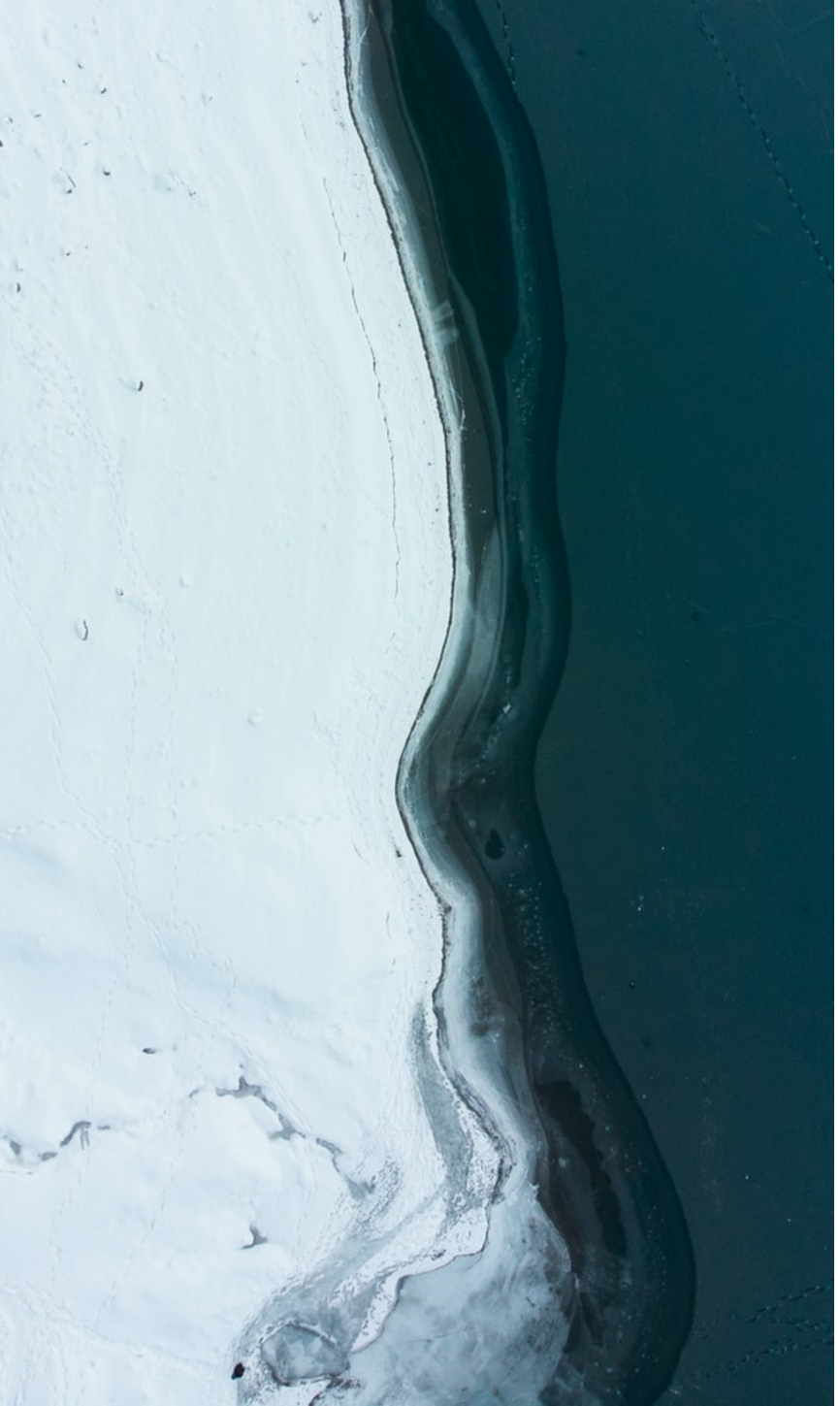


# An Introduction to Post-Quantum Cryptography

CRYPTOGRAPHIC ENGINEERING RESEARCH GROUP @ GMU

Duc Tri Nguyen

dnguye69@gmu.edu

# About me

- 5th-year Ph.D. Student in Applied Post-Quantum Cryptography at George Mason University, advised by Prof.Kris Gaj
- Hardware: FPGA, GPU
- Software: Assembly (x86, ARMv8, NEON, AVX2/AVX512)
- Security: Cryptography, Reverse Engineering

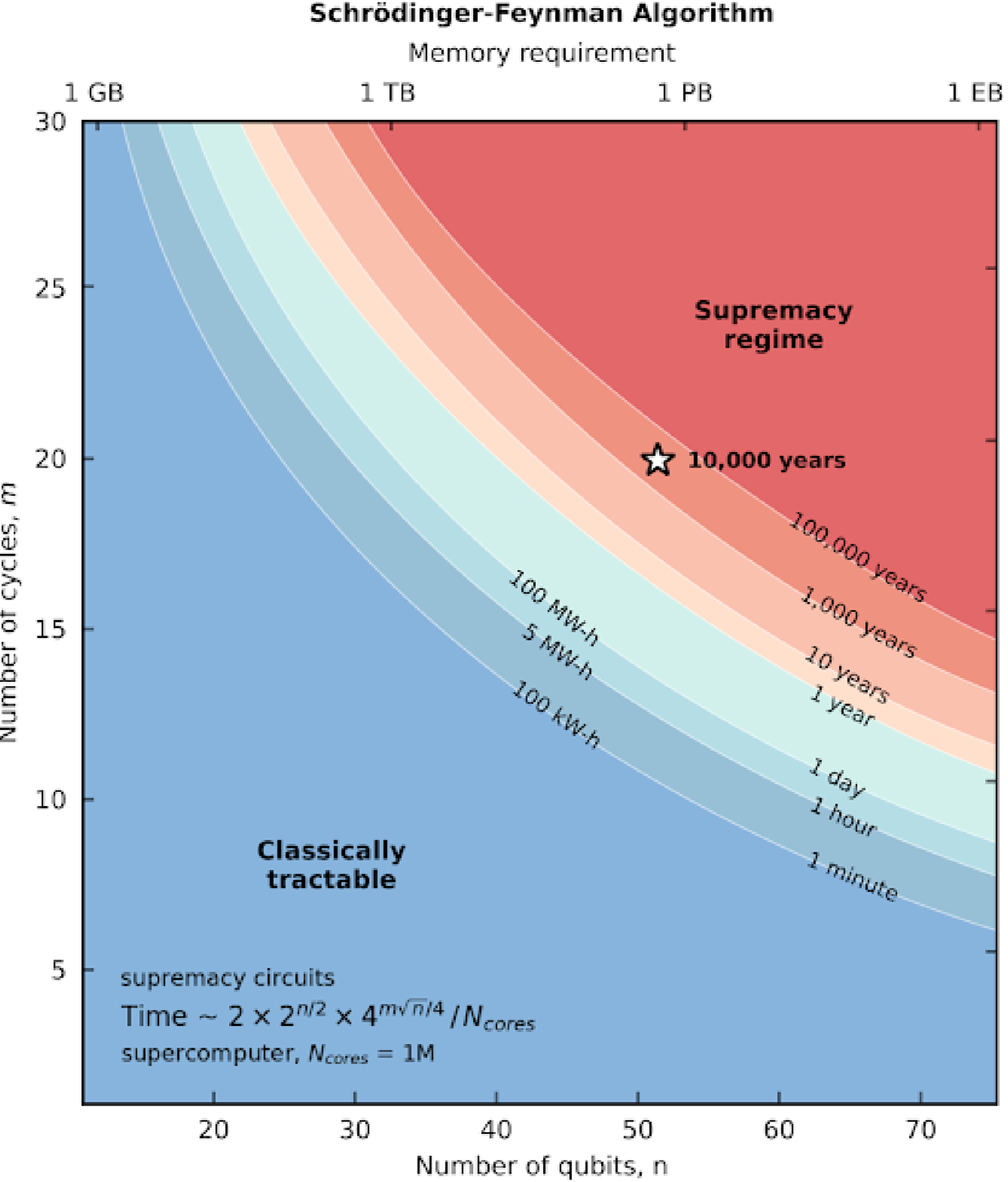
Lattice-based cryptographic implementation:

- Saber, Kyber, NTRU

**Awards**:

- International Olympiad in Cryptography
- U.S National Collegiate Cyber Defense Competition
- International Capture The Flag Security Competition

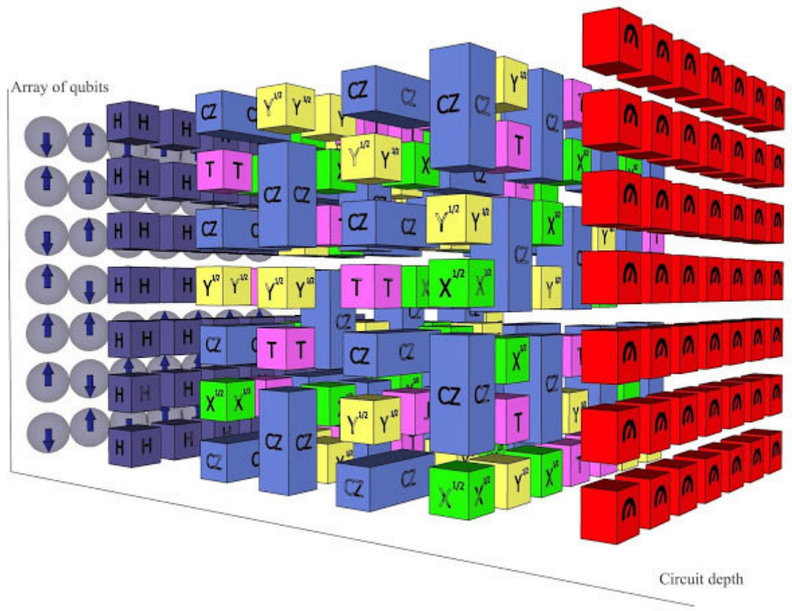
# Quantum Computing and NIST Post-Quantum Cryptography Competition




# The Threat of Quantum Computing

"Quantum supremacy," as proposed by John Preskill in 2012, was to describe the point where quantum computers can do things that classical computers can't.

In **2019**, Google experiment Schrodinger-Feynman Algorithm, within 53 quantum-bits (qubit), researchers at Google claimed they could demonstrate that quantum computer can solve the problem that takes classical computer 10,000 years to complete.




# What makes Quantum Computing a threat?

Superposition, Inference, and Entanglement

**Superposition**:

- Two states $|1>$ and $|0>$ happen at the same time
- For short, 10 classical bits are approximately equivalent to 5 qubits.

