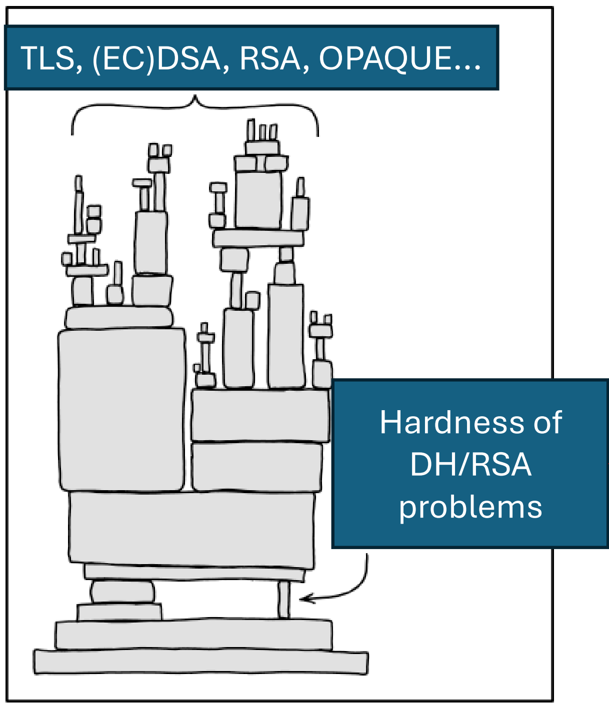
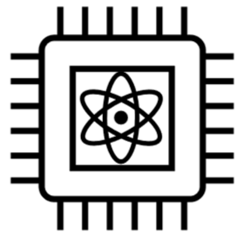
Cryptography Engineering

- Lecture 11 (Jan 29, 2025)
- Today's notes:
  - Background on Post-quantum Cryptography
  - Introduction to Lattice-based Cryptography
  - From the Pre-quantum World to the Post-quantum World

Post-quantum Cryptography

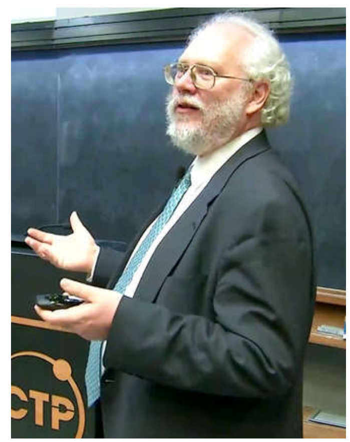


Source: xkcd/2347 and Nadia Heninger’s talk in PKC2024



Recent progress in Quantum Computers/Mechanisms…

Shor’s algorithm



Peter Williston Shor
(image from Wikipedia)

Post-quantum Cryptography

- Post-Quantum Cryptography
  - Cryptographic algorithms run on classical computers, but **remain secure against future quantum computers**...
- Still follow the methodology of modern cryptography: **Assumptions** => Schemes.

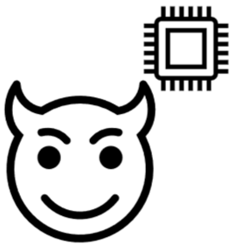
- **What assumptions can we rely on now?**
  - **Lattices**
  - Isogeny (of Elliptic Curves)
  - Code-based
  - ...

- NIST PQC Standardization (https://csrc.nist.gov/Projects/post-quantum-cryptography/news)

Impact on Cryptography

- In the **pre**-quantum world...

- Symmetric-key cryptography
  - Hash functions: SHA2, SHA3,...
  - Symmetric-key (authenticated) encryption: AES, AES-GCM...
  - KDF, MAC, PRNG,...

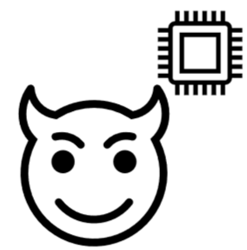


Impact on Cryptography

- In the **pre**-quantum world…

- Symmetric-key cryptography
  - Hash functions: SHA2, SHA3,…
  - Symmetric-key (authenticated) encryption: AES, AES-GCM…
  - KDF, MAC, PRNG,…

- Basis of confidence: **Extensively studied, publicly reviewed, …**
  - (Or we could say that they themselves are assumptions…)

