An introduction to Post-Quantum Cryptography (PQC)

Jean-Christophe Deneuville

<jean-christophe.deneuville@enac.fr>

Fall 2020

TLS-SEC

Outline

Outline

What you've learnt so far (should have)

- What is cryptography

What you've learnt so far (should have)

- What is cryptography
- How it relates to information security

What you've learnt so far (should have)

- What is cryptography
- How it relates to information security
- Examples of ancestral constructions (Scytale, Cæsar, Vigenere, ...)

What you've learnt so far (should have)

- What is cryptography
- How it relates to information security
- Examples of ancestral constructions (Scytale, Cæsar, Vigenere, ...)
- Their weaknesses/cryptanalysis (Al-Kindi (frequency analysis), Babbage, ...)

What you've learnt so far (should have)

- What is cryptography
- How it relates to information security
- Examples of ancestral constructions (Scytale, Cæsar, Vigenere, ...)
- Their weaknesses/cryptanalysis (Al-Kindi (frequency analysis), Babbage, ...)
- Recent symmetric constructions (OTP, Enigma, Sigaba, ...)

What you've learnt so far (should have)

- What is cryptography
- How it relates to information security
- Examples of ancestral constructions (Scytale, Cæsar, Vigenere, ...)
- Their weaknesses/cryptanalysis (Al-Kindi (frequency analysis), Babbage, ...)
- Recent symmetric constructions (OTP, Enigma, Sigaba, ...)
- Modern symmetric constructions (Stream/Block cipher, DES, AES, Blowfish, RC4, ...)

What you've learnt so far (should have)

- What is cryptography
- How it relates to information security
- Examples of ancestral constructions (Scytale, Cæsar, Vigenere, ...)
- Their weaknesses/cryptanalysis (Al-Kindi (frequency analysis), Babbage, ...)
- Recent symmetric constructions (OTP, Enigma, Sigaba, ...)
- Modern symmetric constructions (Stream/Block cipher, DES, AES, Blowfish, RC4, ...)
- Asymmetric cryptography (RSA, Diffie-Hellman, ElGamal, ...)

What you've learnt so far (should have)

- What is cryptography
- How it relates to information security
- Examples of ancestral constructions (Scytale, Cæsar, Vigenere, ...)
- Their weaknesses/cryptanalysis (Al-Kindi (frequency analysis), Babbage, ...)
- Recent symmetric constructions (OTP, Enigma, Sigaba, ...)
- Modern symmetric constructions (Stream/Block cipher, DES, AES, Blowfish, RC4, ...)
- Asymmetric cryptography (RSA, Diffie-Hellman, ElGamal, ...)
- (Collision Resistant) Hash Functions (and birthday paradox)

What you've learnt so far (should have)

- What is cryptography
- How it relates to information security
- Examples of ancestral constructions (Scytale, Cæsar, Vigenere, ...)
- Their weaknesses/cryptanalysis (Al-Kindi (frequency analysis), Babbage, ...)
- Recent symmetric constructions (OTP, Enigma, Sigaba, ...)
- Modern symmetric constructions (Stream/Block cipher, DES, AES, Blowfish, RC4, ...)
- Asymmetric cryptography (RSA, Diffie-Hellman, ElGamal, ...)
- (Collision Resistant) Hash Functions (and birthday paradox)
- Digital Signatures (RSA, DSA, ECDSA, ...)

What you've learnt so far (should have)

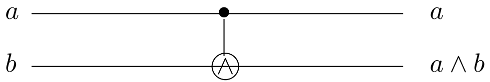
- What is cryptography
- How it relates to information security
- Examples of ancestral constructions (Scytale, Cæsar, Vigenere, ...)
- Their weaknesses/cryptanalysis (Al-Kindi (frequency analysis), Babbage, ...)
- Recent symmetric constructions (OTP, Enigma, Sigaba, ...)
- Modern symmetric constructions (Stream/Block cipher, DES, AES, Blowfish, RC4, ...)
- Asymmetric cryptography (RSA, Diffie-Hellman, ElGamal, ...)
- (Collision Resistant) Hash Functions (and birthday paradox)
- Digital Signatures (RSA, DSA, ECDSA, ...)
- Security models

Outline

Classical Boolean Circuits

source: J. Royer

We view them as naming maps $\{0,1\}^n \to \{0,1\}^n$

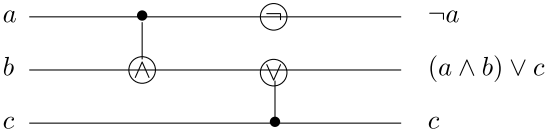


control

target

$\Rrightarrow$ current flow $\Rrightarrow$

Now consider



We can describe this by either of:

- $b \leftarrow a \wedge b; \quad a \leftarrow \neg a; \quad b \leftarrow b \vee c$
- $|a, b, c\rangle \quad \mapsto \quad |a, a \wedge b, c\rangle \quad \mapsto \quad |\neg a, a \wedge b, c\rangle \quad \mapsto \quad |\neg a, (a \wedge b) \vee c, c\rangle$

$|x, y, z\rangle$ = *state vector*

Classical computing

A classical computer (Turing machine) processes (through a language) classical boolean circuits.

The quantity of information is measured through Shannon's entropy, data can eventually be compressed, and there exist efficient algorithms for error correction.

Some circuits are computable *i.e.* the machine eventually halts (*e.g.* primality problem), some others aren't (*e.g.* the halting problem).

Current security

Current security vs. classical computing power (2020)

Current security vs. classical computing power (2020)

1 standard machine: 64 bits architecture

2^6

Current security vs. classical computing power (2020)

1 standard machine: 8 cores

$2^6 \times 2^3$

Current security vs. classical computing power (2020)

1 standard machine: 4 GHz

$2^6 \times 2^3 \times 2^2 \times 10^9$

Current security vs. classical computing power (2020)

1 standard machine: running 1 month

$2^6 \times 2^3 \times 2^2 \times 10^9 \times 60 \times 60 \times 24 \times 30$

Current security vs. classical computing power (2020)

NSA $\geq$ 10 000 standard machines?

$2^6 \times 2^3 \times 2^2 \times 10^9 \times 60 \times 60 \times 24 \times 30 \times 10^4$

Current security vs. classical computing power (2020)

NSA $\geq$ 10 000 standard machines?

$2^6 \times 2^3 \times 2^2 \times 10^9 \times 60 \times 60 \times 24 \times 30 \times 10^4 \approx 2^{80}$ elementary operations

Current security vs. classical computing power (2020)

NSA $\geq$ 10 000 standard machines? **(without possible GPU, ASICS, ...)**

$2^6 \times 2^3 \times 2^2 \times 10^9 \times 60 \times 60 \times 24 \times 30 \times 10^4 \approx 2^{80}$ elementary operations

Current security vs. classical computing power (2020)

NSA $\geq$ 10 000 standard machines? (without possible GPU, ASICS, ...)

$2^6 \times 2^3 \times 2^2 \times 10^9 \times 60 \times 60 \times 24 \times 30 \times 10^4 \approx 2^{80}$ elementary operations

A concrete example

During 2018, there were 2^{89} SHA-256 hashes computed on the blockchain BitCoin...

