Cryptography in a Post-Quantum World

Dustin Moody

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

Crypto Technology Group
Computer Security Division
Information Technology Lab

NIST

- Non-regulatory federal agency within U.S. Department of Commerce.
- Founded in 1901, known as the National Bureau of Standards (NBS) prior to 1988.
  - Origins in the Constitution: “Congress shall have power to …. fix the standard of weights and measures…”
- Headquarters in Gaithersburg, Maryland, and laboratories in Boulder, Colorado.
- Employs around 6,000 employees and associates.
- At least 5 Nobel prizes

The Computer Security Division

Conducts research, development and outreach necessary to provide standards and guidelines, mechanisms, tools, metrics and practices to protect nation's information and information systems.

CSD Publications

- Federal Information Processing Standards (FIPS): Specify approved crypto standards.
- NIST Special Publications (SPs): Guidelines, technical specifications, recommendations and reference materials, including multiple sub-series.
- NIST Internal or Interagency Reports (NISTIR): Reports of research findings, including background information for FIPS and SPs.
- NIST Information Technology Laboratory (ITL) Bulletins: Monthly overviews of NIST's security and privacy publications, programs and projects.

FIPS PUB 186-3

FEDERAL INFORMATION PROCESSING STANDARDS PUBLICATION

Digital Signature Standard (DSS)

CATEGORY: COMPUTER SECURITY SUBCATEGORY: CRYPTOGRAPHY

Information Technology Laboratory
National Institute of Standards and Technology
Gaithersburg, MD 20899-8900

Issued June, 2009

U.S. Department of Commerce
Gary Locke, Secretary
National Institute of Standards and Technology
Patrick Gallagher, Deputy Director

NIST-Approved Crypto Standards

Block Ciphers

- AES with 128, 192 and 256 bit keys (FIPS 197)
- Triple DES*(SP 800-67) and SKIPJACK*(FIPS 185)

Modes of Operation (SP 800 38 series)

- For confidentiality/authentication: ECB, CBC, CFB, OFB, XTS-AES, CCM, GCM
- Format preserving encryption modes: FF1, FF3

Hash Functions

- SHA-1*, SHA-2 family (FIPS 180), SHA-3 family (FIPS 202), TupleHash and ParallelHash (SP 800-185)

MAC

- CMAC, GMAC based on block ciphers
- HMAC, KMAC based on hash functions

Other standards

- Signatures, key agreement, key derivation, random bit generation etc.

NIST museum https://www.nist.gov/nist-museum/about

Public-Key

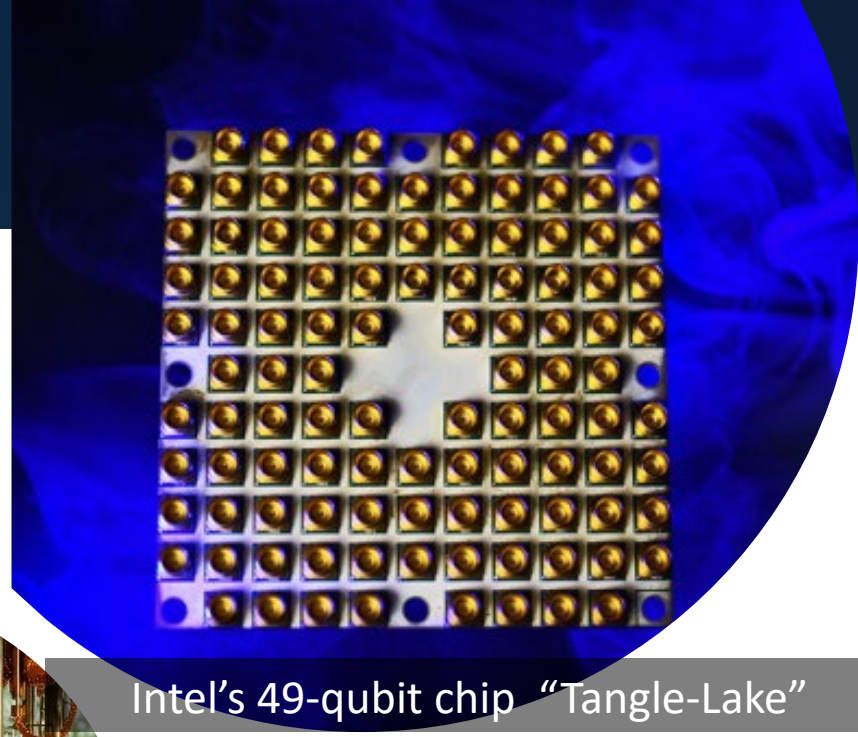
- RSA (encryption and signatures)
- ECDSA
- EC Diffie-Hellman
- Finite field Diffie-Helman

FIPS 186
SP 800-56A and 56B

Quantum Computers

- Exploit quantum mechanics to process information
- ”Qubits” instead of bits
- Potential to vastly increase computational power beyond classical computing limit

- Limitations:
  - When a measurement is made on quantum system, superposition collapses
  - Only good at certain problems
  - Quantum states are very fragile and must be extremely well isolated



Intel’s 49-qubit chip “Tangle-Lake”

IBM’s 50-qubit quantum computer

Google’s 72-qubit chip “Bristlecone”

Quantum Algorithms

- 1994, Peter Shor created a quantum algorithm that would give an exponential speed-up over classical computers
  - Factoring large integers
  - Finding discrete logarithms

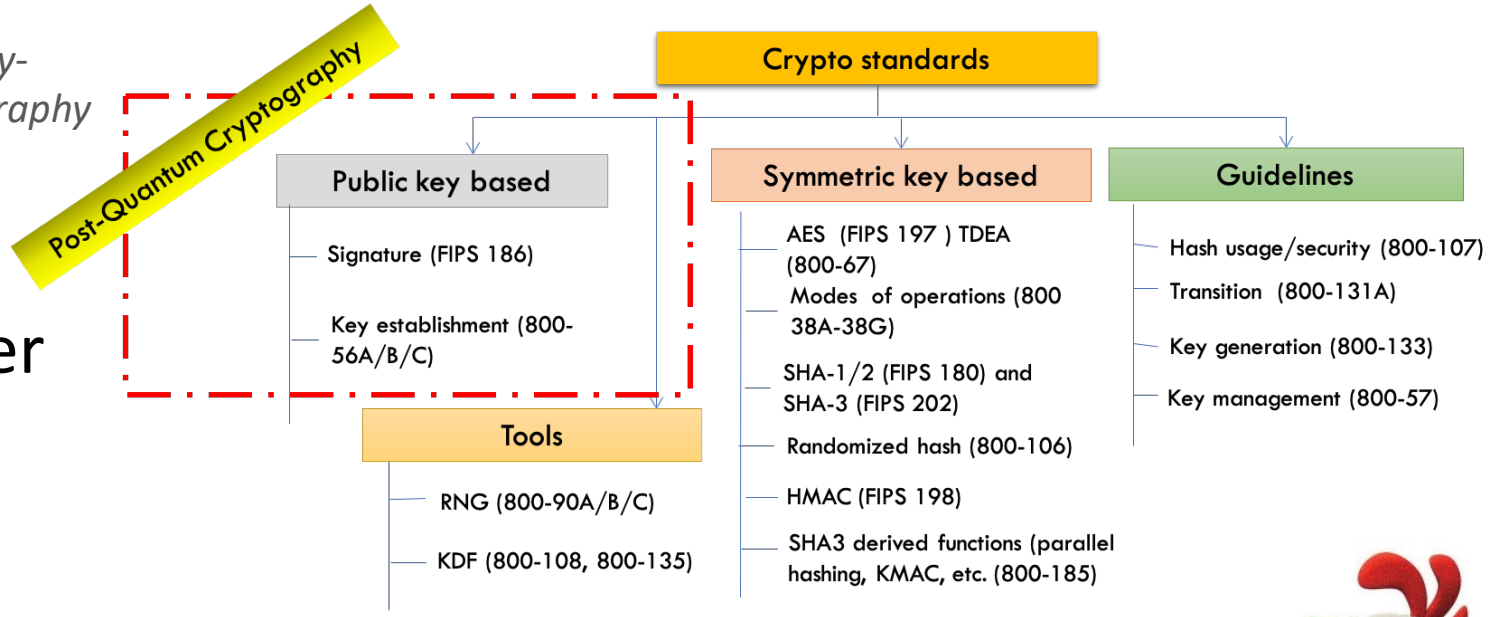
- Grover's algorithm – polynomial speed-up in unstructured search, from O(N) to O($\sqrt{N}$)

- Simulating the dynamics of molecules, superconductors, photosynthesis, among many, many others
  - see https://quantumalgorithmzoo.org/