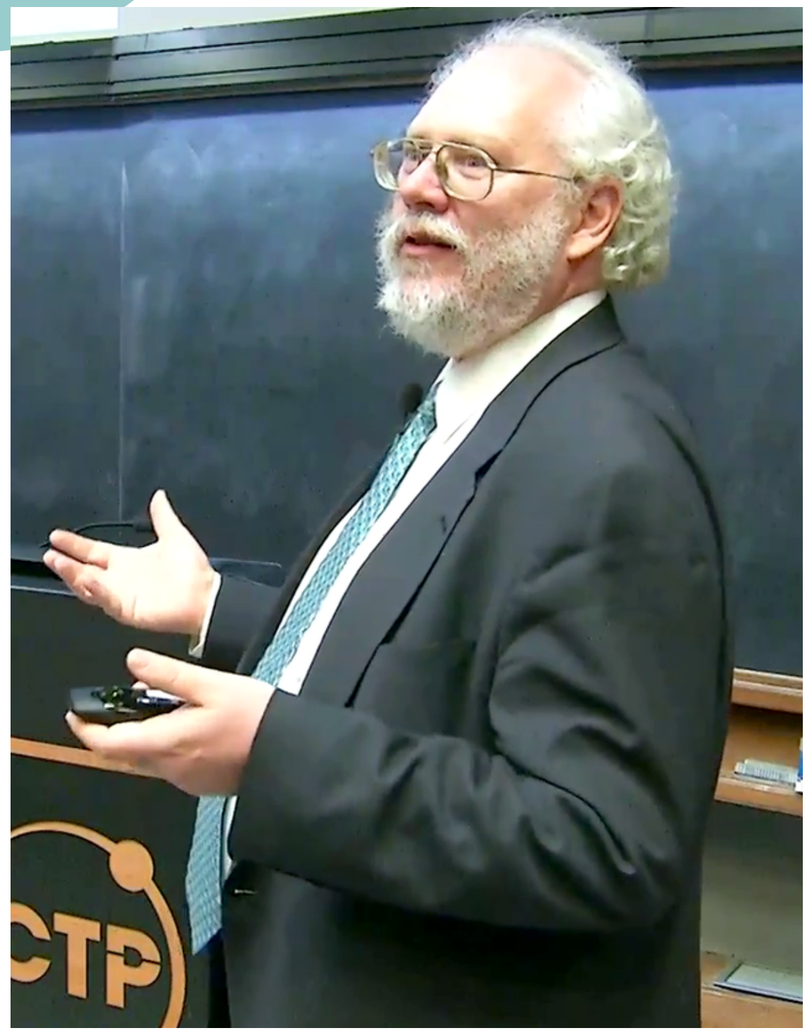
**Interference**

- Qubit states can interference with each other.

**Entanglement**:

- An effect over a single quantum bit can propagate through an entire array of qubits

# Threat of Quantum Algorithm over Public Key Cryptography



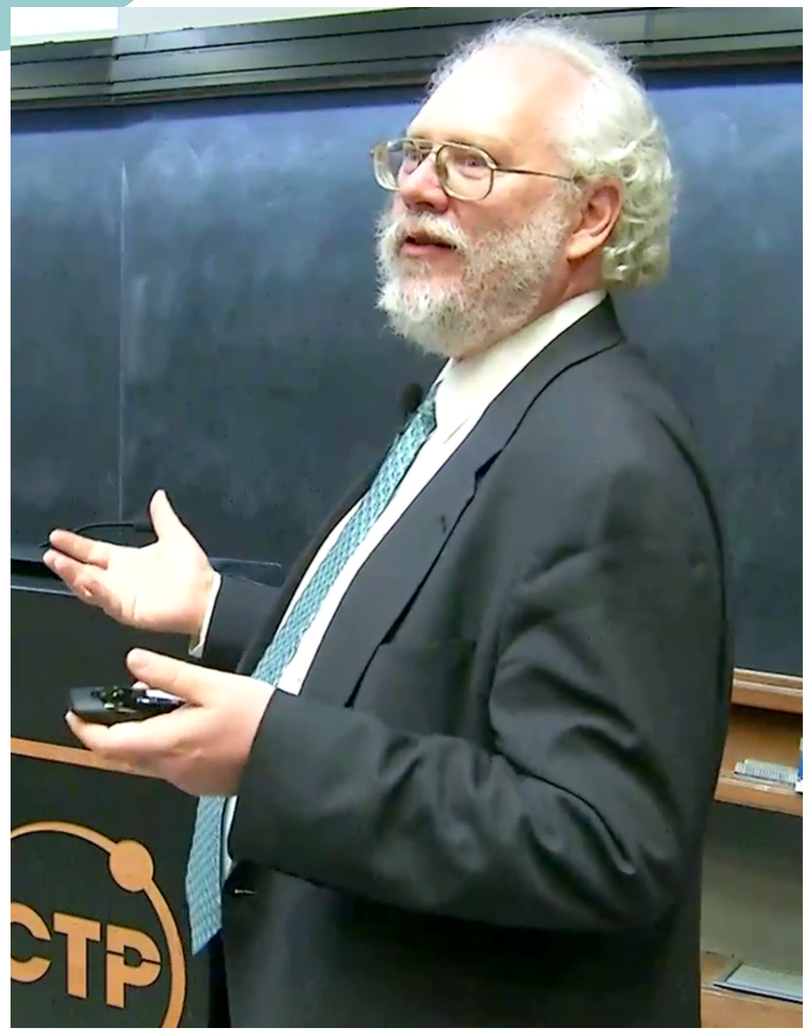
In the picture: Peter Shor

Shor’s algorithm can solve these problems in super polynomial time:

- Integer factorization
- Discrete log in finite field
- Discrete log in elliptic-curve groups

Modern cryptography nowadays cannot withstand the threat of large-scale quantum computers

# Threat of Quantum Algorithm over Public Key Cryptography



In the picture: Peter Shor

Shor’s algorithm can solve these problems in super polynomial time:

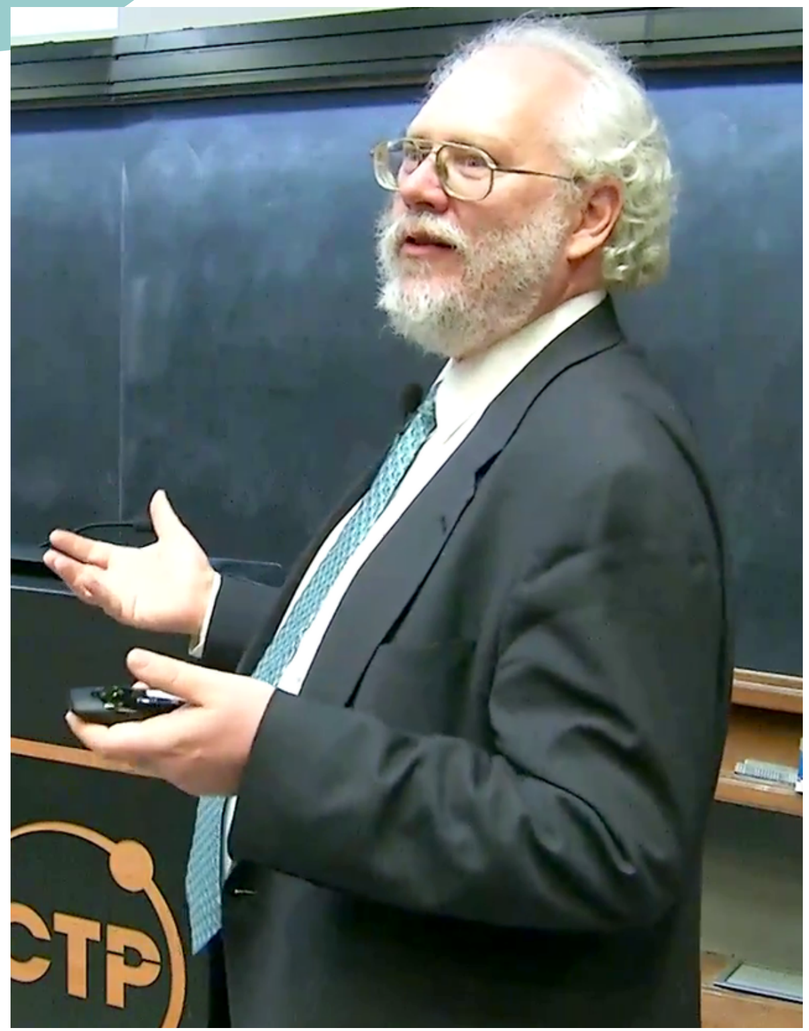
- Integer factorization

  => RSA is dead
- Discrete log in finite field

- Discrete log in elliptic-curve groups

Modern cryptography nowadays cannot withstand the threat of large-scale quantum computers

# Threat of Quantum Algorithm over Public Key Cryptography



In the picture: Peter Shor

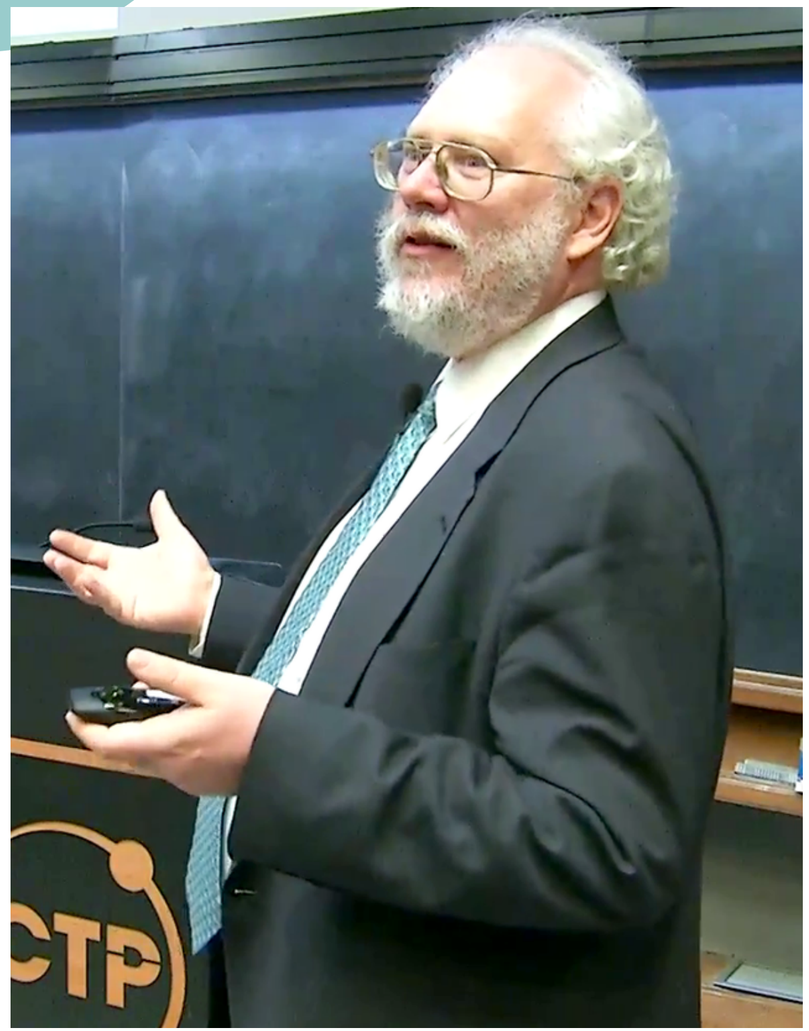
Shor’s algorithm can solve these problems in super polynomial time:

- Integer factorization

  => RSA is dead
- Discrete log in finite field

  => DSA, ElGamal are dead
- Discrete log in elliptic-curve groups

Modern cryptography nowadays cannot withstand the threat of large-scale quantum computers




# Threat of Quantum Algorithm over Public Key Cryptography

In the picture: Peter Shor

Shor's algorithm can solve these problems in super polynomial time:

- Integer factorization

  => RSA is dead
- Discrete log in finite field

  => DSA, ElGamal are dead
- Discrete log in elliptic-curve groups

  => ECC is dead

Modern cryptography nowadays cannot withstand the threat of large-scale quantum computers

# Threat of Quantum Algorithm over Private Key Cryptography

In the picture: Lov Grover

Grover’s algorithm can cut classical security bit by a half

- AES-128 is only 64 security bits in quantum computing. Thus, AES-192 is only 96 quantum security bit.
- SHA0, SHA1 were broken, SHA2 suffer length extension attack due to its Merkle–Damgård structure, SHA3-224 is insecure in quantum era.
- SHA3-256, SHA3-384, SHA3-512 (published in 2015) are not widely deployed yet.