Current security vs. classical computing power (2020)

NSA $\geq$ 10 000 standard machines? **(without possible GPU, ASICS, ...)**

$2^6 \times 2^3 \times 2^2 \times 10^9 \times 60 \times 60 \times 24 \times 30 \times 10^4 \approx 2^{80}$ elementary operations

A concrete example

During 2018, there were 2^{89} SHA-256 hashes computed on the blockchain BitCoin...

Security in 2020

Setting parameters so that best known attacks have complexity (at least) 2^{128}.

Current security vs. classical computing power (2020)

NSA $\geq$ 10 000 standard machines? (without possible GPU, ASICS, ...)
$2^6 \times 2^3 \times 2^2 \times 10^9 \times 60 \times 60 \times 24 \times 30 \times 10^4 \approx 2^{80}$ elementary operations

A concrete example

During 2018, there were 2^{89} SHA-256 hashes computed on the blockchain BitCoin...

Security in 2020

Setting parameters so that best known attacks have complexity (at least) 2^{128}.

Classical best known attacks:

- Symmetric primitives: brute-force
- Asymmetric primitives: GNFS, sub-exponential complexity

Quantum computing

Quantum computing has been suggested by Paul Benioff in 1980. He proposed a quantum mechanical model of the Turing machine, using two ingredients:

Quantum computing

Quantum computing has been suggested by Paul Benioff in 1980. He proposed a quantum mechanical model of the Turing machine, using two ingredients:

- Superposition: while a bit can be either in a state 0 or 1, a quantum bit (*qubit*) can be in any *superposition* of states $|0\rangle$ and $|1\rangle$.

Quantum computing

Quantum computing has been suggested by Paul Benioff in 1980. He proposed a quantum mechanical model of the Turing machine, using two ingredients:

- Superposition: while a bit can be either in a state 0 or 1, a quantum bit (*qubit*) can be in any *superposition* of states $|0\rangle$ and $|1\rangle$.
- Entanglement: the capability of two qubits to be *correlated*. If Alice and Bob both get one of two entangled qubits, and if Alice measures a $|0\rangle$ at some point, then necessarily Bob must measure the same, as $|00\rangle$ is the only state where Alice's qubit is a $|0\rangle$.

Quantum computing

Quantum computing has been suggested by Paul Benioff in 1980. He proposed a quantum mechanical model of the Turing machine, using two ingredients:

- Superposition: while a bit can be either in a state 0 or 1, a quantum bit (*qubit*) can be in any *superposition* of states $|0\rangle$ and $|1\rangle$.
- Entanglement: the capability of two qubits to be *correlated*. If Alice and Bob both get one of two entangled qubits, and if Alice measures a $|0\rangle$ at some point, then necessarily Bob must measure the same, as $|00\rangle$ is the only state where Alice's qubit is a $|0\rangle$.

Qubits can be "implemented" using the spin of an electron, or the polarization of a photon, ...

Quantum computing

As a consequence:

- a vector of n entangled qubits can be in a superposition of any 2^n possible states at the same time,

Quantum computing

As a consequence:

- a vector of n entangled qubits can be in a superposition of any 2^n possible states at the same time,
- against 1 among the 2^n possible states for a classical vector of n bits.

Quantum computing

As a consequence:

- a vector of n entangled qubits can be in a superposition of any 2^n possible states at the same time,
- against 1 among the 2^n possible states for a classical vector of n bits.

It is however not possible to observe these states all together at the same time.

Quantum computing

As a consequence:

- a vector of n entangled qubits can be in a superposition of any 2^n possible states at the same time,
- against 1 among the 2^n possible states for a classical vector of n bits.

It is however not possible to observe these states all together at the same time.
A quantum algorithm solving a problem needs to make the correct solution (state) exponentially more likely than the other states (cf. quantum annealing / wave function collapsing).

Outline

Shor's algorithm

SIAM J. COMPUT.
Vol. 26, No. 5, pp. 1484–1509, October 1997

POLYNOMIAL-TIME ALGORITHMS FOR PRIME FACTORIZATION AND DISCRETE LOGARITHMS ON A QUANTUM COMPUTER*

PETER W. SHOR†

Abstract. A digital computer is generally believed to be an efficient universal computing device; that is, it is believed able to simulate any physical computing device with an increase in computation time by at most a polynomial factor. This may not be true when quantum mechanics is taken into consideration. This paper considers factoring integers and finding discrete logarithms, two problems which are generally thought to be hard on a classical computer and which have been used as the basis of several proposed cryptosystems. Efficient randomized algorithms are given for these two problems on a hypothetical quantum computer. These algorithms take a number of steps polynomial in the input size, e.g., the number of digits of the integer to be factored.

Shor's algorithm: how it works

Algorithm 1: ShorAlgorithm(N)

Input: N

Output: p, q such that $N = pq$

Shor's algorithm: how it works

Algorithm 1: ShorAlgorithm(N)

Input: N

Output: p, q such that $N = pq$

1 Pick $g \in \mathbb{Z}_N$ at random;

Shor's algorithm: how it works

Algorithm 1: ShorAlgorithm(N)

Input: N

Output: p, q such that $N = pq$

```
Pick g ∈ Z_N at random;
if gcd(g, N) ≠ 1 then
    then return (p = gcd(g, N), q = N/p)
```

Shor’s algorithm: how it works

Algorithm 1: ShorAlgorithm(N)

Input: N

Output: p, q such that $N = pq$

Pick $g \in \mathbb{Z}_N$ at random;
if $\gcd(g, N) \neq 1$ **then**
  then **return** $(p = \gcd(g, N), q = N/p)$
Find r such that $g^r \equiv 1[N]$;

Shor's algorithm: how it works

Algorithm 1: ShorAlgorithm(N)

Input: N

Output: p, q such that $N = pq$

```
1 Pick g ∈ ℤ_N at random;
2 if gcd(g, N) ≠ 1 then
3 |   then return (p = gcd(g, N), q = N/p)
4 Find r such that g^r ≡ 1[N];
5 if r ≡ 0[2] then
6 |   return gcd(g^{r/2} ± 1, N)
7 else
8 |   go to 1
```

Shor's algorithm: how it works

"Find r such that $g^r \equiv 1[N]$;"

Shor's algorithm: how it works

"Find r such that $g^r \equiv 1[N]$;"

- First question: How does finding r such that $g^r \equiv 1[N]$ help factoring?