The Quantum Threat

- NIST public-key crypto standards
  - **SP 800-56A**: *Recommendation for Pair-Wise Key-Establishment Schemes Using Discrete Logarithm Cryptography*
  - **SP 800-56B**: *Recommendation for Pair-Wise Key-Establishment Using Integer Factorization Cryptography*
  - **FIPS 186**: *The Digital Signature Standard*

  vulnerable to attacks from a (large-scale) quantum computer



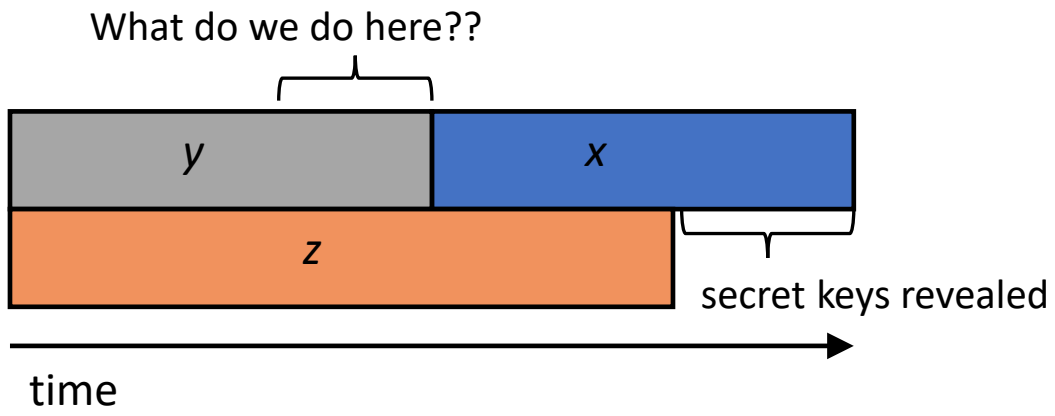
- Shor's algorithm would break RSA, ECDSA, (EC)DH, DSA
- Symmetric-key crypto standards would also be affected, but less dramatically

Post-Quantum Cryptography

- Post-Quantum Cryptography (PQC)
  - Cryptosystems which run on classical computers, and are believed to be resistant to attacks from both classical and quantum computers

- How soon do we need to worry?

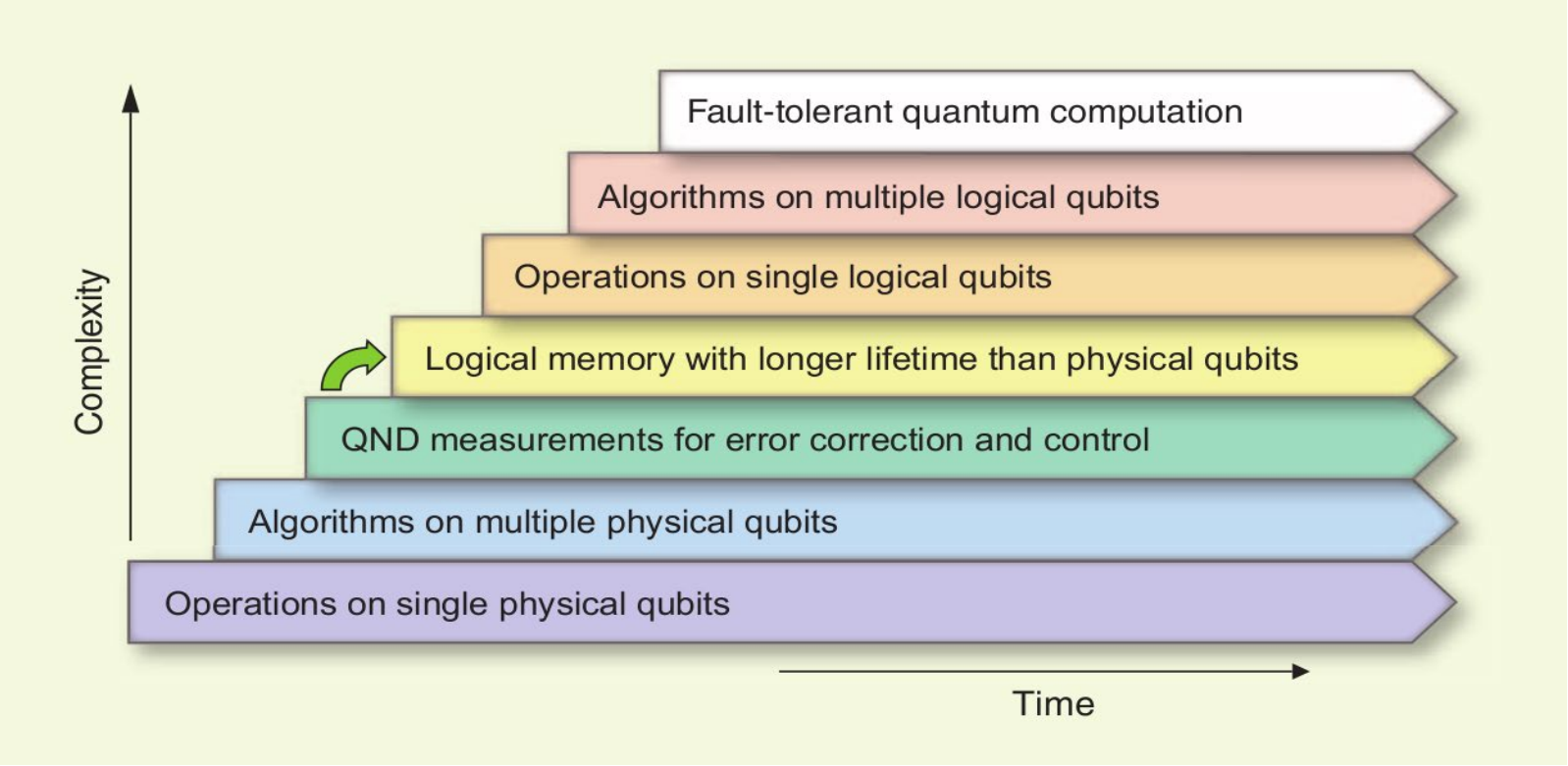
Theorem (Mosca): If $x + y > z$, then worry



x – time of maintaining data security
y – time for PQC standardization and adoption
z – time for quantum computer to be developed

Quantum Computing Progress

- A lot of progress, but still a long way to go



[Image credit: M. Devoret and R. Schoelkopf]

When will a Quantum Computer be Built?

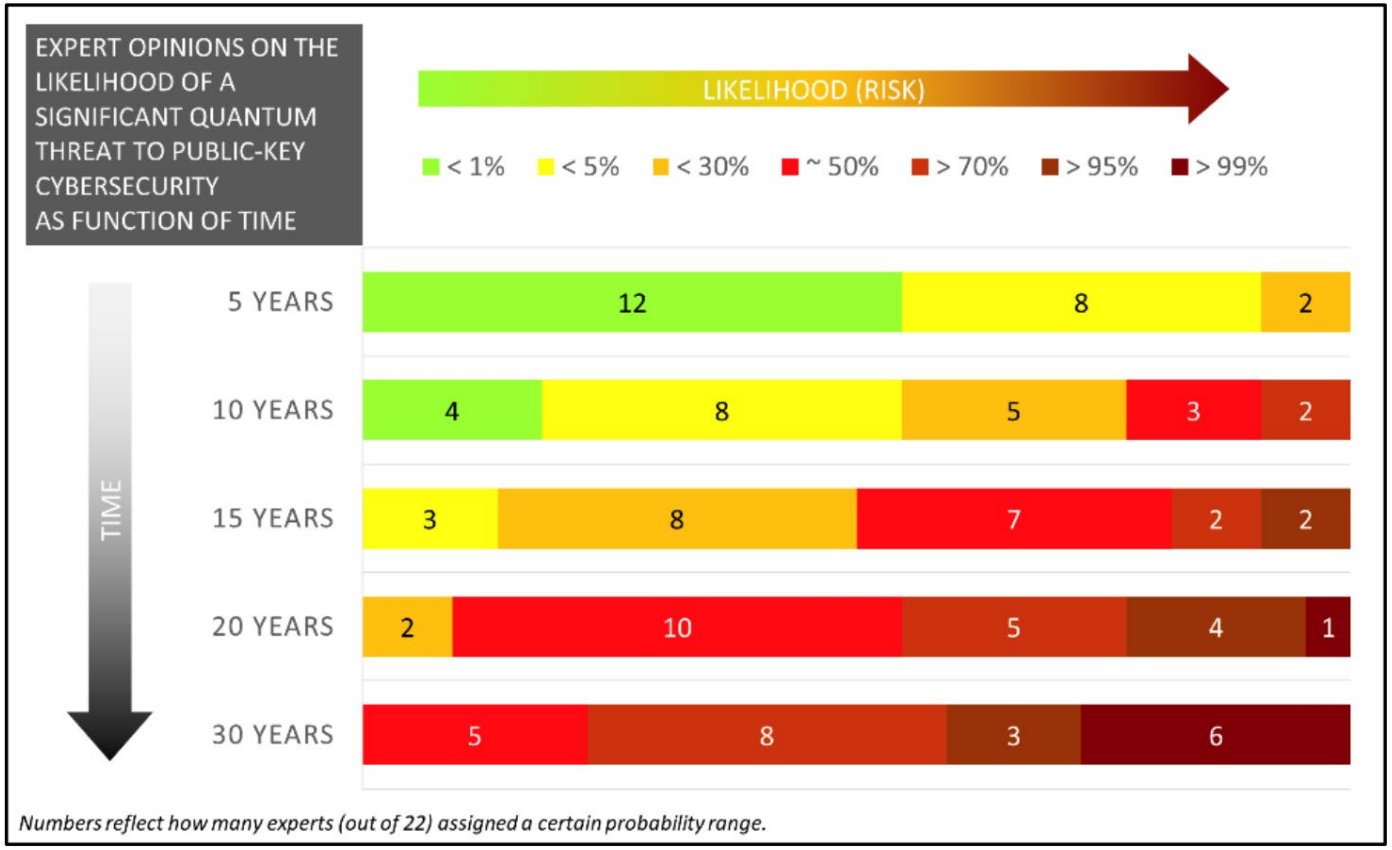
Quantum computers are 20 years in the future and always will be