NSA Guidance: NSA invalidated AES-128, AES-192 in the government sector. Move to AES-256, deploy SHA3.

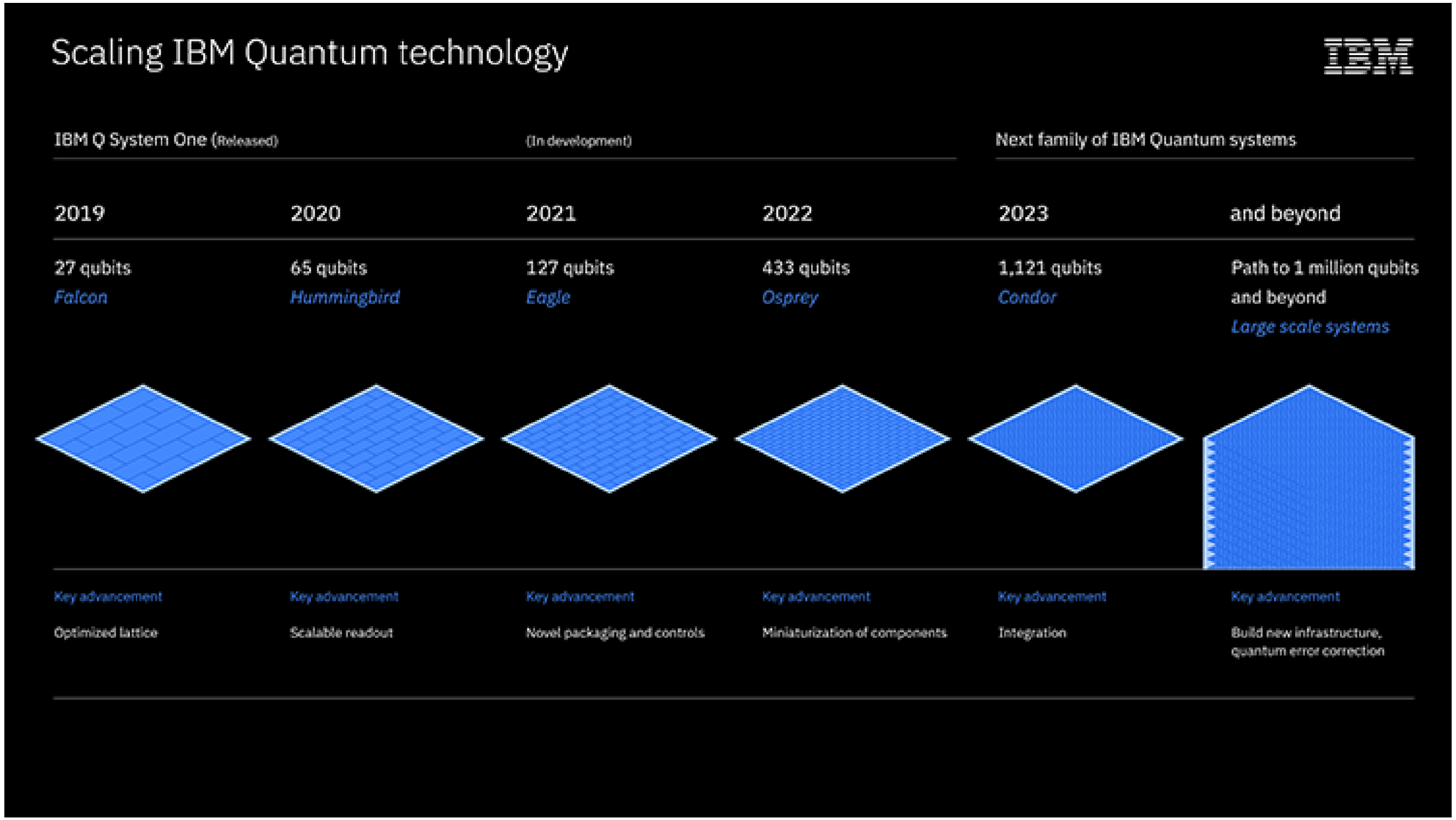
“nobody knows exactly when quantum computing will become a reality, but when and if it does, it will signal the end of traditional cryptography”

Sergai Boukhonine, **Cryptography: a security tool of the information age (1998)**. URL: https://pdfs.semanticscholar.org/3932/ 8253d692f791b37c425e776f6cee0b8c3e56.pdf.

“It is already proven that quantum computers will allow to break public key cryptography.”

Franck Leprévost, **The end of public key cryptography or does God play dices?**, PricewaterhouseCoopers Cryptographic Centre of Excellence Quaterly Journal (1999). URL: http://tinyurl.com/jdkkxc3.

# "QUANTUM SUPREMACY" MAY NOT HAPPEN NOW, BUT IT'S EXPECTED



# Cryptography Security before 2024

- NIST (National Institute of Standards and Technology) is currently in the process of standardizing a quantum-safe public-key encryption system which will **be completed by 2024.**
- The NSA announced its Commercial National Security Algorithm Suite (CNSA Suite) which meant to be a transitionary encryption standard before quantum-safe encryption becomes standardized.
- Symmetric key encryption: The NSA has withdrawn its support of AES-128 and AES-192 as cryptographically secure long-term encryption schemes because of quantum computing.

# Cryptography Security before 2024

- NIST (National Institute of Standards and Technology) is currently in the process of standardizing a quantum-safe public-key encryption system which will **be completed by 2024.**
- The NSA announced its Commercial National Security Algorithm Suite (CNSA Suite) which meant to be a transitionary encryption standard before quantum-safe encryption becomes standardized.
- Symmetric key encryption: The NSA has withdrawn its support of AES-128 and AES-192 as cryptographically secure long-term encryption schemes because of quantum computing.

The NSA has noted that it takes approximately **20 years** to fully deploy any cryptographic algorithm across national security system.

# Should we act now?

Let’s do a simple math

- Suppose we require confidentiality: $C$ years
- Suppose quantum computing takes $Q$ years to break RSA, ECC

If $C > Q$, then we are in trouble **today**.

- Suppose the transition to Post-Quantum Cryptography is $T$ years.

If $T > Q - C$, it means the transition time takes longer than the development of quantum computing.

The ideal case is: $T + C < Q$, the transition, and confidentiality requirement is finished before a large quantum computer is built.

# Should we act now?

Let’s do a simple math with number

- NSA says it takes $T = 20$ years for transition across national security systems. Let say we can successfully perform the transition in $T = 5$ years across all GE products.
- Let’s say we require confidentiality of industrial-grade product is $C = 20$ years.
- IBM claims they can reach 1,121 qubits in 2023 and 1 million qubits at the end of this decade ( $Q = 10$). Conservatively, let’s assume the time it takes to break RSA and ECC is $Q = 20$ years.

Total transition and confidientiality time $T + C = 5 + 20 = 25$

Time to break current public key system $Q = 20$

Then, quantum computing will shorten the confidentiality guarantee to 15 years.

# Should we act now?

Let’s do a simple math with number

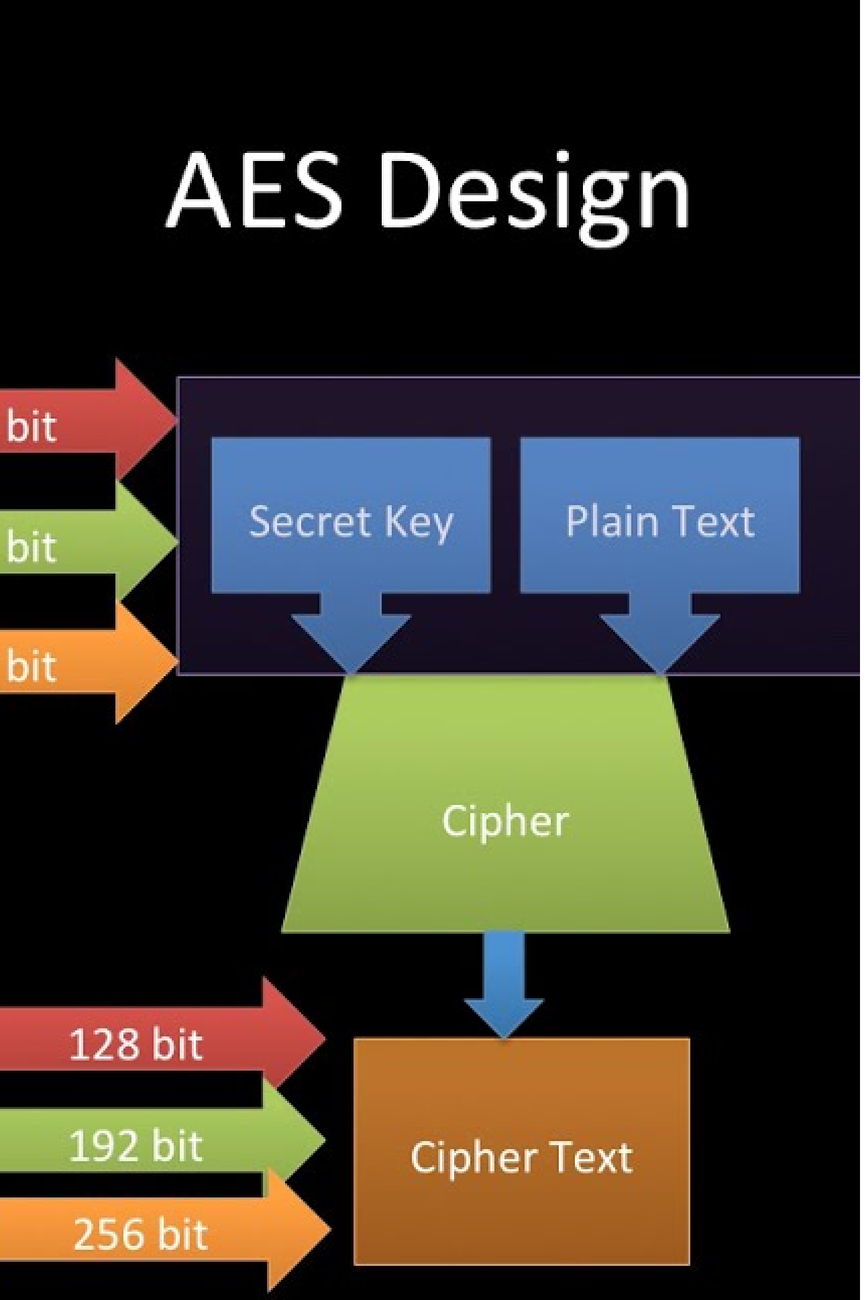
- NSA says it takes $T = 20$ years for transition across national security systems. Let say we can successfully perform the transition in $T = 5$ years across all GE products.
- Let’s say we require confidentiality of industrial-grade product is $C = 20$ years.
- IBM claims they can reach 1,121 qubits in 2023 and 1 million qubits at the end of this decade ( $Q = 10$). Conservatively, let’s assume the time it takes to break RSA and ECC is $Q = 20$ years.

