Isogeny Problem:** Given two (random) elliptic curves E_1 and E_2, find an isogeny of prime-power degree ℓ^n between them.
   - **Fact:** There are $\approx \frac{(p+1)}{12}$ *supersingular* curves E over $\mathbb{F}_{p^2}$ for prime p.
   - **Fact:** we can build a $(\ell+1)$-regular *isogeny graph* $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ as

$$\mathcal{V} = \{E\colon \text{ supersingular over } \mathbb{F}_{p^2}\}$$
$$\mathcal{E} = \{(E_1, E_2)\colon \phi\colon E_1 \to E_2 \text{ is an isogeny for } E_1, E_2 \in \mathcal{V}\}.$$

   - An isogeny of degree ℓ^n corresponds to the composition of n isogenies of degree ℓ. A *path* in the isogeny graph $\mathcal{G}$ of length n corresponds to an isogeny of degree ℓ^n.

2. **The ℓ-IsogenyPath Problem:** Given supersingular E_1 and E_2, find an ℓ-isogeny path from E_1 to E_2.
   - I.e., find a path from E_1 to E_2 in $\mathcal{G}$.

Hard Isogeny Problems

1. **The Isogeny Problem:** Given two (random) elliptic curves E_1 and E_2, find an isogeny of prime-power degree ℓ^n between them.
   - **Fact:** There are $\approx \frac{(p+1)}{12}$ *supersingular* curves E over $\mathbb{F}_{p^2}$ for prime p.
   - **Fact:** we can build a $(\ell+1)$-regular *isogeny graph* $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ as

$$\mathcal{V} = \{E\colon \text{ supersingular over } \mathbb{F}_{p^2}\}$$
$$\mathcal{E} = \{(E_1, E_2)\colon \phi\colon E_1 \to E_2 \text{ is an isogeny for } E_1, E_2 \in \mathcal{V}\}.$$

   - An isogeny of degree ℓ^n corresponds to the composition of n isogenies of degree ℓ. A *path* in the isogeny graph $\mathcal{G}$ of length n corresponds to an isogeny of degree ℓ^n.

2. **The ℓ-IsogenyPath Problem:** Given supersingular E_1 and E_2, find an ℓ-isogeny path from E_1 to E_2.
   - I.e., find a path from E_1 to E_2 in $\mathcal{G}$.

Remark

Isogeny problem $\iff$ ℓ-IsogenyPath Problem

Hard Isogeny Problems

3. **The EndRing Problem:** Given supersingular E, find 4 generators of the endomorphism ring $\mathsf{End}(E)$, where $\mathsf{End}(E)$ is set of endomorphisms of E.

Hard Isogeny Problems

3. **The EndRing Problem:** Given supersingular E, find 4 generators of the endomorphism ring $\mathsf{End}(E)$, where $\mathsf{End}(E)$ is set of endomorphisms of E.
   - For supersingular E, $(\mathsf{End}(E), +)$ is a *lattice* of dimension 4.

Hard Isogeny Problems

3. **The EndRing Problem:** Given supersingular E, find 4 generators of the endomorphism ring $\mathsf{End}(E)$, where $\mathsf{End}(E)$ is set of endomorphisms of E.
   - For supersingular E, $(\mathsf{End}(E), +)$ is a *lattice* of dimension 4.

4. **The OneEnd Problem:** Given supersingular E, find *one* endomorphism $\phi \in \mathsf{End}(E) \setminus \mathbb{Z}$.

Hard Isogeny Problems

3. **The EndRing Problem:** Given supersingular E, find 4 generators of the endomorphism ring $\mathsf{End}(E)$, where $\mathsf{End}(E)$ is set of endomorphisms of E.
   - For supersingular E, $(\mathsf{End}(E), +)$ is a *lattice* of dimension 4.

4. **The OneEnd Problem:** Given supersingular E, find *one* endomorphism $\phi \in \mathsf{End}(E) \setminus \mathbb{Z}$.

Remark

$$\text{EndRing} \iff \text{OneEnd}$$

Hard Isogeny Problems

3. **The EndRing Problem:** Given supersingular E, find 4 generators of the endomorphism ring $\mathsf{End}(E)$, where $\mathsf{End}(E)$ is set of endomorphisms of E.
   - For supersingular E, $(\mathsf{End}(E), +)$ is a *lattice* of dimension 4.

4. **The OneEnd Problem:** Given supersingular E, find *one* endomorphism $\phi \in \mathsf{End}(E) \setminus \mathbb{Z}$.

Remark

$$\text{EndRing} \iff \text{OneEnd}$$

Remark

$$\text{2-IsogenyPath} \implies \text{EndRing, OneEnd} \implies \text{Isogeny}$$

Hard Isogeny Problems

3. **The EndRing Problem:** Given supersingular E, find 4 generators of the endomorphism ring $\mathsf{End}(E)$, where $\mathsf{End}(E)$ is set of endomorphisms of E.
   - For supersingular E, $(\mathsf{End}(E), +)$ is a *lattice* of dimension 4.

4. **The OneEnd Problem:** Given supersingular E, find *one* endomorphism $\phi \in \mathsf{End}(E) \setminus \mathbb{Z}$.

Remark

$$\text{EndRing} \iff \text{OneEnd}$$

Remark

$$\text{2-IsogenyPath} \implies \text{EndRing, OneEnd} \implies \text{Isogeny}$$

Corollary 1

$$\textit{EndRing} \iff \textit{Isogeny}$$

NIST Candidate Isogeny Constructions

NIST Candidate Isogeny Constructions

- KEM Candidate: *Supersingular Isogeny Key Exchange* (SIKE).

NIST Candidate Isogeny Constructions

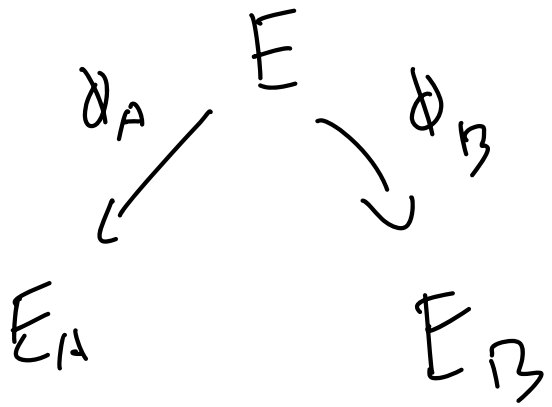
- KEM Candidate: *Supersingular Isogeny Key Exchange* (SIKE).
  - Basic Idea: DDH key exchange for isogenies.

NIST Candidate Isogeny Constructions

- KEM Candidate: *Supersingular Isogeny Key Exchange* (SIKE).
  - Basic Idea: DDH key exchange for isogenies.
  - There is a public curve E with some fixed points P_A, P_B, Q_A, Q_B.

NIST Candidate Isogeny Constructions

- KEM Candidate: *Supersingular Isogeny Key Exchange* (SIKE).
  - Basic Idea: DDH key exchange for isogenies.
  - There is a public curve E with some fixed points P_A, P_B, Q_A, Q_B.
  - To agree upon a secret key, Alice samples random isogeny $\phi_A \colon E \to E_A$ and Bob $\phi_B \colon E \to E_B$.



NIST Candidate Isogeny Constructions

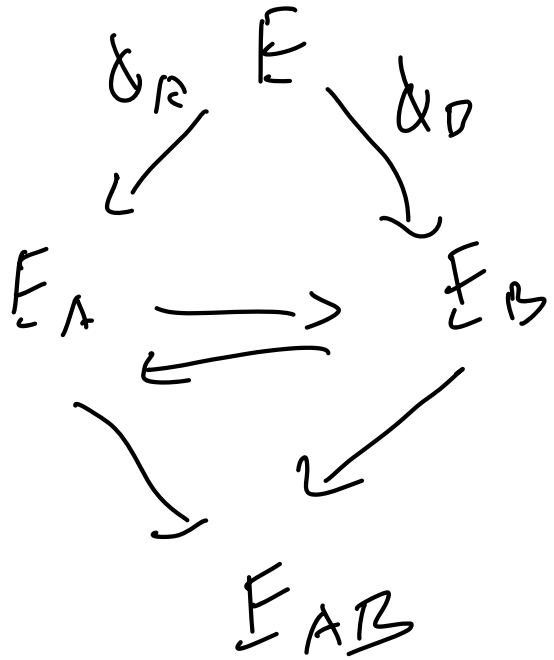
- KEM Candidate: *Supersingular Isogeny Key Exchange* (SIKE).
  - Basic Idea: DDH key exchange for isogenies.
  - There is a public curve E with some fixed points P_A, P_B, Q_A, Q_B.
  - To agree upon a secret key, Alice samples random isogeny $\phi_A \colon E \to E_A$ and Bob $\phi_B \colon E \to E_B$.
  - Alice computes $(E_A, \phi_A(P_B), \phi_A(Q_B))$ and Bob computes $(E_B, \phi_B(P_A), \phi_B(Q_A))$.

NIST Candidate Isogeny Constructions

- KEM Candidate: *Supersingular Isogeny Key Exchange* (SIKE).
  - Basic Idea: DDH key exchange for isogenies.
  - There is a public curve E with some fixed points P_A, P_B, Q_A, Q_B.
  - To agree upon a secret key, Alice samples random isogeny $\phi_A \colon E \to E_A$ and Bob $\phi_B \colon E \to E_B$.
  - Alice computes $(E_A, \phi_A(P_B), \phi_A(Q_B))$ and Bob computes $(E_B, \phi_B(P_A), \phi_B(Q_A))$. Parties then exchange their tuples.

NIST Candidate Isogeny Constructions

- KEM Candidate: *Supersingular Isogeny Key Exchange* (SIKE).
  - Basic Idea: DDH key exchange for isogenies.
  - There is a public curve E with some fixed points P_A, P_B, Q_A, Q_B.
  - To agree upon a secret key, Alice samples random isogeny $\phi_A \colon E \to E_A$ and Bob $\phi_B \colon E \to E_B$.
  - Alice computes $(E_A, \phi_A(P_B), \phi_A(Q_B))$ and Bob computes $(E_B, \phi_B(P_A), \phi_B(Q_A))$. Parties then exchange their tuples.
  - Using this information, each party can compute some common curve E_{AB} and use its j-invariant as the secret key.



NIST Candidate Isogeny Constructions

- KEM Candidate: *Supersingular Isogeny Key Exchange* (SIKE).
  - Basic Idea: DDH key exchange for isogenies.
  - There is a public curve E with some fixed points P_A, P_B, Q_A, Q_B.
  - To agree upon a secret key, Alice samples random isogeny $\phi_A \colon E \to E_A$ and Bob $\phi_B \colon E \to E_B$.
  - Alice computes $(E_A, \phi_A(P_B), \phi_A(Q_B))$ and Bob computes $(E_B, \phi_B(P_A), \phi_B(Q_A))$. Parties then exchange their tuples.
  - Using this information, each party can compute some common curve E_{AB} and use its j-invariant as the secret key.

- Digital Signature Candidate: SQIsign

NIST Candidate Isogeny Constructions

- KEM Candidate: *Supersingular Isogeny Key Exchange* (SIKE).
  - Basic Idea: DDH key exchange for isogenies.
  - There is a public curve E with some fixed points P_A, P_B, Q_A, Q_B.
  - To agree upon a secret key, Alice samples random isogeny $\phi_A \colon E \to E_A$ and Bob $\phi_B \colon E \to E_B$.
  - Alice computes $(E_A, \phi_A(P_B), \phi_A(Q_B))$ and Bob computes $(E_B, \phi_B(P_A), \phi_B(Q_A))$. Parties then exchange their tuples.
  - Using this information, each party can compute some common curve E_{AB} and use its j-invariant as the secret key.

- Digital Signature Candidate: SQIsign
  - Round 2 candidate from the additional signatures call.