“There is a 1 in 5 chance that some fundamental public-key crypto will be broken by quantum by 2029.”

– Dr. Michele Mosca, U. of Waterloo (2020)

See also: https://globalriskinstitute.org/publications/quantum-threat-timeline/

When will a Quantum Computer be Built?



Source: M. Mosca, M. Piani, Quantum Threat Timeline Report, Oct 2019
available at: https://globalriskinstitute.org/publications/quantum-threat-timeline/

Quantum Cryptography aka QKD

Using quantum technology to build cryptosystems

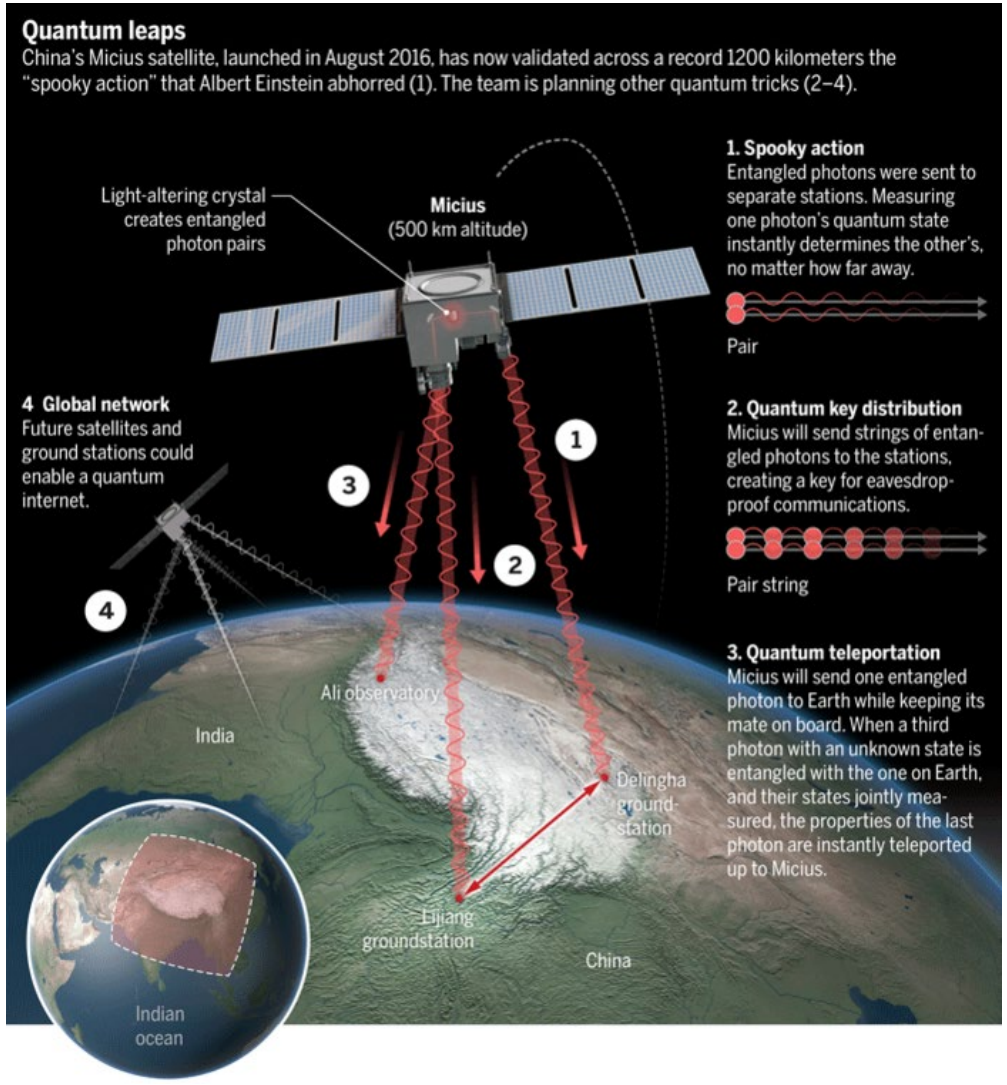
- Theoretically unconditional security guaranteed by the laws of physics

Limitations

- Can do encryption, but not authentication
- Quantum networks not very scalable
- Expensive and needs special hardware

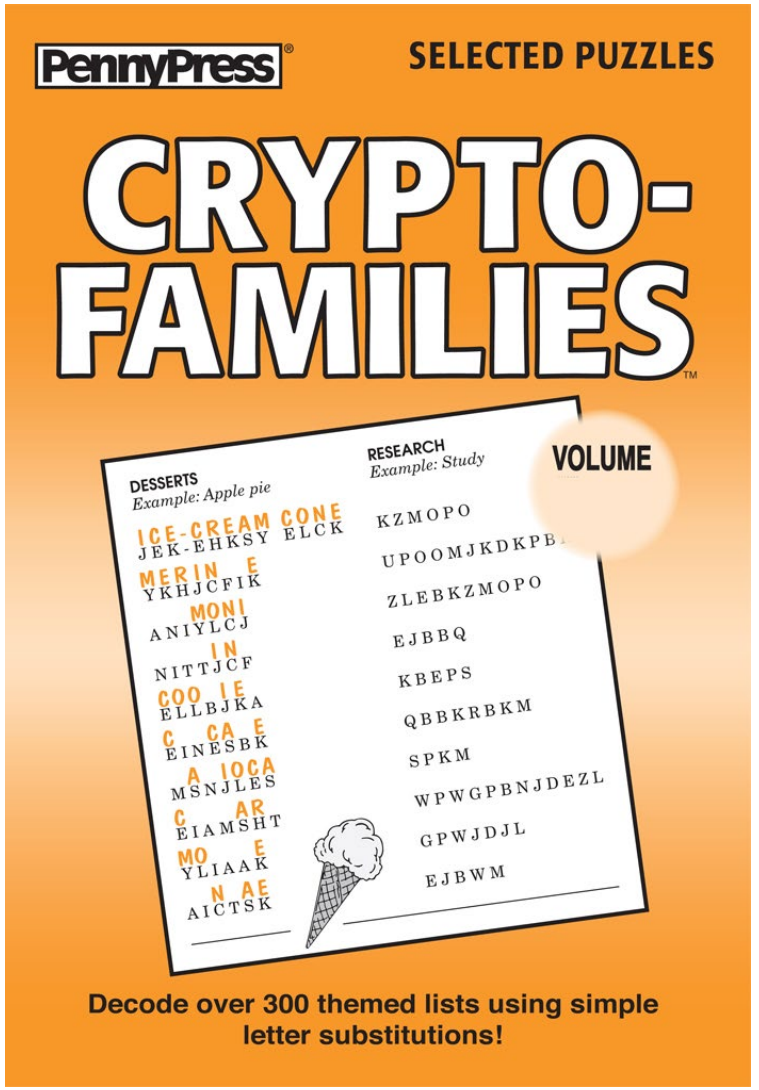
Lots of money being spent on “quantum”