Total transition and confidientiality time $T + C = 5 + 20 = 25$

Time to break current public key system $Q = 20$

Then, quantum computing will shorten the confidentiality guarantee to 15 years.

## If the requirement for the confidentiality of industrial-grade products is less than 15 years, then we are fine.




# What are we protecting?

- Confidentiality of data in transit. Example: TLS 1.3 over the internet
- Confidentiality of data at rest. Examples: AES, ChaCha cipher,...
- Data access. Examples: firmware updates

Consider a long-lived IoT device. To allow software to be patched over time the system allows remote firmware upgrade. The firmware itself has to be signed by the vendor to be accepted by the device, and special firmware verification code is responsible for checking the validity of the signature. If this firmware verification code itself is immutable, a quantum-safe signature mechanism must be deployed.

# NIST Post-Quantum Cryptography Competition

National Institute of Standards and Technology
U.S. Department of Commerce

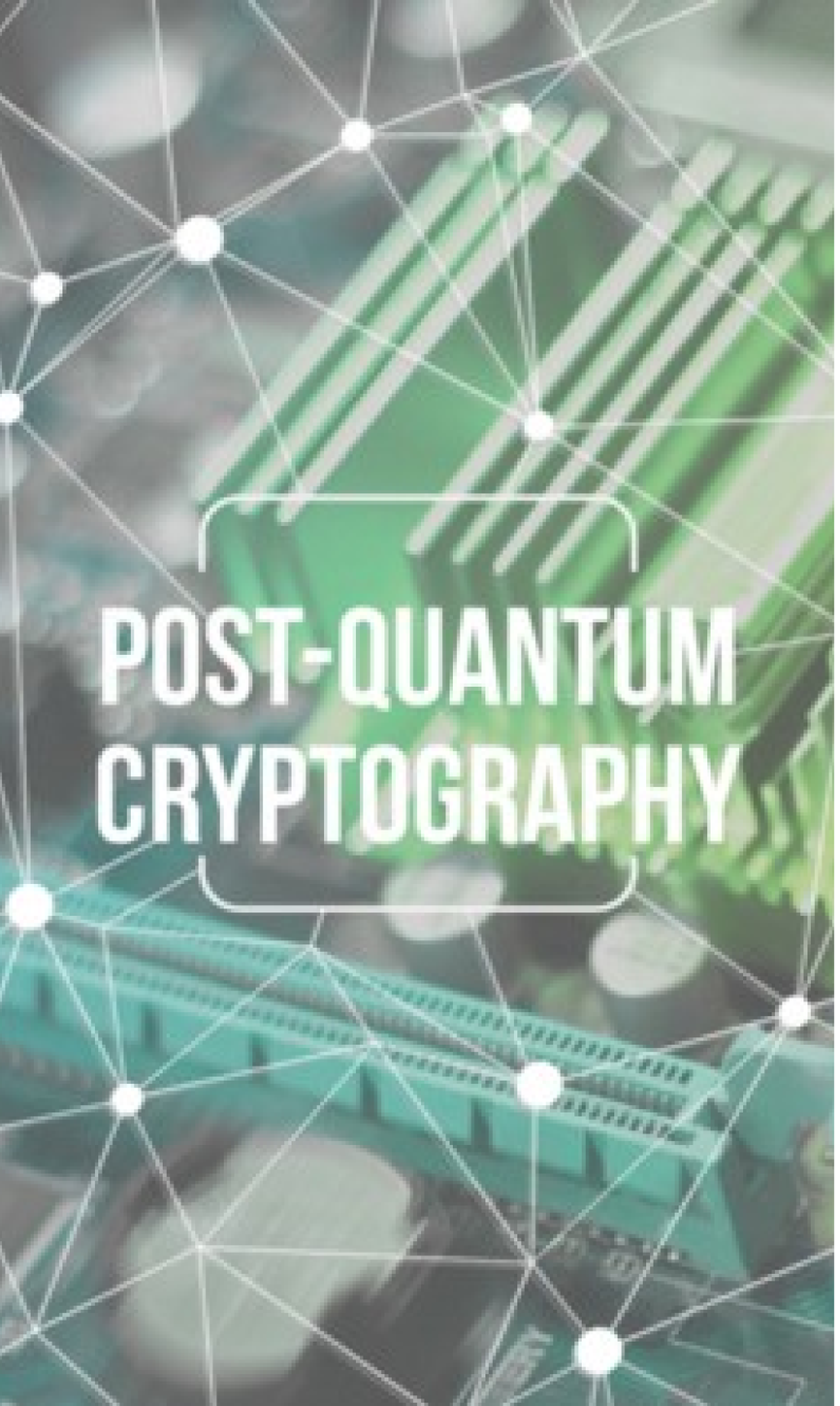
Timeline:

- Call for proposals: Dec' 16 - Nov' 17
- Round 1: Dec' 17 - Jan' 19, 69 submissions
- Round 2: Jan' 19 - Jul' 20, 26 candidates remaining
- Round 3: Jul' 20 - Mar' 22, 7 finalists, 8 alternatives.
- Draft of standard: 2022 - 2023

In the next 8 months, we will know NIST final decision for 7 finalists:

- 1 in 4 Key Encapsulation Mechanism (KEM) will be selected.
- 1 in 3 Digital Signatures will be selected.

# Introduction to Post-Quantum Cryptography

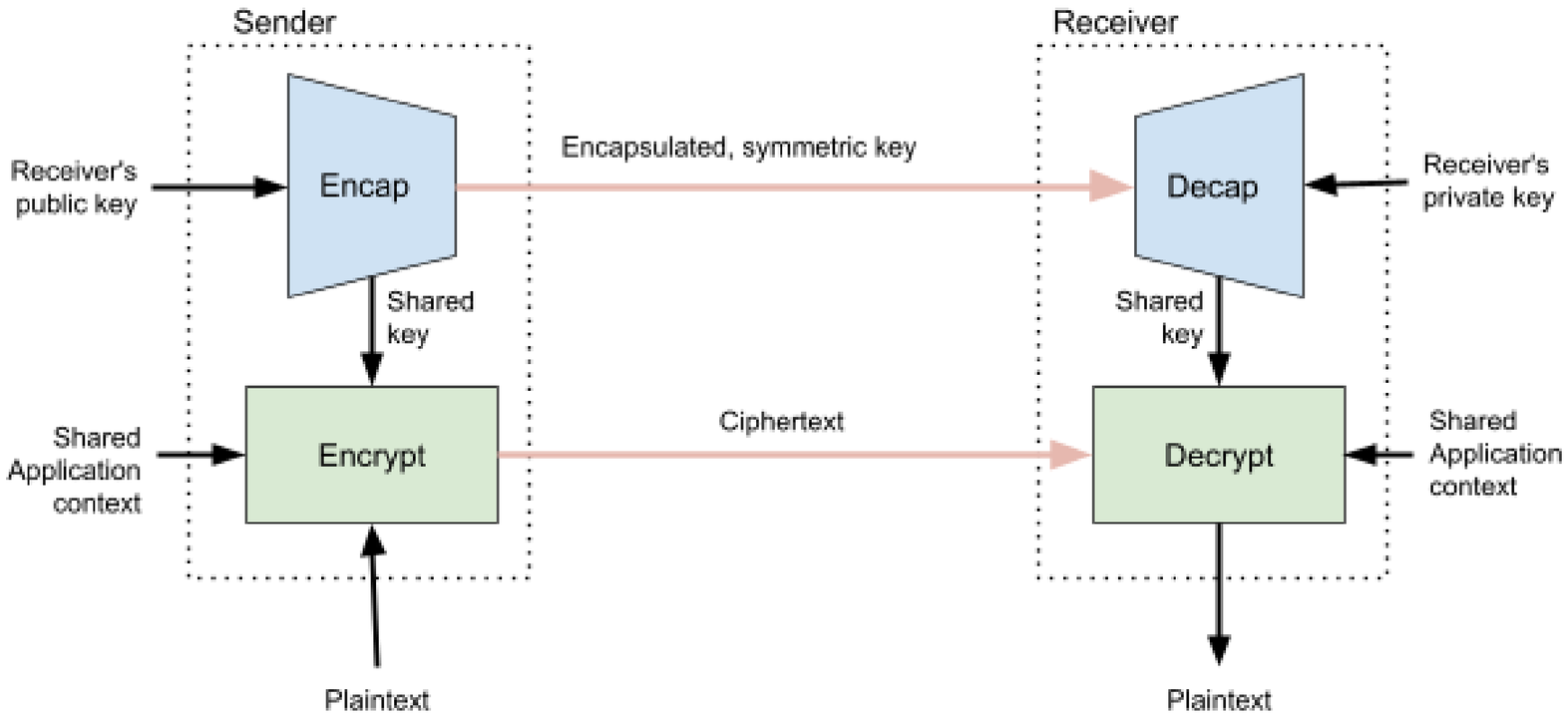



# What is Post-Quantum Cryptography?

- Post-Quantum Cryptography (PQC) is public-key cryptography that runs on classical computers, not quantum computer
- Security is based on hard cryptography problems that cannot be solved efficiently by quantum computing.
- The goal of Post-Quantum Cryptography is to replace current public-key cryptography standards: Public Key Exchange (Diffie-Hellman), Public Encryption (ECC, RSA), Digital Signature (ECDSA, DSA).