NIST Candidate Isogeny Constructions

- KEM Candidate: *Supersingular Isogeny Key Exchange* (SIKE).
  - Basic Idea: DDH key exchange for isogenies.
  - There is a public curve E with some fixed points P_A, P_B, Q_A, Q_B.
  - To agree upon a secret key, Alice samples random isogeny $\phi_A \colon E \to E_A$ and Bob $\phi_B \colon E \to E_B$.
  - Alice computes $(E_A, \phi_A(P_B), \phi_A(Q_B))$ and Bob computes $(E_B, \phi_B(P_A), \phi_B(Q_A))$. Parties then exchange their tuples.
  - Using this information, each party can compute some common curve E_{AB} and use its j-invariant as the secret key.

- Digital Signature Candidate: SQIsign
  - Round 2 candidate from the additional signatures call.
  - Hardness based on EndRing.

NIST Candidate Isogeny Constructions

- KEM Candidate: *Supersingular Isogeny Key Exchange* (SIKE).
  - Basic Idea: DDH key exchange for isogenies.
  - There is a public curve E with some fixed points P_A, P_B, Q_A, Q_B.
  - To agree upon a secret key, Alice samples random isogeny $\phi_A \colon E \to E_A$ and Bob $\phi_B \colon E \to E_B$.
  - Alice computes $(E_A, \phi_A(P_B), \phi_A(Q_B))$ and Bob computes $(E_B, \phi_B(P_A), \phi_B(Q_A))$. Parties then exchange their tuples.
  - Using this information, each party can compute some common curve E_{AB} and use its j-invariant as the secret key.

- Digital Signature Candidate: SQIsign
  - Round 2 candidate from the additional signatures call.
  - Hardness based on EndRing.
  - Signature from the Fiat-Shamir transformation of a Σ-protocol Proof of Knowledge of $\mathsf{End}(E)$, where E is a public curve.

NIST Candidate Isogeny Constructions

NIST Candidate Isogeny Constructions


COULDA BEEN A CONTENDER

Post-quantum encryption contender is taken out by single-core PC and 1 hour

Leave it to mathematicians to muck up what looked like an impressive new algorithm.

DAN GOODIN – AUG 2, 2022 7:31 AM | 132

An efficient key recovery attack on SIDH

Wouter Castryck[1,2] and Thomas Decru[1]

[1] imec-COSIC, KU Leuven, Belgium
[2] Vakgroep Wiskunde: Algebra en Meetkunde, Universiteit Gent, Belgium

Abstract. We present an efficient key recovery attack on the Supersingular Isogeny Diffie–Hellman protocol (SIDH). The attack is based on Kani's "reducibility criterion" for isogenies from products of elliptic curves and strongly relies on the torsion point images that Alice and Bob exchange during the protocol. If we assume knowledge of the endomorphism ring of the starting curve then the classical running time is polynomial in the input size (heuristically), apart from the factorization of a small number of integers that only depend on the system parameters. The attack is particularly fast and easy to implement if one of the parties uses 2-isogenies and the starting curve comes equipped with a non-scalar endomorphism of very small degree; this is the case for SIKE, the instantiation of SIDH that recently advanced to the fourth round of NIST's standardization effort for post-quantum cryptography. Our Magma implementation breaks `SIKEp434`, which aims at security level 1, in about ten minutes on a single core.

Hash-based PQ Cryptography

Hash-based PQ Cryptography

- Crypto based on cryptographic hash functions $H\colon \{0,1\}^* \to \{0,1\}^\lambda$.

Hash-based PQ Cryptography

- Crypto based on cryptographic hash functions $H : \{0,1\}^* \to \{0,1\}^\lambda$.
- We require the following properties (via the definition of "cryptographic" above).

Hash-based PQ Cryptography

- Crypto based on cryptographic hash functions $H : \{0,1\}^* \to \{0,1\}^\lambda$.

- We require the following properties (via the definition of "cryptographic" above).
  - **Pre-image Resistance:** given $x \xleftarrow{\$} \{0,1\}^*$, it is computationally infeasible to find x' such that $H(x) = H(x')$.

Hash-based PQ Cryptography

- Crypto based on cryptographic hash functions $H : \{0,1\}^* \to \{0,1\}^\lambda$.

- We require the following properties (via the definition of "cryptographic" above).
  - **Pre-image Resistance:** given $x \xleftarrow{\$} \{0,1\}^*$, it is computationally infeasible to find x' such that $H(x) = H(x')$.
  - **Second Pre-image Resistance:** Given $x \xleftarrow{\$} \{0,1\}^*$, it is computationally infeasible to find $x' \neq x$ such that $H(x) = H(x')$.

Hash-based PQ Cryptography

- Crypto based on cryptographic hash functions $H : \{0,1\}^* \to \{0,1\}^\lambda$.

- We require the following properties (via the definition of "cryptographic" above).
  - **Pre-image Resistance:** given $x \xleftarrow{\$} \{0,1\}^*$, it is computationally infeasible to find x' such that $H(x) = H(x')$.
  - **Second Pre-image Resistance:** Given $x \xleftarrow{\$} \{0,1\}^*$, it is computationally infeasible to find $x' \neq x$ such that $H(x) = H(x')$.
  - **Collision Resistance:** Given H, it is computationally infeasible to find $x \neq x'$ such that $H(x) = H(x')$.

Hash-based PQ Cryptography

- Crypto based on cryptographic hash functions $H : \{0,1\}^* \to \{0,1\}^\lambda$.

- We require the following properties (via the definition of "cryptographic" above).
  - **Pre-image Resistance:** given $x \xleftarrow{\$} \{0,1\}^*$, it is computationally infeasible to find x' such that $H(x) = H(x')$.
  - **Second Pre-image Resistance:** Given $x \xleftarrow{\$} \{0,1\}^*$, it is computationally infeasible to find $x' \neq x$ such that $H(x) = H(x')$.
  - **Collision Resistance:** Given H, it is computationally infeasible to find $x \neq x'$ such that $H(x) = H(x')$.

- With these properties, we can construct digital signature schemes.

Hash-based PQ Cryptography

- Crypto based on cryptographic hash functions $H : \{0,1\}^* \to \{0,1\}^\lambda$.

- We require the following properties (via the definition of "cryptographic" above).
  - **Pre-image Resistance:** given $x \xleftarrow{\$} \{0,1\}^*$, it is computationally infeasible to find x' such that $H(x) = H(x')$.
  - **Second Pre-image Resistance:** Given $x \xleftarrow{\$} \{0,1\}^*$, it is computationally infeasible to find $x' \neq x$ such that $H(x) = H(x')$.
  - **Collision Resistance:** Given H, it is computationally infeasible to find $x \neq x'$ such that $H(x) = H(x')$.

- With these properties, we can construct digital signature schemes.

- In order to get PKE or KEM, we need *trapdoor hash functions*.

Hash-based PQ Cryptography

- Crypto based on cryptographic hash functions $H : \{0,1\}^* \to \{0,1\}^\lambda$.

- We require the following properties (via the definition of "cryptographic" above).
  - **Pre-image Resistance:** given $x \xleftarrow{\$} \{0,1\}^*$, it is computationally infeasible to find x' such that $H(x) = H(x')$.
  - **Second Pre-image Resistance:** Given $x \xleftarrow{\$} \{0,1\}^*$, it is computationally infeasible to find $x' \neq x$ such that $H(x) = H(x')$.
  - **Collision Resistance:** Given H, it is computationally infeasible to find $x \neq x'$ such that $H(x) = H(x')$.

- With these properties, we can construct digital signature schemes.

- In order to get PKE or KEM, we need *trapdoor hash functions*.
  - There is a trapdoor τ_i such that given $y = H(x)$, τ_i allows one to recover x_i.

Hardness of Hashing

Hardness of Hashing

- As discussed last lecture, hardness of hashing reduces to finding collisions (e.g., violating collision resistance).

Hardness of Hashing

- As discussed last lecture, hardness of hashing reduces to finding collisions (e.g., violating collision resistance).

- Grover's Algorithm allows us to do this in $O(2^{\lambda/3})$ quantum time with $O(2^{\lambda/3})$ samples (so $\lambda \cdot 2^{\lambda/3}$-bits approximately).

Hardness of Hashing

- As discussed last lecture, hardness of hashing reduces to finding collisions (e.g., violating collision resistance).

- Grover's Algorithm allows us to do this in $O(2^{\lambda/3})$ quantum time with $O(2^{\lambda/3})$ samples (so $\lambda \cdot 2^{\lambda/3}$-bits approximately).
  - E.g., for 128-bits of security, we set $\lambda = 384$.

Hardness of Hashing

- As discussed last lecture, hardness of hashing reduces to finding collisions (e.g., violating collision resistance).

- Grover's Algorithm allows us to do this in $O(2^{\lambda/3})$ quantum time with $O(2^{\lambda/3})$ samples (so $\lambda \cdot 2^{\lambda/3}$-bits approximately).
  - E.g., for 128-bits of security, we set $\lambda = 384$. Grover's algorithm takes $O(2^{128})$ quantum operations and uses approximately $384 \cdot 2^{128} \approx 2^{137}$ bits of space (at least).

Hardness of Hashing

- As discussed last lecture, hardness of hashing reduces to finding collisions (e.g., violating collision resistance).

- Grover's Algorithm allows us to do this in $O(2^{\lambda/3})$ quantum time with $O(2^{\lambda/3})$ samples (so $\lambda \cdot 2^{\lambda/3}$-bits approximately).
  - E.g., for 128-bits of security, we set $\lambda = 384$. Grover's algorithm takes $O(2^{128})$ quantum operations and uses approximately $384 \cdot 2^{128} \approx 2^{137}$ bits of space (at least). This is $\approx 2 \times 10^{25}$ *Petabytes*.

Hardness of Hashing

- As discussed last lecture, hardness of hashing reduces to finding collisions (e.g., violating collision resistance).

- Grover's Algorithm allows us to do this in $O(2^{\lambda/3})$ quantum time with $O(2^{\lambda/3})$ samples (so $\lambda \cdot 2^{\lambda/3}$-bits approximately).
  - E.g., for 128-bits of security, we set $\lambda = 384$. Grover's algorithm takes $O(2^{128})$ quantum operations and uses approximately $384 \cdot 2^{128} \approx 2^{137}$ bits of space (at least). This is $\approx 2 \times 10^{25}$ *Petabytes*.

Important Remark!

Hardness of Hashing

- As discussed last lecture, hardness of hashing reduces to finding collisions (e.g., violating collision resistance).

- Grover's Algorithm allows us to do this in $O(2^{\lambda/3})$ quantum time with $O(2^{\lambda/3})$ samples (so $\lambda \cdot 2^{\lambda/3}$-bits approximately).
  - E.g., for 128-bits of security, we set $\lambda = 384$. Grover's algorithm takes $O(2^{128})$ quantum operations and uses approximately $384 \cdot 2^{128} \approx 2^{137}$ bits of space (at least). This is $\approx 2 \times 10^{25}$ *Petabytes*.

Important Remark!

Remember, hash functions are secure *so long as* we model them as *random functions*!

NIST Candidate Hash Constructions

NIST Candidate Hash Constructions

- No KEM Candidates, but there are Digital Signature Candidates.

NIST Candidate Hash Constructions

- No KEM Candidates, but there are Digital Signature Candidates.
- **SPHINCS+** (standardized in FIPS 205 as *Stateless Hash-based Digital Signature Algorithm (SLH-DSA)*).

NIST Candidate Hash Constructions

- No KEM Candidates, but there are Digital Signature Candidates.

- **SPHINCS+** (standardized in FIPS 205 as *Stateless Hash-based Digital Signature Algorithm (SLH-DSA)*).
  - Security based only on the underlying hash function being a cryptographic hash.

NIST Candidate Hash Constructions

- No KEM Candidates, but there are Digital Signature Candidates.

