Standards for Post-Quantum Cryptography

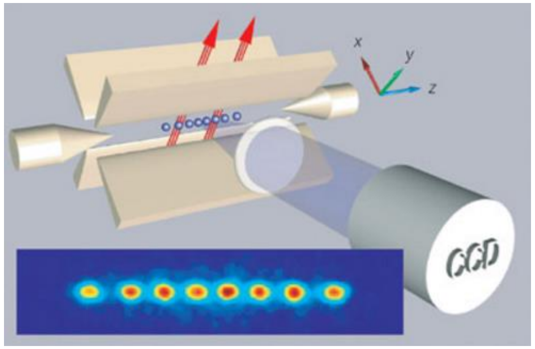
Yi-Kai Liu / NIST PQC team

http://indianajones.wikia.com/wiki/Raiders_of_the_Lost_Ark

Quantum Computers

- Quantum mechanics
  - Behavior of small objects: atoms, electrons, photons
  - Quantum superpositions: $|\psi_{cat}\rangle = |alive\rangle + |dead\rangle$, $|\psi_{qubit}\rangle = |0\rangle + |1\rangle$
  - Interference: combine $|0\rangle + |1\rangle$ with $|0\rangle - |1\rangle$, get $|0\rangle$

  - When an object is observed, the quantum superposition collapses
  - This is why large objects do not behave quantumly
  - Major challenge in building a quantum computer



R.Blatt & D. Wineland, Nature 453, 1008-1015 (19 June 2008)

Quantum Computers

- Potentially much more powerful than classical computers
  - Conjecture: A classical computer needs **exponential time** to simulate a quantum computer (in the general case)
- Exponential speedups for some interesting problems
  - Simulating the dynamics of molecules, superconductors, photosynthesis…?
  - **Factoring large integers (Shor's algorithm)**
  - **Discrete logarithms in any abelian group (Shor's algorithm)**
- And some polynomial speedups
  - **Unstructured search (Grover's alg.), collision finding**

Who Cares?

- Quantum computers would break most of our public-key crypto
  - RSA, Diffie-Hellman key exchange, elliptic curve crypto
  - TLS, digital certificates, IPSec
- Symmetric crypto would be affected, but not broken
  - “Keep using AES, but double the key length”
  - (Actually, it’s more complicated than that)

Who Cares?

- Fortunately, large quantum computers don't exist yet
  - Small ones do exist, but can they scale up?
  - Michele Mosca (http://eprint.iacr.org/2015/1075): "1/2 chance of breaking RSA-2048 by 2031"

- Unfortunately, 2031 is not that far away
  - How long does today's data need to remain secure? 5-10 years?
  - How long does it take to deploy new crypto software? 5-10 years?

Post-Quantum Cryptography

Cryptosystems	Hard problem	Trapdoor
Lattice-based	Finding short vectors in a high-dimensional lattice	Nice basis for the lattice (short, almost-orthogonal vectors)
Code-based	Decoding a random binary linear code	Linear transformations that reveal structure of the code
Multivariate	Solving a random system of multivariate quadratic equations over a finite field	Linear transformations that reveal structure of the equations

Post-Quantum Cryptography

- **Hash-based signatures**
  - Simple: uses only a hash function, doesn’t need a trapdoor
  - Caveat: signing algorithm has to update an internal data structure every time it signs a message
- **Isogenies of supersingular elliptic curves**
  - Useful for key exchange?
- **Quantum key distribution**
  - Information-theoretic security
  - Requires optical fiber, distance limited to ~200 km

Post-Quantum Cryptography

- **How do we know a cryptosystem is secure?**
  - Cryptanalysis: what are the best known attacks?
  - Security proofs: based on some hardness assumption?

- **How well do these cryptosystems work in practice?**
  - Size of keys, time needed for each operation
  - Ease of implementation, how to set the parameters
  - Does it fit nicely with TLS, other higher-level protocols?
  - Vulnerabilities to side channel attacks?

- **There's a conference about this:**

PQCrypto 2016
Fukuoka, Japan
February 24-26, 2016

Lattice-Based Cryptography

Lattice-Based Encryption Schemes

- **NTRUEncrypt**
  - Developed circa 1996 by Hofstein, Pipher and Silverman, commercially available
- **Regev's encryption scheme**
  - Based on LWE problem ("learning with errors") (2005)
    - Solving a noisy system of linear equations modulo p
  - Theoretical security guarantees
    - Solving average-case instances of LWE is at least as hard as solving worst-case instances of SIVP ("lattice short independent vectors problem")
  - When instantiated with ideal lattices, this looks sort of like NTRUEncrypt
    - Ideal lattice: an ideal in a ring, for example, Z[X] / (X^n+1)
    - This gives smaller key sizes, without compromising security?

LWE Problem (“learning with errors”)

- Secret s in $(Z_q)^n$
  - q = poly(n)
- Given samples (a,b) in $(Z_q)^n$ x Z_q
  - a is uniformly random
  - $b = a^T s + e$, where e is Gaussian distributed, w/ std dev q/poly(n)
- Can we determine s?
  - “Decoding a random linear code over Z_q”

- **Claim: samples (a,b) look pseudorandom!**

Regev's Encryption Scheme

- Private key: s in $(Z_q)^n$
- Public key: LWE samples (a_i, b_i) in $(Z_q)^n$ x Z_q (for i = 1,…,m)
  - Where we let m ~ n log n
  - Recall $b_i = a_i^T s + e_i$

- Encryption: Given a single bit x in {0,1}
  - Choose a random subset S of {1,…,m}
  - Output $a = \Sigma_{i \text{ in } S}\, a_i$ and $b = (0.5)(q-1)x + \Sigma_{i \text{ in } S}\, b_i$

- Decryption: Given (a,b)
  - Compute $b – a^T s = (0.5)(q-1)x + \Sigma_{i \text{ in } S}\, e_i$
  - Round this to either 0 or (0.5)(q-1), mod q
  - Output either x = 0 or x = 1, accordingly

Lattice-Based Signatures

- **“Hash-then-sign” approach (GGH ’97)**
- Lattice L
- Public key: A “hard” basis B
- Private key: A “good” basis T (the “trapdoor”)

- Signing: Given message m,
  - Hash it to a point x in R^n
  - Find the lattice vector v in L that lies closest to x
  - Output (x,v)

- Verification: Given (m,x,v),
  - Check that m hashes to x, v is in L, and v is close to x

Lattice-Based Signatures

- **NTRUSign**
  - Developed circa 2003
  - Broken by Nguyen and Regev in 2006 (“learning a parallelipiped”) – each signature leaks some information about the secret key
  - Patched by adding “perturbations” to the signatures

- **GPV signatures**
  - Uses “Gaussian sampling” (Gentry, Peikert, Vaikuntanathan, 2007)
    - Provably secure variant of NTRUSign, but less efficient
    - Based on SIS problem (“small integer solutions”) – random subset sum with vectors modulo p
    - Has worst-case to average-case reduction from lattice problems

Lattice-Based Signatures

- **Signatures using Fiat-Shamir heuristic**
  - More efficient than GPV approach
  - Provably secure based on hardness of SIS problem, in random oracle model
  - Lyubashevsky (2011), and several follow-on works…

Cryptanalysis

- Lattice basis reduction (in polynomial time)
  - Try to find a basis consisting of short, nearly-orthogonal vectors
  - LLL algorithm: finds a $2^{O(n)}$-approximation to the shortest vector in the lattice
  - Block-KZ reduction, follow-on work by Schnorr, Nguyen…
- Sieving, enumeration (in exponential time)
  - Find the shortest vector in the lattice
  - Extreme pruning (Gama, Nguyen, Regev, 2010)
- Algorithms for LWE and SIS problems
  - List merging (Lyubashevsky, 2004)
  - Linearization (Arora, Ge, 2011)

Quantum Cryptanalysis?

- Quantum algorithms for problems in number fields
  - Unit group, class group, principal ideal problem
  - Running time is polynomial in the degree
  - (Eisentrager, Hallgren, Kitaev, Song, 2014; Biasse, Song, 2016)

- Quantum attack on the Soliloquy cryptosystem
  - (Campbell, Groves, Shepherd, 2014)
    - Commentary: http://web.eecs.umich.edu/~cpeikert/soliloquy.html

- Quantum speed-ups of classical lattice algorithms
  - (Laarhoven, Mosca, van de Pol, 2013)

Issues and Open Questions

- Are ideal lattices just as hard as general lattices?
  - Clearly there is some additional structure there…
  - In the security proofs, we assume these problems are hard

- How hard are the LWE and SIS problems, for the parameters we use in practice?
  - Parameters are chosen based on experimental cryptanalysis
  - Worst-case to average-case reduction doesn't say anything meaningful in this regime

- How to implement Gaussian samplers?
  - Need good entropy, how to test this, what about discretization errors, need constant-time implementations to resist side-channel attacks…

Multivariate Quantum-Resistant Cryptography

Daniel Smith-Tone

NIST & UofL

3 February, 2016

Multivariate Public Key Cryptography

Nonlinear Systems

Base the security of the cryptographic scheme on the difficulty of finding a preimage of some element in the range of a system of nonlinear equations.

Multivariate Public Key Cryptography

Nonlinear Systems

Base the security of the cryptographic scheme on the difficulty of finding a preimage of some element in the range of a system of nonlinear equations.