# Key Encapsulation Mechanism (KEM)

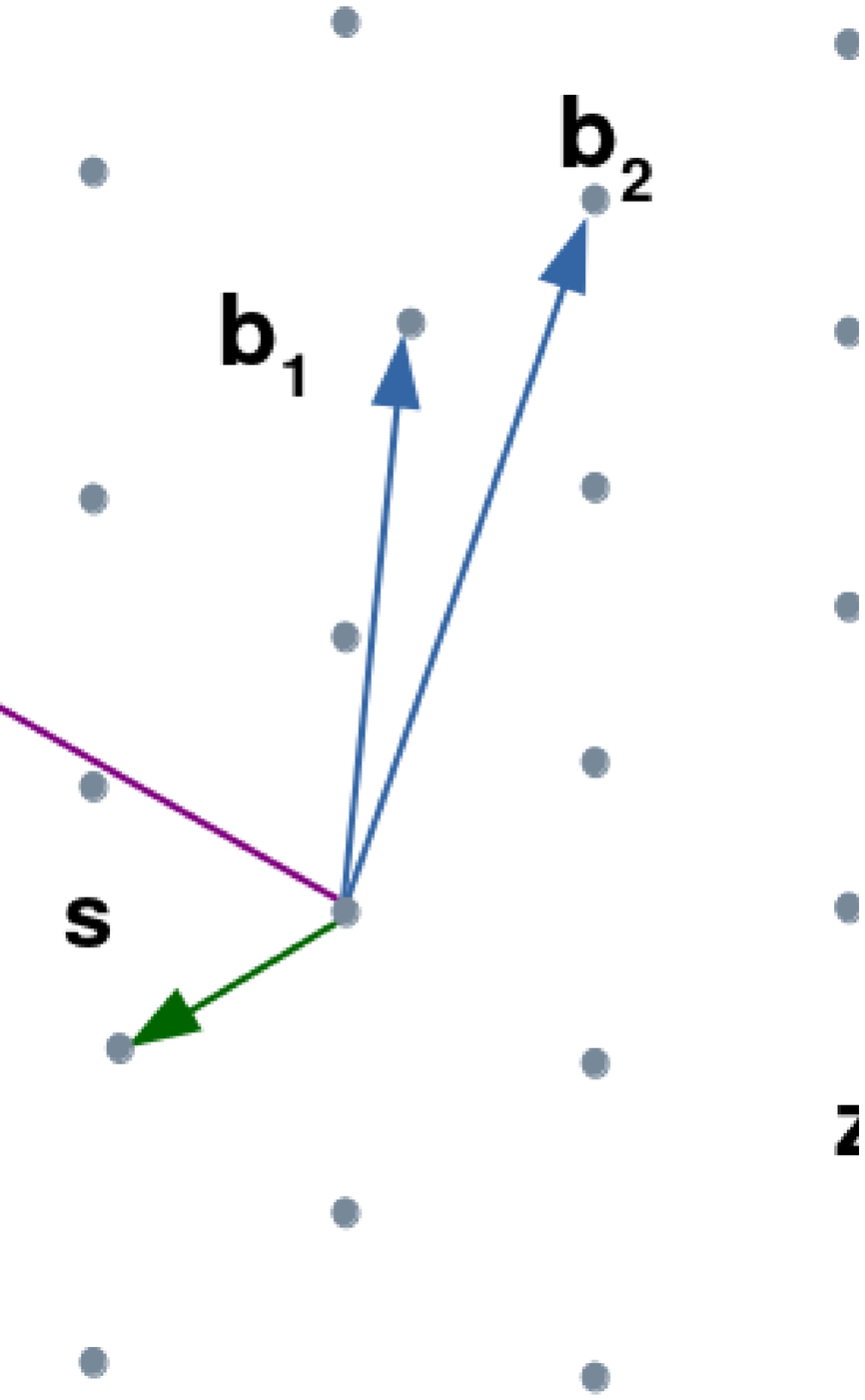
Simple, better than key exchange




Encapsulation: Encrypt shared key with receiver’s public key and send to the receiver.

Decapsulation: Decrypt ciphertext with his private key to figure out the shared key.

Advantages: **Two trips in total.**




# Lattice-based Cryptography

Security problem relied on Shortest/Closest Vector Problem

Lattice-based NIST Finalists KEM selection:

- CRYSTALS-Kyber
- SABER
- NTRU

Lattice-based NIST Finalists Signature selection:

- CRYSTALS-Dilithium
- Falcon

In terms of bandwidth and execution speed, lattice-based candidates have better performance compare to other categories.

# Code-based Cryptography

Security problem relied on syndrome decoding

NIST finalists KEM selection:

- Classic McEliece

No code-based submission in signature category due to large signature size. (possibly hundred of Mb)

Code-based Cryptography is only suitable for KEM.

# Multivariate Cryptography

Security problem relied on solving multivariate equations

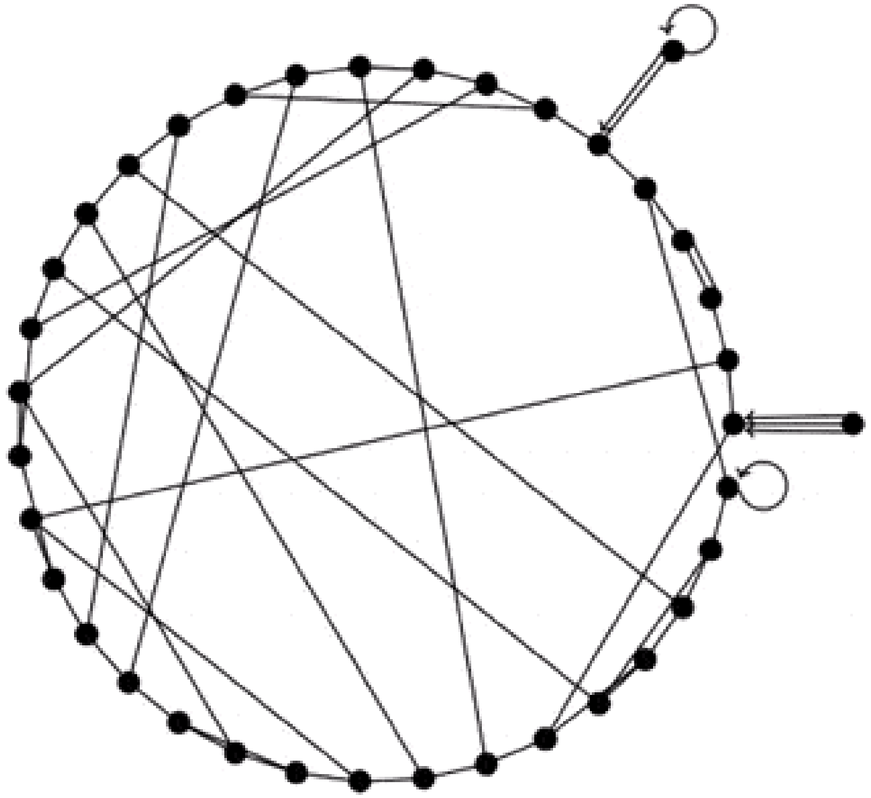
No multivariate submission in KEM category, only suitable for signature

NIST finalists Signature selection:

- Rainbow (severe security reduction attack)

# Isogeny-based Cryptography

Security problem relied on pseudo-random walks in supersingular isogeny graphs



# Isogeny-based Cryptography

Security problem relied in pseudo-random walks in supersingular isogeny graphs

Isogeny uses Elliptic Curve Cryptography (ECC), however, its security does not rely on security of ECC.

NIST finalists Signature selection:

- SIKE

The only advantage is small public keysize. In terms of performance, SIKE is very slow and impractical in embedded devices.

# Hash-based Cryptography

Security problem relied on security of hashing algorithm

NIST alternatives Signature selection:

- SPHINCS+

In terms of speed, SPHINCS+ has acceptable speed on modern CPUs.

Conservative security, potential Digital Signature algorithm for firmware update.

Has slow execution time and requires large memory on embedded devices.

# Post-Quantum Cryptography Deployment and Implementation Efforts

# PQC Deployment Efforts

Leave obsolete ciphers behind, time for Post-Quantum Cryptography

- 1996: SSLv3
- 1999: TLS 1.0
- 2006: TLS 1.1
- 2008: TLS 1.2
- 2017: Quantum Threat emerged. PQC experimental deployment started.

In 2018:

- TLS 1.3
- Google first benchmarked Post-Quantum Cryptography on Chrome browser.
- Round 2 Software ranking on multiple platforms: AVX2 Intel/AMD, Cortex-M4

# PQC Deployment Efforts

Leave obsolete ciphers behind, time for Post-Quantum Cryptography

In 2020:

- Cloudflare, Amazon experimented with PQC benchmarking on its infrastructure.
- High-speed hardware implementation on FPGA, Round 3 Hardware ranking from CERG GMU.

In 2021:

- Real-world benchmark on Mobile devices use *unoptimized implementation*.
- High-speed software implementation on ARM CPU, Round 3 Hardware ranking from CERG GMU.

# PQC Implementation Efforts

In 2018-2021:

- Software/Hardware implementation: ARM, RISC-V
- Hardware implementation: ASIC, FPGA
- Side-channel protected implementation: NewHope, Kyber, Saber on FPGA and Cortex-M4.
- Post-Quantum TLS from Microsoft.

The most important challenging implementation:

- Side-channel Resistant or Whitebox Cryptography for IoT devices.