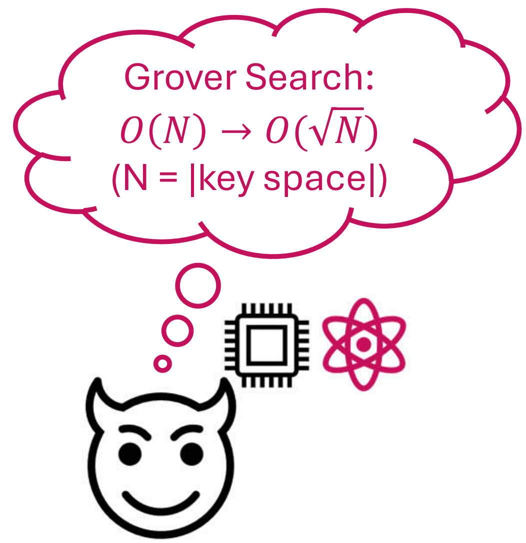


Impact on Cryptography

- In the **post**-quantum world…

- Symmetric-key cryptography
  - Hash functions: SHA2, SHA3,…
  - Symmetric-key (authenticated) encryption: AES, AES-GCM…
  - KDF, MAC, PRNG,…

- Basis of confidence: **Extensively studied, publicly reviewed, …**

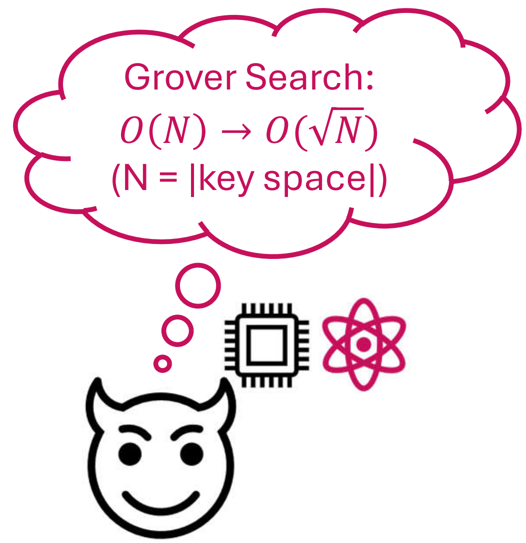


Impact on Cryptography

- In the **post**-quantum world…

- Symmetric-key cryptography
  - Hash functions: SHA2, SHA3,…
  - Symmetric-key (authenticated) encryption: AES, AES-GCM…
  - KDF, MAC, PRNG,…

- Basis of confidence: **Extensively studied, publicly reviewed, …**
- Solution: **Double the key size**… (not always true)

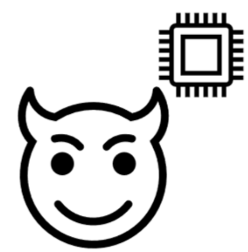


Impact on Cryptography

- In the **pre**-quantum world…

- Public-key cryptography
  - Key exchange: (EC)DHKE, TLS, …
  - Public-key encryption: ElGamal encryption, DHIES, …
  - Signature: DSA, RSA, …
  - …

- **Basis of confidence:**
  - Provable security (e.g., rigorous security proofs, …)
  - Well-studied and publicly reviewed hardness assumptions
  - **Classical assumptions: DH (from discrete-log), RSA (from factoring), …**



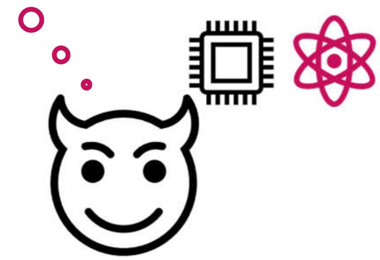
Impact on Cryptography

- In the **post**-quantum world…

- Public-key cryptography
  - Key exchange: (EC)DHKE, TLS, …
  - Public-key encryption: ElGamal encryption, DHIES, …
  - Signature: DSA, RSA, …
  - …

Quantum Fourier transform (QFT):
solve DLOG and Factoring.
$N^{O(1)} \rightarrow \boldsymbol{O(log(N))},$
where N = group/ modulus size

- **Basis of confidence:**
  - Provable security (e.g., rigorous security proofs, …)
  - Well-studied and publicly reviewed hardness assumptions
  - **Classical assumptions: DH (from discrete-log), RSA (from factoring), …**

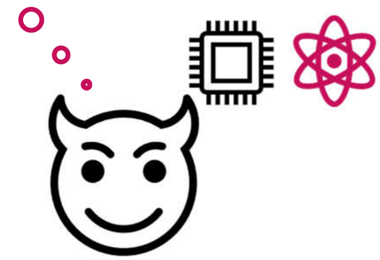


Impact on Cryptography

- In the **post**-quantum world…

- Public-key cryptography
  - Key exchange: (EC)DHKE, TLS, …
  - Public-key encryption: ElGamal encryption, DHIES, …
  - Signature: DSA, RSA, …
  - …

Quantum Fourier transform (QFT):
solve DLOG and Factoring.
$N^{O(1)} \rightarrow \boldsymbol{O(log(N))},$
where N = group/ modulus size