The fundamental problem has been studied for at least hundreds of years and seems difficult.

Multivariate Public Key Cryptography

Nonlinear Systems

Base the security of the cryptographic scheme on the difficulty of finding a preimage of some element in the range of a system of nonlinear equations.

The fundamental problem has been studied for at least hundreds of years and seems difficult.
Relies in essense on algebraic geometry.

Systems of Quadratic Equations

We can restrict ourselves to systems of quadratic equations.

Systems of Quadratic Equations

We can restrict ourselves to systems of quadratic equations.

Key Size

A system of m quadratic equations in n unknowns consists of $m(\binom{n}{2}+n)$ monomials. Key sizes are (in general) proportional to mn^2. If $m \approx n$, key sizes scale like n^3.

Systems of Quadratic Equations

We can restrict ourselves to systems of quadratic equations.

Key Size

A system of m quadratic equations in n unknowns consists of $m(\binom{n}{2} + n)$ monomials. Key sizes are (in general) proportional to mn^2. If $m \approx n$, key sizes scale like n^3.

Underlying Problem

The $\mathcal{MQ}$ problem of solving systems of quadratic equations over a field is NP-complete.
At least there is a chance that cryptanalysis may be difficult.

Multivariate Public Key Cryptosystem

Mechanics

A Multivariate scheme includes a few values publicly known: A polynomial ring, $R[x_1, \ldots, x_n]$, and a collection of polynomials:

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \end{bmatrix} = \begin{bmatrix} \alpha_{1,1,1}x_1^2 + \alpha_{1,2,1}x_1x_2 + \cdots + \alpha_{6,6,1}x_6^2 \\ \alpha_{1,1,2}x_1^2 + \alpha_{1,2,2}x_1x_2 + \cdots + \alpha_{6,6,2}x_6^2 \\ \alpha_{1,1,3}x_1^2 + \alpha_{1,2,3}x_1x_2 + \cdots + \alpha_{6,6,3}x_6^2 \\ \alpha_{1,1,4}x_1^2 + \alpha_{1,2,4}x_1x_2 + \cdots + \alpha_{6,6,4}x_6^2 \\ \alpha_{1,1,5}x_1^2 + \alpha_{1,2,5}x_1x_2 + \cdots + \alpha_{6,6,5}x_6^2 \\ \alpha_{1,1,6}x_1^2 + \alpha_{1,2,6}x_1x_2 + \cdots + \alpha_{6,6,6}x_6^2 \end{bmatrix}$$

Prototypical Multivariate Public Key Scheme

Isomorphism of Polynomials

Let f be an efficiently invertible (in some sense) system of m quadratic formulae in n variables over some field $\mathbb{F}_q$. Let U and T be $\mathbb{F}_q$-linear maps of dimension n and m, respectively.
Let $P = T \circ f \circ U$.

Prototypical Multivariate Public Key Scheme

Isomorphism of Polynomials

Let f be an efficiently invertible (in some sense) system of m quadratic formulae in n variables over some field $\mathbb{F}_q$. Let U and T be $\mathbb{F}_q$-linear maps of dimension n and m, respectively.
Let $P = T \circ f \circ U$.

Since P is simply a representation of f (consider choosing different bases for the input and output spaces), $y = P(x)$ is not an arbitrary instance of $\mathcal{MQ}$.

Relevant Problems

Sub-Disciplines

Relevant Problems

Sub-Disciplines

1. Special Complexity Theoretic Problems

Relevant Problems

Sub-Disciplines

1. Special Complexity Theoretic Problems
2. Gröbner Basis Algorithms

Relevant Problems

Sub-Disciplines

1. Special Complexity Theoretic Problems
2. Gröbner Basis Algorithms
3. Minrank Analysis

Relevant Problems

Sub-Disciplines

1. Special Complexity Theoretic Problems
2. Gröbner Basis Algorithms
3. Minrank Analysis
4. Differential

Unbalanced Oil-Vinegar

The Core Map

Let $f : \mathbb{F}_q^{o+v} \rightarrow \mathbb{F}_q^o$ be a random quadratic map such that given random constants $c_1, \ldots, c_v \in \mathbb{F}_q$, $f(x_1, \ldots, x_o, c_1, \ldots, c_v)$ is affine in $x_1, \ldots, x_o$.

Unbalanced Oil-Vinegar

The Core Map

Let $f : \mathbb{F}_q^{o+v} \to \mathbb{F}_q^o$ be a random quadratic map such that given random constants $c_1, \ldots, c_v \in \mathbb{F}_q$, $f(x_1, \ldots, x_o, c_1, \ldots, c_v)$ is affine in $x_1, \ldots, x_o$.

The Entire Map

The public map, P, is defined by $P = f \circ U$ for some affine map, U.

Unbalanced Oil-Vinegar

The Core Map

Let $f : \mathbb{F}_q^{o+v} \rightarrow \mathbb{F}_q^o$ be a random quadratic map such that given random constants $c_1, \ldots, c_v \in \mathbb{F}_q$, $f(x_1, \ldots, x_o, c_1, \ldots, c_v)$ is affine in $x_1, \ldots, x_o$.

The Entire Map

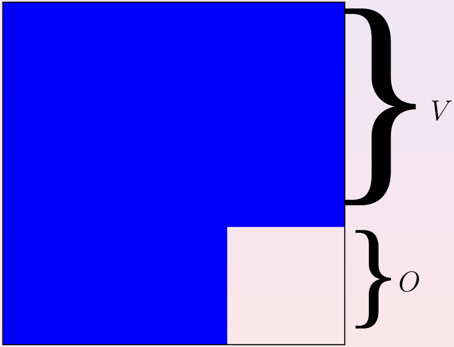
The public map, P, is defined by $P = f \circ U$ for some affine map, U.

Inversion of f

Randomly choose $c_1, \ldots, c_o$, solve $y = f(x_1, \ldots, x_o, c_1, \ldots, c_v)$.

Visualization

The following diagram illustrates the differential structure.



UOV Performance Data

Scheme	PK(kB)	Sign (ms)	Ver. (ms)
UOV(31,33,66)	108.5		1.75
UOV(256,28,56)	99.9		0.98
cyclicUOV(31,33,66)	17.1		0.32
cyclicUOV(256,28,56)	16.5		0.19

Machine ???

Rainbow

First, create a sequence of partitions of the plaintext variables like so:

$$\begin{gathered} V_1 = \{x_1, \ldots, x_{v_1}\},\ O_1 = \{x_{v_1+1}, \ldots, x_{v_2}\} \\ V_2 = \{x_1, \ldots, x_{v_2}\},\ O_2 = \{x_{v_2+1}, \ldots, x_{v_3}\} \\ \vdots \\ V_u = \{x_1, \ldots, x_{v_u}\},\ O_u = \{x_{v_u+1}, \ldots, x_n\} \end{gathered}$$

with $v_1 < v_2 < \cdots < v_u < n$.

Rainbow

First, create a sequence of partitions of the plaintext variables like so:

$$V_1 = \{x_1, \dots, x_{v_1}\},\ O_1 = \{x_{v_1+1}, \dots, x_{v_2}\}$$
$$V_2 = \{x_1, \dots, x_{v_2}\},\ O_2 = \{x_{v_2+1}, \dots, x_{v_3}\}$$
$$\vdots$$
$$V_u = \{x_1, \dots, x_{v_u}\},\ O_u = \{x_{v_u+1}, \dots, x_n\}$$

with $v_1 < v_2 < \cdots < v_u < n$.

The Core Map

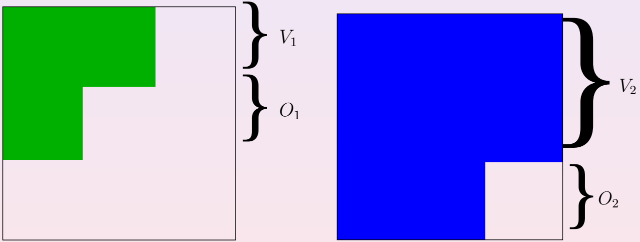
Let $f^k : \mathbb{F}_q^n \to \mathbb{F}_q$ be a random quadratic map of the form

$$f^k(\mathbf{x}) = \sum_{x_i \in O_l, x_j \in V_l} \alpha_{ij}^k x_i x_j + \sum_{x_i, x_j \in V_l} \beta_{ij}^k x_i x_j + \sum_{x_i \in O_l \cup V_l} \gamma_i^k x_i + \delta^k,$$

where l is the unique integer such that $x_k \in O_l$.

Visualization

The following diagram illustrates the differential structure.



Rainbow Performance Data

Scheme	PK(kB)	Sign (ms)	Ver. (ms)
Rainbow(31,14,19,14)	25.3		0.44
Rainbow(256,17,13,13)	25.1		0.26
cyclicRainbow(31,14,19,14)	9.5		0.12
cyclicRainbow(256,17,13,13)	9.5		0.12

Machine???

A “Fair” Comparison

Scheme	PK(bytes)	Sign (cycles)	Ver. (cycles)
Rainbow(31,24,20,20)	57600	64516	24742
Rainbow(256,18,12,12)	30240	14166	10608

2015 Intel Core i5-6600, quad×3310MHz, eBATS

C^* Scheme

Construction

$$\left.\begin{array}{c} k \\ | \\ \mathbb{F}_q \end{array}\right] n$$

C^* Scheme

Construction

$$\begin{tikzcd} k \arrow[d, dash, "n"] \\ \mathbb{F}_q \end{tikzcd}$$

We can identify $\mathbf{x} \in k$ with $x \in \mathbb{F}_q^n$.