This is NOT our focus

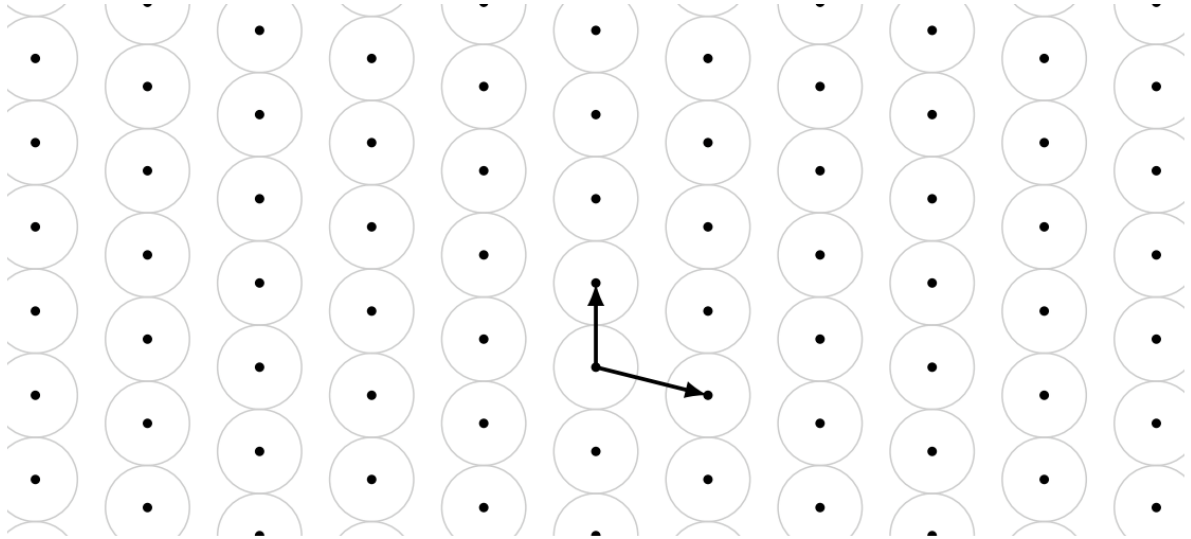


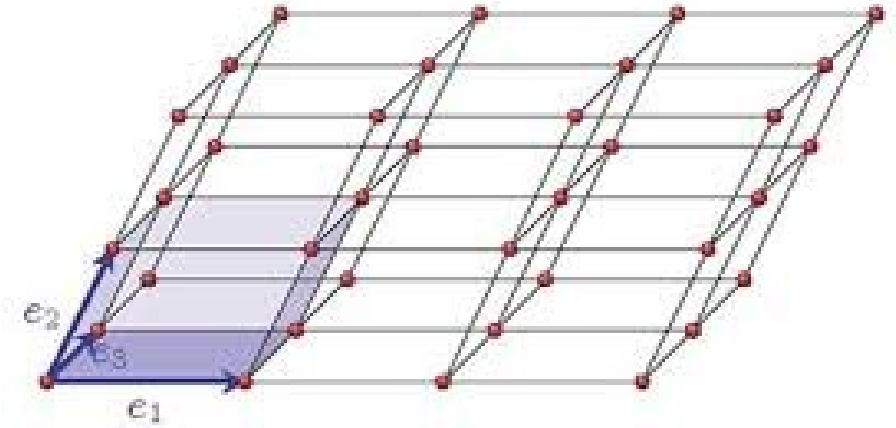
The Main Families

- Lattice-based crypto
- Code-based crypto
- Multivariate crypto
- Isogeny-based crypto
- Hash-based crypto
- Other….



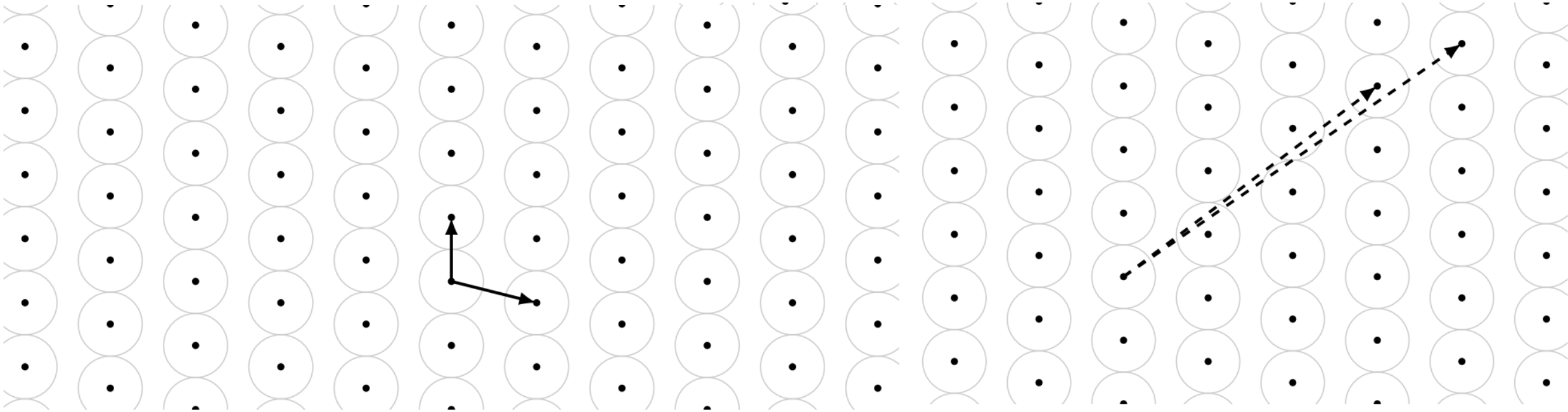
Intro to Lattices





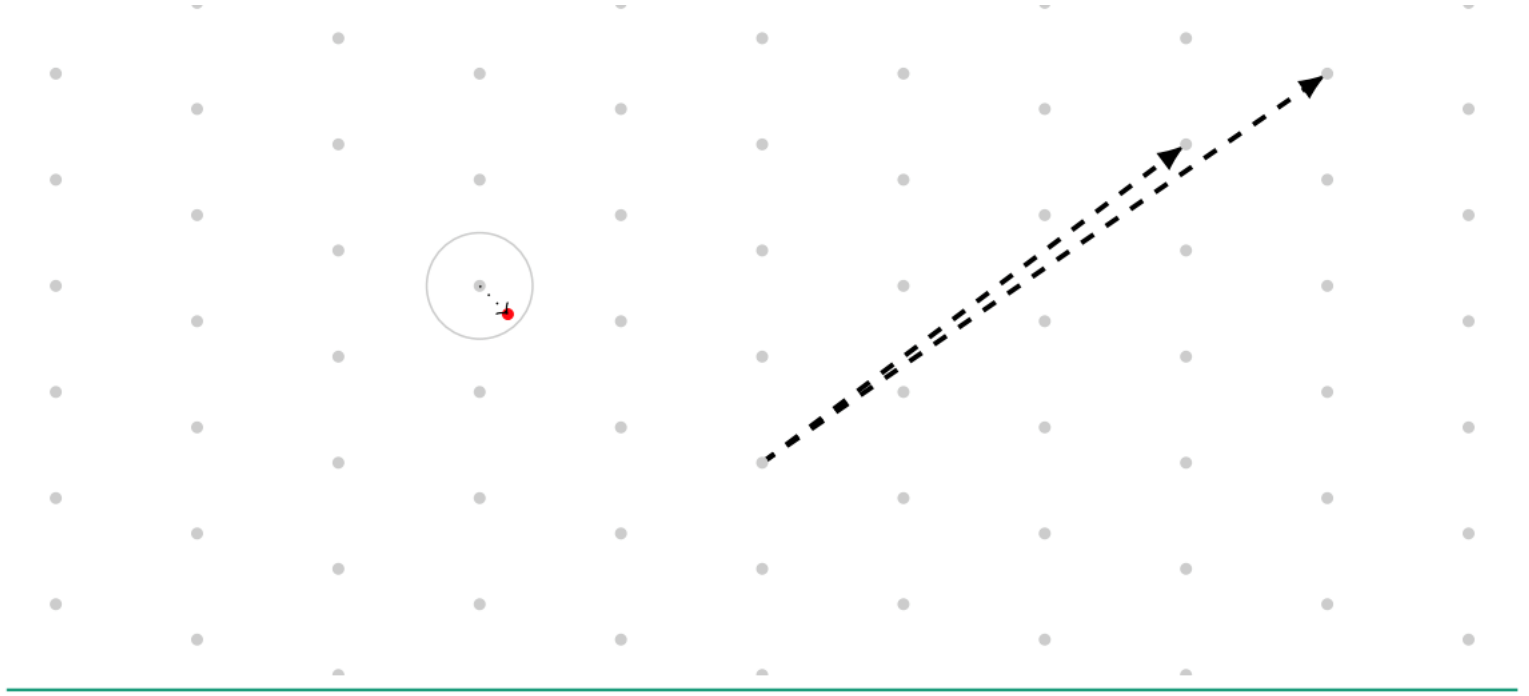
Basis vectors

Basis vectors



Any lattice point can be represented as a linear combination of the basis vectors