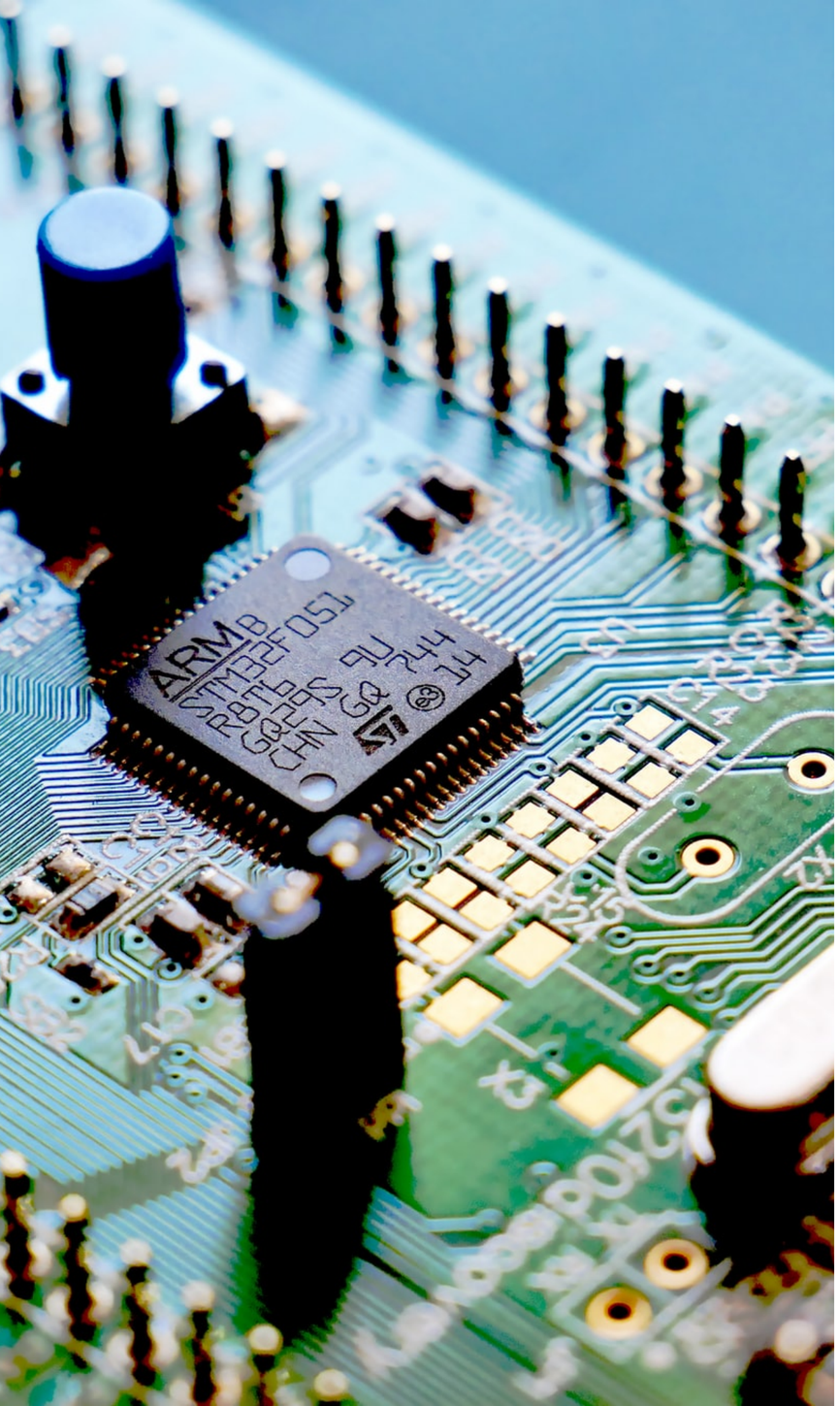
# Why so many deployments and benchmarking efforts?

Answer: There is no "one size fits all"

- The takeaway is that there are **no** candidates for PQC which come close to current public key cryptography in terms of both size of cryptographic material and performance.

For example:

- ECC Curve25519 and Ed25519 use as small public key as 32 bytes Keysize of Lattice-based Cryptography is no less than 500 bytes.
- **Isogeny-based** SIKE Digital signature at security level 1, has very small private and public key: 374, 330 bytes respectively. However, the performance is unbearable slow.
- **Lattice-based** CRYSTALS-Kyber KEM at security level 1, has the private key and public key size: 800, 1632 bytes. However, the execution speed of CRYSTALS-Kyber is faster than SIKE $680\times$ in decapsulation. Would you trade keysize for performance?



# Post-Quantum Cryptography performance on IoT device

Implementation challenges:

- Keysize may not fit in RAM <= Stack size
- Too slow due to complicated algorithm <= Speed
- Power consumption is a big concern <= Energy
- Bandwidth is limited <= Key size
- Side channel attack <= Whitebox implementation

# Post-Quantum Cryptography in WireGuard VPN


The key to observe here:

- Is the speed become faster or slower if we replace current cryptography with Post-Quantum Cryptography?
- Is the latency, keysize make a huge difference?

PQ-WireGuard

PQ-WireGuard: We did it again

# Post-Quantum Cryptography in WireGuard VPN


| VPN Software | Packet Number | Traffic (bytes) | Client time (ms) | Server Time (ms) |
|---|---|---|---|---|
| WireGuard | 2 | 324 | 0.572 | 0.005 |
| **PQ-WireGuard** | 2 | **2492** | 0.573 | 0.027 |
| OpenVPN (NIST P-256) | 19 | 5408 | 242.014 | 253.582 |
| OpenVPN (RSA-2048) | 21 | 7535 | 244.288 | 251.304 |

On high-performance CPU, it’s more about the protocol design rather than the performance of PQC.

# Post-Quantum Cryptography Optimized Implementation on ARMv8


# Post-Quantum Cryptography Optimized Implementation on ARMv8

- Took 8 months for me to complete NEON vectorize implementation of all lattice-based KEM finalists.

Two accepted papers:

- Duc Tri Nguyen and Kris Gaj. Optimized software implementations of CRYSTALS-Kyber, NTRU, and saber using NEON-based special instructions of ARMv8, 2021.**NIST Third PQC Standardization Conference.**

- Nguyen, Duc Tri, and Kris Gaj. "Fast NEON-Based Multiplication for Lattice-Based NIST Post-quantum Cryptography Finalists" **International Conference on Post-Quantum Cryptography**. Springer, Cham, 2021.

# ARMv8 NEON implementation ranking

- The ranking is consistent between Intel/AMD (x86) architecture and ARM architecture.
- Performance on Apple M1 ($900 laptop) can run on pair with AMD EPYC 7742 ($6000 CPU for Server).

My contribution:

- Speed up the deployment of Post-Quantum Cryptography on mobile and embedded devices. Without optimized implementation, the PQC adoption will be slow.
- State-of-the-art PQC ARMv8 implementation. https://github.com/GMUCERG/PQC_NEON
- Hopefully, my research work will be used by millions mobile devices.

The keys to take away from performance ranking:

- Fast implementation is likely to be standardized.
- Help NIST make decision in the final round.

# Apple M1 @ 3200 Mhz ranking

- Unit is microsecond ($\mu s$).

SECURITY LEVEL 3

| Rank | neon | Encapsulation | Ratio | neon | Decapsulation | Ratio |
|---|---|---|---|---|---|---|
| 1 | Kyber768 | 15 | 1.00 | Kyber768 | 13 | 1.00 |
| 2 | saber | 18 | 1.20 | saber | 17 | 1.30 |
| 3 | ntru-hps821 | 24 | 1.62 | ntru-hps821 | 22 | 1.66 |

SECURITY LEVEL 5

| Rank | neon | Encapsulation | Ratio | neon | Decapsulation | Ratio |
|---|---|---|---|---|---|---|
| 1 | Kyber1024 | 21 | 1.00 | Kyber1024 | 20 | 1.00 |
| 2 | firesaber | 27 | 1.21 | firesaber | 26 | 1.29 |

# Cortex-A72 @ 1500 Mhz ranking

- Unit is microsecond ($\mu s$).

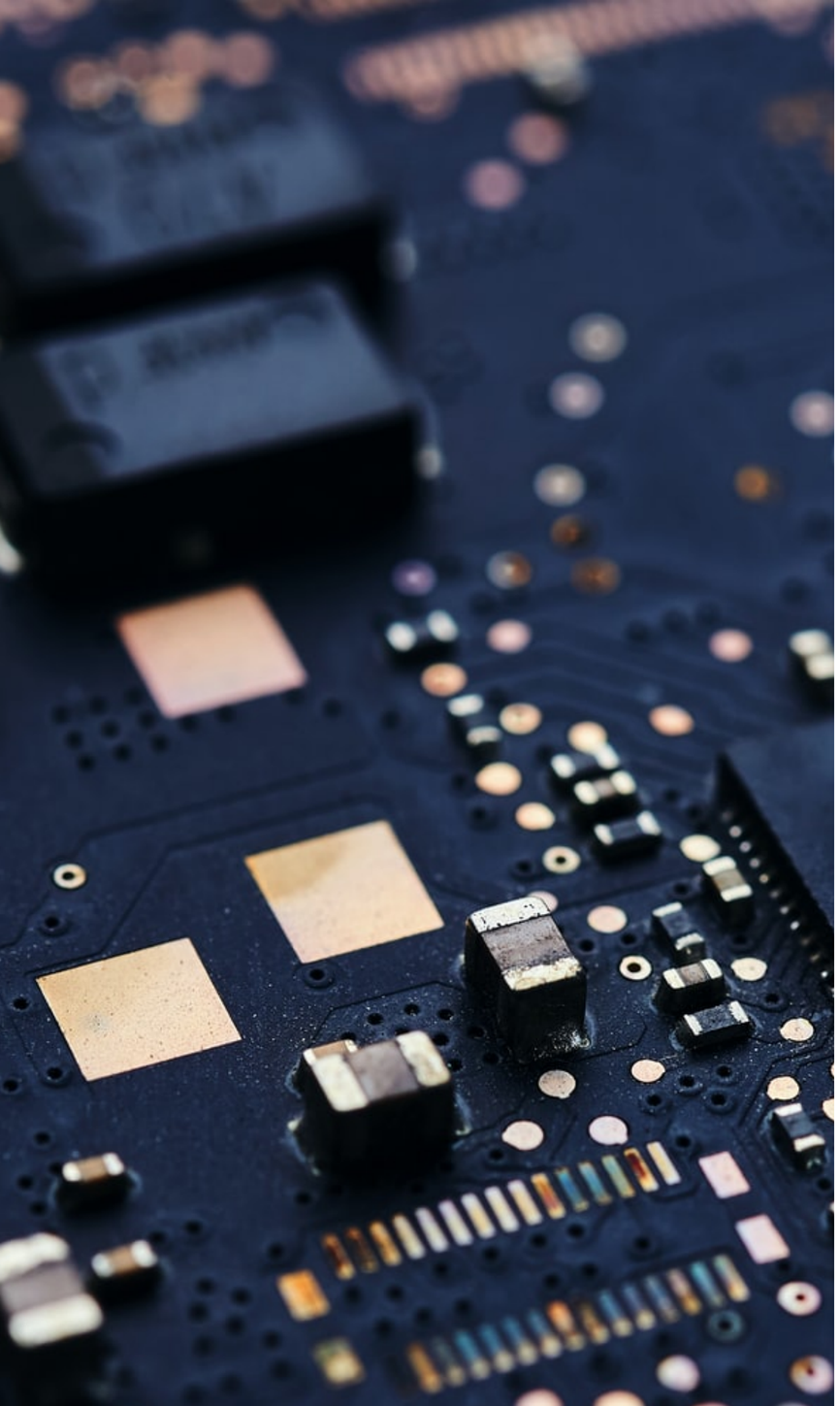
SECURITY LEVEL 3

| Rank | neon | Encapsulation | Ratio | neon | Decapsulation | Ratio |
|---|---|---|---|---|---|---|
| 1 | Kyber768 | 93 | 1.00 | Kyber768 | 90 | 1.00 |
| 2 | saber | 121 | 1.30 | saber | 124 | 1.36 |
| 3 | ntru-hps821 | 154 | 1.66 | ntru-hps821 | 181 | 2.02 |

SECURITY LEVEL 5

| Rank | neon | Encapsulation | Ratio | neon | Decapsulation | Ratio |
|---|---|---|---|---|---|---|
| 1 | Kyber1024 | 142 | 1.00 | Kyber1024 | 139 | 1.00 |
| 2 | firesaber | 174 | 1.22 | firesaber | 182 | 1.30 |

# High-speed Post-Quantum Cryptography on FPGA



# High-speed Post-Quantum Cryptography on FPGA

(My group research, I'm not the author of this paper)

Contribution:

- NTRU: First pure hardware implementation.
- CRYSTALS-Kyber, Saber: Produce the most-efficient hardware implementation in terms of both speed and area.