C^* Scheme

Construction

$$\left.\begin{array}{c} k \\ | \\ \mathbb{F}_q \end{array}\right] n$$

We can identify $\mathbf{x} \in k$ with $x \in \mathbb{F}_q^n$.

Encryption Scheme

Algebraically, $y = (T \circ f \circ U)x$ where $f(x) = x^{q^\theta+1}$.

HFE

Central Map

Let k be a degree n extension field of F_q and let $f : k \to k$ be defined by $f(x) = \sum_{(i,j)\in I} \alpha_{(i,j)} x^{q^i+q^j}$ where I is some index set such that the pairs satisfy some degree bound $q^i + q^j \leq d$.

HFE

Central Map

Let k be a degree n extension field of F_q and let $f : k \to k$ be defined by $f(x) = \sum_{(i,j)\in I} \alpha_{(i,j)} x^{q^i+q^j}$ where I is some index set such that the pairs satisfy some degree bound $q^i + q^j \leq d$.

HFEv-

Add v new variables $\{\tilde{x}_1, \ldots, \tilde{x}_v\} = V$ and define

$$f(\mathbf{x}) = \sum_{(i,j)\in I} \alpha_{(i,j)} x^{q^i+q^j} + \sum_{i,q^i<D} \beta_i(\tilde{x}_1, \ldots, \tilde{x}_v) x^{q^i} + \gamma(\tilde{x}_1, \ldots, \tilde{x}_v),$$

where β_i are linear and γ is quadratic. Remove some equations.

HFE

Central Map

Let k be a degree n extension field of F_q and let $f : k \to k$ be defined by $f(x) = \sum_{(i,j)\in I} \alpha_{(i,j)} x^{q^i+q^j}$ where I is some index set such that the pairs satisfy some degree bound $q^i + q^j \leq d$.

HFEv-

Add v new variables $\{\tilde{x}_1, \dots, \tilde{x}_v\} = V$ and define

$$f(\mathbf{x}) = \sum_{(i,j)\in I} \alpha_{(i,j)} x^{q^i+q^j} + \sum_{i,q^i<D} \beta_i(\tilde{x}_1, \dots, \tilde{x}_v) x^{q^i} + \gamma(\tilde{x}_1, \dots, \tilde{x}_v),$$

where β_i are linear and γ is quadratic. Remove some equations.

Provably secure against differential adversaries.

HFEv- Performance

Scheme	PK(Bytes)	Sign (k-cycles)	Ver. (k-cycles)
Gui-96(2^{96},5,6,6)	63036	238	62
Gui-95(2^{95},9,5,5)	60600	602	58
Gui-94(2^{94},17,4,4)	58212	2495	71

Intel Xeon E3-1245 v3, 3.4 GHz

Time Constant Implementations!

PFLASH

Central Map

Let $f : k \to k$ be defined by $f(x) = x^{q^\theta+1}$.

PFLASH

Central Map

Let $f : k \to k$ be defined by $f(x) = x^{q^\theta + 1}$.

Morphism

Choose T and U both **singular** affine transformations and compute $P = T \circ f \circ U$.

PFLASH

Central Map

Let $f : k \rightarrow k$ be defined by $f(x) = x^{q^\theta+1}$.

Morphism

Choose T and U both **singular** affine transformations and compute $P = T \circ f \circ U$.

Provably secure against differential adversaries.

Performance Comparison

Scheme	Security (bits)	PK(Bytes)	Sign (k-cycles)	Ver. (k-cycles)
PFLASH(16,62,22,1)	80	39040	288	17
PFLASH(16,74,22,1)	104	72124	509	24
PFLASH(16,94,30,1)	128	142848	634	38
Gui-96(2^{96},5,6,6)	80	63036	238	62
Gui-95(2^{95},9,5,5)	80	60600	602	58
Gui-94(2^{94},17,4,4)	80	58212	2495	71

Intel Xeon E3-1245 v3, 3.4 GHz

Time Constant Implementations!

Encryption?

Some intriguing new schemes. Too immature.

Code Based Crypto

- Encryption
  - McEliece
  - QC-MDPC
  - QC-LRPC
- Signature (Less Mature)
  - CFS
  - RankSign
  - Stern/Cayrel

How does McEliece work?

- Encryption
  - Public key is a binary linear transformation from k to n bits: $\hat{G}$
  - To encrypt a message compute $m\hat{G} + e$
    - e is a binary vector of weight at most t

How does McEliece work? (2)

- Decryption
  - $\hat{G}$ is secretly constructed as SGP
    - S (Scramble) is a random kxk invertible matrix
    - P is an nxn permutation matrix
    - G (Generator) is a generator matrix for an error-correcting Code
      - All we need to know here is that the decryptor can compute x (and e') from $Gx + e'$ as long as e' has weight less than t.
  - First invert P
    - Now you have $G(Sm) + eP^{-1}$
  - Then use the error correcting code
    - Now you have Sm
  - Now invert the scrambler
    - And you have m

Some Coding Theory

- Generator matrix (Systematic form)
  - *nxk*

$$G = [I_k \mid C]$$

- Parity Check matrix
  - *nx(n-k)*

$$H = [-C^T \mid I_{n-k}]$$

    - Note that $GH^T = 0$
- Codewords may either be defined as
  - *n*-bit vectors that can be expressed as *mG* for *k*-bit *m*
  - Solutions to $Hx = 0$
- Syndrome: $s = H(mG + e)^T = H(e^T)$
  - Mapping *s* to minimal weight *e* is sometimes easy but NP hard in general.

The Classic Scheme McEliece 1978

- Uses an algebraic code (Goppa Code)
- Advantages:
  - Still secure (with slightly larger parameters)
  - Apparently Quantum Resistant
  - Fast
    - (like RSA) Encryption is about 10x faster than Decryption
    - Encryption, Decryption, Key Generation are faster and scale better than RSA
- Drawbacks:
  - Giant public keys (~ 1 million bits)
  - Not well suited to signatures

McEliece Key Size Reduction (Motivation)

- Classic McEliece has giant keys
  - 1,537,536 bits for 128-bit security (Bernstein, Peters, Lange 2008)

- Structured (e.g. QC, QD) algebraic codes are often vulnerable to attack
  - Structural Cryptanalysis (Otmani, Tillich, Dallot 2008)
  - Countermeasure – shortened codes (Berger, Cayrel, Gaborit, Otmani 2009)
  - Algebraic cryptanalysis (Faugere, Otmani, Perret, Tillich 2010)

- Non-algebraic (LRPC, MDPC) codes seem less likely to interact badly with structure
  - Secret is not hidden algebraic structure
  - Secret is low-weight basis for parity check matrix row-space

Cyclic Matrices

- Used widely for key size reduction
  - Lattices (NTRU), Coding (QC-MDPC)

$$\begin{pmatrix} a & b & c & d & e & f \\ f & a & b & c & d & e \\ e & f & a & b & c & d \\ d & e & f & a & b & c \\ c & d & e & f & a & b \\ b & c & d & e & f & a \end{pmatrix}$$

- Cyclic Matrices are efficient because they can be reconstructed from their top row $\begin{pmatrix} a & b & c & d & e & f \end{pmatrix}$

MDPC codes

(Misoczki, Tillich, Sendrier, Barreto 2012)

- Key generation:

$$H = (H_0|H_1)$$
$$H_{pub} = {H_1}^{-1}H$$

- Example Parameters (128-bit classical security): n=19714, k=9857, t=134
  - Row weight 142

Rank metric; LRPC Codes
(Gaborit, Murat, Ruatta, Zemor 2013)

- Applies to vectors over $GF(q^m)$
- Use a basis over $GF(q)$ to rewrite an n-dimensional vector over $GF(q^m)$ as an $n \times m$ matrix over $GF(q)$
  - Compute the rank.
- E.g. under the basis $(1, x, x^2)$ for $GF(2^3)$:

$$(1+x, 0, 1+x^2, 1+x, x^2+x)$$

becomes

$$\begin{pmatrix} 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \end{pmatrix}$$

Which has rank 2.

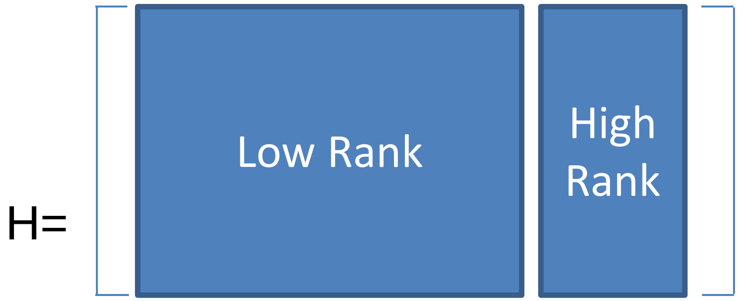
- QC-LRPC keys are constructed exactly like QC-MDPC keys except substitute Rank Metric for Hamming Metric.
- Key sizes can be somewhat smaller (e.g. 3000 bits instead of 10,000 bits) than QC-MDPC, but need to be more careful about folding attacks, decryption failures.