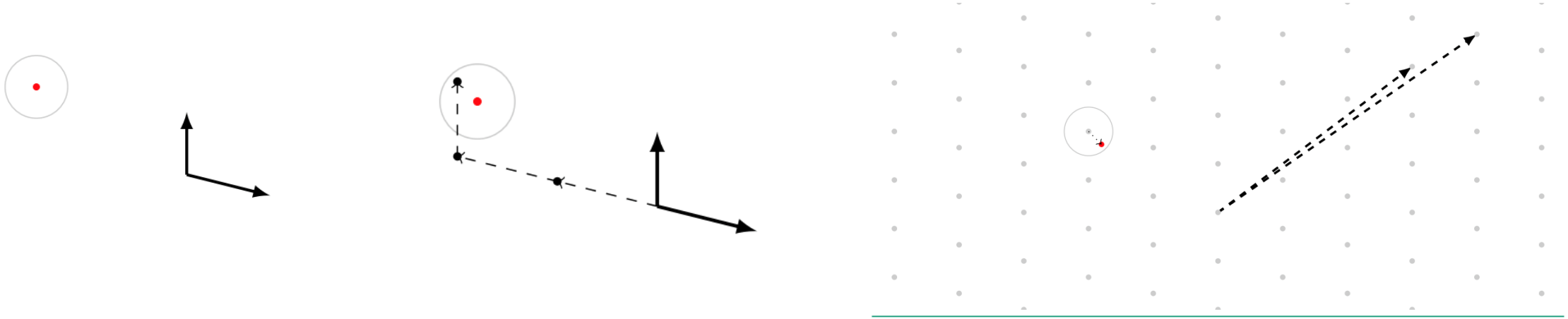
Closest Vector Problem



Given a random point, can we find the closest lattice point?

Closest Vector Problem: Given a point, and a basis, find the closest lattice point

Good and Bad bases



- Closest Vector Problem: Given a point, and a basis, find the closest lattice point
- The problem is much easier with a “good” basis

Linear Algebra

- We can represent the basis vectors of a lattice as a matrix
- Writing a lattice point as a linear combination of basis vectors is then linear algebra

Solving linear systems is easy

(use Gaussian elimination, polynomial time)

- Given

$$
\begin{aligned}
1s_1 + 2s_2 + 5s_3 + 2s_4 &= 9 \, mod \, 13 \\
12s_1 + 1s_2 + 1s_3 + 6s_4 &= 7 \, mod \, 13 \\
6s_1 + 10s_2 + 3s_3 + 6s_4 &= 1 \, mod \, 13 \\
10s_1 + 4s_2 + 12s_3 + 8s_4 &= 0 \, mod \, 13 \, .
\end{aligned}
$$

- Find s_1, s_2, s_3, s_4

Closest Vector Problem

- Given an arbitrary point – how do find the closest lattice point?

Solving linear systems with errors is hard

- Given

$$1s_1 + 2s_2 + 5s_3 + 2s_4 \approx 9 \, mod \, 13$$
$$12s_1 + 1s_2 + 1s_3 + 6s_4 \approx 7 \, mod \, 13$$
$$6s_1 + 10s_2 + 3s_3 + 6s_4 \approx 1 \, mod \, 13$$
$$10s_1 + 4s_2 + 12s_3 + 8s_4 \approx 0 \, mod \, 13 \, .$$

- Find s_1, s_2, s_3, s_4 , knowing that the solution is incorrect by ± 1 …
- The problem is called Learning With Errors (LWE)
- The associated one-way function is

$$f(s, e) = As + e$$

Where $s = (s_1, \ldots, s_4)$, A is the coefficient matrix, e is a vector of small errors

A (simplified) LWE Cryptosystem

- KeyGen()
  - Let *A* be a matrix for a lattice. Everything here is mod *q* (for some prime *q*)
  - Choose secret "short" vector *s* and "short" vector *e*. Compute $b = As + e$
  - The public key is *A* and *b*. The secret key is *s*
- Encrypt()
  - Choose "short" s' and e',e''. Compute $u = A^T s' + e'$ and $v = b^T s' + e'' + m * [q/2]$
  - Ciphertext is (*u, v*)
- Decrypt()
  - Alice computes $v - s^T u = b^T s' + e'' + m * [q/2] - s^T(As' + e')$

$$= (As + e)^T s' + e'' + m * \left[\frac{q}{2}\right] - s^T A^T s' + s^T e'$$

$$= s^T A^T s' + e^T s' + e'' + m * \left[\frac{q}{2}\right] - s^T A^T s' + s^T e'$$

$$= m * \left[\frac{q}{2}\right] + e^T s' + e'' + s^T e'$$

  - The error is "small" so *m* can be recovered

Lattice-based cryptosytems

- A lot of research work on lattices
- A huge number of crypto functionalities can be implemented via lattices
- Formal security proofs to hard mathematical problems
  - Though not for parameters used in cryptosystems!
- Can add structure to lattices to reduce key sizes
  - Increased avenue for attacks
  - Structured lattices seem to be the most promising general-purpose post-quantum cryptosystems
- Efficient to implement in practice

Intro to code-based crypto

- Error-correcting codes are used in telecommunications to correct errors
- Repetition code: encode a message m = 10110010 as

$$1111000011111111000000011110000$$

- This code can correct up to 1 error (per encoded message bit)
- How could we modify the encoding so it corrects more errors?

Generator Matrices

- For the repetition code, a generator matrix is just $G = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$
- Represent the message as a vector m = [1 0 1 1 0 0 1 0]
- Then $Gm = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \end{bmatrix}$
- There exist much more efficient codes: Goppa codes, Reed-Solomon codes, etc
- Codes have decoding algorithms, which take an arbitrary vector, and find the closest codeword.

A (simplified) code-based encryption system

- KeyGen()
  - Alice chooses a code, i.e. a generator matrix *G* with an efficient decoding algorithm
  - She hides it by setting her public key to be $\hat{G} = SGP$, where *S* is invertible, and *P* is a permutation matrix
- Encrypt()
  - Bob encrypts a message *m* by computing $m\hat{G}$
  - Bob selects an error vector *e*, and the ciphertext is $c = m\hat{G} + e$
- Decrypt()
  - Alice computes $cP^{-1} = m\hat{G}P^{-1} + eP^{-1}$
    $= mSG + e'$
  - Alice can correct for *e'*, obtaining *mSG*. She then decodes to obtain *mS*. As she knows S^{-1}, she can recover *m*

- An attacker has to try and find a decoding algorithm from the scrambled generator matrix, which appears to look like a random matrix

Code-based Cryptosystems

- Old: The McEliece cryptosystem was proposed in 1979, and is still unbroken
- Code-based schemes tend to have large public keys, but small ciphertexts
- Can add more structure to the codes, and get smaller keys
  - Run a risk of additional structure leads to a new attack surface
- Almost all code-based signature schemes have been broken
- Implementations are efficient, since everything is linear algebra
- The ideas behind code-based schemes are very similar to the ideas in lattice-based crypto

Multivariate Crypto

Solving a system of m multivariate polynomial equations in n variables over $\mathbb{F}_q$.

This is called the

MP Problem

the MP problem is an *NP-Complete* problem even for multivariate *quadratic* system and $q = 2$

Example with $m = 3, n = 3$:

$$5x_1^3x_2x_3^2 + 17x_2^4x_3 + 23x_1^2x_2^4 + 13\,x_1 + 12x_2 + 5 = 0$$

$$12x_1^3x_2^3x_3 + 15x_1x_3^3 + 25x_2x_3^3 + 5\,x_1 + 6x_3 + 12 = 0$$

$$28x_1x_2x_3^4 + 14x_2^3x_3^2 + 16x_1x_3 + 32\,x_2 + 7x_3 + 10 = 0$$

It is very easy to evaluate multivariate functions

A multivariate signature scheme

- Keygen()
  - Choose a “random” multivariate *f* such that f^{-1} is secretly known
  - The public key is f. The secret key is f^{-1}
- Signing()
  - Given a message m, compute $s = f^{-1}(m)$
  - The signature is *s*
- Verifying()
  - Given *s*, compute $f(s) = f\left(f^{-1}(m)\right) = m$
  - Accept if you get *m* and reject otherwise

- How to choose such an *f*?
  - Many failed attempts
  - Over $\mathbb{F}_{q^n}$, the map induced by $x \to x^q$ is a linear map. Can show $g: x \to x^{q^\alpha+1}$ is invertible for certain α. You then scramble *g* by composing it with invertible maps on the left and right.