Shor's algorithm: how it works

"Find r such that $g^r \equiv 1[N]$;"

- First question: How does finding r such that $g^r \equiv 1[N]$ help factoring?

$$g^r \equiv 1[N] \quad \Leftrightarrow \quad \exists k \in \mathbb{N}^* \text{ such that } g^r = kN + 1$$

Shor's algorithm: how it works

"Find r such that $g^r \equiv 1[N]$;"

- First question: How does finding r such that $g^r \equiv 1[N]$ help factoring?

$$\begin{aligned} g^r \equiv 1[N] \quad &\Leftrightarrow \quad \exists k \in \mathbb{N}^* \text{ such that } g^r = kN + 1 \\ &\Leftrightarrow \quad g^r - 1 = kN \end{aligned}$$

Shor's algorithm: how it works

"Find r such that $g^r \equiv 1[N]$;"

- First question: How does finding r such that $g^r \equiv 1[N]$ help factoring?

$$
\begin{aligned}
g^r \equiv 1[N] \quad &\Leftrightarrow \quad \exists k \in \mathbb{N}^* \text{ such that } g^r = kN + 1 \\
&\Leftrightarrow \quad g^r - 1 = kN \\
\textbf{(assuming } r \textbf{ is even)} \quad &\Leftrightarrow \quad \left(g^{r/2} - 1\right)\left(g^{r/2} + 1\right) = kN
\end{aligned}
$$

Shor's algorithm: how it works

"Find r such that $g^r \equiv 1[N]$;"

- First question: How does finding r such that $g^r \equiv 1[N]$ help factoring?

$$
\begin{aligned}
g^r \equiv 1[N] &\Leftrightarrow \exists k \in \mathbb{N}^* \text{ such that } g^r = kN + 1 \\
&\Leftrightarrow g^r - 1 = kN \\
\textbf{(assuming } r \textbf{ is even)} &\Leftrightarrow \left(g^{r/2} - 1\right)\left(g^{r/2} + 1\right) = kN
\end{aligned}
$$

Meaning that there is a non-negligible probability that $g^{r/2} \pm 1$ shares non trivial factors with N.

Shor's algorithm: how it works

Example with $N = 314191$, find p, q

(source: minutephysics)

Shor's algorithm: how it works

Example with $N = 314191$, find p, q

(source: minutephysics)

- step 1. $g \leftarrow 101$

Shor's algorithm: how it works

Example with $N = 314191$, find p, q

(source: minutephysics)

- step 1. $g \leftarrow 101$
- step 2. $r \leftarrow 4347$

Shor's algorithm: how it works

Example with $N = 314191$, find p, q

(source: minutephysics)

- step 1. $g \leftarrow 101$
- step 2. $r \leftarrow 4347$
- step 3. r is **odd**... go to 1

Shor's algorithm: how it works

Example with $N = 314191$, find p, q

(source: minutephysics)

- step 1. $g \leftarrow 101$
- step 2. $r \leftarrow 4347$
- step 3. r is **odd**... go to 1
- step 1. $g \leftarrow 127$

Shor's algorithm: how it works

Example with $N = 314191$, find p, q

(source: minutephysics)

- step 1. $g \leftarrow 101$
- step 2. $r \leftarrow 4347$
- step 3. r is **odd**... go to 1
- step 1. $g \leftarrow 127$
- step 2. $r \leftarrow 17388$

Shor's algorithm: how it works

Example with $N = 314191$, find p, q **(source: minutephysics)**

- step 1. $g \leftarrow 101$
- step 2. $r \leftarrow 4347$
- step 3. r is **odd**... go to 1
- step 1. $g \leftarrow 127$
- step 2. $r \leftarrow 17388$
- step 3. let us denote $g_p = g^{17388/2} + 1$ and $g_q = g^{17388/2} - 1$

Shor's algorithm: how it works

Example with $N = 314191$, find p, q

(source: minutephysics)

- step 1. $g \leftarrow 101$
- step 2. $r \leftarrow 4347$
- step 3. r is **odd**... go to 1
- step 1. $g \leftarrow 127$
- step 2. $r \leftarrow 17388$
- step 3. let us denote $g_p = g^{17388/2} + 1$ and $g_q = g^{17388/2} - 1$
  we have that $\gcd(g_p, N) = 829 =: p$ and $\gcd(g_q, N) = 379 =: q$
  and indeed, $p \cdot q = 829 \times 379 = 314191 = N$

Shor's algorithm: how it works

- Second question: "*Wait a minute, I was expecting some magic quantum trick out there. Where the is the quantum part?*"

Shor's algorithm: how it works

- Second question: "*Wait a minute, I was expecting some magic quantum trick out there. Where the is the quantum part?*"

$$\text{"Find } r \text{ such that } g^r \equiv 1[N];\text{"}$$

Shor's algorithm: how it works

- Second question: "*Wait a minute, I was expecting some magic quantum trick out there. Where the is the quantum part?*"

"**Quantumly** find r such that $g^r \equiv 1[N]$;"

Shor's algorithm: how it works

- Second question: "*Wait a minute, I was expecting some magic quantum trick out there. Where the is the quantum part?*"

"**Quantumly** find r such that $g^r \equiv 1[N]$;"

The complexity to find the *period* of the function $g \mapsto g^x \mod N$ is:

Shor's algorithm: how it works

- Second question: "*Wait a minute, I was expecting some magic quantum trick out there. Where the is the quantum part?*"

"**Quantumly** find r such that $g^r \equiv 1[N]$;"

The complexity to find the *period* of the function $g \mapsto g^x \mod N$ is:

- Classically $\mathcal{O}(N)$

Shor's algorithm: how it works

- Second question: "*Wait a minute, I was expecting some magic quantum trick out there. Where the is the quantum part?*"

"**Quantumly** find r such that $g^r \equiv 1[N]$;"

The complexity to find the *period* of the function $g \mapsto g^x \mod N$ is:

- Classically $\mathcal{O}(N)$
- Quantumly $\mathcal{O}\left(\log(N)^3\right)$

Shor's algorithm: how it works

- Second question: "*Wait a minute, I was expecting some magic quantum trick out there. Where the is the quantum part?*"

"**Quantumly** find r such that $g^r \equiv 1[N]$;"

The complexity to find the *period* of the function $g \mapsto g^x \mod N$ is:

- Classically $\mathcal{O}(N)$
- Quantumly $\mathcal{O}\left(\log(N)^3\right)$. That's an **exponential** speedup!

Quantum period finding

How does it work? Why is it much much faster quantumly?

Fourier Transform is THE tool to analyse frequencies. Fortunately, it has a quantum equivalent: QFT.

Quantum computing allows to provide QFT a superposition of every possible states (assuming enough qubits).