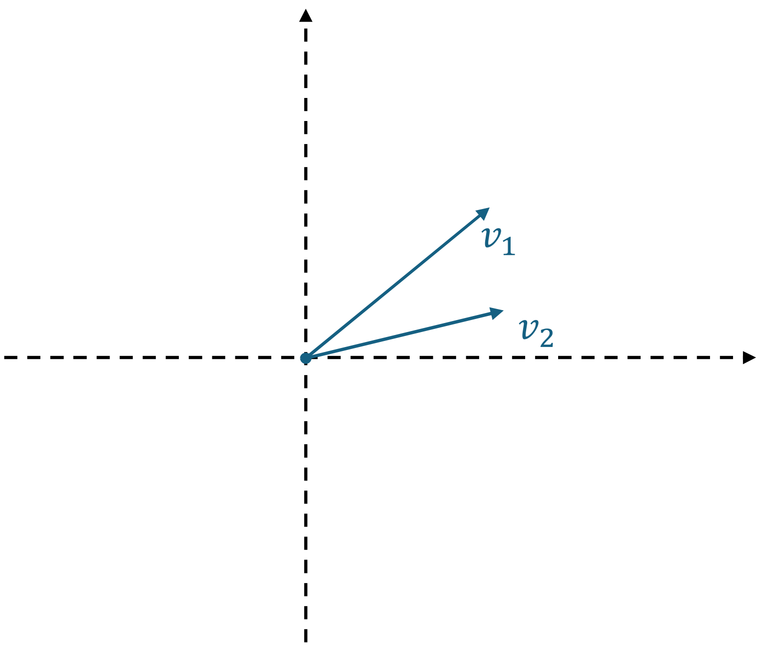
- **Basis of confidence:**
  - Provable security (e.g., rigorous security proofs, …)
  - Well-studied and publicly reviewed hardness assumptions
  - ~~Classical assumptions: DH (from discrete-log), RSA (from factoring), …~~
  - **New assumptions are needed.**

Post-quantum Assumptions

- Assumptions that are believed to be **quantum-secure:**
  - Lattice-based
  - Isogeny-based
  - Code-based
  - …

Post-quantum Assumptions

- A brief introduction of **lattice-based** assumptions

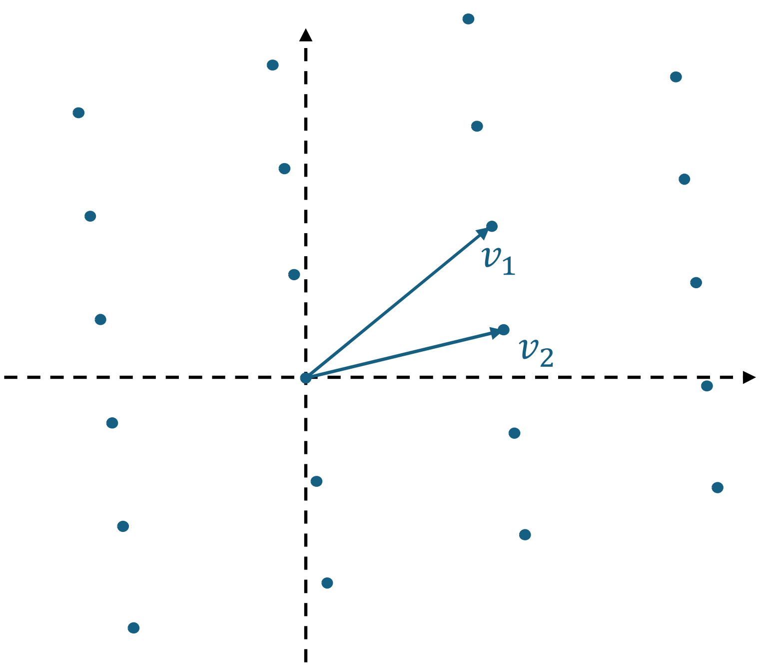


- **Integer combinations**
  - "Grid" structure

- Basis: $\{\boldsymbol{v_1}, \boldsymbol{v_2}\} \in \mathbb{R}^2$

Post-quantum Assumptions

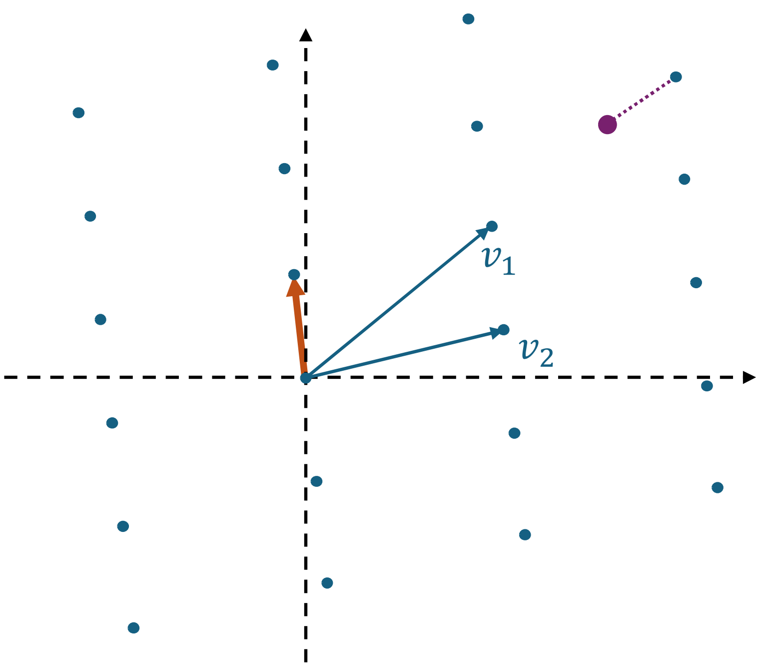
- A brief introduction of **lattice-based** assumptions