Advantages and disadvantages

- Multivariate crypto is very efficient – particularly verification
- Security rests on known hard problem – the MQ problem
- Multivariate systems tend to have large public keys and small signatures
- As usual, can introduce some structure to get the keys smaller
- Pretty much all attempts at multivariate encryption have failed
- Many multivariate signature schemes have been broken, so many are nervous about the field
  - There are several unbroken schemes that have been around awhile, e.g. UOV, HFEv-

NIST PQC Milestones and Timelines

2016

Determined criteria and requirements, published NISTIR 8105

Announced call for proposals

2017

Received 82 submissions

Announced 69 1st round candidates

2018

Held the 1st NIST PQC standardization Conference

2019

Announced 26 2nd round candidates, NISTIR 8240

Held the 2nd NIST PQC Standardization Conference

2020

Announced 3rd round 7 finalists and 8 alternate candidates. NISTIR 8309

2021

Hold the 3rd NIST PQC Standardization Conference

2022-2023

Release draft standards and call for public comments

Evaluation Criteria

Security – against both classical and quantum attacks

Level	Security Description
I	At least as hard to break as AES128 (exhaustive key search)
II	At least as hard to break as SHA256 (collision search)
III	At least as hard to break as AES192 (exhaustive key search)
IV	At least as hard to break as SHA384 (collision search)
V	At least as hard to break as AES256 (exhaustive key search)

NIST asked submitters to focus on levels 1,2, and 3. (Levels 4 and 5 are for very high security)

Performance – measured on various classical platforms

Other properties: Drop-in replacements, Perfect forward secrecy, Resistance to side-channel attacks, Simplicity and flexibility, Misuse resistance, etc.

A Worldwide Effort

25 Countries

16 States

6 Continents

The 1st Round

- A lot of schemes quickly attacked!
- Many similar schemes (esp. lattice KEMs)
- 1st NIST PQC Standardization workshop
- Over 300 ”official comments” and 900 posts on the pqc-forum
- Research and performance numbers

- After a year: 26 schemes move on

	Signatures	KEM/Encryption	Overall
Lattice-based	5	21	26
Code-based	2	17	19
Multi-variate	7	2	9
Stateless Hash or Symmetric based	3		3
Other	2	5	7
Total	**19**	**45**	**64**

The 2nd Round

- 4 merged submissions
- Maintained diversity of algorithms
- Cryptanalysis continues
- LAC, LEDAcrypt, RQC, Rollo, MQDSS, qTESLA, LUOV all broken
- 2nd NIST PQC Standardization workshop
- More benchmarking and real world experiments

- After 18 months: 15 submissions move on

	Signatures	KEM/Encryption	Overall
Lattice-based	3	9	12
Code-based		7	7
Multi-variate	4		4
Stateless Hash or Symmetric based	2		2
Isogeny		1	1
Total	**10**	**16**	**26**

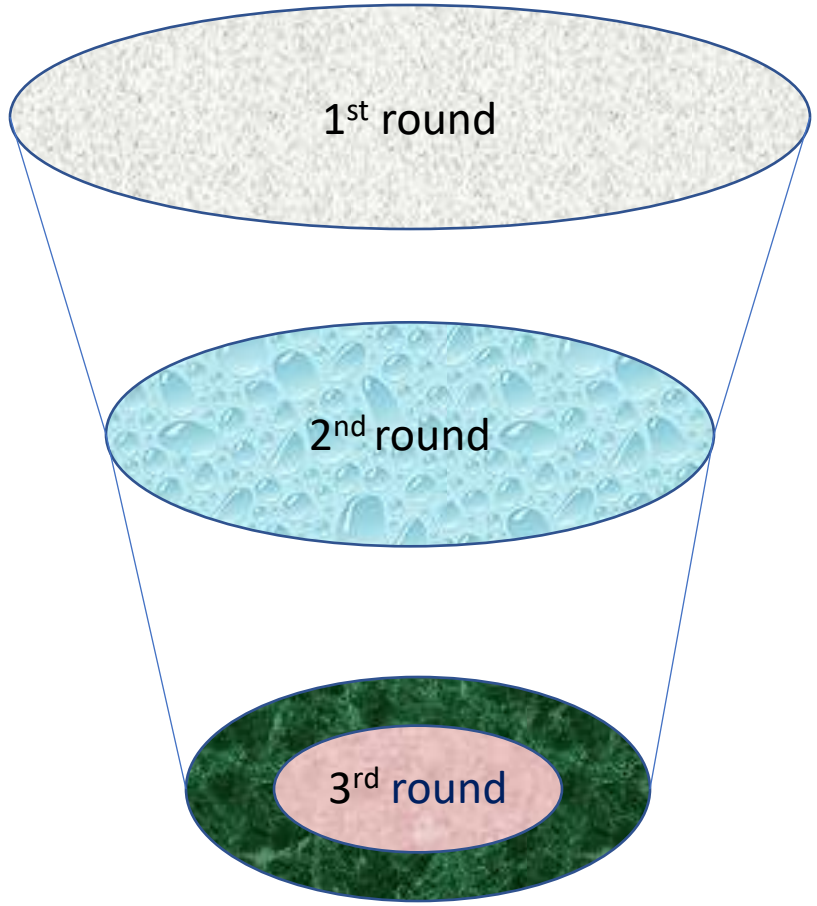
Challenges and Considerations in Selecting Algorithms

Security
- Security levels offered
- (confidence in) security proof
- Any attacks
- Classical/quantum complexity

Performance
- Size of parameters
- Speed of KeyGen, Enc/Dec, Sign/Verify
- Decryption failures

Algorithm and implementation characteristics
- IP issues
- Side channel resistance
- Simplicity and clarity of documentation
- Flexible

Other
- Round 2 changes
- Official comments/pqc-forum discussion
- Papers published/presented



The 3rd Round Finalists and Alternates

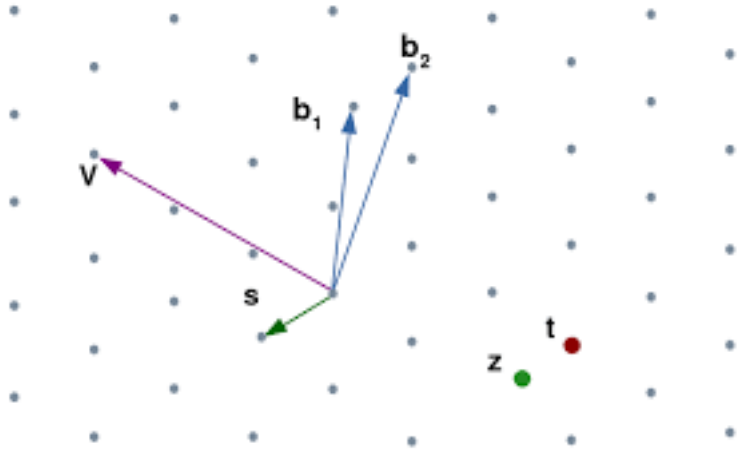
- NIST selected 7 Finalists and 8 Alternates
  - Finalists: most promising algorithms we expect to be ready for standardization at end of 3rd round
  - Alternates: candidates for potential standardization, most likely after another (4th) round
- KEM finalists: Kyber, NTRU, SABER, Classic McEliece
- Signature finalists: Dilithium, Falcon, Rainbow

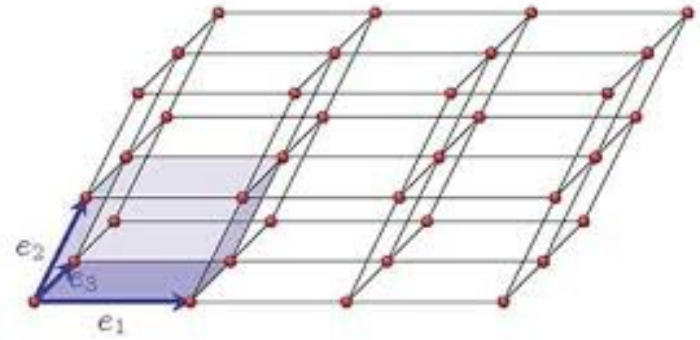
- KEM alternates: Bike, FrodoKEM, HQC, NTRUprime, SIKE
- Signature alternates: GeMSS, Picnic, Sphincs+

	Signatures		KEM/Encryption		Overall	
Lattice-based	2		3	2	5	2
Code-based			1	2	1	2
Multi-variate	1	1			1	1
Stateless Hash or Symmetric based		2				2
Isogeny				1		1
Total	3	**3**	4	**5**	7	**8**