Consequences of Shor's algorithm on PKC

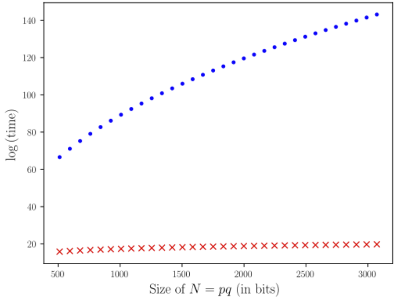
- Factoring becomes polynomial-time
  $\mathcal{O}\left((\log N)^2 (\log\log N) (\log\log\log N)\right)$

Consequences of Shor's algorithm on PKC

- Factoring becomes polynomial-time $\mathcal{O}\left((\log N)^2 (\log\log N)(\log\log\log N)\right)$ against $\exp\left(1.9(\log N)^{1/3}(\log\log N)^{2/3}\right)$ classically

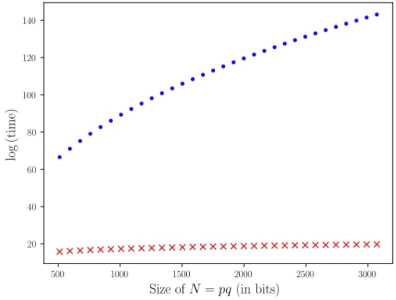
Consequences of Shor's algorithm on PKC

- Factoring becomes polynomial-time $\mathcal{O}\left((\log N)^2 (\log\log N)(\log\log\log N)\right)$ against $\exp\left(1.9(\log N)^{1/3}(\log\log N)^{2/3}\right)$ classically



Consequences of Shor's algorithm on PKC

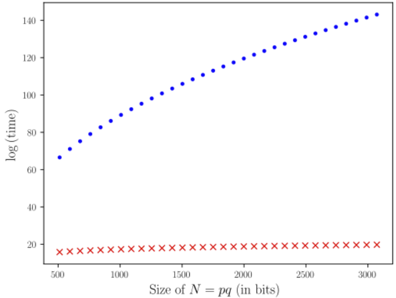
- Factoring becomes polynomial-time
  $\mathcal{O}\left((\log N)^2 (\log \log N) (\log \log \log N)\right)$
  against $\exp\left(1.9(\log N)^{1/3}(\log \log N)^{2/3}\right)$ classically
- Discrete logarithm becomes almost polynomial-time



Consequences of Shor's algorithm on PKC

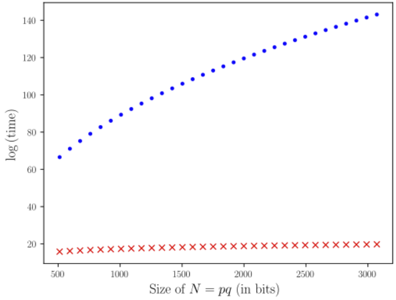
- Factoring becomes polynomial-time
  $\mathcal{O}\left((\log N)^2 (\log\log N)(\log\log\log N)\right)$
  against $\exp\left(1.9(\log N)^{1/3}(\log\log N)^{2/3}\right)$ classically
- Discrete logarithm becomes almost polynomial-time

No more RSA, DSA, ECDSA, ElGamal, ...



Consequences of Shor's algorithm on PKC

- Factoring becomes polynomial-time $\mathcal{O}\left((\log N)^2 (\log\log N)(\log\log\log N)\right)$ against $\exp\left(1.9(\log N)^{1/3}(\log\log N)^{2/3}\right)$ classically
- Discrete logarithm becomes almost polynomial-time



No more RSA, DSA, ECDSA, ElGamal, ...

In other words, **security as we know it collapses**...

Grover's algorithm

A fast quantum mechanical algorithm for database search

Lov K. Grover
3C-404A, Bell Labs
600 Mountain Avenue
Murray Hill NJ 07974
lkgrover@bell-labs.com

Summary

Imagine a phone directory containing N names arranged in completely random order. In order to find someone's phone number with a probability of $\frac{1}{2}$, any classical algorithm (whether deterministic or probabilistic) will need to look at a minimum of $\frac{N}{2}$ names. Quantum mechanical systems can be in a superposition of states and simultaneously examine multiple names. By properly adjusting the phases of various operations, successful computations reinforce each other while others interfere randomly. As a result, the desired phone number can be obtained in only $O(\sqrt{N})$ steps. The algorithm is within a small constant factor of the fastest possible quantum mechanical algorithm.

Consequences of Grover's algorithm

(n-entries unsorted) Database search takes $\mathcal{O}(\sqrt{n})$ queries instead of $\mathcal{O}(n)$.

Consequences of Grover's algorithm

(n-entries unsorted) Database search takes $\mathcal{O}\left(\sqrt{n}\right)$ queries instead of $\mathcal{O}\left(n\right)$.

Consequence over symmetric crypto:

Consequences of Grover's algorithm

(n-entries unsorted) Database search takes $\mathcal{O}\left(\sqrt{n}\right)$ queries instead of $\mathcal{O}\left(n\right)$.

Consequence over symmetric crypto:

$\rightarrow$ The length of the secret key must be **doubled** to preserve the same level of security

Consequences of Grover's algorithm

(n-entries unsorted) Database search takes $\mathcal{O}\left(\sqrt{n}\right)$ queries instead of $\mathcal{O}\left(n\right)$.

Consequence over symmetric crypto:

$\rightarrow$ The length of the secret key must be **doubled** to preserve the same level of security

Consequence over hash functions:

Consequences of Grover's algorithm

(n-entries unsorted) Database search takes $\mathcal{O}\left(\sqrt{n}\right)$ queries instead of $\mathcal{O}\left(n\right)$.

Consequence over symmetric crypto:

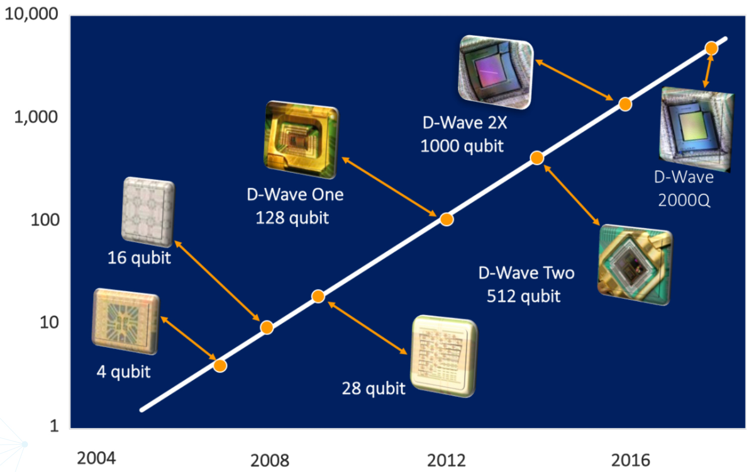
$\rightarrow$ The length of the secret key must be **doubled** to preserve the same level of security

Consequence over hash functions:

$\rightarrow$ More tricky (depending on the model, the size of the quantum computer, ...), at least $+33\%$ to preserve the security level

Outline

How far are we from a large-scale quantum computer?



A quantum analog to Moore's law: the number of qubits (y-axe) approximately doubles every year (x-axe). (Source: D-Wave)

Large-scale quantum computing: a caveat

This analog to Moore's law has several drawbacks:

Large-scale quantum computing: a caveat

This analog to Moore's law has several drawbacks:

- essentially corresponds to multiple 32 qubits architectures mounted in parallel

Large-scale quantum computing: a caveat

This analog to Moore's law has several drawbacks:

- essentially corresponds to multiple 32 qubits architectures mounted in parallel
- fault-tolerance remains an open problem

Large-scale quantum computing: a caveat

This analog to Moore's law has several drawbacks:

- essentially corresponds to multiple 32 qubits architectures mounted in parallel
- fault-tolerance remains an open problem
- still far from what is required to factor 2048 bits moduli

Large-scale quantum computing: a caveat

This analog to Moore's law has several drawbacks:

- essentially corresponds to multiple 32 qubits architectures mounted in parallel
- fault-tolerance remains an open problem
- still far from what is required to factor 2048 bits moduli

In 2020, the largest quantum computer features 72 qubits (Google).