- **Integer combinations**
  - "Grid" structure
- Basis: $\{\boldsymbol{v_1}, \boldsymbol{v_2}\} \in \mathbb{R}^2$
- $\boldsymbol{\mathcal{L}}(\boldsymbol{v_1}, \boldsymbol{v_2}) = \{x \cdot \boldsymbol{v_1} + y \cdot \boldsymbol{v_2} |\ x, y \in \mathbb{Z}\}$

Post-quantum Assumptions

- A brief introduction of **lattice-based** assumptions

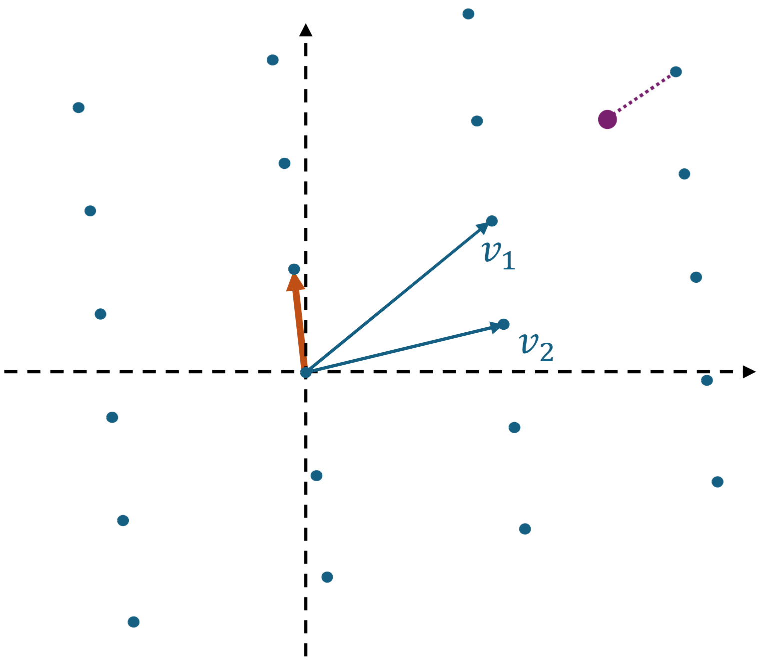


- **Integer combinations**
  - "Grid" structure
- Basis: $\{\boldsymbol{v_1}, \boldsymbol{v_2}\} \in \mathbb{R}^2$
- $\boldsymbol{\mathcal{L}}(\boldsymbol{v_1}, \boldsymbol{v_2}) = \{x \cdot \boldsymbol{v_1} + y \cdot \boldsymbol{v_2} \mid x, y \in \mathbb{Z}\}$

- **Shortest vector problem (SVP)**
- **Closest vector problem (CVP)**

Post-quantum Assumptions

- A brief introduction of **lattice-based** assumptions



- **Integer combinations**
  - “Grid” structure
- Basis: $\{\boldsymbol{v_1}, \boldsymbol{v_2}\} \in \mathbb{R}^2$
- $\boldsymbol{\mathcal{L}}(\boldsymbol{v_1}, \boldsymbol{v_2}) = \{x \cdot \boldsymbol{v_1} + y \cdot \boldsymbol{v_2} \mid x, y \in \mathbb{Z}\}$

- Shortest vector problem (SVP)
- Closest vector problem (CVP)

- **Both are easy in dimension 2**

  // Lagrange’s lattice reduction algorithm

Post-quantum Assumptions

- Case n > 2: Let $\{\boldsymbol{v_1}, \boldsymbol{v_2}, \ldots, \boldsymbol{v_n}\}$ be a basis, define $\boldsymbol{\mathcal{L}}(\boldsymbol{v_1}, \ldots, \boldsymbol{v_n}) = \{x_1 \cdot \boldsymbol{v_1} + \cdots + x_n \cdot \boldsymbol{v_n} | \, x_1, \ldots, x_n \in \mathbb{Z}\}$

