Post-Quantum Manifesto

Philippe Lamontagne
1. Public-key Cryptography Recap
2. How Quantum Mechanics Breaks Encryption
3. Post-Quantum Cryptography
4. Quantum Key Distribution
PUBLIC KEY CRYPTOGRAPHY
The Public Key Revolution (’70s)

New Directions in Cryptography
Invited Paper
WHITFIELD DIFFIE AND MARTIN E. HELLMAN, MEMBER, IEEE

A Method for Obtaining Digital Signatures and Public-Key Cryptosystems
R.L. Rivest, A. Shamir, and L. Adleman*
Reminder

**Encryption**
- **Encrypt**
  - Public key
- **Decrypt**
  - Private key

**Digital Signatures**
- **Verify**
  - Public key
- **Sign**
  - Private key
Public Key Cryptography

Many wonderful tools

- Public key encryption
- Key exchange
- Digital signatures
- Public key certificates

Assumes “one-way functions”

- Factoring (RSA) \( N = p \cdot q \)
- Discrete logarithm (DH, ECC)

\[ x \text{ such that } b^x \equiv a \pmod{n} \]
The Public Key Infrastructure

- Root CA
- Intermediate Certificate
- Developer Certificate
- Web pages, binaries, etc.

Trust
Signatures
QUANTUM COMPUTING
Incomplete History of Quantum Mechanics

• ‘20s – ’30s: emerged as a new formalism of physics.
• 1935: Einstein dismisses QM (incomplete theory).
• 1964: John Stewart Bell “proves” Einstein wrong.
• 1983: Wiesner’s quantum money.
• 1984: Bennet and Brassard invent quantum key distribution.
• 1996: Peter Shor is factoring and discrete-logging on quantum computers.
Quantum Computing: Superposition

“Schrödinger’s cat”

If then = 42

Collapse

= 16 cat combinations

Credits: Wikipedia user Dhatfield, tumblr user Igor Canova
Quantum Computing: Entanglement

Teleporting an Unknown Quantum State via Dual Classical and Einstein-Podolsky-Rosen Channels

Charles H. Bennett,\(^{(1)}\) Gilles Brassard,\(^{(2)}\) Claude Crépeau,\(^{(2),(3)}\)

Quantum Pseudo-Telepathy

Gilles Brassard,\(^{1,*}\) Anne Broadbent,\(^{1,†}\) and Alain Tapp\(^{1,‡}\)

“Spooky action at a distance”
Shor’s Algorithm

“... To find a factor of an odd number n, given a method for computing the order r of x, choose a random x (mod n), find its order r, and compute gcd(x^r/2 - 1, n).”

- Order: number r such that x^r \equiv 1 \pmod{n}
- Build superposition of:

| x^1 \pmod{n} > + | x^2 \pmod{n} > + \cdots + | x^q \pmod{n} >

- Quantum Fourier Transform (QFT): $O(\log^2 n)$
- Fast Fourier Transform (FFT): $O(n \cdot \log n)$
## The Quantum Aftermath

<table>
<thead>
<tr>
<th>Quantum Algorithm</th>
<th>Impacts</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Shor</strong> (factoring)</td>
<td>• RSA broken</td>
</tr>
</tbody>
</table>
| **Shor** (discrete logarithm) | • (EC)Diffie-Hellman broken  
• (EC)DSA broken |
| **Grover** (unstructured search)  
$O(N) \rightarrow O(\sqrt{N})$ | • Symmetric Cryptography AES-128 -> AES-256  
• Hash functions SHA-256 -> SHA-512 |
Why aren’t we panicking?

Yet we should do something about it now.

Time data needs to be secret ➔ Time to transition to PQC
Time to large-scale quantum computer

“Store now, decrypt later”
POST-QUANTUM CRYPTOGRAPHY
Finding New Hard Problems

Trapdoor permutation

\[ x \xrightarrow{f} f(x) \text{ Easy} \]

\[ x \xleftarrow{f} f(x) \text{ Hard} \]

\[ x \xleftarrow{f} f(x) \text{ Easy} \]
Lattice-based Cryptography

Key generation
- Private key is “good” basis of lattice
- Public key is “bad” basis

Encryption/Decryption
- Message is lattice point
- To encrypt, add noise vector to a properly chosen lattice point
- To decrypt, find the closest lattice point to noisy vector

Shortest Vector Problem

Closest Vector Problem
Multivariate Cryptography

Hardness of inverting systems of multivariate quadratic polynomials

1. Choose \( F \) easy to invert
2. Hide \( F \) with \( S \) and \( T \) easy to invert
3. Public key \( P \) is hard to invert, but easy knowing private key \((S, F, T)\).

\[
F(x, y) = 2xy + 4x^2 + 2y + 7
\]

\[
P = S \circ F \circ T = \]

Decryption/signature generation

Encryption/signature verification
Code-based Cryptography

- Hardness of decoding a random linear error correcting code
- Private key is an efficient decoder for an ECC
- Public key is “random” generator matrix for the same ECC
- Encrypt by adding errors. Decrypt by correcting.