- **SPHINCS+** (standardized in FIPS 205 as *Stateless Hash-based Digital Signature Algorithm (SLH-DSA)*).
  - Security based only on the underlying hash function being a cryptographic hash.
  - Uses a variety of hash-based techniques, including some one-time signature scheme, a few-time signature scheme, and Merkle trees.

NIST Candidate Hash Constructions

- No KEM Candidates, but there are Digital Signature Candidates.

- **SPHINCS+** (standardized in FIPS 205 as *Stateless Hash-based Digital Signature Algorithm (SLH-DSA)*).
  - Security based only on the underlying hash function being a cryptographic hash.
  - Uses a variety of hash-based techniques, including some one-time signature scheme, a few-time signature scheme, and Merkle trees.

- **PICNIC**: another DS scheme.

NIST Candidate Hash Constructions

- No KEM Candidates, but there are Digital Signature Candidates.

- **SPHINCS+** (standardized in FIPS 205 as *Stateless Hash-based Digital Signature Algorithm (SLH-DSA)*).
  - Security based only on the underlying hash function being a cryptographic hash.
  - Uses a variety of hash-based techniques, including some one-time signature scheme, a few-time signature scheme, and Merkle trees.

- **PICNIC**: another DS scheme.
  - Private key is random x, public key is (f, y) where f is a one-way function and $y = f(x)$.

NIST Candidate Hash Constructions

- No KEM Candidates, but there are Digital Signature Candidates.

- **SPHINCS+** (standardized in FIPS 205 as *Stateless Hash-based Digital Signature Algorithm (SLH-DSA)*).
  - Security based only on the underlying hash function being a cryptographic hash.
  - Uses a variety of hash-based techniques, including some one-time signature scheme, a few-time signature scheme, and Merkle trees.

- **PICNIC**: another DS scheme.
  - Private key is random x, public key is (f, y) where f is a one-way function and $y = f(x)$.
  - To sign message m, the *MPC-in-the-Head* paradigm is used to generate a non-interactive zero-knowledge proof showing that the signer "knows" x such that $y = f(x)$, and uses m to generate randomness used in the NIZK generation.

Multivariate-based PQ Cryptography

Multivariate-based PQ Cryptography

- Multivariate cryptography is based on the hardness of the *MQ problem* (which is **NP**-complete).

Multivariate-based PQ Cryptography

- Multivariate cryptography is based on the hardness of the *MQ problem* (which is **NP**-complete).

- The MQ problem (in)formally is as follows:

Multivariate-based PQ Cryptography

- Multivariate cryptography is based on the hardness of the *MQ problem* (which is **NP**-complete).

- The MQ problem (in)formally is as follows:
  - Given a system of m (random) quadratic (or larger degree) polynomials $p_1, \ldots, p_m$ in n variables $X_1, \ldots, X_n$ over $\mathbb{F}_q$ (denoted as $\mathcal{A}$), find $\mathbf{x} \in \mathbb{F}_q^n$ such that $p_i(\mathbf{x}) = 0$ for all i.

Multivariate-based PQ Cryptography

- Multivariate cryptography is based on the hardness of the *MQ problem* (which is **NP**-complete).

- The MQ problem (in)formally is as follows:
  - Given a system of m (random) quadratic (or larger degree) polynomials $p_1, \ldots, p_m$ in n variables $X_1, \ldots, X_n$ over $\mathbb{F}_q$ (denoted as $\mathcal{A}$), find $\mathbf{x} \in \mathbb{F}_q^n$ such that $p_i(\mathbf{x}) = 0$ for all i.

- Grover search only gives a quadratic speed-up for solving this problem (from $O(q^n)$ to $O(q^{n/2})$) when brute-force is the best-known algorithm for the instance $\mathcal{A}$.

Multivariate-based PQ Cryptography

- Multivariate cryptography is based on the hardness of the *MQ problem* (which is **NP**-complete).

- The MQ problem (in)formally is as follows:
  - Given a system of m (random) quadratic (or larger degree) polynomials $p_1, \ldots, p_m$ in n variables $X_1, \ldots, X_n$ over $\mathbb{F}_q$ (denoted as $\mathcal{A}$), find $\mathbf{x} \in \mathbb{F}_q^n$ such that $p_i(\mathbf{x}) = 0$ for all i.

- Grover search only gives a quadratic speed-up for solving this problem (from $O(q^n)$ to $O(q^{n/2})$) when brute-force is the best-known algorithm for the instance $\mathcal{A}$.
  - As little as 92 qbits can break MQ instances over $\mathbb{F}_2$ (MQ_2) at 80-bits of classical security.

Multivariate-based PQ Cryptography

- Multivariate cryptography is based on the hardness of the *MQ problem* (which is **NP**-complete).

- The MQ problem (in)formally is as follows:
  - Given a system of m (random) quadratic (or larger degree) polynomials $p_1, \ldots, p_m$ in n variables $X_1, \ldots, X_n$ over $\mathbb{F}_q$ (denoted as $\mathcal{A}$), find $\mathbf{x} \in \mathbb{F}_q^n$ such that $p_i(\mathbf{x}) = 0$ for all i.

- Grover search only gives a quadratic speed-up for solving this problem (from $O(q^n)$ to $O(q^{n/2})$) when brute-force is the best-known algorithm for the instance $\mathcal{A}$.
  - As little as 92 qbits can break MQ instances over $\mathbb{F}_2$ (MQ_2) at 80-bits of classical security.
  - There is a quantum Las-Vegas algorithm solving MQ_2 using $O(2^{0.462 \cdot n})$ quantum gates.

Multivariate-based PQ Cryptography

- There are also faster classical algorithms for solving MQ.

Multivariate-based PQ Cryptography

- There are also faster classical algorithms for solving MQ.
  - Faster brute-force using Gray code enumeration.

Multivariate-based PQ Cryptography

- There are also faster classical algorithms for solving MQ.
  - Faster brute-force using Gray code enumeration.
  - If $\mathcal{A}$ is under-determined:

Multivariate-based PQ Cryptography

- There are also faster classical algorithms for solving MQ.
  - Faster brute-force using Gray code enumeration.
  - If $\mathcal{A}$ is under-determined:
    - If $\mathbb{F}$ is a binary field and $n > m(m+1)$, the problem is solvable in polynomial time.

Multivariate-based PQ Cryptography

- There are also faster classical algorithms for solving MQ.
  - Faster brute-force using Gray code enumeration.
  - If $\mathcal{A}$ is under-determined:
    - If $\mathbb{F}$ is a binary field and $n > m(m+1)$, the problem is solvable in polynomial time.
    - If $n \geqslant m(m+3)/2$, there is an algorithm running in time

Multivariate-based PQ Cryptography

- There are also faster classical algorithms for solving MQ.
  - Faster brute-force using Gray code enumeration.
  - If $\mathcal{A}$ is under-determined:
    - If $\mathbb{F}$ is a binary field and $n > m(m+1)$, the problem is solvable in polynomial time.
    - If $n \geqslant m(m+3)/2$, there is an algorithm running in time

$$\begin{cases} O(n^k m) & q \equiv 0 \bmod 2 \\ O(2^m n^k m) & q \equiv 1 \bmod 2 \end{cases}.$$

Multivariate-based PQ Cryptography

- There are also faster classical algorithms for solving MQ.
  - Faster brute-force using Gray code enumeration.
  - If $\mathcal{A}$ is under-determined:
    - If $\mathbb{F}$ is a binary field and $n > m(m+1)$, the problem is solvable in polynomial time.
    - If $n \geqslant m(m+3)/2$, there is an algorithm running in time

$$\begin{cases} O(n^k m) & q \equiv 0 \bmod 2 \\ O(2^m n^k m) & q \equiv 1 \bmod 2 \end{cases}.$$

    - If $n \geqslant m(m+1)/2$, there is an algorithm running in time

Multivariate-based PQ Cryptography

- There are also faster classical algorithms for solving MQ.
  - Faster brute-force using Gray code enumeration.
  - If $\mathcal{A}$ is under-determined:
    - If $\mathbb{F}$ is a binary field and $n > m(m+1)$, the problem is solvable in polynomial time.
    - If $n \geqslant m(m+3)/2$, there is an algorithm running in time

$$\begin{cases} O(n^k m) & q \equiv 0 \bmod 2 \\ O(2^m n^k m) & q \equiv 1 \bmod 2 \end{cases}.$$

    - If $n \geqslant m(m+1)/2$, there is an algorithm running in time

$$q(\log(q))^2 \cdot \begin{cases} O(n^k m) & q \equiv 0 \bmod 2 \\ O(2^m n^k m) & q \equiv 1 \bmod 2 \end{cases}.$$

- There are also faster classical algorithms for solving MQ.
  - Faster brute-force using Gray code enumeration.
  - If $\mathcal{A}$ is under-determined:
    - If $\mathbb{F}$ is a binary field and $n > m(m+1)$, the problem is solvable in polynomial time.
    - If $n \geqslant m(m+3)/2$, there is an algorithm running in time

$$\begin{cases} O(n^k m) & q \equiv 0 \bmod 2 \\ O(2^m n^k m) & q \equiv 1 \bmod 2 \end{cases}.$$

    - If $n \geqslant m(m+1)/2$, there is an algorithm running in time

$$q(\log(q))^2 \cdot \begin{cases} O(n^k m) & q \equiv 0 \bmod 2 \\ O(2^m n^k m) & q \equiv 1 \bmod 2 \end{cases}.$$

    - For $k = \min(\lfloor m/2 \rfloor, \lfloor (n/2 - \sqrt{n/2})^{1/2} \rfloor)$ satisfying $m - 2k < 2k^2 \leqslant n - 2k$, then there is an algorithm which (on average) runs in time $O(2k\binom{n-k}{2} q^{m-k})$

Multivariate-based PQ Cryptography

- There are also faster classical algorithms for solving MQ.
  - Faster brute-force using Gray code enumeration.
  - If $\mathcal{A}$ is under-determined:
    - If $\mathbb{F}$ is a binary field and $n > m(m+1)$, the problem is solvable in polynomial time.
    - If $n \geqslant m(m+3)/2$, there is an algorithm running in time

$$\begin{cases} O(n^k m) & q \equiv 0 \bmod 2 \\ O(2^m n^k m) & q \equiv 1 \bmod 2 \end{cases}.$$

    - If $n \geqslant m(m+1)/2$, there is an algorithm running in time

$$q(\log(q))^2 \cdot \begin{cases} O(n^k m) & q \equiv 0 \bmod 2 \\ O(2^m n^k m) & q \equiv 1 \bmod 2 \end{cases}.$$

    - For $k = \min(\lfloor m/2 \rfloor, \lfloor (n/2 - \sqrt{n/2})^{1/2} \rfloor)$ satisfying $m - 2k < 2k^2 \leqslant n - 2k$, then there is an algorithm which (on average) runs in time $O(2k\binom{n-k}{2} q^{m-k})$
- See https://eprint.iacr.org/2022/708.pdf for more details.

Formula for MQ Crypto

Formula for MQ Crypto

- Crypto based on the MQ problem has the following general outline.

Formula for MQ Crypto

- Crypto based on the MQ problem has the following general outline.
  - **Public Key:** Set $\mathcal{A}$ of quadratic polynomials $p_1, \ldots, p_m \in \mathbb{F}[X_1, \ldots, X_n]$.

Formula for MQ Crypto

- Crypto based on the MQ problem has the following general outline.
  - **Public Key:** Set $\mathcal{A}$ of quadratic polynomials $p_1, \ldots, p_m \in \mathbb{F}[X_1, \ldots, X_n]$.
  - **Private Key:** (Knowledge of) a trapdoor to efficiently compute the system's solution.