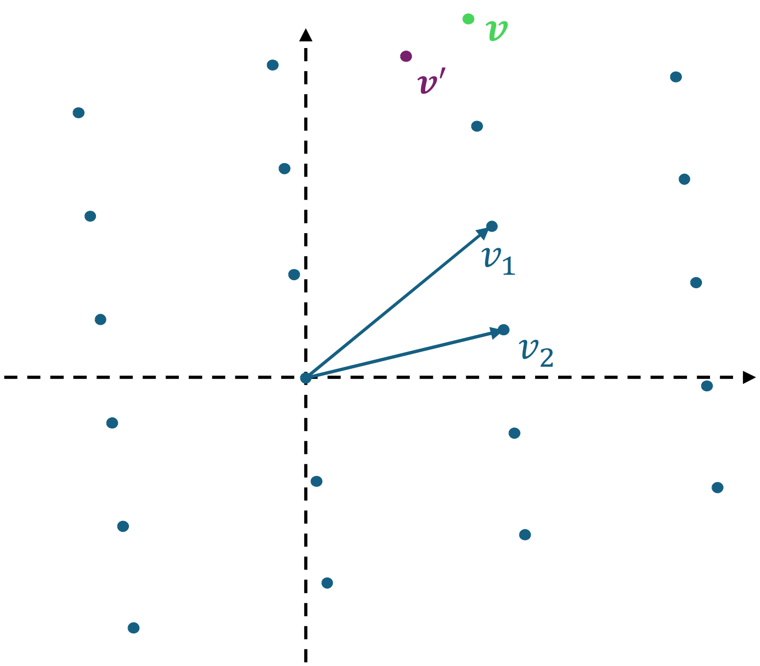
- Computational hardness of SVP/CVP over $\boldsymbol{\mathcal{L}}$: Depends on $\boldsymbol{n}$ and the **quality** of the given basis (informally)

- No efficient algorithms have been found for SVP and CVP
  - Some lattice reduction algorithms(e.g., given a lattice basis, outputs a "good" basis): LLL, BKZ, ...
  - The CVP problem can be **NP-hard** in the "worst case"
  - **SVP/CVP assumptions**: They cannot be solved in quantum polynomial time...

- Other "cryptographically-friendly" assumptions derived from SVP/CVP:
  - **Learning-with-error (LWE)**, Short-integer-solution (SIS), ...

Post-quantum Assumptions

- A very brief introduction about LWE

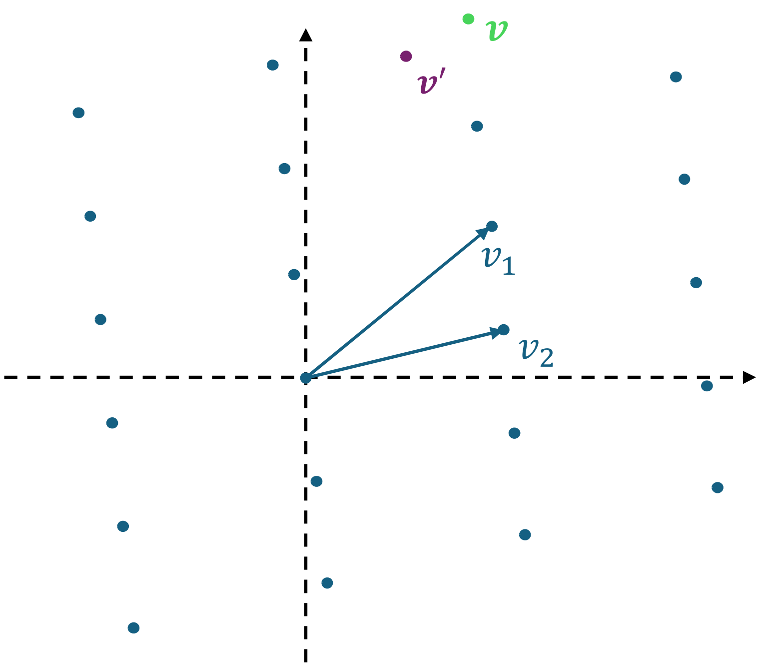


- $\boldsymbol{A} = \{\boldsymbol{v_1}, \boldsymbol{v_2}\} \in \mathbb{R}^2, \boldsymbol{\mathcal{L}}(\boldsymbol{A}) = \{x \cdot \boldsymbol{v_1} + y \cdot \boldsymbol{v_2} \mid x, y \in \mathbb{Z}\}$
- Let $\boldsymbol{s} = (x^*, y^*)$ be a random secret vector.
- $\boldsymbol{v} = \boldsymbol{As} = x^* \cdot \boldsymbol{v_1} + y^* \cdot \boldsymbol{v_2}$
- Let χ be some distribution of "short" vectors
- Let $\boldsymbol{e} \leftarrow \chi, \boldsymbol{v'} = \boldsymbol{v} + \boldsymbol{e}$