CFS Signature

- Attempt to “Decrypt” a message digest (like RSA)
  - Problem: most “ciphertexts” are not uniquely decodable
  - Solution:
    - Choose extreme (e.g. n = 65536, k =65392, t=9) code
    - try decrypting H(m||counter) until it works.
  - Downsides: SLOW, even bigger keys than standard McEliece

RankSign

- Use special form key to allow (non-unique) decoding of all hashes



H=

Low Rank

High Rank

- This can be effectively disguised since rank metric is preserved by arbitrary invertible linear transformations on column space
  - (not just permutations. G_{pub} = SGL not just SGP.)

- However, QC structure no longer works for key size reduction.
  - A factor of 2 is ok, but more is insecure.

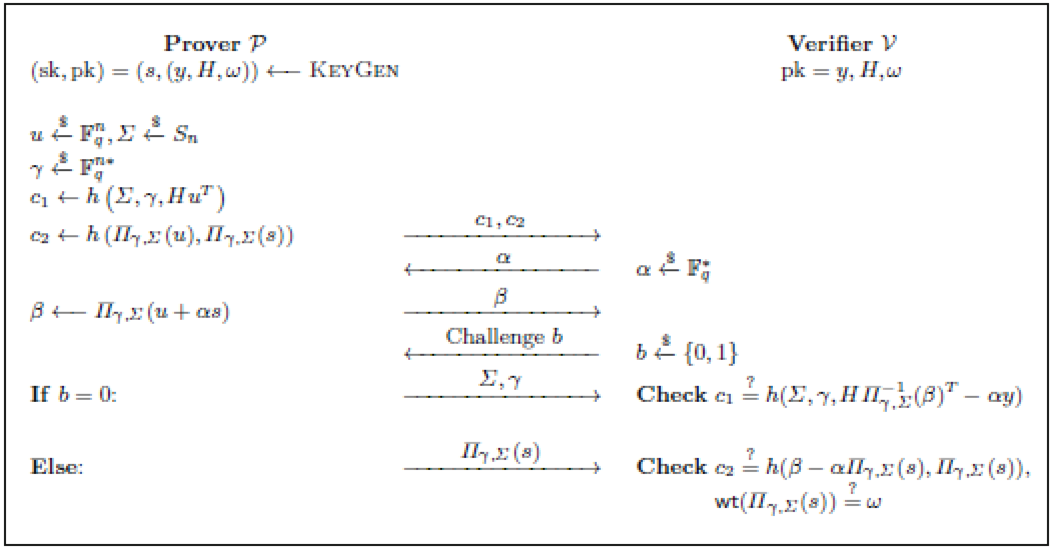
- Example Sizes: Public key 28300 bits, Signature 8640 bits, 128 bits of security

Stern/Cayrel

(Stern 1993)

(Cayrel, Veron, Alaoui 2011)

- Uses Fiat-Shamir instead of hash-then-sign
- Secret key, low-weight *s*
- Public Key *H*, y = Hs^{T}



Prover $\mathcal{P}$		**Verifier** $\mathcal{V}$
$(\mathrm{sk}, \mathrm{pk}) = (s, (y, H, \omega)) \longleftarrow \textsc{KeyGen}$		$\mathrm{pk} = y, H, \omega$
$u \overset{\$}{\leftarrow} \mathbb{F}_q^n, \Sigma \overset{\$}{\leftarrow} S_n$		
$\gamma \overset{\$}{\leftarrow} \mathbb{F}_q^{n*}$		
$c_1 \leftarrow h\left(\Sigma, \gamma, Hu^T\right)$		
$c_2 \leftarrow h\left(\Pi_{\gamma,\Sigma}(u), \Pi_{\gamma,\Sigma}(s)\right)$	$\xrightarrow{c_1, c_2}$	
	$\xleftarrow{\alpha}$	$\alpha \overset{\$}{\leftarrow} \mathbb{F}_q^*$
$\beta \longleftarrow \Pi_{\gamma,\Sigma}(u + \alpha s)$	$\xrightarrow{\beta}$	
	$\xleftarrow{\text{Challenge } b}$	$b \overset{\$}{\leftarrow} \{0, 1\}$
If $b = 0$:	$\xrightarrow{\Sigma, \gamma}$	**Check** $c_1 \overset{?}{=} h(\Sigma, \gamma, H\Pi_{\gamma,\Sigma}^{-1}(\beta)^T - \alpha y)$
Else:	$\xrightarrow{\Pi_{\gamma,\Sigma}(s)}$	**Check** $c_2 \overset{?}{=} h(\beta - \alpha\Pi_{\gamma,\Sigma}(s), \Pi_{\gamma,\Sigma}(s))$, $\mathrm{wt}(\Pi_{\gamma,\Sigma}(s)) \overset{?}{=} \omega$

- Approximate sizes (public key: 80,000 bits, signature 400,000 bits, 128 bits of security)

Hash-Based Signatures

- Lamport-Diffie-Merkle-Winternitz
  - Draft-McGrew (Leighton-Micali)
  - Draft-Huelsing (XMSS)
- SPHINCS

Basic One-Time-Signature

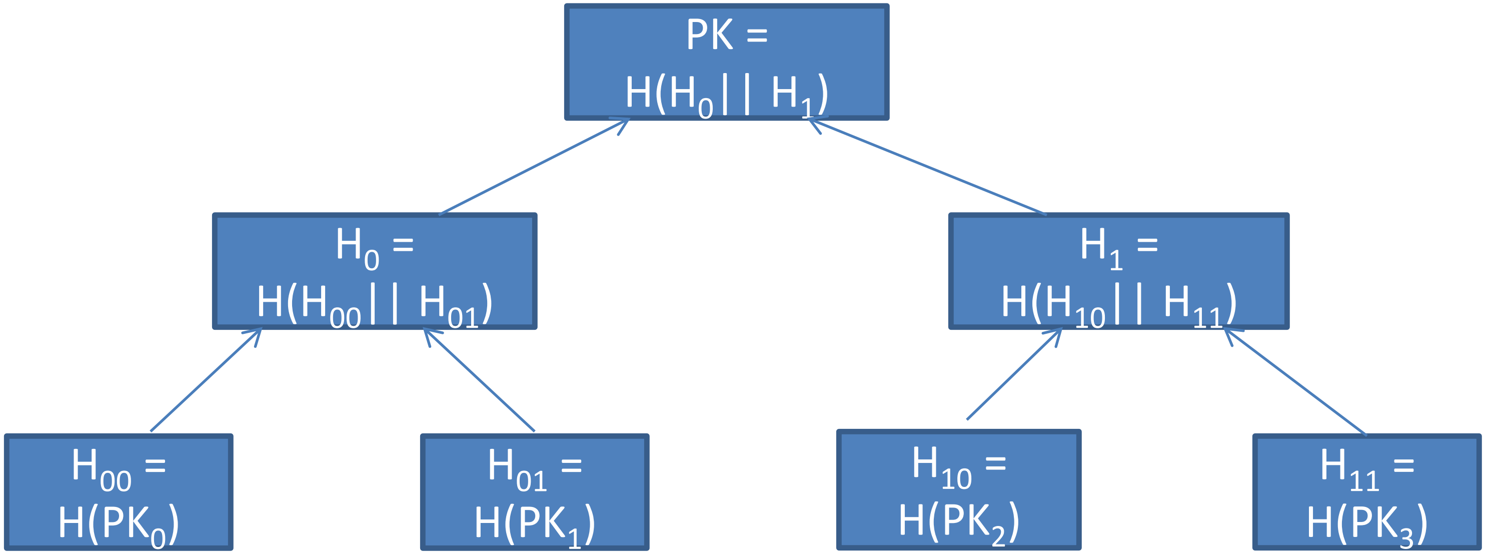
- Signing a bit
  - Public key: H(s_0)||H(s_1)
  - Secret key: s_0, s_1
  - Signature for 0: s_0
  - Signature for 1: s_1

- To sign a message digest, simply concatenate 256 one-bit public keys/ secret keys / signatures
  - One for each bit of the digest:
    - Public key: H($s_{0,0}$)||H($s_{0,1}$) || H($s_{1,0}$)||H($s_{1,1}$) || … || H($s_{255,0}$)||H($s_{255,1}$)
  - Note that with a signature on as few as two digests (e.g. 111…1, 000…0) the adversary can forge any signature. (Hence, One-Time Signature.)