Some perspective

A bit of perspective regarding quantum stuffs.

(full story here)

Some perspective

A bit of perspective regarding quantum stuffs.

(full story here)

- 1968: Wiesner describes conjugate (quantum) coding

Some perspective

A bit of perspective regarding quantum stuffs.

(full story here)

- 1968: Wiesner describes conjugate (quantum) coding
- 1984: Bennett & Brassard use Wiesner's coding for QKD

Some perspective

A bit of perspective regarding quantum stuffs.

(full story here)

- 1968: Wiesner describes conjugate (quantum) coding
- 1984: Bennett & Brassard use Wiesner's coding for QKD
- 1994: Shor's algorithm

Some perspective

A bit of perspective regarding quantum stuffs.

(full story here)

- 1968: Wiesner describes conjugate (quantum) coding
- 1984: Bennett & Brassard use Wiesner's coding for QKD
- 1994: Shor's algorithm
- 1995: 1st US DoD workshop on Quantum Crypto / Shor proposes quantum error correction

Some perspective

A bit of perspective regarding quantum stuffs. **(full story here)**

- 1968: Wiesner describes conjugate (quantum) coding
- 1984: Bennett & Brassard use Wiesner's coding for QKD
- 1994: Shor's algorithm
- 1995: 1st US DoD workshop on Quantum Crypto / Shor proposes quantum error correction
- 1996: Grover's algorithm

Some perspective

A bit of perspective regarding quantum stuffs. **(full story here)**

- 1968: Wiesner describes conjugate (quantum) coding
- 1984: Bennett & Brassard use Wiesner's coding for QKD
- 1994: Shor's algorithm
- 1995: 1st US DoD workshop on Quantum Crypto / Shor proposes quantum error correction
- 1996: Grover's algorithm
- 2001: First execution of Shor's algorithm, factoring 15

Some perspective

A bit of perspective regarding quantum stuffs. **(full story here)**

- 1968: Wiesner describes conjugate (quantum) coding
- 1984: Bennett & Brassard use Wiesner's coding for QKD
- 1994: Shor's algorithm
- 1995: 1st US DoD workshop on Quantum Crypto / Shor proposes quantum error correction
- 1996: Grover's algorithm
- 2001: First execution of Shor's algorithm, factoring 15
- 2002: Creation of the Institute for Quantum Computing

Some perspective

A bit of perspective regarding quantum stuffs. **(full story here)**

- 1968: Wiesner describes conjugate (quantum) coding
- 1984: Bennett & Brassard use Wiesner's coding for QKD
- 1994: Shor's algorithm
- 1995: 1st US DoD workshop on Quantum Crypto / Shor proposes quantum error correction
- 1996: Grover's algorithm
- 2001: First execution of Shor's algorithm, factoring 15
- 2002: Creation of the Institute for Quantum Computing
- 2003: DARPA's quantum network operational

Some perspective

A bit of perspective regarding quantum stuffs. (full story here)

- 1968: Wiesner describes conjugate (quantum) coding
- 1984: Bennett & Brassard use Wiesner's coding for QKD
- 1994: Shor's algorithm
- 1995: 1st US DoD workshop on Quantum Crypto / Shor proposes quantum error correction
- 1996: Grover's algorithm
- 2001: First execution of Shor's algorithm, factoring 15
- 2002: Creation of the Institute for Quantum Computing
- 2003: DARPA's quantum network operational
- 2009: qubits lifetime $\simeq 100$ ms

Some perspective

A bit of perspective regarding quantum stuffs. **(full story here)**

- 1968: Wiesner describes conjugate (quantum) coding
- 1984: Bennett & Brassard use Wiesner's coding for QKD
- 1994: Shor's algorithm
- 1995: 1st US DoD workshop on Quantum Crypto / Shor proposes quantum error correction
- 1996: Grover's algorithm
- 2001: First execution of Shor's algorithm, factoring 15
- 2002: Creation of the Institute for Quantum Computing
- 2003: DARPA's quantum network operational
- 2009: qubits lifetime $\simeq 100$ ms
- 2012: D-Wave claims a quantum computation using 84 qubits (24 computational)

Some perspective

A bit of perspective regarding quantum stuffs. **(full story here)**

- 1968: Wiesner describes conjugate (quantum) coding
- 1984: Bennett & Brassard use Wiesner's coding for QKD
- 1994: Shor's algorithm
- 1995: 1st US DoD workshop on Quantum Crypto / Shor proposes quantum error correction
- 1996: Grover's algorithm
- 2001: First execution of Shor's algorithm, factoring 15
- 2002: Creation of the Institute for Quantum Computing
- 2003: DARPA's quantum network operational
- 2009: qubits lifetime $\simeq 100$ ms
- 2012: D-Wave claims a quantum computation using 84 qubits (24 computational)
- 2013: coherence time 39mins at room temperature / 1st Snowden revelations

Some perspective

A bit of perspective regarding quantum stuffs.

(full story here)

- 1968: Wiesner describes conjugate (quantum) coding
- 1984: Bennett & Brassard use Wiesner's coding for QKD
- 1994: Shor's algorithm
- 1995: 1st US DoD workshop on Quantum Crypto / Shor proposes quantum error correction
- 1996: Grover's algorithm
- 2001: First execution of Shor's algorithm, factoring 15
- 2002: Creation of the Institute for Quantum Computing
- 2003: DARPA's quantum network operational
- 2009: qubits lifetime $\simeq 100$ ms
- 2012: D-Wave claims a quantum computation using 84 qubits (24 computational)
- 2013: coherence time 39mins at room temperature / 1st Snowden revelations
- 2014: NSA project "Penetrating Hard Target" aims at breaking quantumly strong crypto

Some perspective

A bit of perspective regarding quantum stuffs.

(full story here)

- 1968: Wiesner describes conjugate (quantum) coding
- 1984: Bennett & Brassard use Wiesner's coding for QKD
- 1994: Shor's algorithm
- 1995: 1st US DoD workshop on Quantum Crypto / Shor proposes quantum error correction
- 1996: Grover's algorithm
- 2001: First execution of Shor's algorithm, factoring 15
- 2002: Creation of the Institute for Quantum Computing
- 2003: DARPA's quantum network operational
- 2009: qubits lifetime $\simeq 100$ ms
- 2012: D-Wave claims a quantum computation using 84 qubits (24 computational)
- 2013: coherence time 39mins at room temperature / 1st Snowden revelations
- 2014: NSA project "Penetrating Hard Target" aims at breaking quantumly strong crypto
- 2015: NSA statement / 2016: NIST preparing PQC standardization

Some perspective

A bit of perspective regarding quantum stuffs. (full story here)