Formula for MQ Crypto

- Crypto based on the MQ problem has the following general outline.
  - **Public Key:** Set $\mathcal{A}$ of quadratic polynomials $p_1, \ldots, p_m \in \mathbb{F}[X_1, \ldots, X_n]$.
  - **Private Key:** (Knowledge of) a trapdoor to efficiently compute the system's solution.
    - Trapdoor usually obtained by building $\mathcal{A}$ as $\mathcal{A} = \mathcal{T} \circ \mathcal{Q} \circ \mathcal{S}$, where $\mathcal{T}, \mathcal{S}$ are two affine maps and $\mathcal{Q}$ is an easily invertible quadratic map.

Formula for MQ Crypto

- Crypto based on the MQ problem has the following general outline.
  - **Public Key:** Set $\mathcal{A}$ of quadratic polynomials $p_1, \ldots, p_m \in \mathbb{F}[X_1, \ldots, X_n]$.
  - **Private Key:** (Knowledge of) a trapdoor to efficiently compute the system's solution.
    - Trapdoor usually obtained by building $\mathcal{A}$ as $\mathcal{A} = \mathcal{T} \circ \mathcal{Q} \circ \mathcal{S}$, where $\mathcal{T}, \mathcal{S}$ are two affine maps and $\mathcal{Q}$ is an easily invertible quadratic map.

- Encryption with MQ

Formula for MQ Crypto

- Crypto based on the MQ problem has the following general outline.
  - **Public Key:** Set $\mathcal{A}$ of quadratic polynomials $p_1, \ldots, p_m \in \mathbb{F}[X_1, \ldots, X_n]$.
  - **Private Key:** (Knowledge of) a trapdoor to efficiently compute the system's solution.
    - Trapdoor usually obtained by building $\mathcal{A}$ as $\mathcal{A} = \mathcal{T} \circ \mathcal{Q} \circ \mathcal{S}$, where $\mathcal{T}, \mathcal{S}$ are two affine maps and $\mathcal{Q}$ is an easily invertible quadratic map.

- Encryption with MQ
  - Need $m \geqslant n$ so every encryption has a unique decryption.

Formula for MQ Crypto

- Crypto based on the MQ problem has the following general outline.
  - **Public Key:** Set $\mathcal{A}$ of quadratic polynomials $p_1, \ldots, p_m \in \mathbb{F}[X_1, \ldots, X_n]$.
  - **Private Key:** (Knowledge of) a trapdoor to efficiently compute the system's solution.
    - Trapdoor usually obtained by building $\mathcal{A}$ as $\mathcal{A} = \mathcal{T} \circ \mathcal{Q} \circ \mathcal{S}$, where $\mathcal{T}, \mathcal{S}$ are two affine maps and $\mathcal{Q}$ is an easily invertible quadratic map.

- Encryption with MQ
  - Need $m \geqslant n$ so every encryption has a unique decryption.
  - Encryption of $\mathbf{m}$ given as $\mathbf{c} = \mathcal{A}(\mathbf{m})$.

Formula for MQ Crypto

- Crypto based on the MQ problem has the following general outline.
  - **Public Key:** Set $\mathcal{A}$ of quadratic polynomials $p_1, \ldots, p_m \in \mathbb{F}[X_1, \ldots, X_n]$.
  - **Private Key:** (Knowledge of) a trapdoor to efficiently compute the system's solution.
    - Trapdoor usually obtained by building $\mathcal{A}$ as $\mathcal{A} = \mathcal{T} \circ \mathcal{Q} \circ \mathcal{S}$, where $\mathcal{T}, \mathcal{S}$ are two affine maps and $\mathcal{Q}$ is an easily invertible quadratic map.

- Encryption with MQ
  - Need $m \geqslant n$ so every encryption has a unique decryption.
  - Encryption of $\mathbf{m}$ given as $\mathbf{c} = \mathcal{A}(\mathbf{m})$.
  - Decryption is $\mathbf{m} = \mathcal{T}^{-1}(\mathcal{Q}^{-1}(\mathcal{S}^{-1}(\mathbf{c})))$.

Formula for MQ Crypto

- Crypto based on the MQ problem has the following general outline.
  - **Public Key:** Set $\mathcal{A}$ of quadratic polynomials $p_1, \ldots, p_m \in \mathbb{F}[X_1, \ldots, X_n]$.
  - **Private Key:** (Knowledge of) a trapdoor to efficiently compute the system's solution.
    - Trapdoor usually obtained by building $\mathcal{A}$ as $\mathcal{A} = \mathcal{T} \circ \mathcal{Q} \circ \mathcal{S}$, where $\mathcal{T}, \mathcal{S}$ are two affine maps and $\mathcal{Q}$ is an easily invertible quadratic map.

- Encryption with MQ
  - Need $m \geqslant n$ so every encryption has a unique decryption.
  - Encryption of $\mathbf{m}$ given as $\mathbf{c} = \mathcal{A}(\mathbf{m})$.
  - Decryption is $\mathbf{m} = \mathcal{T}^{-1}(\mathcal{Q}^{-1}(\mathcal{S}^{-1}(\mathbf{c})))$.

Remark

Most/nearly all MQ-based encryption schemes are all broken because they do not sufficiently hide the trapdoor from the attacker.

Formula for MQ Crypto

- Crypto based on the MQ problem has the following general outline.
  - **Public Key:** Set $\mathcal{A}$ of quadratic polynomials $p_1, \ldots, p_m \in \mathbb{F}[X_1, \ldots, X_n]$.
  - **Private Key:** (Knowledge of) a trapdoor to efficiently compute the system's solution.
    - Trapdoor usually obtained by building $\mathcal{A}$ as $\mathcal{A} = \mathcal{T} \circ \mathcal{Q} \circ \mathcal{S}$, where $\mathcal{T}, \mathcal{S}$ are two affine maps and $\mathcal{Q}$ is an easily invertible quadratic map.

Formula for MQ Crypto

- Crypto based on the MQ problem has the following general outline.
  - **Public Key:** Set $\mathcal{A}$ of quadratic polynomials $p_1, \ldots, p_m \in \mathbb{F}[X_1, \ldots, X_n]$.
  - **Private Key:** (Knowledge of) a trapdoor to efficiently compute the system's solution.
    - Trapdoor usually obtained by building $\mathcal{A}$ as $\mathcal{A} = \mathcal{T} \circ \mathcal{Q} \circ \mathcal{S}$, where $\mathcal{T}, \mathcal{S}$ are two affine maps and $\mathcal{Q}$ is an easily invertible quadratic map.

- Digital Signatures with MQ

Formula for MQ Crypto

- Crypto based on the MQ problem has the following general outline.
  - **Public Key:** Set $\mathcal{A}$ of quadratic polynomials $p_1, \ldots, p_m \in \mathbb{F}[X_1, \ldots, X_n]$.
  - **Private Key:** (Knowledge of) a trapdoor to efficiently compute the system's solution.
    - Trapdoor usually obtained by building $\mathcal{A}$ as $\mathcal{A} = \mathcal{T} \circ \mathcal{Q} \circ \mathcal{S}$, where $\mathcal{T}, \mathcal{S}$ are two affine maps and $\mathcal{Q}$ is an easily invertible quadratic map.

- Digital Signatures with MQ
  - Need $m \leqslant n$ so one can sign any message.

Formula for MQ Crypto

- Crypto based on the MQ problem has the following general outline.
  - **Public Key:** Set $\mathcal{A}$ of quadratic polynomials $p_1, \ldots, p_m \in \mathbb{F}[X_1, \ldots, X_n]$.
  - **Private Key:** (Knowledge of) a trapdoor to efficiently compute the system's solution.
    - Trapdoor usually obtained by building $\mathcal{A}$ as $\mathcal{A} = \mathcal{T} \circ \mathcal{Q} \circ \mathcal{S}$, where $\mathcal{T}, \mathcal{S}$ are two affine maps and $\mathcal{Q}$ is an easily invertible quadratic map.

- Digital Signatures with MQ
  - Need $m \leqslant n$ so one can sign any message.
  - Signature of $\mathbf{m}$ given as $\sigma = \mathcal{T}^{-1}(\mathcal{Q}^{-1}(\mathcal{S}^{-1}(H(\mathbf{m}))))$, where H is a hash function.

Formula for MQ Crypto

- Crypto based on the MQ problem has the following general outline.
  - **Public Key:** Set $\mathcal{A}$ of quadratic polynomials $p_1, \ldots, p_m \in \mathbb{F}[X_1, \ldots, X_n]$.
  - **Private Key:** (Knowledge of) a trapdoor to efficiently compute the system's solution.
    - Trapdoor usually obtained by building $\mathcal{A}$ as $\mathcal{A} = \mathcal{T} \circ \mathcal{Q} \circ \mathcal{S}$, where $\mathcal{T}, \mathcal{S}$ are two affine maps and $\mathcal{Q}$ is an easily invertible quadratic map.

- Digital Signatures with MQ
  - Need $m \leqslant n$ so one can sign any message.
  - Signature of $\mathbf{m}$ given as $\sigma = \mathcal{T}^{-1}(\mathcal{Q}^{-1}(\mathcal{S}^{-1}(H(\mathbf{m}))))$, where H is a hash function.
  - Signature verification checks if $H(m) = \mathcal{A}(\sigma)$.

Formula for MQ Crypto

- Crypto based on the MQ problem has the following general outline.
  - **Public Key:** Set $\mathcal{A}$ of quadratic polynomials $p_1, \ldots, p_m \in \mathbb{F}[X_1, \ldots, X_n]$.
  - **Private Key:** (Knowledge of) a trapdoor to efficiently compute the system's solution.
    - Trapdoor usually obtained by building $\mathcal{A}$ as $\mathcal{A} = \mathcal{T} \circ \mathcal{Q} \circ \mathcal{S}$, where $\mathcal{T}, \mathcal{S}$ are two affine maps and $\mathcal{Q}$ is an easily invertible quadratic map.

- Digital Signatures with MQ
  - Need $m \leqslant n$ so one can sign any message.
  - Signature of $\mathbf{m}$ given as $\sigma = \mathcal{T}^{-1}(\mathcal{Q}^{-1}(\mathcal{S}^{-1}(H(\mathbf{m}))))$, where H is a hash function.
  - Signature verification checks if $H(m) = \mathcal{A}(\sigma)$.

Remark

MQ schemes have the shortest signatures among all PQ schemes, at the cost of large public keys (10 to 100 kB).

NIST Candidate MQ Constructions

NIST Candidate MQ Constructions

- No KEM, but a few signatures.

NIST Candidate MQ Constructions

- No KEM, but a few signatures.
- **GeMSS** (A Great Multivariate Short Signature).

NIST Candidate MQ Constructions

- No KEM, but a few signatures.

- **GeMSS** (A Great Multivariate Short Signature).
  - Alternate 3rd round candidate of NIST competition.

NIST Candidate MQ Constructions

- No KEM, but a few signatures.

- **GeMSS** (A Great Multivariate Short Signature).
  - Alternate 3rd round candidate of NIST competition.
  - Built using so-called *Hidden Field Equations* (HFE) cryptosystem (1996), with some so-called minus and vinegar modifiers (denoted as HFEv-).

NIST Candidate MQ Constructions

- No KEM, but a few signatures.

- **GeMSS** (A Great Multivariate Short Signature).
  - Alternate 3rd round candidate of NIST competition.
  - Built using so-called *Hidden Field Equations* (HFE) cryptosystem (1996), with some so-called minus and vinegar modifiers (denoted as HFEv-).
  - I could not find a clean and simple explanation of this scheme!

NIST Candidate MQ Constructions

- No KEM, but a few signatures.

- **GeMSS** (A Great Multivariate Short Signature).
  - Alternate 3rd round candidate of NIST competition.
  - Built using so-called *Hidden Field Equations* (HFE) cryptosystem (1996), with some so-called minus and vinegar modifiers (denoted as HFEv-).
  - I could not find a clean and simple explanation of this scheme!

- **Rainbow Signature Scheme**

NIST Candidate MQ Constructions

- No KEM, but a few signatures.