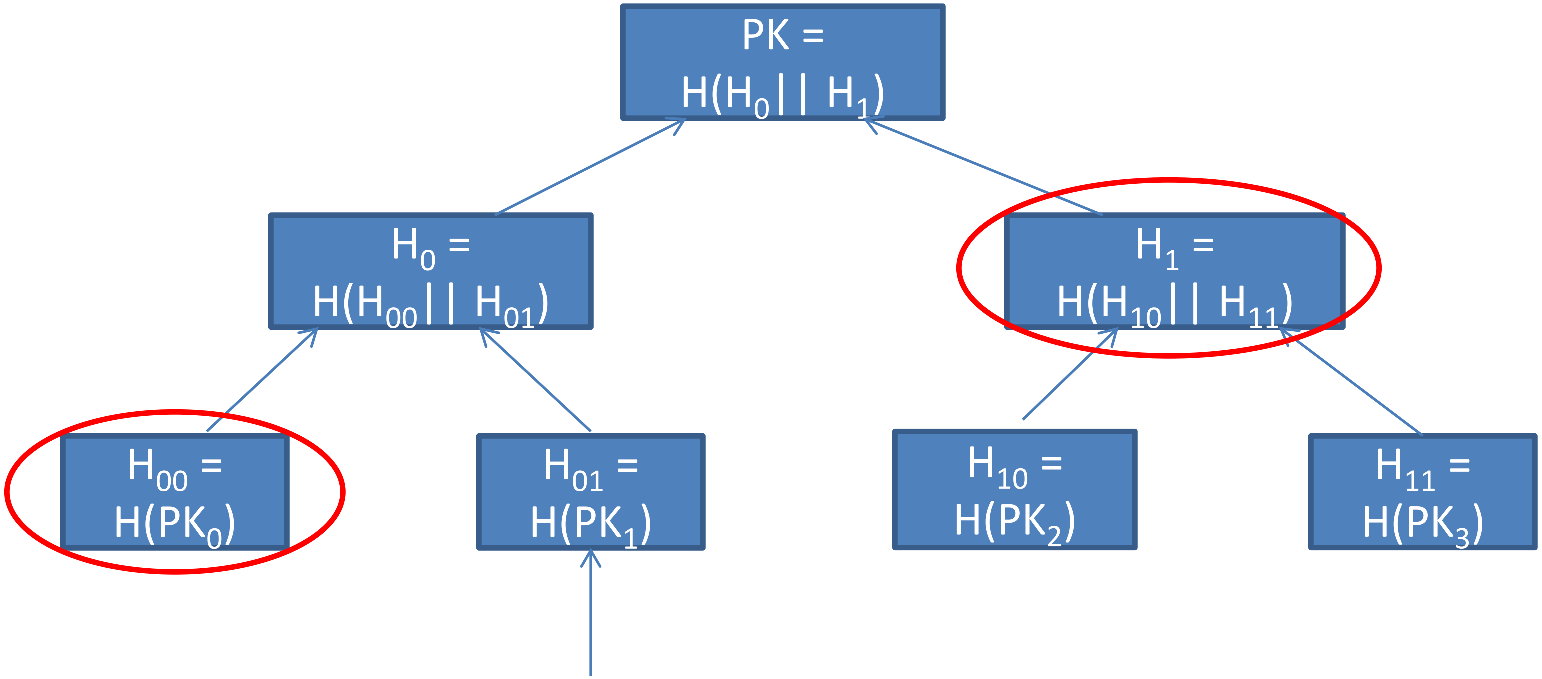
Improvements (Winternitz)

- Save space with a checksum
  - E.g. Only release a secret for bits of the digest that are ones.
  - Now an adversary can change ones to zeros, but not vice versa.
  - To fix this problem, append to the digest a binary representation of the number of zeroes in the digest.
  - Now the public key size goes from $2n$ hashes to $n + \log n$
- Use a hash chain to go from binary representation of the digest to base w.
  - Public key for each digit is a secret hashed w times.
  - To sign a digit, d, release the secret hashed $w – d$ times.
  - Now the checksum is $n \cdot w/\log(w)$ – <Sum of the Digits>.
  - The Winternitz parameter w presents a time-space tradeoff.

Many Time Signatures (Merkle)



Many Time Signatures (Merkle)



- Signature: $OTS_{sk1}(m) || PK_1 || H_{00} || H_1$

Key Generation Times and “Certificate Chains”

- With standard Merkle signature, you have to precompute the whole tree before you can sign anything!
  - Allowing more signatures under one key has:
    - Log overhead in signature length/signing time
    - Linear overhead in key generation time.

- Solution, use the Merkle tree to sign the root of another Merkle tree.
  - Taken to the extreme, this can enable stateless signatures. (More later)

XMSS and McGrew’s draft

- Both are IRTF drafts
  - XMSS is a work item and McGrew’s draft is a personal draft (I think.)
- XMSS has a standard model proof (second-preimage resistance.)
  - McGrew’s draft (Leighton-Micali signatures) has a random oracle proof.
- Leighton-Micali is old enough that it can’t still be in patent, although I think XMSS is not patented.

- **Importantly, both drafts are *stateful.***
  - This might be ok for things like code signing, where strong version control is assumed, but will make trouble for
    - Software processes where memory failure due to unexpected reboot is a real possibility.
    - Online signing services that are duplicated on several systems.
    - Etc.

SPHINCS (stateless hash-based signatures)

- Signature is structured like a cert-chain with
  - many layers (12)
  - of small Merkle Trees (32 nodes)
- Sample tree index randomly
- Use Few-Time Signature (HORST) rather than One-Time Signature to sign messages.
  - (OTS is fine for signing Merkle Tree roots.)
- Signature size: 328,000 bits
  - This compares to a typical size of 15,000 bits per layer for stateful schemes.

Outliers

Isogeny-based

- Defined on the space of elliptic curves.
- Less studied and do worse than lattice based.
- We propose to ignore them.

Outliers

Based on braid group

- Very pretty
- Groups are infinite
- The hard question is whether a braid can be turned into another by pre-pending a braid S and appending its inverse S^{-1} .
- Some proposals have been shown insecure.
- We propose to ignore them.

One-way functions for key-exchange

- Suppose F is one-way for quantum computers.
- Also suppose F^n (X) can be calculated fast even for exponential n.
- Then key exchange a la DH is possible: For random X, n, m
    - Alice sends X , F^n (X)
    - Bob replies with F^m (X)
    - Both compute F^{n+m} (X)
- The point is that no trapdoor seems necessary.
- Should we leave the door open for this type of construction?

Practical Questions

- Which are most important in practice?
  - Public and private key sizes
  - Key pair generation time
  - Ciphertext size
  - Encryption/Decryption speed
  - Signature size
  - Signature generation/verification time

- Not a lot of benchmarks in this area

Encryption Schemes

Algorithm	KeyGen Time (RSA sign=1)	Decrypt Time (RSA sign=1)	Encrypt Time (RSA sign=1)	Public Key Size (bits)	Private Key Size (bits)	Ciphertext Size (bits)	Time* Scaling	Key* Scaling
NTRUEncrypt	10	0.1	0.1	~3000	~4000	~3000	k^2	k
McEliece	5	1	0.02	651264	1098256	1660	k^2	k^2
Quasi-Cyclic McEliece	5	1	0.02	4801	9602	9602	k^2	k
RSA	50	1	0.02	1024	1024	1024	k^6	k^3
DH	0.5	0.5	0.5	1024	480	1024	k^4	k^3
ECC	0.1	0.1	0.1	320	480	320	k^2	k

- **Disclaimer** – these are rough estimates for comparison purposes only, not benchmarks. Numbers are for 80 bits of security.

* Time and key scaling ignore log *k* factors

Signature Schemes

Algorithm	KeyGen Time (RSA sign=1)	Sign Time (RSA sign=1)	Verify Time (RSA sign=1)	Limited Lifetime?	Public Key Size	Private Key Size	Signature Size (bits)	Time* Scaling	Key * Scaling
Winternitz-Merkle signatures	200 10000 500000	1 1 2	0.2 0.2 0.2	2^{20} 2^{30} 2^{40}	368 368 368	15200 22304 29344	17024 18624 20224	k^2	k^2
GLP signatures (lattice-based)	0.01	0.5	0.02		11800	1620	8950	k^2	k
CFS signature (code based)	5	2000	0.02		9437184	~15000000	144	exp(o(k))	exp(o(k))
Psflash signature (multivariate)	50	1	0.1		576992	44400	296	k^3	k^3
Quartz signature (multivariate)	100	2	0.05		126000	11500	80	k^3	k^3
RSA	50	1	0.02		1024	1024	1024	k^6	k^3
DSA	0.5	0.5	0.5		1024	480	320	k^4	k^3
ECDSA	0.1	0.1	0.1		320	480	320	k^2	k

- **Disclaimer** – these are rough estimates for comparison purposes only, not benchmarks. Numbers are for 80 bits of security.

* Time and key scaling ignore log *k* factors

Observations

- For most of the potential PQC replacements, the times needed for encryption, decryption, signing, and verification are acceptable

- Some key sizes are significantly increased
  - For most protocols, if the public keys do not need to be exchanged, it may not be a problem

- Some ciphertext sizes and signature sizes are not quite plausible

- Key-pair generation time for the encryption schemes is not bad at all

- **No easy "drop-in" replacements**

- Would be nice to have more benchmarks

Quantum Computers…When?

We do not yet have large scale general-purpose quantum computers, though many approaches are being pursued.

Quantum computers are 25 years in the future and always will be.

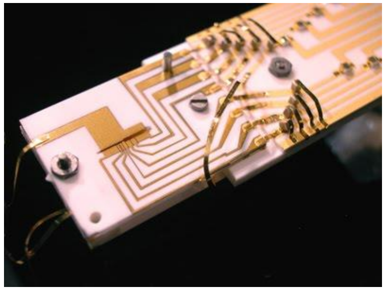
?

Quantum Computers…When?

We do not yet have large scale general-purpose quantum computers, though many approaches are being pursued.

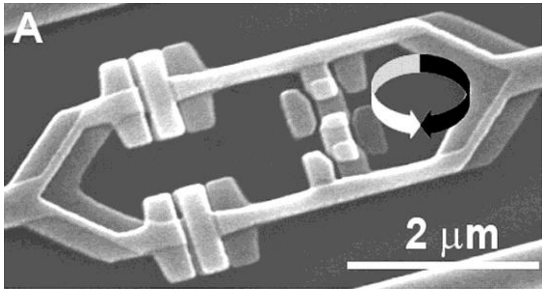
Quantum computers are 25 years in the future and always will be. ?