- 1968: Wiesner describes conjugate (quantum) coding
- 1984: Bennett & Brassard use Wiesner's coding for QKD
- 1994: Shor's algorithm
- 1995: 1st US DoD workshop on Quantum Crypto / Shor proposes quantum error correction
- 1996: Grover's algorithm
- 2001: First execution of Shor's algorithm, factoring 15
- 2002: Creation of the Institute for Quantum Computing
- 2003: DARPA's quantum network operational
- 2009: qubits lifetime $\simeq 100$ ms
- 2012: D-Wave claims a quantum computation using 84 qubits (24 computational)
- 2013: coherence time 39mins at room temperature / 1st Snowden revelations
- 2014: NSA project "Penetrating Hard Target" aims at breaking quantumly strong crypto
- 2015: NSA statement / 2016: NIST preparing PQC standardization
- 2017: industry race for largest quantum computer / D-Wave 2000Q / NIST PQC starts

Some perspective

A bit of perspective regarding quantum stuffs. **(full story here)**

- 1968: Wiesner describes conjugate (quantum) coding
- 1984: Bennett & Brassard use Wiesner's coding for QKD
- 1994: Shor's algorithm
- 1995: 1st US DoD workshop on Quantum Crypto / Shor proposes quantum error correction
- 1996: Grover's algorithm
- 2001: First execution of Shor's algorithm, factoring 15
- 2002: Creation of the Institute for Quantum Computing
- 2003: DARPA's quantum network operational
- 2009: qubits lifetime $\simeq 100$ ms
- 2012: D-Wave claims a quantum computation using 84 qubits (24 computational)
- 2013: coherence time 39mins at room temperature / 1st Snowden revelations
- 2014: NSA project "Penetrating Hard Target" aims at breaking quantumly strong crypto
- 2015: NSA statement / 2016: NIST preparing PQC standardization
- 2017: industry race for largest quantum computer / D-Wave 2000Q / NIST PQC starts
- 2018: Google announces a 72-qubit quantum chip / 2019: quantum supremacy

Hot news!

Hot news!

TECH • QUANTUM COMPUTING

Google Claims 'Quantum Supremacy,' Marking a Major Milestone in Computing

By Robert Hackett September 20, 2019

Hot news!

TECH • QUANTUM COMPUTING

Google Claims 'Quantum Supremacy,' Marking Major Milestone in Computing

By Robert Hackett September 20, 2019

Google's supposed milestone achievement became public last month when a preprint scientific paper accidentally leaked on the website of NASA, a collaborator, as *Fortune* reported at the time. Google has said nothing about the potentially historic experiment since then, lending credence to whispers that its researchers are bound to silence under the terms of a news embargo by a major science journal, unable to disclose more information until a certain date which is presumed to be imminent.

Hot news!

Google Cla

N

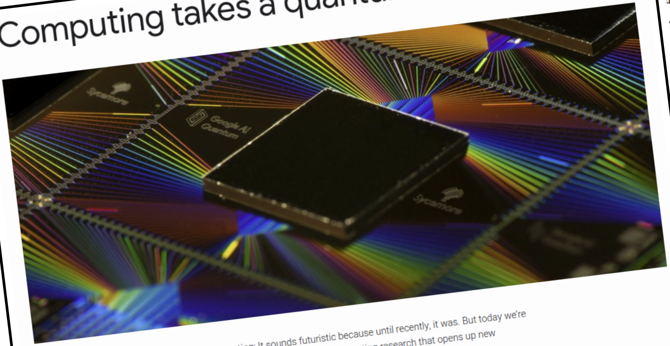
...rking

Google's supposed milesto...

preprint ...

...ecame public last month when a ... on the website of NASA, a collaborator, ...aid nothing about the potentially ...nce to whispers that its researchers are ...bargo by a major science journal, ...ain date which is presumed to be

Computing takes a quantum leap forward



Hartmut Neven
Engineering Director, Google AI Quantum Team

Published Oct 23, 2019

Quantum computing: It sounds futuristic because until recently, it was. But today we're marking a major milestone in quantum computing research that opens up new possibilities for this technology.

Unlike classical computing, which runs everything from your cell phone to a supercomputer, quantum computing is based on the properties of quantum mechanics. As a result, quantum computers could potentially solve problems that would be too difficult or even impossible for classical computers—like designing better batteries,

What is quantum supremacy?

What is quantum supremacy?

Quantum supremacy refers to the moment where a functional quantum computer can effectively solve a problem that is not solvable (within decent time frame, *e.g.* 100 years) with any (super) computer.

Wha

Quantum supre … ntum computer can
effectively solve … frame, *e.g.* 100 years) with
any (super) cor

Why is Google's quantum supremacy experiment impressive?

Asked 13 days ago Active 11 days ago Viewed 12k times

129

21

In the Nature paper published by Google, they say,

> To demonstrate quantum supremacy, we compare our quantum processor against state-of-the-art classical computers in the task of sampling the output of a pseudo-random quantum circuit. Random circuits are a suitable choice for benchmarking because they do not possess structure and therefore allow for limited guarantees of computational hardness. We design the circuits to entangle a set of quantum bits (qubits) by repeated application of single-qubit and two-qubit logical operations. Sampling the quantum circuit's output produces a set of bitstrings, for example {0000101, 1011100, …}. Owing to quantum interference, the probability distribution of the bitstrings resembles a speckled intensity pattern produced by light interference in laser scatter, such that some bitstrings are much more likely to occur than others. Classically computing this probability distribution becomes exponentially more difficult as the number of qubits (width) and number of gate cycles (depth) grow.

So, from what I can tell, they configure their qubits into a pseudo-randomly generated circuit, which, when run, puts the qubits into a state vector that represents a probability distribution over 2^{53} possible states of the qubits, but that distribution is intractable to calculate, or even estimate via sampling using a classical computer simulation. But they sample it by "looking" at the state of the qubits after running the circuit many times.

Isn't this just an example of creating a system whose output is intractable to calculate, and then "calculating" it by simply observing the output of the system?

It sounds similar to saying:

> If I spill this pudding cup on the floor, the exact pattern it will form is very chaotic, and intractable for any supercomputer to calculate. But I just invented a new special type of computer: this pudding cup. And I'm going to do the calculation by spilling it on the floor and observing the result. I have achieved pudding supremacy.

What i...

Quantum suprem... effectively solve a... any (super) comp... ...antum computer can ...e frame, *e.g.* 100 years) with

Why is Google's quantum supremacy experiment impressive?

Asked 13 days ago Active 11 days ago Viewed 12k times

129

21

In the Nature paper published by Google, they say,

> To demonstrate quantum supremacy, we compare our quantum processor against state-of-the-art classical computers in the task of sampling the output of a pseudo-random quantum circuit. Random circuits are a suitable choice for benchmarking because they do not possess structure and therefore allow for limited guarantees of computational hardness. We design the circuits to entangle a set of quantum bits (qubits) by repeated application of single-qubit and two-qubit logical operations. Sampling the quantum circuit's output produces a set of bitstrings, for example {0000101, 1011100, …}. Owing to quantum interference, the probability distribution of the bitstrings resembles a speckled intensity pattern produced by light interference in laser scatter, such that some bitstrings are much more likely to occur than others. Classically computing this probability distribution becomes exponentially more difficult as the number of qubits (width) and number of gate cycles (depth) grow.