UNIKASSEL VERSITÄT

Post-quantum Assumptions

- A very brief introduction about LWE



- $\boldsymbol{A} = \{\boldsymbol{v_1}, \boldsymbol{v_2}\} \in \mathbb{R}^2, \boldsymbol{\mathcal{L}}(\boldsymbol{A}) = \{x \cdot \boldsymbol{v_1} + y \cdot \boldsymbol{v_2} \mid x, y \in \mathbb{Z}\}$
- Let $\boldsymbol{s} = (x^*, y^*)$ be a random secret vector.
- $\boldsymbol{v} = \boldsymbol{As} = x^* \cdot \boldsymbol{v_1} + y^* \cdot \boldsymbol{v_2}$
- Let χ be some distribution of "short" vectors
- Let $\boldsymbol{e} \leftarrow \chi, \boldsymbol{v'} = \boldsymbol{v} + \boldsymbol{e}$

- **LWE assumption (very informally!):**
  - The vector $\boldsymbol{v'} = \boldsymbol{As} + \boldsymbol{e}$ "looks" like a random vector
  - (i.e., it is generated uniformly at random, rather than by using the vector $\boldsymbol{s}$ and the distribution.
  - Does not hold if n = 2...
  - ...but for n > 2: **LWE** $\approx_{\text{hardness}}$ **SVP**
- Concrete hardness depends on: **Dimensions**, the **quality of the basis**, and the **error distribution**...

Post-quantum Assumptions

- Different types of lattices:
  - Lattices with indefinite points: Lattices over $\mathbb{R}^n, \mathbb{Z}^n$, …
  - Integer lattices mod q: Lattices over $\mathbb{Z}_q^n$, … **(LWE, SIS, …)**
  - Ideal lattices: Lattices based on ideals in rings…**(Ring-LWE, Ring-SIS, NTRU, …)**
  - Module lattices: **Module-LWE, Module-SIS, …**

- Ring/Module lattices:
  - Higher computational efficiency
  - Shorter key pairs, ciphertexts, signatures, …

Post-quantum Assumptions

- Isogeny-based assumptions
  - Isogenies of Elliptic Curves
  - **CSIDH**
  - Structure similar to DH: Could be a drop-in replacement of DHKE

- Code-based cryptosystem
  - Based on error-correcting code
  - **Classic McEliece**: based on random binary Goppa code

Post-quantum Cryptographic Algorithms

- NIST standardization of Post-Quantum Cryptography (2016 - Now)

- Some candidate algorithms:
  - CRYSTALS-Kyber: Public-key Encryption based on MLWE
  - CRYSTALS-Dilithium: Signature Scheme based on MLWE and MSIS
  - FALCON: Signature Scheme based on NTRU
  - SPHINCS+: Hash-based signature scheme
  - Classic-McEliece: Public-key Encryption based on random binary Goppa code
  - ...

- Standardizing:
  - **ML-KEM**: based on CRYSTALS-Kyber
  - **ML-DSA**: based on CRYSTALS-Dilithium
  - Stateless Hash-Based Digital Signature: based on SPHINCS+

Transition from Pre-Quantum to Post-Quantum

- Should we immediately change everything to be post-quantum?

- Efficiency of classical algorithms v.s. post-quantum algorithms: (e.g., ECDSA v.s. CRYSTALS-Dilithium)

	ECDSA	Dilithium
sk size	~32B	~1.3KB
pk size	~32B	~2.5KB
signature size	~64B	~2.5KB
Running time	t	10~100*t

- Studies on classical cryptography: since 1970s
- Large-scale studies on post-quantum cryptography: since 2010s

Transition from Pre-Quantum to Post-Quantum

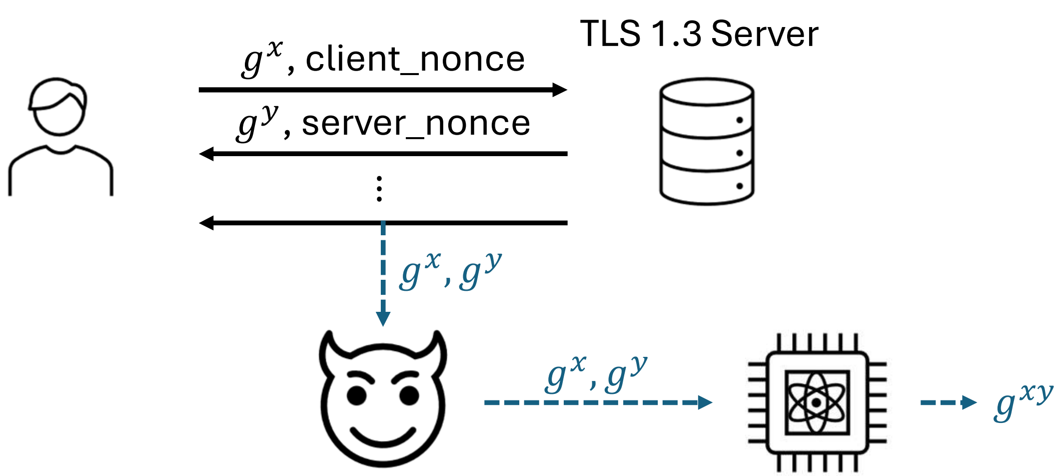
- Should we wait until the first large-scale quantum computer appears?