Trapped Ions



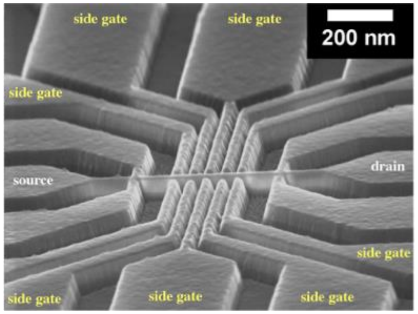
[Wineland group, NIST]

Superconducting Circuits



[Mooij group, TU Delft]

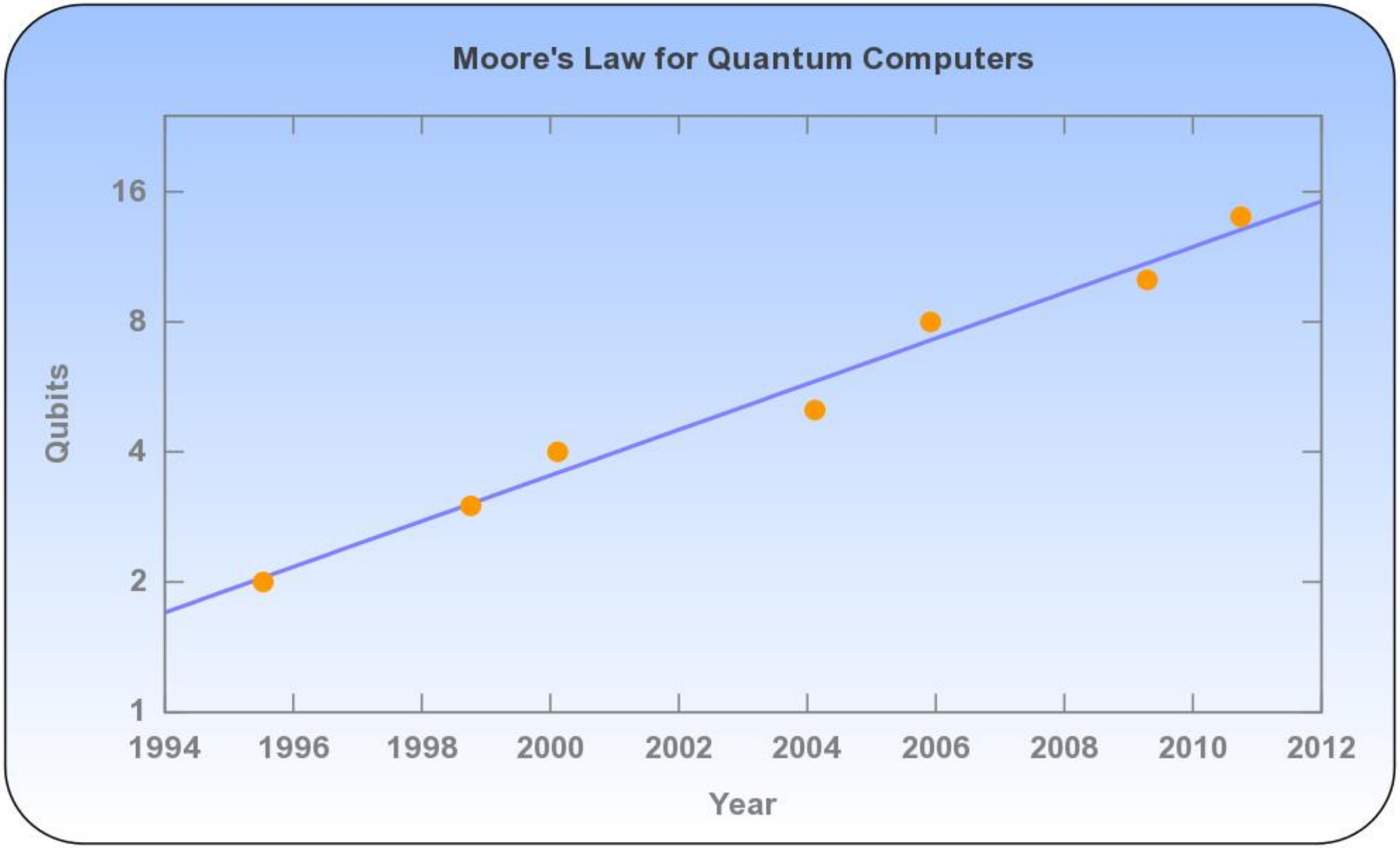
Quantum Dots



…

[Paul group, U. Glasgow]

A Frequently Made Argument



Quantum Moore's law: Number of qubits doubles every 6 years.

My opinion: Number of qubits is the wrong metric.

Why is it hard to build a quantum computer?

Quantum states are very fragile and must be extremely well isolated.

In the early days, many prominent scientists thought that quantum computation was doomed for this reason. (analog computing, anyone?)

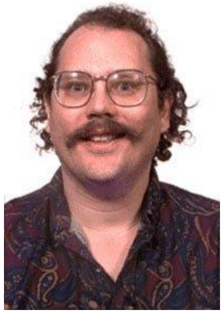
A 1996/1997 breakthrough convinced all but diehard skeptics that quantum computation is scalable, in principle.

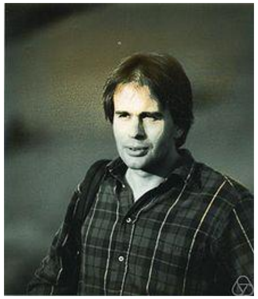
Threshold Theorem

Theorem (loosely stated): If error per quantum operation can be brought below 0.5%, arbitrarily long quantum computations can be performed by correcting errors as you go.

Aharonov

Ben-Or


Shor


Calderbank

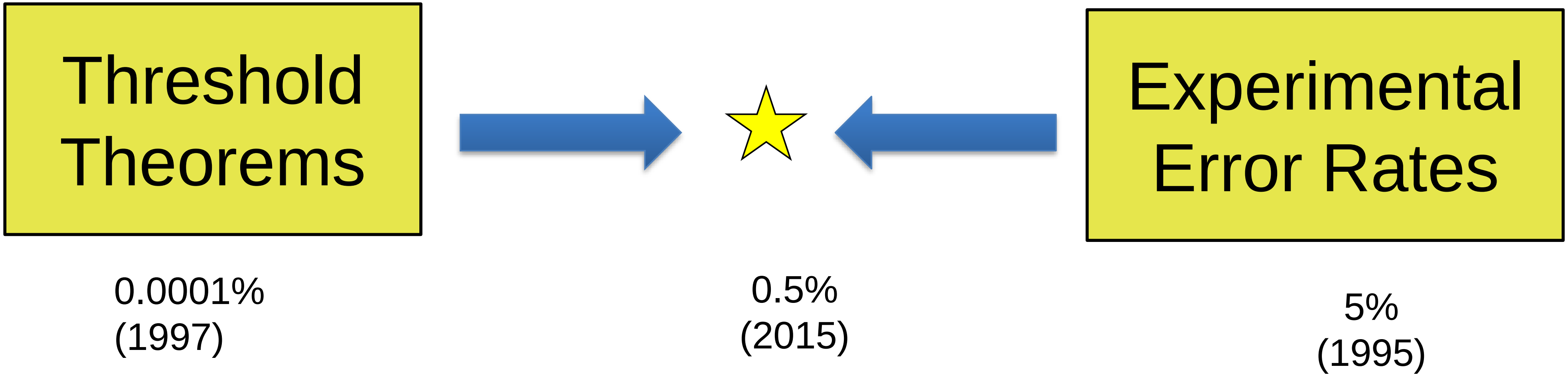
Steane

Knill

Kitaev

Laflamme

Progress in Quantum Computing



Threshold Theorems

Experimental Error Rates

0.0001%
(1997)

0.5%
(2015)