So, from what I can tell, they configure their qubits into a pseudo-randomly generated circuit, which, when run, puts the qubits into a state vector that represents a probability distribution over 2^{53} possible states of the qubits, but that distribution is intractable to calculate, or even estimate via sampling using a classical computer simulation. But they sample it by "looking" at the state of the qubits after running the circuit many times.

Isn't this just an example of creating a system whose output is intractable to calculate, and then "calculating" it by simply observing the output of the system?

It sounds similar to saying:

> If I spill this pudding cup on the floor, the exact pattern it will form is very chaotic, and intractable for any supercomputer to calculate. But I just invented a new special type of computer: this pudding cup. And I'm going to do the calculation by spilling it on the floor and observing the result. I have achieved pudding supremacy.

What is quantum supremacy?

Quantum supremacy refers to the moment where a functional quantum computer can effectively solve a problem that is not solvable (within decent time frame, *e.g.* 100 years) with any (super) computer.

This result is a bit biased and overselled: it was obtained using a very specific (ad-hoc) problem that was purposely designed to behave much much better quantumly than classicaly...

What is quantum supremacy?

Quantum supremacy refers to the moment where a functional quantum computer can effectively solve a problem that is not solvable (within decent time frame, *e.g.* 100 years) with any (super) computer.

This result is a bit biased and overselled: it was obtained using a very specific (ad-hoc) problem that was purposely designed to behave much much better quantumly than classicaly...

It however remains impressive, since no regular computer can do that efficiently. A bit weaker than supremacy is “quantum advantage”, where a quantum computer simply performs better than any computer.

Open challenges towards quantum computing

More work is required to embrace a large scale quantum computer:

Open challenges towards quantum computing

More work is required to embrace a large scale quantum computer:

- developing quantum error-correcting codes for error-free quantum computing

Open challenges towards quantum computing

More work is required to embrace a large scale quantum computer:

- developing quantum error-correcting codes for error-free quantum computing
- developing reliable quantum memories

Open challenges towards quantum computing

More work is required to embrace a large scale quantum computer:

- developing quantum error-correcting codes for error-free quantum computing
- developing reliable quantum memories
- quantum measurement (wave function collapsing) is probabilistic

Open challenges towards quantum computing

More work is required to embrace a large scale quantum computer:

- developing quantum error-correcting codes for error-free quantum computing
- developing reliable quantum memories
- quantum measurement (wave function collapsing) is probabilistic
- building architectures and interfaces between quantum computers and communication systems

Open challenges towards quantum computing

More work is required to embrace a large scale quantum computer:

- developing quantum error-correcting codes for error-free quantum computing
- developing reliable quantum memories
- quantum measurement (wave function collapsing) is probabilistic
- building architectures and interfaces between quantum computers and communication systems
- developing quantum programming languages, compilers and middle-ware stack

Open challenges towards quantum computing

More work is required to embrace a large scale quantum computer:

- developing quantum error-correcting codes for error-free quantum computing
- developing reliable quantum memories
- quantum measurement (wave function collapsing) is probabilistic
- building architectures and interfaces between quantum computers and communication systems
- developing quantum programming languages, compilers and middle-ware stack

Still, a Sword of Damocles hanging over our heads, and **now** is the time for designing **quantum-safe** alternatives.

Outline

Possible alternative: physical cryptography

- Imagine a lockable-briefcase salesman proposing a "locked-briefcase Internet"

Possible alternative: physical cryptography

- Imagine a lockable-briefcase salesman proposing a "locked-briefcase Internet"
- using "provably secure locked-briefcase cryptography":
  - Alice puts secret information into a lockable briefcase.
  - Alice locks the briefcase.
  - A courier transports the briefcase from Alice to Bob.
  - Bob unlocks the briefcase and retrieves the information.
  - There is a mathematical proof that the information is hidden!
  - Throw away algorithmic cryptography!

Possible alternative: physical cryptography

- Imagine a lockable-briefcase salesman proposing a "locked-briefcase Internet"
- using "provably secure locked-briefcase cryptography":
  - Alice puts secret information into a lockable briefcase.
  - Alice locks the briefcase.
  - A courier transports the briefcase from Alice to Bob.
  - Bob unlocks the briefcase and retrieves the information.
  - There is a mathematical proof that the information is hidden!
  - Throw away algorithmic cryptography!

- Most common reactions from security experts:
  - This would make security much worse.
  - You can't do signatures.
  - This would be insanely expensive.
  - We should not dignify this proposal with a response

Security advantages of algorithmic cryptography

- Keep secrets heavily shielded inside authorized computers.

Security advantages of algorithmic cryptography

- Keep secrets heavily shielded inside authorized computers.
- Reduce trust in third parties:
  - Reduce reliance on closed-source software and hardware.
  - Increase comprehensiveness of audits.
  - Increase comprehensiveness of formal verification.
  - Design systems to be secure even if algorithm and public keys are public.
  - Critical example: signed software updates.

Security advantages of algorithmic cryptography

- Keep secrets heavily shielded inside authorized computers.
- Reduce trust in third parties:
  - Reduce reliance on closed-source software and hardware.
  - Increase comprehensiveness of audits.
  - Increase comprehensiveness of formal verification.
  - Design systems to be secure even if algorithm and public keys are public.
  - Critical example: signed software updates.
- Understand security as thoroughly as possible:
  - Publish comprehensive specifications.
  - Build large research community with clear security goals.
  - Publicly document attack efforts.
  - Require systems to convincingly survive many years of analysis.

Confidence-inspiring crypto takes time to build

- Many stages of research from cryptographic design to deployment:
  - Explore space of cryptosystems.
  - Study algorithms for the attackers.
  - Focus on secure cryptosystems.
  - Study algorithms for the users.
  - Study implementations on real hardware.
  - Study side-channel attacks, fault attacks, etc.
  - Focus on secure, reliable implementations.
  - Focus on implementations meeting performance requirements.
  - Integrate securely into real-world applications.

Confidence-inspiring crypto takes time to build

- Many stages of research from cryptographic design to deployment:
  - Explore space of cryptosystems.
  - Study algorithms for the attackers.
  - Focus on secure cryptosystems.
  - Study algorithms for the users.
  - Study implementations on real hardware.
  - Study side-channel attacks, fault attacks, etc.
  - Focus on secure, reliable implementations.
  - Focus on implementations meeting performance requirements.
  - Integrate securely into real-world applications.
- Example: ECC introduced 1985; big advantages over RSA. Robust ECC started to take over the Internet in 2015.

Confidence-inspiring crypto takes time to build

- Many stages of research from cryptographic design to deployment:
  - Explore space of cryptosystems.
  - Study algorithms for the attackers.
  - Focus on secure cryptosystems.
  - Study algorithms for the users.
  - Study implementations on real hardware.
  - Study side-channel attacks, fault attacks, etc.
  - Focus on secure, reliable implementations.
  - Focus on implementations meeting performance requirements.
  - Integrate securely into real-world applications.
- Example: ECC introduced 1985; big advantages over RSA. Robust ECC started to take over the Internet in 2015.
- Can't wait for quantum computers before finding a solution!