Lattice-based KEMs

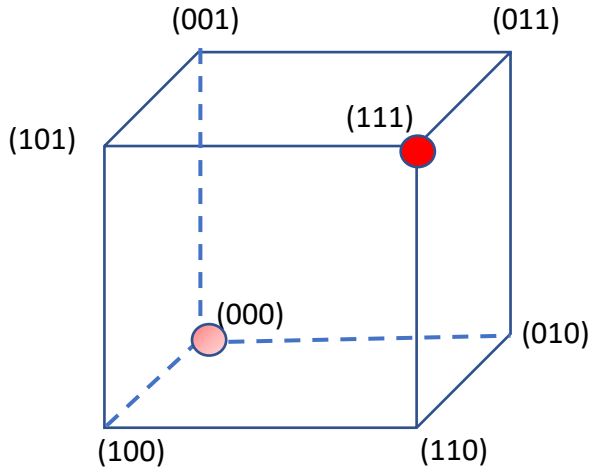
- Crystals-Kyber
  - Great all-around → Finalist
- Saber
  - Great all-around → Finalist
- NTRU
  - Not quite as efficient, but older, IP situation → Finalist
- NTRUprime
  - Different design choice and security model → Alternate
- FrodoKEM
  - Conservative/Backup → Alternate

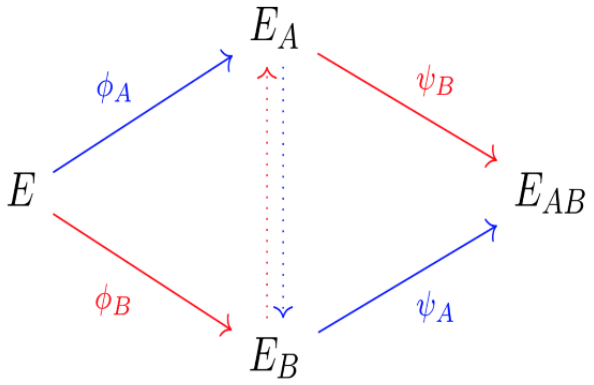




Isogeny- and Code-based KEMs

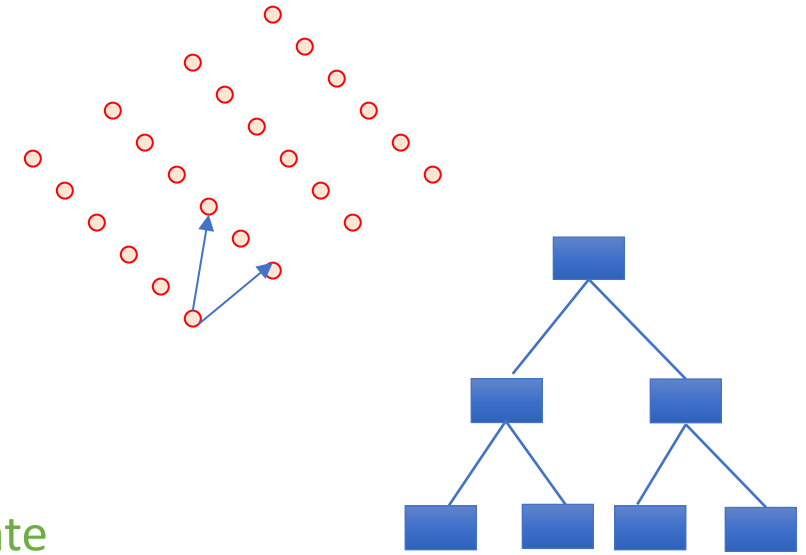
- Classic McEliece
  - Oldest submission, large public keys but small ciphertexts→ Finalist
- BIKE
  - Good performance, CCA security?, more time to be stable → Alternate
- HQC
  - Better security analysis/larger keys (than BIKE) → Alternate
- SIKE
  - Newer security problem, an order slower → Alternate





The Signatures

- Dilithium and Falcon
  - Both balanced, efficient lattice-based signatures
  - coreSVP security higher?
  - → Finalists



- SPHINCS+ and Picnic
  - SPHINCS+ is stable, conservative security, larger/slower → Alternate
  - Picnic not stable yet, but has lots of potential → Alternate

- Rainbow and GeMSS
  - Both have large public keys, small signatures.
    Rainbow a bit better → Finalist, GeMSS → Alternate
  - There have been recent attacks on both Rainbow and GeMSS

$$
\begin{aligned}
p^{(1)}(x_1,\ldots,x_n) &= \sum_{i=1}^{n}\sum_{j=i}^{n} p_{ij}^{(1)}\cdot x_i x_j \quad + \quad \sum_{i=1}^{n} p_i^{(1)}\cdot x_i + p_0^{(1)} \\
p^{(2)}(x_1,\ldots,x_n) &= \sum_{i=1}^{n}\sum_{j=i}^{n} p_{ij}^{(2)}\cdot x_i x_j \quad + \quad \sum_{i=1}^{n} p_i^{(2)}\cdot x_i + p_0^{(2)} \\
&\vdots \\
p^{(m)}(x_1,\ldots,x_n) &= \sum_{i=1}^{n}\sum_{j=i}^{n} p_{ij}^{(m)}\cdot x_i x_j \quad + \quad \sum_{i=1}^{n} p_i^{(m)}\cdot x_i + p_0^{(m)}
\end{aligned}
$$

Timeline

- The 3rd round will last 12-18 months
  - NIST will then select which finalist algorithms to standardize
  - NIST will also select which alternates to keep studying in a 4th round (*)
  - The 4th round will similarly be 12-18 months
  - NIST may decide to consider new schemes – details to come
- NIST will hold a 3rd PQC Standardization workshop ~ spring 2021
- We expect to release draft standards for public comment in 2022-2023
- The finalized standard will hopefully be ready by 2024

Research Challenges

- Many important topics to be studied:
  - Security proofs in both the ROM and QROM
  - Does the specific ring/module/field choice matter for security?
    - Or choice of noise distribution?
    - Does “product” or “quotient” style LWE matter?
  - Finer-grained metrics for security of lattice-based crypto (coreSVP vs. real-world security)
  - Are there any important attack avenues that have gone unnoticed?
  - Side-channel attacks/resistant implementations for finalists and alternates
  - More hardware implementations
  - Ease of implementations – decryption failures, floating point arithmetic, noise sampling, etc.
- Specific algorithm questions
  - Decoding analysis for BIKE, category 1 security levels for Kyber/Saber/Dilithium, algebraic cryptanalysis of cyclotomics for lattices, etc…

Other Challenges

- Many other challenges to work on
  - IP issues
  - Continued performance benchmarking in different platforms and environments
    - For hardware – NIST suggested Artix-7 and Cortex M4 (with all options) for easier comparison
  - Real world experiments
    - How do these algorithms work in actual protocols and applications.
      - Are some key sizes too large?

Stateful Hash Based Signatures for Early Adoption

Stateful hash-based signatures were proposed in 1970s

- Rely on assumptions on hash functions, that is, not on number theory complexity assumptions
- It is essentially limited-time signatures, which require state management

NIST specification on stateful hash-based signatures

- NIST SP 800-208 "*Recommendation for Stateful Hash-Based Signature Schemes*"

Internet Engineering Task Force (IETF) has released two RFCs on hash-based signatures

- RFC 8391 "XMSS: eXtended Merkle Signature Scheme" (By Internet Research Task Force (IRTF))
- RFC 8554 "Leighton-Micali Hash-Based Signatures" (By Internet Research Task Force (IRTF))