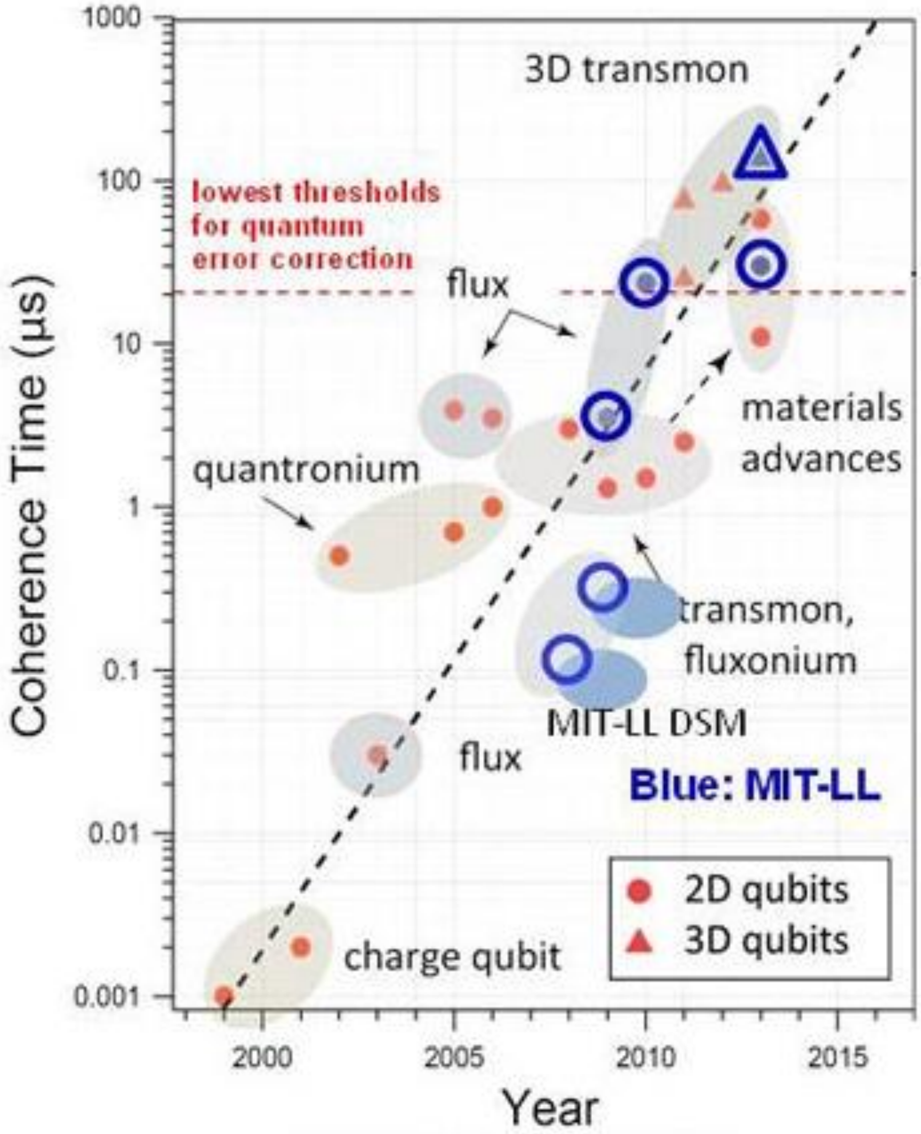
5%
(1995)

Theorists improve error correction schemes
to tolerate higher error rates

Experimentalists achieve lower error rates.

When these numbers meet we can think about scaling up.

A Real Quantum Moore's Law



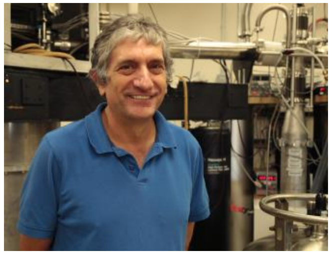
Oliver & Welander, MRS Bulletin (2013)

Superconducting Qubits:

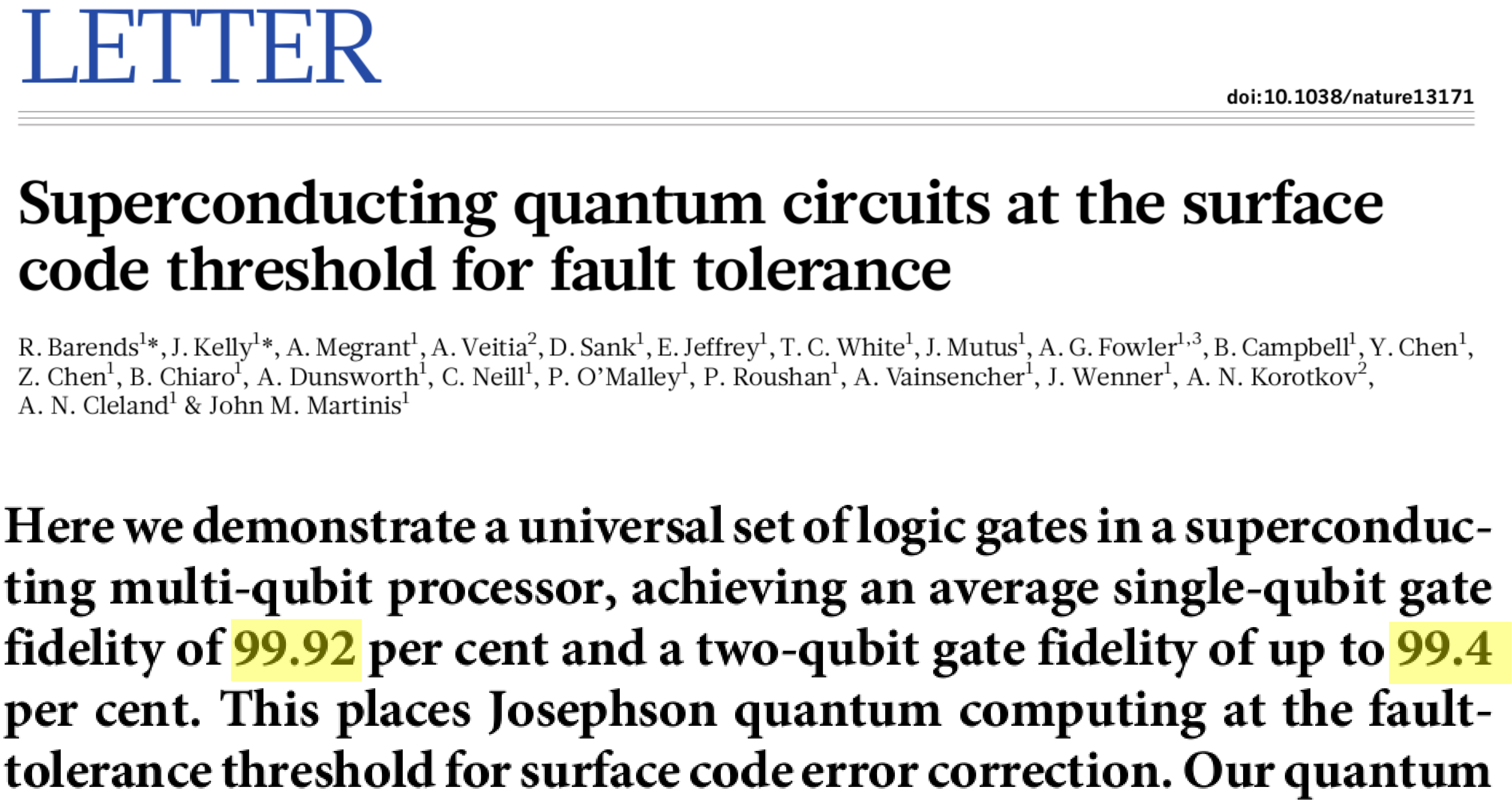
Coherence time doubles every 11 months.

Roughly equivalent:

Error rate halves every 11 months.



April, 2014



LETTER

doi:10.1038/nature13171

Superconducting quantum circuits at the surface code threshold for fault tolerance

R. Barends[1]*, J. Kelly[1]*, A. Megrant[1], A. Veitia[2], D. Sank[1], E. Jeffrey[1], T. C. White[1], J. Mutus[1], A. G. Fowler[1,3], B. Campbell[1], Y. Chen[1], Z. Chen[1], B. Chiaro[1], A. Dunsworth[1], C. Neill[1], P. O'Malley[1], P. Roushan[1], A. Vainsencher[1], J. Wenner[1], A. N. Korotkov[2], A. N. Cleland[1] & John M. Martinis[1]

Here we demonstrate a universal set of logic gates in a superconducting multi-qubit processor, achieving an average single-qubit gate fidelity of 99.92 per cent and a two-qubit gate fidelity of up to 99.4 per cent. This places Josephson quantum computing at the fault-tolerance threshold for surface code error correction. Our quantum

-March, 2014 (Trapped ion qubits):
Lockheed-Martin/University of Maryland quantum engineering center announced