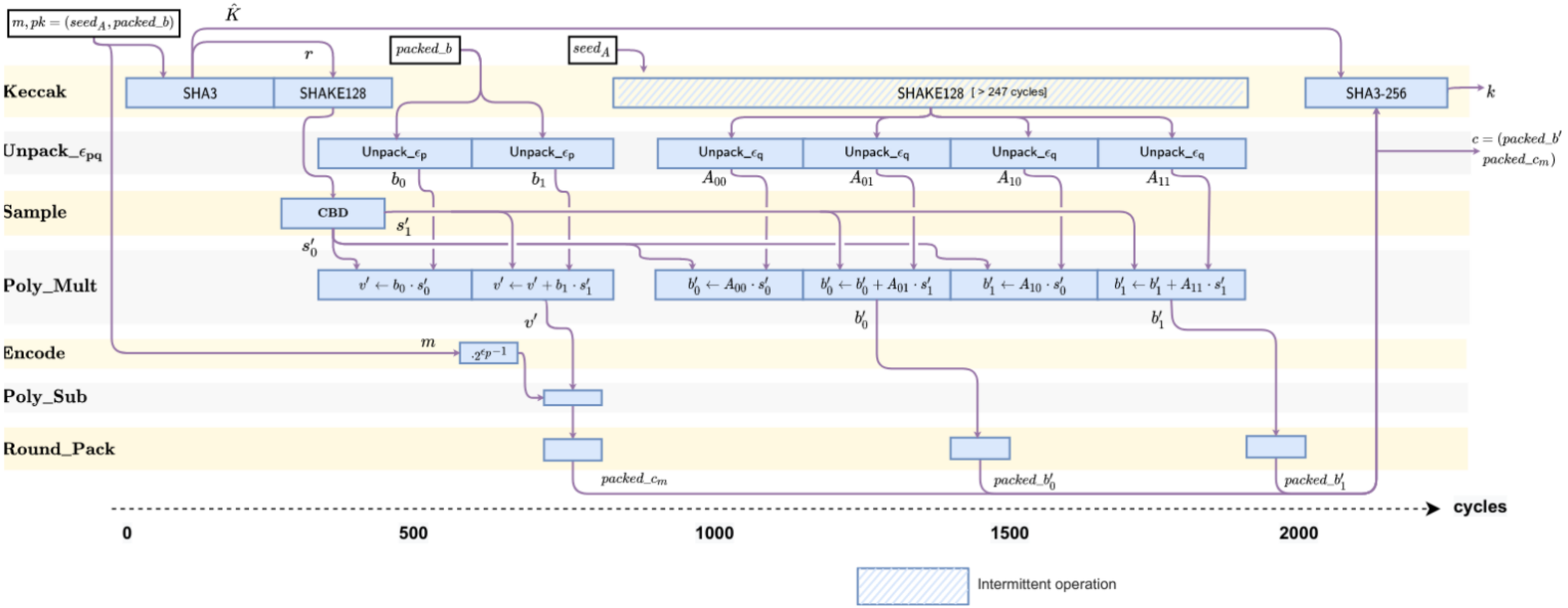
Accepted paper:

- Viet Ba Dang, Kamyar Mohajerani and Kris Gaj. High-Speed Hardware Architectures and Fair FPGA Benchmarking of CRYSTALS-Kyber, NTRU, and Saber, 2021.**NIST Third PQC Standardization Conference.**

# High-speed Post-Quantum Cryptography on FPGA

What makes hardware designs from CERG GMU the most efficient in terms of speed and area?

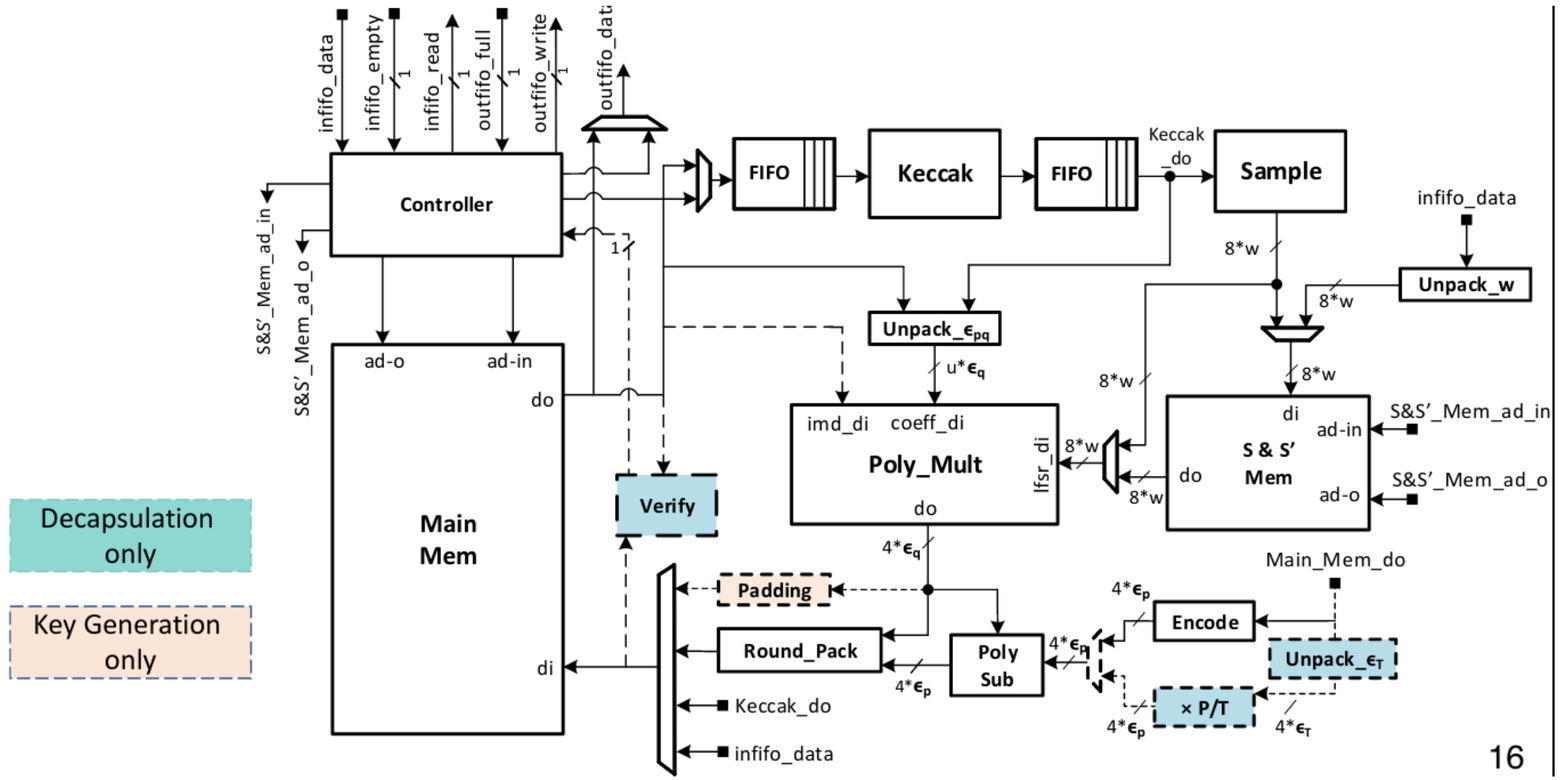
## Optimal operation scheduling



# High-speed Post-Quantum Cryptography on FPGA

What makes hardware designs from CERG GMU is the most efficient in terms of speed and area?

Optimal design into detailed



# High-speed Post-Quantum Cryptography Finalists Ranking

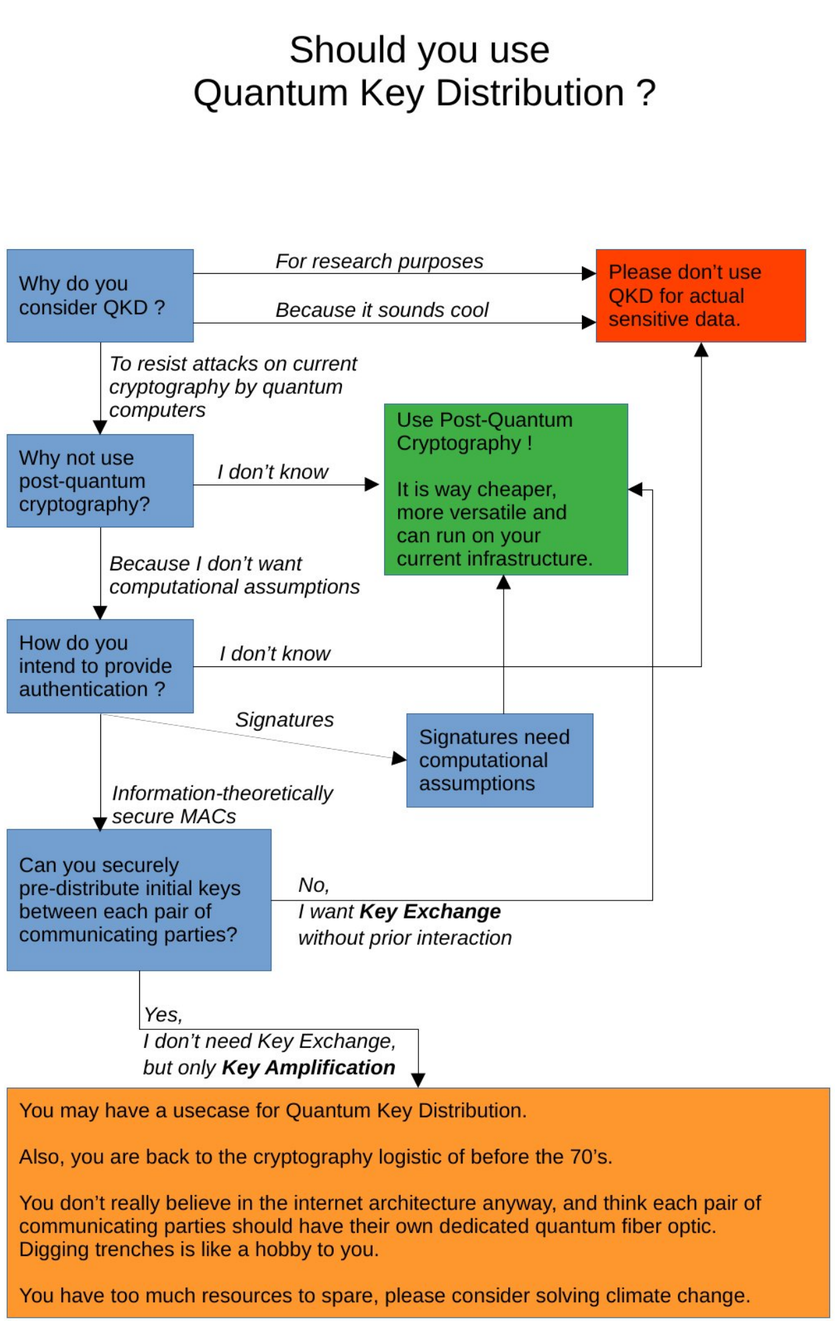
1st is Kyber, 2nd is Saber, and 3rd is NTRU