- "Harvest Now, Decrypt Later": The adversary stores today's encrypted data (harvest now). In the future, quantum computers decrypt this data (decrypt later)

Transition from Pre-Quantum to Post-Quantum

- Should we wait until the first large-scale quantum computer appears?

- “Harvest Now, Decrypt Later”: The adversary stores today’s encrypted data (harvest now). In the future, quantum computers decrypt this data (decrypt later)



Transition from Pre-Quantum to Post-Quantum

- Hybrid Cryptography
  - Classical algorithms + post-quantum algorithms
  - Example: ECDH in TLS 1.3 -> ECDH + Kyber in TLS

Transition from Pre-Quantum to Post-Quantum

- Hybrid Cryptography
  - Classical algorithms + post-quantum algorithms
  - Example: ECDH in TLS 1.3 -> ECDH + Kyber in TLS

The ECDH in TLS 1.3

$\xrightarrow{g^x\text{, client\_nonce}}$

$\xleftarrow{g^y\text{, server\_nonce, …}}$

⋮

A simple KE
based on Kyber KEM

$(epk, esk) \leftarrow \mathrm{KeyGen}$ $\xrightarrow{\quad epk \quad}$

$\xleftarrow{\quad c \quad}$ $(c, K) \leftarrow \mathrm{Encaps}(epk)$

$K \leftarrow \mathrm{Decaps}(esk, c)$

- Advantages: Classical security provided by ECDH + Quantum security provided by Kyber

Transition from Pre-Quantum to Post-Quantum

- Hybrid Cryptography
  - Classical algorithms + post-quantum algorithms
  - Example: ECDH in TLS 1.3 -> ECDH + Kyber in TLS

ECDH+ Kyber KEM

$(epk, esk) \leftarrow \text{KeyGen}$ $\quad \xrightarrow{g^x,\ epk,\ \text{client\_nonce}}$

$\xleftarrow{g^y,\ c,\ \text{server\_nonce},\ \dots}$ $\quad (c, K) \leftarrow \text{Encaps}(epk)$

$K \leftarrow \text{Decaps}(esk, c)$

$\vdots$

Keys = KeySchedule(…|| g^{xy} || K ||…)

- Advantages: Classical security provided by ECDH + Quantum security provided by Kyber

Transition from Pre-Quantum to Post-Quantum

- Post-quantum Encryption + classical signature schemes:
    - Resist "Forge now, decrypt later" attacks by quantum computers
    - Example: TLS 1.3 -> "Semi-PQ" TLS
        - The classical signature scheme ensures that the adversary cannot impersonate a server **now**…
        - The PQ KEM scheme ensures the adversary cannot decrypt in the **future**…

$(epk, esk) \leftarrow$ KeyGen

g^x, epk, client_nonce $\longrightarrow$

$\longleftarrow$ g^y, c, server_nonce

$(c, K) \leftarrow$ Encaps(epk)

$K \leftarrow$ Decaps(esk, c)

$\longleftarrow$ $cert[pk_S], Sign\big(sk_s, (g^x, g^y, epk, c, \text{nonces})\big)$

⋮

Keys = KeySchedule(…|| g^{xy} || K ||…)

Transition from Pre-Quantum to Post-Quantum

- Post-quantum Encryption + classical signature schemes:
  - Example: (Simplified) PQXDH: X3DH + **PQ-secure KEM**

Alice — Server — Bob

Server ← Bob: $IPK_B, SPK_B, \sigma_B, \{OPK_B^1, \dots, OPK_B^{100}\}$

where (σ_B = Sign(ik_B, SPK_B))