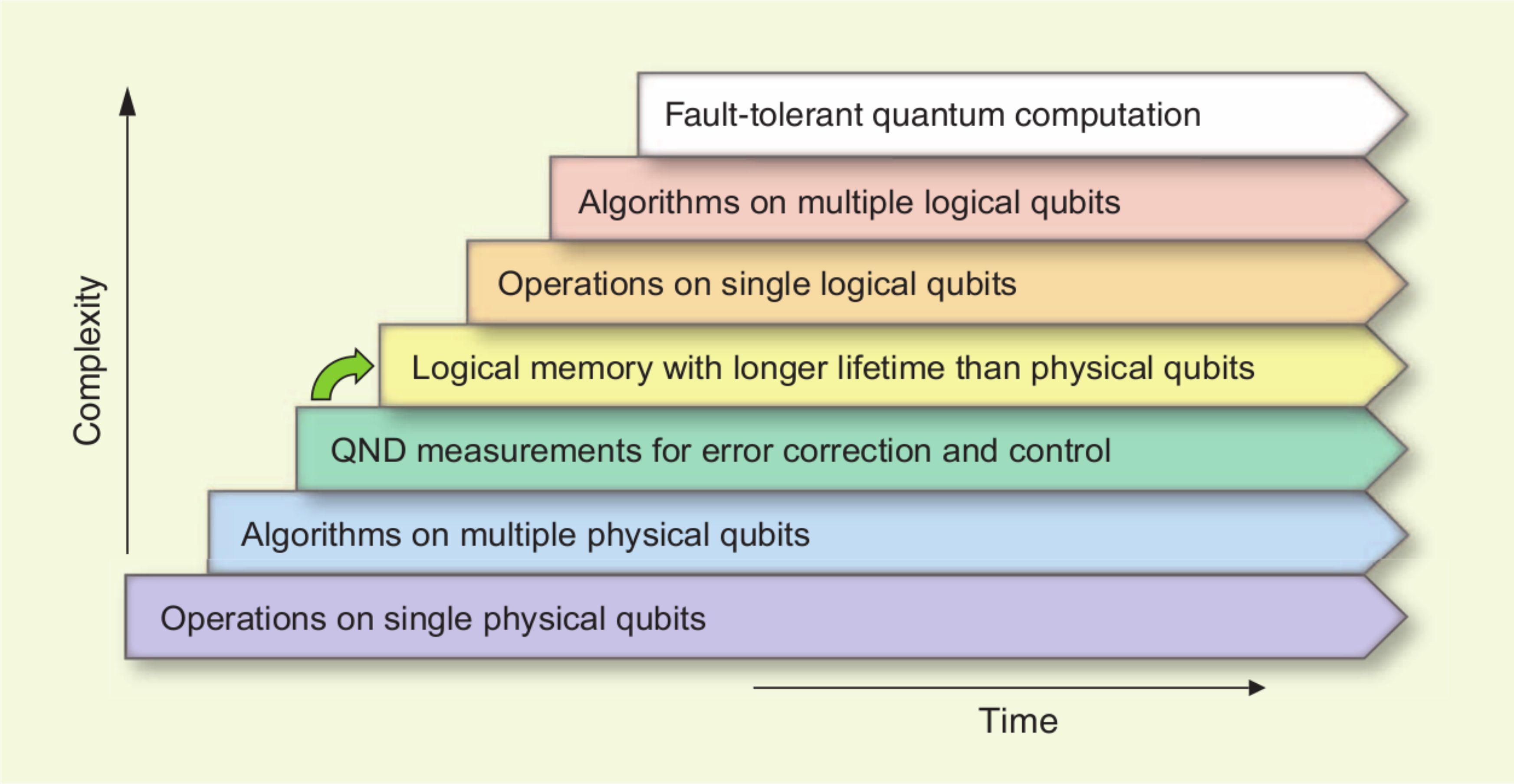
-April, 2014 (Superconducting qubits):
Martinis threshold paper

-September, 2014 (Superconducting qubits):
Google buys Martinis Lab

-October, 2014 (Silicon-based qubits):
Morello & Dzurak at University of New South Wales announce 99% gate fidelities

-November, 2014 (Trapped ion qubits):
Oxford announces Q20:20 project

-April, 2015 (Superconducting qubits):
IBM demonstrates error detecting codes
So does Delft University of technology


Fault-tolerant quantum computation
Algorithms on multiple logical qubits
Operations on single logical qubits
Logical memory with longer lifetime than physical qubits
QND measurements for error correction and control
Algorithms on multiple physical qubits
Operations on single physical qubits
Complexity
Time

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

We've made a lot of progress, but we've still got a long way to go.

So what?

The NIST PQC Project

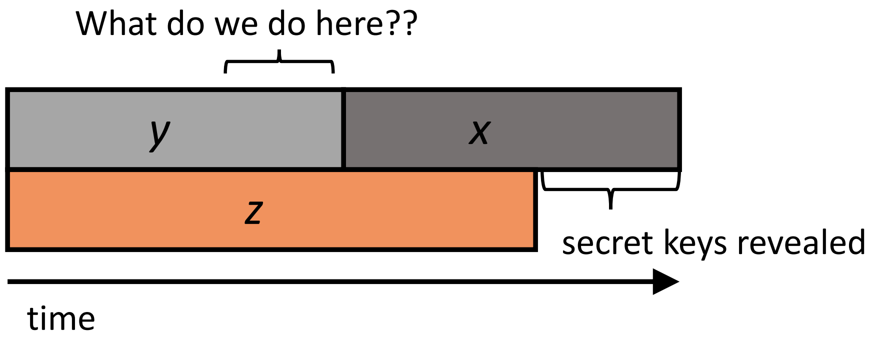
- Objectives
  - Examine quantum-resistant public key cryptosystems
  - Monitor quantum computing progress and applicability of known quantum algorithms
- Biweekly seminars since 2012
- Publications and presentations
  - Journals, conferences, workshops
- Collaboration:
  - Hosting academic visitors
  - CryptoWorks 21(U. of Waterloo)
  - Joint Center for Quantum Information and Computer Science, University of Maryland
- NIST Workshop on Cybersecurity in a Post-Quantum World

  http://www.nist.gov/itl/csd/ct/post-quantum-crypto-workshop-2015.cfm

How soon do we need to worry?

- How long does encryption need to be secure (*x* years)
- How long to re-tool existing infrastructure with quantum safe solution (*y* years)
- How long until large-scale quantum computer is built (*z* years)

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



- NSA is transitioning in the “not too distant” future <https://www.nsa.gov/ia/programs/suiteb_cryptography/>
- European PQCrypto project
- ETSI work
- NIST report - <http://csrc.nist.gov/publications/drafts/nistir-8105/nistir_8105_draft.pdf >

Call for Proposals

- NIST is calling for quantum-resistant cryptographic algorithms to be considered for new public-key cryptographic standards
  - Digital signatures
  - Encryption/key-establishment

- We do not expect to “pick a winner”
  - Ideally, several algorithms will emerge as ‘good choices’

- We may pick one (or more) for standardization

Timeline

- Fall 2016 – formal Call For Proposals
- Nov 2017 – Deadline for submissions
- 3-5 years – Analysis phase
    - NIST will report its findings
- 2 years later - Draft standards ready

- Workshops
    - Early 2018 – submitter's presentations
    - One or two during the analysis phase

Differences with AES/SHA-3 competitions

- This is not a competition
  - We see our role as managing a process of achieving community consensus in a **transparent** and timely manner

- Post-quantum cryptography is more complicated than AES or SHA-3
  - No silver bullet - each candidate has some disadvantage
  - Not enough research on quantum algorithms to ensure confidence for some schemes

- We do not expect to “pick a winner”
  - Ideally, several algorithms will emerge as ‘good choices’

- We may narrow our focus at some point
  - This does not mean algorithms are “out”

Requirements

- The formal Call will have detailed submission requirements
  - A complete written specification of the algorithms shall be included, consisting of all necessary mathematical operations, equations, tables, diagrams, and parameters that are needed to implement the algorithms. The document shall include design rationale and an explanation for all the important design decisions that are made.

- Minimal acceptability requirements
  - Publicly disclosed and available with no IPR
  - Implementable in wide range of platforms
  - Provides at least one of: signature, encryption, or key exchange
  - Theoretical and empirical evidence that provides justification for claims about security

Specification

- Implementation
  - Reference version
  - Optimized version

- Cryptographic API will be provided
  - Can call approved hash functions, block ciphers, modes, etc…

- Known Answer and Monte Carlo tests

- Optional – constant time implementation

Intellectual Property

- Signed statements
  - Submitted algorithm
  - Implementations

- Disclose known patent information

- Available worldwide without royalties during the process
  - If algorithm is not chosen for standardization, the rights will be returned to the submitters

Evaluation criteria

- To be detailed in the formal Call
  - Security
  - Cost (computation and memory)
  - Algorithm and implementation characteristics
- Draft criteria will be open for public comment
- We strongly encourage public evaluation and publication of results concerning submissions
- NIST will summarize the evaluation results and report publicly

Security Analysis

- Target security levels
    - 128 bits classical security
    - 64/80/96/128 bits quantum security?
- Correct security definitions?
    - IND-CCA2 for encryption
    - EUF-CMA for signatures
    - CK best for key exchange?
- Quantum/classical algorithm complexity
    - Stability of best known attack complexity
    - Precise security claim against quantum computation
    - Parallelism?
    - Attacks on multiple keys?
    - How many chosen ciphertext queries allowed?
- Security proofs (not required?)
- Quality and quantity of prior cryptanalysis

Cost

- Computational efficiency
  - Hardware and software
    - Key generation
    - Encryption/Decryption
    - Signing/Verification
    - Key exchange

- Memory requirements
  - Concrete parameter sets and key sizes for target security levels
  - Ciphertext/signature size

Algorithm and Implementation Characteristics

- Ease of implementation
    - Tunable parameters
    - Implementable on wide variety of platforms and applications
    - Parallelizable
    - Resistance to side-channel attacks
- Ease of use
    - How does it fit in existing protocols (such as TLS or IKE)
    - Misuse resistance
- Simplicity

Questions

- How is the timeline? Too fast? Too slow?
  - Do we need an ongoing process, or is one time enough?
- How to determine if a candidate is mature enough for standardization
- Should we just focus on encryption and signatures, or should we also consider other functionalities?
- How many "bits of security" do we need against quantum attacks?
- How can we encourage more work on quantum cryptanalysis? Maybe we need more "challenge problems"?
- How can we encourage people to study practical impacts on the existing protocols?
  - For example, key sizes may be too big

So What?

- Summary
  - Quantum computers will break today’s PKC
  - Many proposals for post-quantum crypto, but no drop-in replacement
  - NIST is going to call for quantum-resistant algorithms
    - Signatures, encryption/key-exchange
  - Hope to have standards ready within 10 years

- This will take a lot of resources
  - Not (quite) as much as SHA-3
  - We will need more help
  - Post-docs/guest researchers wanted