| **Encapsulation** | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| *Level 1* | | | *Level 3* | | | *Level 5* | | |
| **Algorithm** | **Time [us]** | **Ratio** | **Algorithm** | **Time [us]** | **Ratio** | **Algorithm** | **Time [us]** | **Ratio** |
| Saber | 12.7 | 1.00 | Kyber | 17.0 | 1.00 | Kyber | 21.7 | 1.00 |
| NTRU-HRSS | 13.9 | 1.09 | Saber | 22.0 | 1.29 | Saber | 34.5 | 1.59 |
| Kyber | 14.7 | 1.16 | NTRU-HPS | 35.2 | 2.07 | | | |
| NTRU-HPS | 28.4 | 2.24 | | | | | | |

| **Decapsulation** | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| *Level 1* | | | *Level 3* | | | *Level 5* | | |
| **Algorithm** | **Time [us]** | **Ratio** | **Algorithm** | **Time [us]** | **Ratio** | **Algorithm** | **Time [us]** | **Ratio** |
| Saber | 16.4 | 1.00 | Kyber | 22.2 | 1.00 | Kyber | 26.4 | 1.00 |
| Kyber | 20.5 | 1.25 | Saber | 27.5 | 1.24 | Saber | 41.9 | 1.59 |
| NTRU-HPS | 47.0 | 2.87 | NTRU-HPS | 63.8 | 2.87 | | | |
| NTRU-HRSS | 55.2 | 3.37 | | | | | | |

# Should we use Quantum Key Distribution?

Fight fire with fire, resist quantum computing by using quantum key distribution




# Should we use Quantum Key Distribution?

The answer is No in most cases.


Source

Quantum cryptography: a practical information security perspective*

# Post-Quantum Cryptography and 5G Security


# Post-Quantum Cryptography and 5G Security

Short answer: "Don’t use current Public Key Encryption".

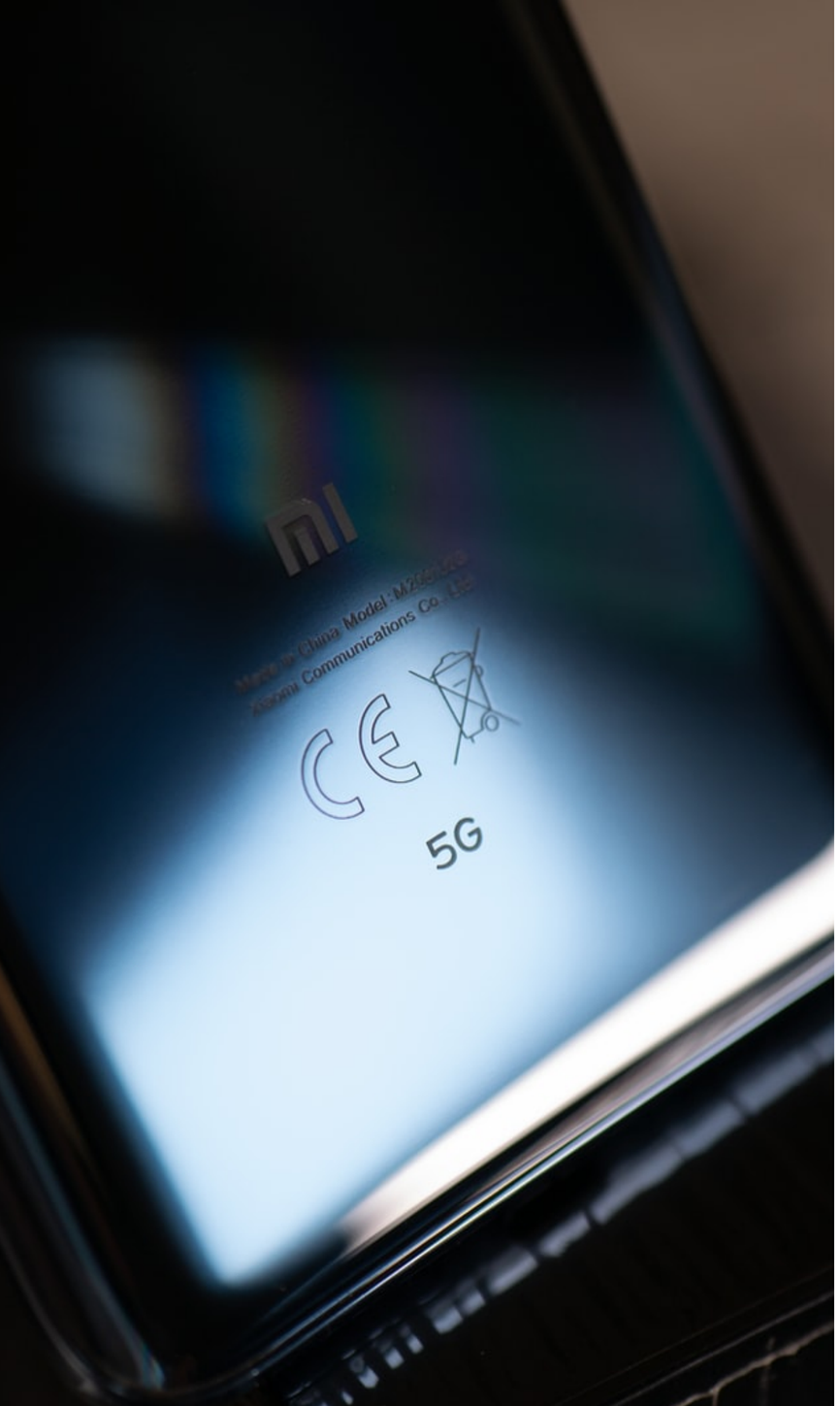
**TUTORIAL: Post-Quantum Cryptography and 5G Security***

T. Charles Clancy
Virginia Tech
Arlington, VA, United States
tcc@vt.edu

Robert W. McGwier
Virginia Tech
Blacksburg, VA, United States
rwmcgwi@vt.edu

Lidong Chen
National Institute of Standards and Technology
Gaithersburg, MD, United States
lily.chen@nist.gov

Source




# Post-Quantum Cryptography and 5G Security

Short answer: "Don’t use current Public Key Encryption".

**Table 1: Comparison of Classical and Quantum Security Levels for Ciphers**

| Cipher | Key Size | Effective Strength | |
|---|---|---|---|
| | | Classical | Quantum |
| RSA | 1024 | 80 | 0 |
| RSA | 2048 | 112 | 0 |
| ECC | 256 | 128 | 0 |
| ECC | 384 | 256 | 0 |
| AES | 128 | 128 | 64 |
| AES | 256 | 256 | 128 |

Source

# What if … we continue to use RSA, ECC in Quantum Era?

# What if … we continue to use RSA, ECC in Quantum Era?

Back to the good old days: "Increase key size"

# What if … we continue to use RSA, ECC in Quantum Era?

Back to the good old days: "Increase key size"

- RSA-512 publicly broken: "Let’s use RSA-768"

# What if … we continue to use RSA, ECC in Quantum Era?

Back to the good old days: "Increase key size"

- RSA-512 publicly broken: "Let’s use RSA-768"
- RSA-768 publicly broken: "Let’s use RSA-1024"

# What if … we continue to use RSA, ECC in Quantum Era?

Back to the good old days: "Increase key size"

- RSA-512 publicly broken: "Let’s use RSA-768"
- RSA-768 publicly broken: "Let’s use RSA-1024"
- RSA-2048 publicly broken by quantum computers: "Yeah, NSA already told us to use RSA-3072."

# What if … we continue to use RSA, ECC in Quantum Era?

Back to the good old days: "Increase key size"

- RSA-512 publicly broken: "Let’s use RSA-768"
- RSA-768 publicly broken: "Let’s use RSA-1024"
- RSA-2048 publicly broken by quantum computers: "Yeah, NSA already told us to use RSA-3072."
- Future: "RSA is life, Hold On for Dear Life (HODL)". Source

# What if … we continue to use RSA, ECC in Quantum Era?

Back to the good old days: "Increase key size"

- RSA-512 publicly broken: "Let's use RSA-768"
- RSA-768 publicly broken: "Let's use RSA-1024"
- RSA-2048 publicly broken by quantum computers: "Yeah, NSA already told us to use RSA-3072."
- Future: "RSA is life, Hold On for Dear Life (HODL)". Source

## The solution can never be simpler. **Just** increase keysize

| | Keysize |
|---|---|
| Private Key | 1 Gb |
| Signature | ~1 Gb |
| KEM Ciphertext | 1 Gb |
| Encrypt Ciphertext | 1 Gb |

# The solution can never be simpler. **Just** increase keysize

The computational time is also increased.

| | Keygen | Decrypt | Encrypt | Quantum Security Bits |
|---|---|---|---|---|
| pqrsa15 | 10.5h | 22m | 3m | - |
| pqrsa20 | 11h | 35m | 6m | - |
| pqrsa25 | 2.2d | 1.5h | 8m | - |
| pqrsa30 | 2.3d | 2.1h | 10m | 192 |

Benchmark on 1 core of 3GHz Intel Skylake with simplified public exponent `` `e = 3` ``

Imagine the speed of IoT and embedded devices.

# Take aways

# Conclusion

- In two years after NIST standardization, Post-Quantum Cryptography will replace existing public-key cryptography.
- The performance of Post-Quantum Cryptography is close to current Public Key Encryption.
- Side-channel resistance implementation is a huge challenge.
- Hardware implementation in FPGA shows that Lattice-based PQC has the potential to be standardized.
- Software implementation in ARM NEON and AVX2 also confirm lattice-based PQC has the potential to be standardized. CRYSTALS-Kyber has the best engineering design. CRYSTALS-Kyber is small, fast, utilize the best algorithm for polynomial multiplication.
- There is no good to increase the key size for RSA, ECC to overcome quantum threat.
- It’s better to have a placeholder for PQC deployment in product design and implementation now.

Thank you for your attention! Questions?