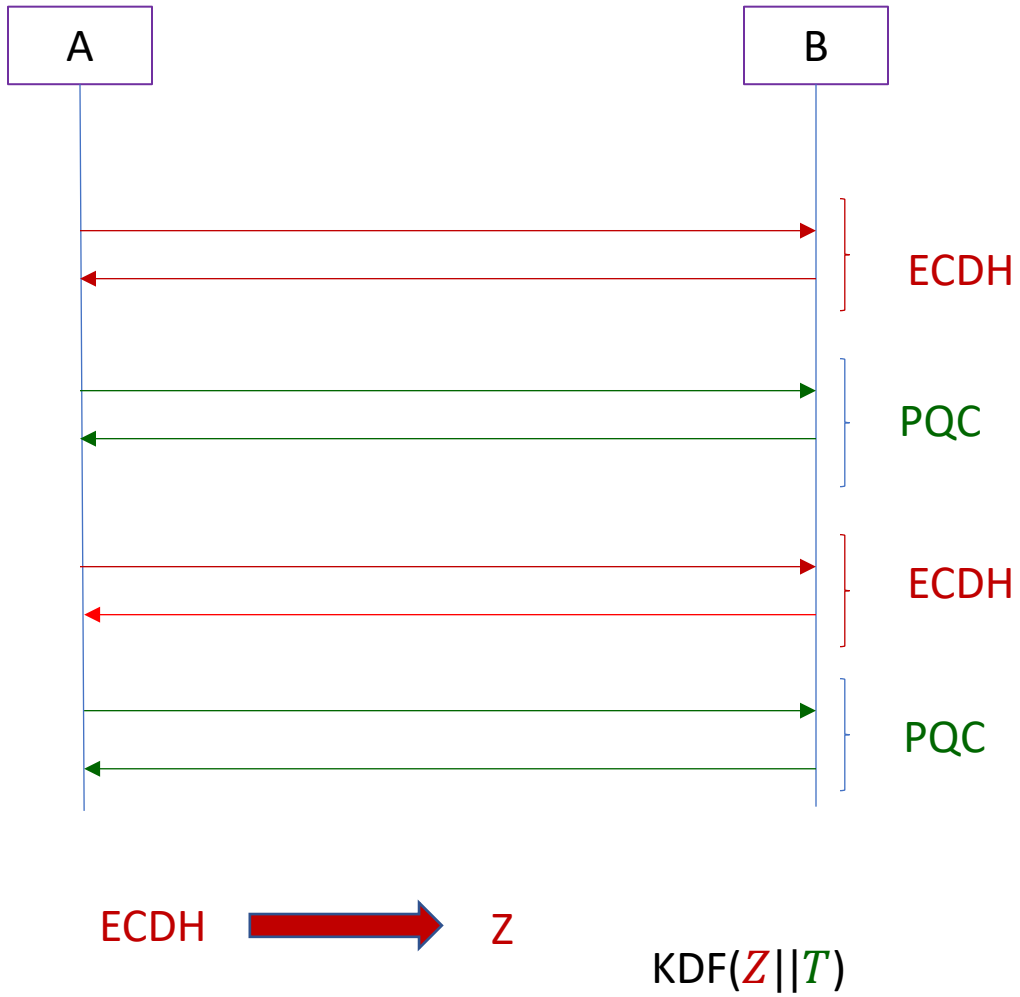
ISO/IEC JTC 1 SC27 WG2 Project on hash-based signatures

- Stateful hash-based signatures will be specified in ISO/IEC 14888 Part 4
- It is in the 1st Working Draft stage

Hybrid mode – An approach for migration

NIST SP800-56C Rev. 2
Recommendation for Key-Derivation Methods in Key-Establishment Schemes
August 2020

> "In addition to the currently approved techniques for the generation of the shared secret *Z* … this Recommendation permits the use of a "hybrid" shared secret of the form Z' = *Z* || *T*, a concatenation consisting of a "standard" shared secret *Z* that was generated during the execution of a key-establishment scheme (as currently specified in [SP 800-56A] or [SP 800-56B]) followed by an auxiliary shared secret *T* that has been generated using some other method"



NIST Transition Guideline for PQC?

NIST has published transition guidelines for algorithms and key lengths

NIST SP 800-131A Revision 2 "Transitioning the Use of Cryptographic Algorithms and Key Lengths" - Examples

- Three-key Triple DES
  - Encryption - Deprecated through 2023 Disallowed after 2023
  - Decryption - Legacy use
- SHA-1
  - Digital signature generation - Disallowed, except where specifically allowed by NIST protocol-specific guidance
  - Digital signature verification - Legacy use
  - Non-digital signature applications – Acceptable
- Key establishment methods with strength < 112 bits (e.g. DH mod p, $|p| < 2048$)
  - Disallowed

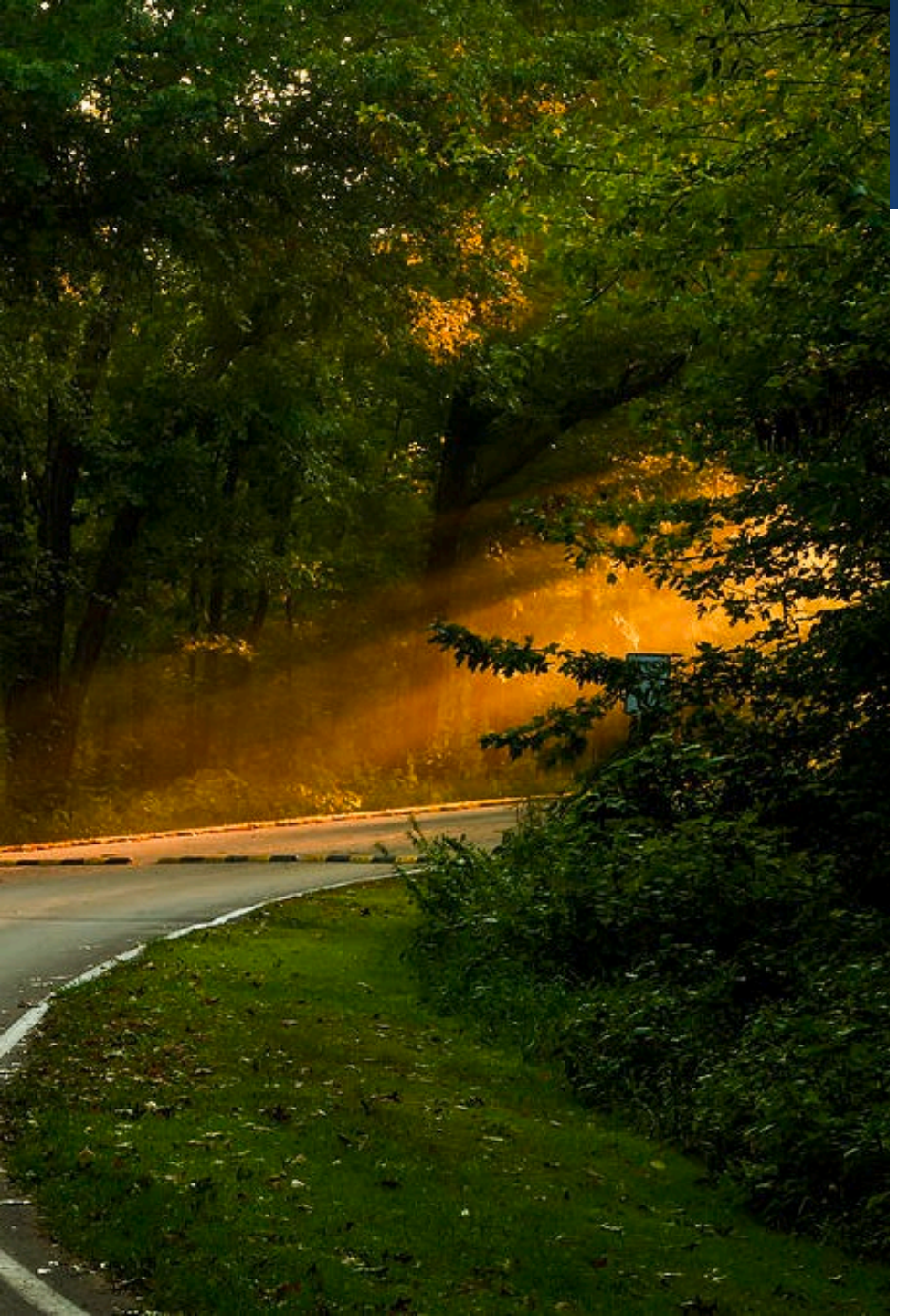
NIST will provide transition guidelines to PQC standards

- The timeframe will be based on a risk assessment of quantum attacks
- NCCoE hosted a workshop on *Considerations in Migrating to Post-Quantum Cryptographic Algorithms* on October 7

What can organizations do now?

- Perform a quantum risk assessment within your organization
  - Identify information assets and their current crypto protection
  - Identify what ‘x’, ‘y’, and ‘z’ might be for you – determine your quantum risk
  - Prioritize activities required to maintain awareness, and to migrate technology to quantum-safe solutions
- Evaluate vendor products with quantum safe features
  - Know which products are not quantum safe
  - Ask vendors for quantum safe features in procurement templates
- Develop an internal knowledge base amongst IT staff
- Track developments in quantum computing and quantum safe solutions, and to establish a roadmap to quantum readiness for your organization

- Act now – it will be less expensive, less disruptive, and less likely to have mistakes caused by rushing and scrambling



Conclusion

- We can start to see the end?
- NIST is grateful for everybody's efforts
- Check out www.nist.gov/pqcrypto
  - Sign up for the pqc-forum for announcements & discussion
  - send e-mail to pqc-comments@nist.gov