- **GeMSS** (A Great Multivariate Short Signature).
  - Alternate 3rd round candidate of NIST competition.
  - Built using so-called *Hidden Field Equations* (HFE) cryptosystem (1996), with some so-called minus and vinegar modifiers (denoted as HFEv-).
  - I could not find a clean and simple explanation of this scheme!

- **Rainbow Signature Scheme**
  - Another MQ based scheme.

NIST Candidate MQ Constructions

- No KEM, but a few signatures.

- **GeMSS** (A Great Multivariate Short Signature).
  - Alternate 3rd round candidate of NIST competition.
  - Built using so-called *Hidden Field Equations* (HFE) cryptosystem (1996), with some so-called minus and vinegar modifiers (denoted as HFEv-).
  - I could not find a clean and simple explanation of this scheme!

- **Rainbow Signature Scheme**
  - Another MQ based scheme.
  - Works similar/like the outline we described previously.

NIST Candidate MQ Constructions

- No KEM, but a few signatures.

- **GeMSS** (A Great Multivariate Short Signature).
  - Alternate 3rd round candidate of NIST competition.
  - Built using so-called *Hidden Field Equations* (HFE) cryptosystem (1996), with some so-called minus and vinegar modifiers (denoted as HFEv-).
  - I could not find a clean and simple explanation of this scheme!

- **Rainbow Signature Scheme**
  - Another MQ based scheme.
  - Works similar/like the outline we described previously.
  - Selected as a NIST Finalist in 2022!

NIST Candidate MQ Constructions

- No KEM, but a few signatures.

- **GeMSS** (A Great Multivariate Short Signature).
  - Alternate 3rd round candidate of NIST competition.
  - Built using so-called *Hidden Field Equations* (HFE) cryptosystem (1996), with some so-called minus and vinegar modifiers (denoted as HFEv-).
  - I could not find a clean and simple explanation of this scheme!

- **Rainbow Signature Scheme**
  - Another MQ based scheme.
  - Works similar/like the outline we described previously.
  - Selected as a NIST Finalist in 2022!

- 4 schemes in NIST's Round 2 of the Additional Signatures call.

NIST Candidate MQ Constructions

- No KEM, but a few signatures.

- **GeMSS** (A Great Multivariate Short Signature).
  - Alternate 3rd round candidate of NIST competition.
  - Built using so-called *Hidden Field Equations* (HFE) cryptosystem (1996), with some so-called minus and vinegar modifiers (denoted as HFEv-).
  - I could not find a clean and simple explanation of this scheme!

- **Rainbow Signature Scheme**
  - Another MQ based scheme.
  - Works similar/like the outline we described previously.
  - Selected as a NIST Finalist in 2022!

- 4 schemes in NIST's Round 2 of the Additional Signatures call.
  - MAYO, QR-UOV, SNOVA, UOV.

NIST Candidate MQ Constructions

- No KEM, but a few signatures.

- **GeMSS** (A Great Multivariate Short Signature).
  - Alternate 3rd round candidate of NIST competition.
  - Built using so-called *Hidden Field Equations* (HFE) cryptosystem (1996), with some so-called minus and vinegar modifiers (denoted as HFEv-).
  - I could not find a clean and simple explanation of this scheme!

- **Rainbow Signature Scheme**
  - Another MQ based scheme.
  - Works similar/like the outline we described previously.
  - Selected as a NIST Finalist in 2022!

- 4 schemes in NIST's Round 2 of the Additional Signatures call.
  - MAYO, QR-UOV, SNOVA, UOV.
  - All based on a so-called *unbalanced oil and vinegar* approach.

Rainbow is not Secure

Breaking Rainbow Takes a Weekend on a Laptop

Ward Beullens

IBM Research, Zurich, Switzerland
wbe@zurich.ibm.com

Abstract. This work introduces new key recovery attacks against the Rainbow signature scheme, which is one of the three finalist signature schemes still in the NIST Post-Quantum Cryptography standardization project. The new attacks outperform previously known attacks for all the parameter sets submitted to NIST and make a key-recovery practical for the SL 1 parameters. Concretely, given a Rainbow public key for the SL 1 parameters of the second-round submission, our attack returns the corresponding secret key after on average 53 hours (one weekend) of computation time on a standard laptop.

Code-based PQ Cryptography

Code-based PQ Cryptography

- Code-based cryptography is based on the **NP**-hard/complete problem of decoding random linear codes.

Code-based PQ Cryptography

- Code-based cryptography is based on the **NP**-hard/complete problem of decoding random linear codes.

- There are two equivalent formulations of this hard problem.

Code-based PQ Cryptography

- Code-based cryptography is based on the **NP**-hard/complete problem of decoding random linear codes.

- There are two equivalent formulations of this hard problem.
  1. Let $G \xleftarrow{\$} \mathbb{F}^{k \times n}$ with $k \leqslant n$ be a random generator matrix of a code C.

Code-based PQ Cryptography

- Code-based cryptography is based on the **NP**-hard/complete problem of decoding random linear codes.

- There are two equivalent formulations of this hard problem.
  1. Let $G \xleftarrow{\$} \mathbb{F}^{k \times n}$ with $k \leqslant n$ be a random generator matrix of a code C. For error parameter t, the following *decoding problem* is assumed to be hard:

Code-based PQ Cryptography

- Code-based cryptography is based on the **NP**-hard/complete problem of decoding random linear codes.

- There are two equivalent formulations of this hard problem.
  1. Let $G \xleftarrow{\$} \mathbb{F}^{k \times n}$ with $k \leqslant n$ be a random generator matrix of a code C. For error parameter t, the following *decoding problem* is assumed to be hard:
     - Sample $\mathbf{m} \xleftarrow{\$} \mathbb{F}^k$ and $\mathbf{e} \xleftarrow{\$} \mathbb{F}^n$ of weight $\leqslant t$.

Code-based PQ Cryptography

- Code-based cryptography is based on the **NP**-hard/complete problem of decoding random linear codes.

- There are two equivalent formulations of this hard problem.
  1. Let $G \xleftarrow{\$} \mathbb{F}^{k \times n}$ with $k \leqslant n$ be a random generator matrix of a code C. For error parameter t, the following *decoding problem* is assumed to be hard:
     - Sample $\mathbf{m} \xleftarrow{\$} \mathbb{F}^k$ and $\mathbf{e} \xleftarrow{\$} \mathbb{F}^n$ of weight $\leqslant t$.
     - Output $\mathbf{z} = \mathbf{m} \cdot G + \mathbf{e}$.

Code-based PQ Cryptography

- Code-based cryptography is based on the **NP**-hard/complete problem of decoding random linear codes.

- There are two equivalent formulations of this hard problem.
  1. Let $G \xleftarrow{\$} \mathbb{F}^{k \times n}$ with $k \leqslant n$ be a random generator matrix of a code C. For error parameter t, the following *decoding problem* is assumed to be hard:
     - Sample $\mathbf{m} \xleftarrow{\$} \mathbb{F}^k$ and $\mathbf{e} \xleftarrow{\$} \mathbb{F}^n$ of weight $\leqslant t$.
     - Output $\mathbf{z} = \mathbf{m} \cdot G + \mathbf{e}$.
     - Given $\mathbf{z}$, find $\mathbf{m}$.

Code-based PQ Cryptography

- Code-based cryptography is based on the **NP**-hard/complete problem of decoding random linear codes.

- There are two equivalent formulations of this hard problem.
  1. Let $G \xleftarrow{\$} \mathbb{F}^{k \times n}$ with $k \leqslant n$ be a random generator matrix of a code C. For error parameter t, the following *decoding problem* is assumed to be hard:
     - Sample $\mathbf{m} \xleftarrow{\$} \mathbb{F}^k$ and $\mathbf{e} \xleftarrow{\$} \mathbb{F}^n$ of weight $\leqslant t$.
     - Output $\mathbf{z} = \mathbf{m} \cdot G + \mathbf{e}$.
     - Given $\mathbf{z}$, find $\mathbf{m}$.
  2. Let $G \xleftarrow{\$} \mathbb{F}^{k \times n}$ with $k \leqslant n$ and let $H \in \mathbb{F}^{(n-k) \times n}$ be the parity check matrix.

Code-based PQ Cryptography

- Code-based cryptography is based on the **NP**-hard/complete problem of decoding random linear codes.

- There are two equivalent formulations of this hard problem.
  1. Let $G \xleftarrow{\$} \mathbb{F}^{k \times n}$ with $k \leqslant n$ be a random generator matrix of a code C. For error parameter t, the following *decoding problem* is assumed to be hard:
     - Sample $\mathbf{m} \xleftarrow{\$} \mathbb{F}^k$ and $\mathbf{e} \xleftarrow{\$} \mathbb{F}^n$ of weight $\leqslant t$.
     - Output $\mathbf{z} = \mathbf{m} \cdot G + \mathbf{e}$.
     - Given $\mathbf{z}$, find $\mathbf{m}$.
  2. Let $G \xleftarrow{\$} \mathbb{F}^{k \times n}$ with $k \leqslant n$ and let $H \in \mathbb{F}^{(n-k) \times n}$ be the parity check matrix. For weight parameter w, the following *syndrome decoding problem* is assumed to be hard:

Code-based PQ Cryptography

- Code-based cryptography is based on the **NP**-hard/complete problem of decoding random linear codes.

- There are two equivalent formulations of this hard problem.
  1. Let $G \xleftarrow{\$} \mathbb{F}^{k \times n}$ with $k \leqslant n$ be a random generator matrix of a code C. For error parameter t, the following *decoding problem* is assumed to be hard:
     - Sample $\mathbf{m} \xleftarrow{\$} \mathbb{F}^k$ and $\mathbf{e} \xleftarrow{\$} \mathbb{F}^n$ of weight $\leqslant t$.
     - Output $\mathbf{z} = \mathbf{m} \cdot G + \mathbf{e}$.
     - Given $\mathbf{z}$, find $\mathbf{m}$.
  2. Let $G \xleftarrow{\$} \mathbb{F}^{k \times n}$ with $k \leqslant n$ and let $H \in \mathbb{F}^{(n-k) \times n}$ be the parity check matrix. For weight parameter w, the following *syndrome decoding problem* is assumed to be hard:
     - Sample and output $\mathbf{s} \xleftarrow{\$} \mathbb{F}^{n-k}$.

Code-based PQ Cryptography

- Code-based cryptography is based on the **NP**-hard/complete problem of decoding random linear codes.

- There are two equivalent formulations of this hard problem.
  1. Let $G \xleftarrow{\$} \mathbb{F}^{k \times n}$ with $k \leqslant n$ be a random generator matrix of a code C. For error parameter t, the following *decoding problem* is assumed to be hard:
     - Sample $\mathbf{m} \xleftarrow{\$} \mathbb{F}^k$ and $\mathbf{e} \xleftarrow{\$} \mathbb{F}^n$ of weight $\leqslant t$.
     - Output $\mathbf{z} = \mathbf{m} \cdot G + \mathbf{e}$.
     - Given $\mathbf{z}$, find $\mathbf{m}$.
  2. Let $G \xleftarrow{\$} \mathbb{F}^{k \times n}$ with $k \leqslant n$ and let $H \in \mathbb{F}^{(n-k) \times n}$ be the parity check matrix. For weight parameter w, the following *syndrome decoding problem* is assumed to be hard:
     - Sample and output $\mathbf{s} \xleftarrow{\$} \mathbb{F}^{n-k}$.
     - Given H and $\mathbf{s}$, find $\mathbf{e} \in \mathbb{F}^n$ of weight $\leqslant w$ such that $H\mathbf{e}^\top = \mathbf{s}^\top$.

Code-based PQ Cryptography

Code-based PQ Cryptography

- McEliece in 1978 proposed a PKE scheme based on (binary) Goppa codes.