Alice ← Server: $\{IPK_B, SPK_B, \sigma_B, OPK_B^1\}$

Alice: ik_A, sk_A

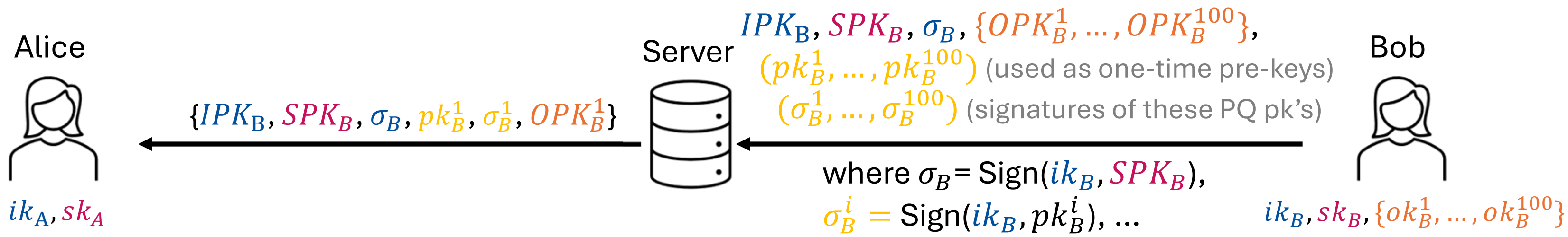
Bob: $ik_B, sk_B, \{ok_B^1, \dots, ok_B^{100}\}$

Alice: $ek_A \leftarrow_\$ \mathbb{Z}_q$

Alice → Bob: $IPK_A = g^{ik_A}$, $EPK_A = g^{ek_A}$, "1st OPK", $\mathrm{AEAD}_{SK_A}(\text{"some message"}, \mathrm{AD} = IPK_A || IPK_B)$

Transition from Pre-Quantum to Post-Quantum

- Post-quantum Encryption + classical signature schemes:
  - Example: (Simplified) PQXDH: X3DH + **PQ-secure KEM**



Transition from Pre-Quantum to Post-Quantum

- Post-quantum Encryption + classical signature schemes:
  - Example: (Simplified) PQXDH: X3DH + **PQ-secure KEM**

Alice

Server

Bob

$IPK_{\mathrm{B}}, SPK_B, \sigma_B, \{OPK_B^1, \dots, OPK_B^{100}\},$

$(pk_B^1, \dots, pk_B^{100})$ (used as one-time pre-keys)

$(\sigma_B^1, \dots, \sigma_B^{100})$ (signatures of these PQ pk's)

where σ_B = Sign(ik_B, SPK_B),

σ_B^i = Sign(ik_B, pk_B^i), …

$ik_B, sk_B, \{ok_B^1, \dots, ok_B^{100}\}$

$\{IPK_{\mathrm{B}}, SPK_B, \sigma_B, pk_B^1, \sigma_B^1, OPK_B^1\}$

ik_{A}, sk_A

$ek_A \leftarrow_\$ \mathbb{Z}_q$

$(c_{KEM}, K_{KEM}) \leftarrow \mathrm{Encaps}(pk_B^1)$

$SK_A = KDF(\dots,$ X3DH_secrets, $K_{KEM})$

$IPK_A = g^{ik_A}, EPK_A = g^{ek_A}$, "1st OPK", "1st KEM pk", c_{KEM},

$\mathrm{AEAD}_{SK_A}(\text{"some message"}, \mathrm{AD} = IPK_A || IPK_B)$

Transition from Pre-Quantum to Post-Quantum

- Post-quantum Encryption + classical signature schemes:
  - Example: (Simplified) PQXDH: X3DH + **PQ-secure KEM**

Alice

Server

Bob

$IPK_B, SPK_B, \sigma_B, \{OPK_B^1, \dots, OPK_B^{100}\},$

$(pk_B^1, \dots, pk_B^{100})$ (used as one-time pre-keys)

$(\sigma_B^1, \dots, \sigma_B^{100})$ (signatures of these PQ pk's)

where σ_B = Sign(ik_B, SPK_B),

σ_B^i = Sign(ik_B, pk_B^i), ...

$\{IPK_B, SPK_B, \sigma_B, pk_B^1, \sigma_B^1, OPK_B^1\}$

ik_A, sk_A

$ik_B, sk_B, \{ok_B^1, \dots, ok_B^{100}\}$

$ek_A \leftarrow_\$ \mathbb{Z}_q$

$(c_{KEM}, K_{KEM}) \leftarrow \text{Encaps}(pk_B^1)$

$SK_A = KDF(\dots,$ X3DH_secrets, $K_{KEM})$

$IPK_A = g^{ik_A}, EPK_A = g^{ek_A}$, "1st OPK", "1st KEM pk", c_{KEM},

$\text{AEAD}_{SK_A}(\text{"some message"}, \text{AD} = IPK_A || IPK_B)$

Next lecture:
More details about LWE, SIS, Crystal-Kyber/Dilithium, and more...

Exercises

- Find available python implementations of CRYSTAL-Kyber and CRYSTAL-Dilithium.

Further Reading

- NIST PQC project: https://csrc.nist.gov/projects/post-quantum-cryptography
- Chris Peikert's paper - *A Decade of Lattice Cryptography*: https://ia.cr/2015/939
- Specification of PQXDH: https://signal.org/docs/specifications/pqxdh/
- iMessage with PQ3: https://security.apple.com/blog/imessage-pq3/