Confidence-inspiring crypto takes time to build

- Many stages of research from cryptographic design to deployment:
  - Explore space of cryptosystems.
  - Study algorithms for the attackers.
  - Focus on secure cryptosystems.
  - Study algorithms for the users.
  - Study implementations on real hardware.
  - Study side-channel attacks, fault attacks, etc.
  - Focus on secure, reliable implementations.
  - Focus on implementations meeting performance requirements.
  - Integrate securely into real-world applications.
- Example: ECC introduced 1985; big advantages over RSA. Robust ECC started to take over the Internet in 2015.
- Can't wait for quantum computers before finding a solution!

Let's move to post-quantum crypto now!

Outline

Clarification

What are the alternatives to classical cryptography in presence of an adversary equipped with a large scale quantum computer?

Clarification

What are the alternatives to classical cryptography in presence of an adversary equipped with a large scale quantum computer?

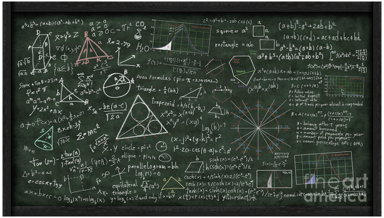
- Quantum Key Exchange (out of the scope of this course)

Clarification

What are the alternatives to classical cryptography in presence of an adversary equipped with a large scale quantum computer?

- Quantum Key Exchange (out of the scope of this course)

- Post-Quantum Cryptography



Post-Quantum Cryptography

What are the ingredients for building quantum-safe cryptographic primitives?

Post-Quantum Cryptography

What are the ingredients for building quantum-safe cryptographic primitives?

- Lattice-based cryptography

Post-Quantum Cryptography

What are the ingredients for building quantum-safe cryptographic primitives?

- Lattice-based cryptography
- (Error-correcting) Code-based cryptography

Post-Quantum Cryptography

What are the ingredients for building quantum-safe cryptographic primitives?

- Lattice-based cryptography
- (Error-correcting) Code-based cryptography
- Hash (function) - based cryptography

Post-Quantum Cryptography

What are the ingredients for building quantum-safe cryptographic primitives?

- Lattice-based cryptography
- (Error-correcting) Code-based cryptography
- Hash (function) - based cryptography
- Multivariate (polynomials) - based cryptography

Post-Quantum Cryptography

What are the ingredients for building quantum-safe cryptographic primitives?

- Lattice-based cryptography
- (Error-correcting) Code-based cryptography
- Hash (function) - based cryptography
- Multivariate (polynomials) - based cryptography
- Isogeny (over elliptic curves) - based cryptography

NIST PQC standardization process

NIST National Institute of Standards and Technologies

NIST PQC standardization process

NIST **N**ational **I**nstitute of **S**tandards and **T**echnologies

- 3rd call for standardization
- Asks for post-quantum cryptographic algorithms
- 3 categories :
  - Encryption
  - Key exchange
  - Signature
- Many candidates:
  - Error correcting codes,
  - Lattices,
  - Multivariate,
  - Hash functions,
  - ...

NIST PQC standardization process

NIST National Institute of Standards and Technologies

- 3rd call for standardization
- Asks for post-quantum cryptographic algorithms
- 3 categories :
  - Encryption
  - Key exchange
  - Signature
- Many candidates:
  - Error correcting codes,
  - Lattices,
  - Multivariate,
  - Hash functions,
  - ...

- November 2016: anouncement
- November 2017: submission deadline (82 submissions)
- December 2017: 1st round: 69 submissions
- April 2018: 1st standardization conference
- January 2019: 2nd round: 26 candidates
- March 2019: tweaks for 2nd round
- August 2019: 2nd standardization conference
- July 2020: 3rd round: 7 finalists, 8 alternates
- 2022 $\rightarrow$ 2024: draft standards ready

Hot topic!

	Signatures	KEM/Encryption	Overall
Lattice-based	4	24	28
Code-based	5	19	24
Multi-variate	7	6	13
Hash-based	4		4
Other	3	10	13
Total	23	59	82

Submissions available at:

- https://csrc.nist.gov/Projects/post-quantum-cryptography/Post-Quantum-Cryptography-Standardization
- https://www.safecrypto.eu/pqclounge/

source:
Dustin Moody, NIST

Hot topic!

Below is a timeline of major events with respect to the NIST PQC Standardization Process.

- April 2-3, 2015 — Workshop on Cybersecurity in a Post-Quantum World, NIST, Gaithersburg, MD
- February 24, 2016 — PQC Standardization: Announcement and outline of NIST's Call for Submissions presentation given at PQCrypto 2016
- April 28, 2016 — NISTIR 8105, Report on Post-Quantum Cryptography, released
- August 2, 2016 — Federal Register Notice - Proposed Requirements and Evaluation Criteria announced for public comment
- December 20, 2016 — Federal Register Notice – Announcing Request for Nominations for Public-Key Post-Quantum Cryptographic Algorithms
- November 30, 2017 — Submission Deadline for NIST PQC Standardization Process
- December 20, 2017 — First-Round Candidates were announced. The public comment period on the first-round candidates began.
- April 11-13, 2018 — First NIST PQC Standardization Conference, Ft. Lauderdale, FL
- January 30, 2019 — The First Round ended and the Second Round began. Second-Round candidates announced. The public comment period on the second-round candidates began.
- March 15, 2019 — Deadline for updated submission packages for the Second Round
- August 22-24, 2019 — 2nd NIST PQC Standardization Conference, Santa Barbara, CA

source:
NIST IR 8240

Hot topic!

Timeline

**This is a tentative timeline, provided for information, and subject to change.*

Date	
Feb 24-26, 2016	NIST Presentation at PQCrypto 2016: *Announcement and outline of NIST's Call for Submissions (Fall 2016), Dustin Moody*
April 28, 2016	NIST releases NISTIR 8105, Report on Post-Quantum Cryptography
Dec 20, 2016	Formal Call for Proposals
Nov 30, 2017	Deadline for submissions
Dec 4, 2017	NIST Presentation at AsiaCrypt 2017: *The Ship Has Sailed: The NIST Post-Quantum Crypto "Competition", Dustin Moody*
Dec 21, 2017	Round 1 algorithms announced (69 submissions accepted as "complete and proper")
Apr 11, 2018	NIST Presentation at PQCrypto 2018: *Let's Get Ready to Rumble - The NIST PQC "Competition"*, *Dustin Moody*
April 11-13, 2018	First PQC Standardization Conference - Submitter's Presentations
January 30, 2019	Second Round Candidates announced (26 algorithms)
March 15, 2019	Deadline for updated submission packages for the Second Round
May 8-10, 2019	NIST Presentation at PQCrypto 2019: Round 2 of the NIST PQC "Competition" - What was NIST Thinking? (Spring 2019), *Dustin Moody*
August 22-24, 2019	Second PQC Standardization Conference
2020/2021	Round 3 begins or select algorithms
2022/2024	Draft Standards Available

Outline