Code-based PQ Cryptography

- McEliece in 1978 proposed a PKE scheme based on (binary) Goppa codes.
  - Let $G \in \mathbb{F}_2^k \to \mathbb{F}_2^n$ be the generator matrix of a *random* binary Goppa code, let $S \in \mathbb{F}^k \to \mathbb{F}^k$ be some random non-singular matrix, and let $P \colon \mathbb{F}^n \to \mathbb{F}^n$ be a random permutation matrix.

Code-based PQ Cryptography

- McEliece in 1978 proposed a PKE scheme based on (binary) Goppa codes.
  - Let $G \in \mathbb{F}_2^k \to \mathbb{F}_2^n$ be the generator matrix of a *random* binary Goppa code, let $S \in \mathbb{F}^k \to \mathbb{F}^k$ be some random non-singular matrix, and let $P \colon \mathbb{F}^n \to \mathbb{F}^n$ be a random permutation matrix.
  - The public key is the matrix $G' = SGP$ (which is also a binary Goppa code) and error parameter t.

Code-based PQ Cryptography

- McEliece in 1978 proposed a PKE scheme based on (binary) Goppa codes.
  - Let $G \in \mathbb{F}_2^k \to \mathbb{F}_2^n$ be the generator matrix of a *random* binary Goppa code, let $S \in \mathbb{F}^k \to \mathbb{F}^k$ be some random non-singular matrix, and let $P \colon \mathbb{F}^n \to \mathbb{F}^n$ be a random permutation matrix.
  - The public key is the matrix $G' = SGP$ (which is also a binary Goppa code) and error parameter t.
  - The private key is (S, P, Dec), where Dec is the decoding algorithm for G.

Code-based PQ Cryptography

- McEliece in 1978 proposed a PKE scheme based on (binary) Goppa codes.
  - Let $G \in \mathbb{F}_2^k \to \mathbb{F}_2^n$ be the generator matrix of a *random* binary Goppa code, let $S \in \mathbb{F}^k \to \mathbb{F}^k$ be some random non-singular matrix, and let $P \colon \mathbb{F}^n \to \mathbb{F}^n$ be a random permutation matrix.
  - The public key is the matrix $G' = SGP$ (which is also a binary Goppa code) and error parameter t.
  - The private key is (S, P, Dec), where Dec is the decoding algorithm for G.
  - $\mathbf{c} = \mathsf{Enc}_{(G',t)}(\mathbf{m}) := \mathbf{m}G' + \mathbf{z}$, where $\mathbf{z} \xleftarrow{\$} \mathbb{F}^n$ such that $\mathsf{wt}(\mathbf{z}) = t$.

Code-based PQ Cryptography

- McEliece in 1978 proposed a PKE scheme based on (binary) Goppa codes.
  - Let $G \in \mathbb{F}_2^k \to \mathbb{F}_2^n$ be the generator matrix of a *random* binary Goppa code, let $S \in \mathbb{F}^k \to \mathbb{F}^k$ be some random non-singular matrix, and let $P \colon \mathbb{F}^n \to \mathbb{F}^n$ be a random permutation matrix.
  - The public key is the matrix $G' = SGP$ (which is also a binary Goppa code) and error parameter t.
  - The private key is (S, P, Dec), where Dec is the decoding algorithm for G.
  - $\mathbf{c} = \mathsf{Enc}_{(G',t)}(\mathbf{m}) := \mathbf{m}G' + \mathbf{z}$, where $\mathbf{z} \xleftarrow{\$} \mathbb{F}^n$ such that $\mathsf{wt}(\mathbf{z}) = t$.
  - $\mathbf{m}' = \mathsf{Decr}_{(S,P,\mathsf{Dec})}(\mathbf{c}) := \mathsf{Dec}(\mathbf{c}P^{-1})S^{-1}$.

Code-based PQ Cryptography

- Courtois, Finiasz, and Sendrier in 2001 showed how to build digital signatures using McEliecee encryption.

Code-based PQ Cryptography

- Courtois, Finiasz, and Sendrier in 2001 showed how to build digital signatures using McEliecee encryption.
  - Idea is like FDH signatures for RSA.

Code-based PQ Cryptography

- Courtois, Finiasz, and Sendrier in 2001 showed how to build digital signatures using McEliecee encryption.
  - Idea is like FDH signatures for RSA.
  - To sign a message $\mathbf{m}$, first we hash it to obtain $\mathbf{y} = H(\mathbf{m})$.

Code-based PQ Cryptography

- Courtois, Finiasz, and Sendrier in 2001 showed how to build digital signatures using McEliecee encryption.
  - Idea is like FDH signatures for RSA.
  - To sign a message $\mathbf{m}$, first we hash it to obtain $\mathbf{y} = H(\mathbf{m})$.
  - Next, try to *decrypt* $\mathbf{y}$ via McEliecee to obtain σ.

Code-based PQ Cryptography

- Courtois, Finiasz, and Sendrier in 2001 showed how to build digital signatures using McEliecee encryption.
  - Idea is like FDH signatures for RSA.
  - To sign a message $\mathbf{m}$, first we hash it to obtain $\mathbf{y} = H(\mathbf{m})$.
  - Next, try to *decrypt* $\mathbf{y}$ via McEliecee to obtain σ.
  - Output $(\mathbf{m}, \sigma)$.

Code-based PQ Cryptography

- Courtois, Finiasz, and Sendrier in 2001 showed how to build digital signatures using McEliecee encryption.
  - Idea is like FDH signatures for RSA.
  - To sign a message $\mathbf{m}$, first we hash it to obtain $\mathbf{y} = H(\mathbf{m})$.
  - Next, try to *decrypt* $\mathbf{y}$ via McEliecee to obtain σ.
  - Output $(\mathbf{m}, \sigma)$.
  - Verification of $(\mathbf{m}, \sigma)$ checks if $\mathbf{Enc}(\sigma) = H(\mathbf{m})$.

Code-based PQ Cryptography

- Courtois, Finiasz, and Sendrier in 2001 showed how to build digital signatures using McEliecee encryption.
  - Idea is like FDH signatures for RSA.
  - To sign a message $\mathbf{m}$, first we hash it to obtain $\mathbf{y} = H(\mathbf{m})$.
  - Next, try to *decrypt* $\mathbf{y}$ via McEliecee to obtain σ.
  - Output $(\mathbf{m}, \sigma)$.
  - Verification of $(\mathbf{m}, \sigma)$ checks if $\mathbf{Enc}(\sigma) = H(\mathbf{m})$.

These types of signatures have many issues.

Code-based PQ Cryptography

- Courtois, Finiasz, and Sendrier in 2001 showed how to build digital signatures using McEliecee encryption.
  - Idea is like FDH signatures for RSA.
  - To sign a message $\mathbf{m}$, first we hash it to obtain $\mathbf{y} = H(\mathbf{m})$.
  - Next, try to *decrypt* $\mathbf{y}$ via McEliecee to obtain σ.
  - Output $(\mathbf{m}, \sigma)$.
  - Verification of $(\mathbf{m}, \sigma)$ checks if $\mathbf{Enc}(\sigma) = H(\mathbf{m})$.

These types of signatures have many issues.

- What happens if $\mathbf{y}$ cannot be decrypted? $y_i = H(m \| r_i)$ $H(m \| i)$

Code-based PQ Cryptography

- Courtois, Finiasz, and Sendrier in 2001 showed how to build digital signatures using McEliecee encryption.
  - Idea is like FDH signatures for RSA.
  - To sign a message $\mathbf{m}$, first we hash it to obtain $\mathbf{y} = H(\mathbf{m})$.
  - Next, try to *decrypt* $\mathbf{y}$ via McEliecee to obtain σ.
  - Output $(\mathbf{m}, \sigma)$.
  - Verification of $(\mathbf{m}, \sigma)$ checks if $\mathbf{Enc}(\sigma) = H(\mathbf{m})$.

These types of signatures have many issues.

- What happens if $\mathbf{y}$ cannot be decrypted?
- How long until you find a decryptable hash $\mathbf{y}_i = H(\mathbf{m} \| \mathbf{r}_i)$?

Code-based PQ Cryptography

- Courtois, Finiasz, and Sendrier in 2001 showed how to build digital signatures using McEliecee encryption.
  - Idea is like FDH signatures for RSA.
  - To sign a message $\mathbf{m}$, first we hash it to obtain $\mathbf{y} = H(\mathbf{m})$.
  - Next, try to *decrypt* $\mathbf{y}$ via McEliecee to obtain σ.
  - Output $(\mathbf{m}, \sigma)$.
  - Verification of $(\mathbf{m}, \sigma)$ checks if $\mathbf{Enc}(\sigma) = H(\mathbf{m})$.

These types of signatures have many issues.

- What happens if $\mathbf{y}$ cannot be decrypted?
- How long until you find a decryptable hash $\mathbf{y}_i = H(\mathbf{m} \| \mathbf{r}_i)$?
- Public/verification keys are huge even for 80-bits of security.

Code-based PQ Cryptography

- Baldi, Chiaraluce, and Santini in 2021 proposed a different code-based signature using the so-called restricted syndrome decoding problem.

Code-based PQ Cryptography

- Baldi, Chiaraluce, and Santini in 2021 proposed a different code-based signature using the so-called restricted syndrome decoding problem.
  - Informally, for parity-check matrix H and target $\mathbf{s}$, find $\mathbf{e}$ of weight at most t with values in $\{\pm i\}_{i=0}^{\gamma}$ such that $H\mathbf{e}^\top = \mathbf{s}$.

Code-based PQ Cryptography

- Baldi, Chiaraluce, and Santini in 2021 proposed a different code-based signature using the so-called restricted syndrome decoding problem.
  - Informally, for parity-check matrix H and target $\mathbf{s}$, find $\mathbf{e}$ of weight at most t with values in $\{\pm i\}_{i=0}^{\gamma}$ such that $H\mathbf{e}^\top = \mathbf{s}$.
  - Idea: suitable random matrix E (from a non-uniform distribution) is the private key, and the public key is $S = EH^\top$, where $H = [I \| P]$ for random matrix P.

Code-based PQ Cryptography

- Baldi, Chiaraluce, and Santini in 2021 proposed a different code-based signature using the so-called restricted syndrome decoding problem.
  - Informally, for parity-check matrix H and target $\mathbf{s}$, find $\mathbf{e}$ of weight at most t with values in $\{\pm i\}_{i=0}^{\gamma}$ such that $H\mathbf{e}^\top = \mathbf{s}$.
  - Idea: suitable random matrix E (from a non-uniform distribution) is the private key, and the public key is $S = EH^\top$, where $H = [I \| P]$ for random matrix P.
  - To sign a message $\mathbf{m}$, sample some appropriate $\mathbf{y}$, obtain $\mathbf{s_y} = \mathbf{y}H^\top$, compute $\mathbf{c} = H(\mathbf{m}, \mathbf{s_y})$, and compute (via rejection sampling) $\mathbf{z} = \mathbf{c}E + \mathbf{y}$.

Code-based PQ Cryptography

- Baldi, Chiaraluce, and Santini in 2021 proposed a different code-based signature using the so-called restricted syndrome decoding problem.
  - Informally, for parity-check matrix H and target $\mathbf{s}$, find $\mathbf{e}$ of weight at most t with values in $\{\pm i\}_{i=0}^{\gamma}$ such that $H\mathbf{e}^\top = \mathbf{s}$.
  - Idea: suitable random matrix E (from a non-uniform distribution) is the private key, and the public key is $S = EH^\top$, where $H = [I \| P]$ for random matrix P.
  - To sign a message $\mathbf{m}$, sample some appropriate $\mathbf{y}$, obtain $\mathbf{s_y} = \mathbf{y}H^\top$, compute $\mathbf{c} = H(\mathbf{m}, \mathbf{s_y})$, and compute (via rejection sampling) $\mathbf{z} = \mathbf{c}E + \mathbf{y}$. Output $\sigma = (\mathbf{z}, \mathbf{c})$.

Code-based PQ Cryptography

- Baldi, Chiaraluce, and Santini in 2021 proposed a different code-based signature using the so-called restricted syndrome decoding problem.
  - Informally, for parity-check matrix H and target $\mathbf{s}$, find $\mathbf{e}$ of weight at most t with values in $\{\pm i\}_{i=0}^{\gamma}$ such that $H\mathbf{e}^\top = \mathbf{s}$.
  - Idea: suitable random matrix E (from a non-uniform distribution) is the private key, and the public key is $S = EH^\top$, where $H = [I \| P]$ for random matrix P.
  - To sign a message $\mathbf{m}$, sample some appropriate $\mathbf{y}$, obtain $\mathbf{s_y} = \mathbf{y}H^\top$, compute $\mathbf{c} = H(\mathbf{m}, \mathbf{s_y})$, and compute (via rejection sampling) $\mathbf{z} = \mathbf{c}E + \mathbf{y}$. Output $\sigma = (\mathbf{z}, \mathbf{c})$.
  - To verify $\sigma = (\mathbf{z}, \mathbf{c})$, verify $\mathbf{z}$ is correctly distributed, then compute $\mathbf{s_y} = \mathbf{z}H^\top - \mathbf{c}S$, and check if $\mathbf{c} = H(m, \mathbf{s_y})$.

Code-based PQ Cryptography

- Baldi, Chiaraluce, and Santini in 2021 proposed a different code-based signature using the so-called restricted syndrome decoding problem.
  - Informally, for parity-check matrix H and target $\mathbf{s}$, find $\mathbf{e}$ of weight at most t with values in $\{\pm i\}_{i=0}^{\gamma}$ such that $H\mathbf{e}^\top = \mathbf{s}$.
  - Idea: suitable random matrix E (from a non-uniform distribution) is the private key, and the public key is $S = EH^\top$, where $H = [I \| P]$ for random matrix P.
  - To sign a message $\mathbf{m}$, sample some appropriate $\mathbf{y}$, obtain $\mathbf{s_y} = \mathbf{y}H^\top$, compute $\mathbf{c} = H(\mathbf{m}, \mathbf{s_y})$, and compute (via rejection sampling) $\mathbf{z} = \mathbf{c}E + \mathbf{y}$. Output $\sigma = (\mathbf{z}, \mathbf{c})$.
  - To verify $\sigma = (\mathbf{z}, \mathbf{c})$, verify $\mathbf{z}$ is correctly distributed, then compute $\mathbf{s_y} = \mathbf{z}H^\top - \mathbf{c}S$, and check if $\mathbf{c} = H(m, \mathbf{s_y})$.

- See https://eprint.iacr.org/2019/544.pdf for an extensive overview of code-based digital signatures.

NIST Candidate Code-based Cryptography

NIST Candidate Code-based Cryptography

- PKE/KEM: Classic McEliece (https://classic.mceliece.org/)

NIST Candidate Code-based Cryptography

- PKE/KEM: Classic McEliece (https://classic.mceliece.org/)
  - Currently a 4th round candidate.

NIST Candidate Code-based Cryptography

- PKE/KEM: Classic McEliece (https://classic.mceliece.org/)
  - Currently a 4th round candidate.
  - Very well-studied, has withstood almost 50 years of analysis and attacks.

NIST Candidate Code-based Cryptography

- PKE/KEM: Classic McEliece (https://classic.mceliece.org/)
  - Currently a 4th round candidate.
  - Very well-studied, has withstood almost 50 years of analysis and attacks.
  - Main drawback: public key sizes.

NIST Candidate Code-based Cryptography

- PKE/KEM: Classic McEliece (https://classic.mceliece.org/)
  - Currently a 4th round candidate.
  - Very well-studied, has withstood almost 50 years of analysis and attacks.
  - Main drawback: public key sizes.

- PKE/KEM: BIKE (Bit Flipping KEM, https://bikesuite.org/)

NIST Candidate Code-based Cryptography

- PKE/KEM: Classic McEliece (https://classic.mceliece.org/)
  - Currently a 4th round candidate.
  - Very well-studied, has withstood almost 50 years of analysis and attacks.
  - Main drawback: public key sizes.

- PKE/KEM: BIKE (Bit Flipping KEM, https://bikesuite.org/)
  - Another 4th round candidate.

NIST Candidate Code-based Cryptography

- PKE/KEM: Classic McEliece (https://classic.mceliece.org/)
  - Currently a 4th round candidate.
  - Very well-studied, has withstood almost 50 years of analysis and attacks.
  - Main drawback: public key sizes.

- PKE/KEM: BIKE (Bit Flipping KEM, https://bikesuite.org/)
  - Another 4th round candidate.
  - Uses a variant of McEliece called Niederreiter (using a parity-check matrix instead of the generator matrix) with Quasi-cyclic Moderate Density Parity Check codes.

NIST Candidate Code-based Cryptography

- PKE/KEM: Classic McEliece (https://classic.mceliece.org/)
  - Currently a 4th round candidate.
  - Very well-studied, has withstood almost 50 years of analysis and attacks.
  - Main drawback: public key sizes.

- PKE/KEM: BIKE (Bit Flipping KEM, https://bikesuite.org/)
  - Another 4th round candidate.
  - Uses a variant of McEliece called Niederreiter (using a parity-check matrix instead of the generator matrix) with Quasi-cyclic Moderate Density Parity Check codes.
  - Combine with the Fujisaki-Okamoto transform to upgrade CPA security to CCA Security.

NIST Candidate Code-based Cryptography

- PKE/KEM: HQC (Hamming Quasi-Cyclic KEM, https://pqc-hqc.org/)

NIST Candidate Code-based Cryptography

- PKE/KEM: HQC (Hamming Quasi-Cyclic KEM, https://pqc-hqc.org/)
  - Selected as the KEM backup algorithm in 2025.

NIST Candidate Code-based Cryptography

- PKE/KEM: HQC (Hamming Quasi-Cyclic KEM, https://pqc-hqc.org/)
  - Selected as the KEM backup algorithm in 2025.
  - Security based on the quasi-cyclic syndrome decoding problem + Fujisaki-Okamoto transform.

NIST Candidate Code-based Cryptography

- PKE/KEM: HQC (Hamming Quasi-Cyclic KEM, https://pqc-hqc.org/)
  - Selected as the KEM backup algorithm in 2025.
  - Security based on the quasi-cyclic syndrome decoding problem + Fujisaki-Okamoto transform.
  - Private keys: $\mathbf{x}, \mathbf{y} \xleftarrow{\$} \mathbb{F}_2^n$ of weight w.

NIST Candidate Code-based Cryptography

- PKE/KEM: HQC (Hamming Quasi-Cyclic KEM, https://pqc-hqc.org/)
  - Selected as the KEM backup algorithm in 2025.
  - Security based on the quasi-cyclic syndrome decoding problem + Fujisaki-Okamoto transform.
  - Private keys: $\mathbf{x}, \mathbf{y} \xleftarrow{\$} \mathbb{F}_2^n$ of weight w.
  - Public keys: $\mathbf{h} \xleftarrow{\$} \mathbb{F}_2^n$, $\mathbf{s} = \mathbf{x} + \mathbf{h} \odot \mathbf{y}$ (point-wise product).

NIST Candidate Code-based Cryptography

- PKE/KEM: HQC (Hamming Quasi-Cyclic KEM, https://pqc-hqc.org/)
  - Selected as the KEM backup algorithm in 2025.
  - Security based on the quasi-cyclic syndrome decoding problem + Fujisaki-Okamoto transform.
  - Private keys: $\mathbf{x}, \mathbf{y} \xleftarrow{\$} \mathbb{F}_2^n$ of weight w.
  - Public keys: $\mathbf{h} \xleftarrow{\$} \mathbb{F}_2^n$, $\mathbf{s} = \mathbf{x} + \mathbf{h} \odot \mathbf{y}$ (point-wise product).
  - Encryption of $\mathbf{m}$: sample appropriately weighted $\mathbf{r}_1, \mathbf{r}_2, \mathbf{e}$, set $\mathbf{c}_1 = \mathbf{r}_1 + \mathbf{h} \odot \mathbf{r}_2$ and $\mathbf{c}_2 = \mathsf{Enc}(\mathbf{m}) + \mathbf{s} \odot \mathbf{r}_2 + \mathbf{e}$.

encoder Enc

NIST Candidate Code-based Cryptography

- PKE/KEM: HQC (Hamming Quasi-Cyclic KEM, https://pqc-hqc.org/)
  - Selected as the KEM backup algorithm in 2025.
  - Security based on the quasi-cyclic syndrome decoding problem + Fujisaki-Okamoto transform.
  - Private keys: $\mathbf{x}, \mathbf{y} \xleftarrow{\$} \mathbb{F}_2^n$ of weight w.
  - Public keys: $\mathbf{h} \xleftarrow{\$} \mathbb{F}_2^n$, $\mathbf{s} = \mathbf{x} + \mathbf{h} \odot \mathbf{y}$ (point-wise product).
  - Encryption of $\mathbf{m}$: sample appropriately weighted $\mathbf{r}_1, \mathbf{r}_2, \mathbf{e}$, set $\mathbf{c}_1 = \mathbf{r}_1 + \mathbf{h} \odot \mathbf{r}_2$ and $\mathbf{c}_2 = \mathsf{Enc}(\mathbf{m}) + \mathbf{s} \odot \mathbf{r}_2 + \mathbf{e}$.
  - Decryption: $\mathbf{m} = \mathsf{Dec}(\mathbf{c}_2 - \mathbf{c}_1 \odot \mathbf{y})$.

NIST Candidate Code-based Cryptography

NIST Candidate Code-based Cryptography

- Digital Signatures: CROSS
  (https://www.cross-crypto.com/cross.html)

NIST Candidate Code-based Cryptography

- Digital Signatures: CROSS (https://www.cross-crypto.com/cross.html)
  - NIST Round 2 candidate of the call for additional digital signatures.

NIST Candidate Code-based Cryptography

- Digital Signatures: CROSS
  (https://www.cross-crypto.com/cross.html)
  - NIST Round 2 candidate of the call for additional digital signatures.
  - Based on the RSDP problem of Baldi, Chiaraluce, and Santini.

NIST Candidate Code-based Cryptography

- Digital Signatures: CROSS
  (https://www.cross-crypto.com/cross.html)
  - NIST Round 2 candidate of the call for additional digital signatures.
  - Based on the RSDP problem of Baldi, Chiaraluce, and Santini.
  - Utilizes a ZKP + Fiat-Shamir to construct the signature.

NIST Candidate Code-based Cryptography

- Digital Signatures: CROSS
  (https://www.cross-crypto.com/cross.html)
  - NIST Round 2 candidate of the call for additional digital signatures.
  - Based on the RSDP problem of Baldi, Chiaraluce, and Santini.
  - Utilizes a ZKP + Fiat-Shamir to construct the signature.

- Digital Signatures: LESS
  (https://www.less-project.com/home/)

NIST Candidate Code-based Cryptography

- Digital Signatures: CROSS
  (https://www.cross-crypto.com/cross.html)
  - NIST Round 2 candidate of the call for additional digital signatures.
  - Based on the RSDP problem of Baldi, Chiaraluce, and Santini.
  - Utilizes a ZKP + Fiat-Shamir to construct the signature.

- Digital Signatures: LESS
  (https://www.less-project.com/home/)
  - Based on the hardness of finding linear isometries between vector spaces (and hence linear codes).

NIST Candidate Code-based Cryptography

- Digital Signatures: CROSS
  (https://www.cross-crypto.com/cross.html)
  - NIST Round 2 candidate of the call for additional digital signatures.
  - Based on the RSDP problem of Baldi, Chiaraluce, and Santini.
  - Utilizes a ZKP + Fiat-Shamir to construct the signature.

- Digital Signatures: LESS
  (https://www.less-project.com/home/)
  - Based on the hardness of finding linear isometries between vector spaces (and hence linear codes).
  - Signature constructed similarly with ZKP + Fiat-Shamir.

Lattice-based PQ Cryptography

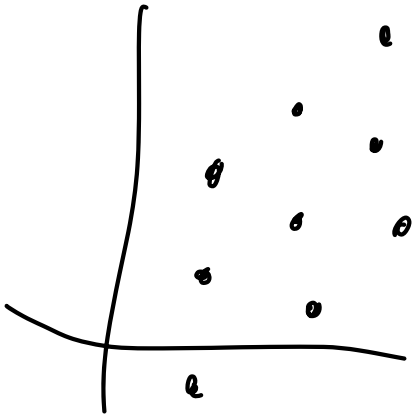
Lattice-based PQ Cryptography

- By far the most popular and studied approach to post-quantum cryptosystems.

Lattice-based PQ Cryptography

- By far the most popular and studied approach to post-quantum cryptosystems.

- A lattice $L \subset \mathbb{R}^n$ is defined via a basis $\{\mathbf{b}_i\}_{i=1}^n$ of $\mathbb{R}^n$ as $L = \{\sum_i a_i \mathbf{b}_i : a_i \in \mathbb{Z}\}$.



Lattice-based PQ Cryptography

- By far the most popular and studied approach to post-quantum cryptosystems.
- A lattice $L \subset \mathbb{R}^n$ is defined via a basis $\{\mathbf{b}_i\}_{i=1}^n$ of $\mathbb{R}^n$ as $L = \{\sum_i a_i \mathbf{b}_i : a_i \in \mathbb{Z}\}$.
- A large variety of hardness assumptions are used in practice.

Lattice-based PQ Cryptography

- By far the most popular and studied approach to post-quantum cryptosystems.

- A lattice $L \subset \mathbb{R}^n$ is defined via a basis $\{\mathbf{b}_i\}_{i=1}^n$ of $\mathbb{R}^n$ as $L = \{\sum_i a_i \mathbf{b}_i : a_i \in \mathbb{Z}\}$.

- A large variety of hardness assumptions are used in practice.
  - The Shortest Vector Problem (**SVP**).

Lattice-based PQ Cryptography

- By far the most popular and studied approach to post-quantum cryptosystems.

- A lattice $L \subset \mathbb{R}^n$ is defined via a basis $\{\mathbf{b}_i\}_{i=1}^n$ of $\mathbb{R}^n$ as $L = \{\sum_i a_i \mathbf{b}_i : a_i \in \mathbb{Z}\}$.

- A large variety of hardness assumptions are used in practice.
  - The Shortest Vector Problem (**SVP**).
  - The Gap **SVP** problem.

Lattice-based PQ Cryptography

- By far the most popular and studied approach to post-quantum cryptosystems.

- A lattice $L \subset \mathbb{R}^n$ is defined via a basis $\{\mathbf{b}_i\}_{i=1}^n$ of $\mathbb{R}^n$ as $L = \{\sum_i a_i \mathbf{b}_i : a_i \in \mathbb{Z}\}$.

- A large variety of hardness assumptions are used in practice.
  - The Shortest Vector Problem (**SVP**).
  - The Gap **SVP** problem.
  - The Closest Vector Problem (**CVP**) and Gap **CVP**.

Lattice-based PQ Cryptography

- By far the most popular and studied approach to post-quantum cryptosystems.

- A lattice $L \subset \mathbb{R}^n$ is defined via a basis $\{\mathbf{b}_i\}_{i=1}^n$ of $\mathbb{R}^n$ as $L = \{\sum_i a_i \mathbf{b}_i : a_i \in \mathbb{Z}\}$.

- A large variety of hardness assumptions are used in practice.
  - The Shortest Vector Problem (**SVP**).
  - The Gap **SVP** problem.
  - The Closest Vector Problem (**CVP**) and Gap **CVP**.
  - The Short Integer Solution (**SIS**), ring-**SIS**, and module-**SIS** problems.

Lattice-based PQ Cryptography

- By far the most popular and studied approach to post-quantum cryptosystems.

- A lattice $L \subset \mathbb{R}^n$ is defined via a basis $\{\mathbf{b}_i\}_{i=1}^n$ of $\mathbb{R}^n$ as $L = \{\sum_i a_i \mathbf{b}_i : a_i \in \mathbb{Z}\}$.

- A large variety of hardness assumptions are used in practice.
  - The Shortest Vector Problem (**SVP**).
  - The Gap **SVP** problem.
  - The Closest Vector Problem (**CVP**) and Gap **CVP**.
  - The Short Integer Solution (**SIS**), ring-**SIS**, and module-**SIS** problems.
  - Learning Parity with Noise (**LPN**; equivalent to syndrome decoding over $\mathbb{F}_2$).

Lattice-based PQ Cryptography

- By far the most popular and studied approach to post-quantum cryptosystems.

- A lattice $L \subset \mathbb{R}^n$ is defined via a basis $\{\mathbf{b}_i\}_{i=1}^n$ of $\mathbb{R}^n$ as $L = \{\sum_i a_i \mathbf{b}_i : a_i \in \mathbb{Z}\}$.

- A large variety of hardness assumptions are used in practice.
  - The Shortest Vector Problem (**SVP**).
  - The Gap **SVP** problem.
  - The Closest Vector Problem (**CVP**) and Gap **CVP**.
  - The Short Integer Solution (**SIS**), ring-**SIS**, and module-**SIS** problems.
  - Learning Parity with Noise (**LPN**; equivalent to syndrome decoding over $\mathbb{F}_2$).
  - Learning with Errors (**LWE**), ring-**LWE**, Learning with Rounding (**LWR**),...

Lattice-based PQ Cryptography

Lattice-based PQ Cryptography

- We know how to construct the following from lattice-based hardness assumptions:

Lattice-based PQ Cryptography

- We know how to construct the following from lattice-based hardness assumptions:
  - Encryption;

Lattice-based PQ Cryptography

- We know how to construct the following from lattice-based hardness assumptions:
  - Encryption;
  - Fully Homomorphic Encryption and other advanced encryption schemes;

Lattice-based PQ Cryptography

- We know how to construct the following from lattice-based hardness assumptions:
  - Encryption;
  - Fully Homomorphic Encryption and other advanced encryption schemes;
  - Hash Functions;

Lattice-based PQ Cryptography

- We know how to construct the following from lattice-based hardness assumptions:
  - Encryption;
  - Fully Homomorphic Encryption and other advanced encryption schemes;
  - Hash Functions;
  - Key Exchange/KEM;

Lattice-based PQ Cryptography

- We know how to construct the following from lattice-based hardness assumptions:
  - Encryption;
  - Fully Homomorphic Encryption and other advanced encryption schemes;
  - Hash Functions;
  - Key Exchange/KEM;
  - Digital Signatures;

Lattice-based PQ Cryptography

- We know how to construct the following from lattice-based hardness assumptions:
  - Encryption;
  - Fully Homomorphic Encryption and other advanced encryption schemes;
  - Hash Functions;
  - Key Exchange/KEM;
  - Digital Signatures;
  - Polynomial Commitments and Vector Commitments;

Lattice-based PQ Cryptography

- We know how to construct the following from lattice-based hardness assumptions:
  - Encryption;
  - Fully Homomorphic Encryption and other advanced encryption schemes;
  - Hash Functions;
  - Key Exchange/KEM;
  - Digital Signatures;
  - Polynomial Commitments and Vector Commitments;
  - $i\mathcal{O}$ (almost).

Lattice-based PQ Cryptography

- We know how to construct the following from lattice-based hardness assumptions:
  - Encryption;
  - Fully Homomorphic Encryption and other advanced encryption schemes;
  - Hash Functions;
  - Key Exchange/KEM;
  - Digital Signatures;
  - Polynomial Commitments and Vector Commitments;
  - $i\mathcal{O}$ (almost).

- 3 of the 5 NIST winners are lattice-based schemes.

Lattice-based PQ Cryptography

- We know how to construct the following from lattice-based hardness assumptions:
  - Encryption;
  - Fully Homomorphic Encryption and other advanced encryption schemes;
  - Hash Functions;
  - Key Exchange/KEM;
  - Digital Signatures;
  - Polynomial Commitments and Vector Commitments;
  - $i\mathcal{O}$ (almost).

- 3 of the 5 NIST winners are lattice-based schemes.
  - One PKE/KEM, two Signatures.

NIST Candidate Lattice-based Cryptography

- PKE/KEM: CRYSTALS-Kyber (renamed ML-KEM, https://en.wikipedia.org/wiki/ML-KEM).

NIST Candidate Lattice-based Cryptography

- PKE/KEM: CRYSTALS-Kyber (renamed ML-KEM, https://en.wikipedia.org/wiki/ML-KEM).
  - NIST standardized in FIPS 203 (2024).

NIST Candidate Lattice-based Cryptography

- PKE/KEM: CRYSTALS-Kyber (renamed ML-KEM, https://en.wikipedia.org/wiki/ML-KEM).
  - NIST standardized in FIPS 203 (2024).
  - Based on the module-**LWE** assumption.

NIST Candidate Lattice-based Cryptography

- PKE/KEM: CRYSTALS-Kyber (renamed ML-KEM, https://en.wikipedia.org/wiki/ML-KEM).
  - NIST standardized in FIPS 203 (2024).
  - Based on the module-**LWE** assumption.
  - Currently implemented in:

NIST Candidate Lattice-based Cryptography

- PKE/KEM: CRYSTALS-Kyber (renamed ML-KEM, https://en.wikipedia.org/wiki/ML-KEM).
  - NIST standardized in FIPS 203 (2024).
  - Based on the module-**LWE** assumption.
  - Currently implemented in:
    - The Signal Protocol;
    - OpenSSL;
    - wolfSSL;
    - libOQS;
    - IAIK-JCE (Java crypto library);
    - Libgcrypt with GNU Privacy Guard.

NIST Candidate Lattice-based Cryptography

- Signature: CRYSTALS-Dilithium (renamed ML-DSA, https://pq-crystals.org/dilithium/index.shtml).

NIST Candidate Lattice-based Cryptography

- Signature: CRYSTALS-Dilithium (renamed ML-DSA, https://pq-crystals.org/dilithium/index.shtml).
  - NIST standardized in FIPS 204 (2024).

NIST Candidate Lattice-based Cryptography

- Signature: CRYSTALS-Dilithium (renamed ML-DSA, https://pq-crystals.org/dilithium/index.shtml).
  - NIST standardized in FIPS 204 (2024).
  - Signature cousin of Kyber.

NIST Candidate Lattice-based Cryptography

- Signature: CRYSTALS-Dilithium (renamed ML-DSA, https://pq-crystals.org/dilithium/index.shtml).
  - NIST standardized in FIPS 204 (2024).
  - Signature cousin of Kyber.

- Signature: FALCON (Fast-Fourier Lattice-based Compact Signatures over NTRU, https://falcon-sign.info/).

NIST Candidate Lattice-based Cryptography

- Signature: CRYSTALS-Dilithium (renamed ML-DSA, https://pq-crystals.org/dilithium/index.shtml).
  - NIST standardized in FIPS 204 (2024).
  - Signature cousin of Kyber.

- Signature: FALCON (Fast-Fourier Lattice-based Compact Signatures over NTRU, https://falcon-sign.info/).
  - To be standardized in FIPS 206.

NIST Candidate Lattice-based Cryptography

- Signature: CRYSTALS-Dilithium (renamed ML-DSA, https://pq-crystals.org/dilithium/index.shtml).
  - NIST standardized in FIPS 204 (2024).
  - Signature cousin of Kyber.

- Signature: FALCON (Fast-Fourier Lattice-based Compact Signatures over NTRU, https://falcon-sign.info/).
  - To be standardized in FIPS 206.
  - Based on the NTRU trapdoor lattice, with hardness based on the **SVP** problem.

Next Time: The Learning-with-Errors Assumption and KEM from LWE