Error Correcting Code

\[ G = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \end{pmatrix} \]

\[ w = 011 \mapsto c = 01110 \]
Hash-based Signatures

- Lamport one-time signatures
- Merkle signature scheme
  - Stateful
  - $2^N$ one-time signatures with a single public key
- PRNG for one-time key generation

\[
\text{sign } m = 101: \quad K_{priv} = \begin{pmatrix}
    x_{0,1} & x_{0,2} & x_{0,3} \\
    x_{1,1} & x_{1,2} & x_{1,3}
\end{pmatrix}
\]

\[\implies \sigma = (x_{1,1}, x_{0,2}, x_{1,3})\]

verify signature $\sigma = (x_1, x_2, x_3)$ for $m = 101$:

\[
\text{sign } m = 101: \quad K_{pub} = \begin{pmatrix}
    y_{1,0} & y_{2,0} & y_{3,0} \\
    y_{1,1} & y_{2,1} & y_{3,1}
\end{pmatrix}
\]

\[\implies H(x_1) = y_{1,1} \quad H(x_2) = y_{2,0} \quad H(x_3) = y_{3,1}\]
ALGORITHMS AND WHERE TO FIND THEM
PQC Standardisation (NIST)

- **2016**
  - Launched at PQCrypto 2016

- **2017**
  - Round 1 (69 accepted submissions)

- **Now**
  - Round 2 (26 algorithms)

- **2020/2021**
  - Round 3 begins

- **2022/2024**
  - Draft standards

<table>
<thead>
<tr>
<th>Type</th>
<th>PKE/KEM</th>
<th>Signature</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lattice</td>
<td>CRYSTALS-KYBER[^4]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>FrodoKEM[^5]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>LAC</td>
<td></td>
</tr>
<tr>
<td></td>
<td>NewHope[^9]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>NTRU (merger of NTRUEncrypt and NTRU-HRSS-KEM[^7])</td>
<td></td>
</tr>
<tr>
<td></td>
<td>NTRU Prime[^8]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Round5 (merger of Round2 and Hiia5, announced 4 August 2019[^9])</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SABER[^10]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Three Bears[^11]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>QTESLA[^13]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CRYSSTRUCT-DILITHIUM[^4]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>FALCON[^12]</td>
<td></td>
</tr>
<tr>
<td>Code-based</td>
<td>BIKE[^14]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Classic McEliece</td>
<td></td>
</tr>
<tr>
<td></td>
<td>HQC[^15]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>LEDAcrypt (merger of LEDAkm[^16] and LEDApkc[^17])</td>
<td></td>
</tr>
<tr>
<td></td>
<td>NTS-KEM[^18]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ROLLO (merger of Ouroboros-R, LAKE and LOCKER) [^19]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RQC[^20]</td>
<td></td>
</tr>
<tr>
<td>Hash-based</td>
<td>SPHINCS+[^21]</td>
<td></td>
</tr>
<tr>
<td>Multivariate</td>
<td>GeMSS[^22]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>LUOV</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MODSS[^23]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Rainbow</td>
<td></td>
</tr>
<tr>
<td>Supersingular Elliptic Curve Isogeny</td>
<td>SIKE[^24]</td>
<td></td>
</tr>
<tr>
<td>Zero-knowledge proofs</td>
<td>Picnic[^25]</td>
<td></td>
</tr>
</tbody>
</table>
Efficiency

- Larger key sizes
- Larger signatures
- More complex operations*

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Type</th>
<th>Public Key</th>
<th>Private Key</th>
<th>Signature</th>
</tr>
</thead>
<tbody>
<tr>
<td>NTRU Encrypt</td>
<td>Lattice</td>
<td>6130 B</td>
<td>6743 B</td>
<td></td>
</tr>
<tr>
<td>Streamlined NTRU Prime</td>
<td>Lattice</td>
<td>1232 B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rainbow</td>
<td>Multivariate</td>
<td>124 KB</td>
<td>95 KB</td>
<td>41 KB</td>
</tr>
<tr>
<td>SPHINCS</td>
<td>Hash Signature</td>
<td>1 KB</td>
<td>1 KB</td>
<td>8 KB</td>
</tr>
<tr>
<td>SPHINCS-</td>
<td>Hash Signature</td>
<td>32 B</td>
<td>64 B</td>
<td></td>
</tr>
<tr>
<td>BLISS-II</td>
<td>Lattice</td>
<td>7 KB</td>
<td>2 KB</td>
<td>5 KB</td>
</tr>
<tr>
<td>GLP-Variant GLYPH Signature</td>
<td>Ring-LWE</td>
<td>2 KB</td>
<td>0.4 KB</td>
<td>1.8 KB</td>
</tr>
<tr>
<td>New Hope</td>
<td>Ring-LWE</td>
<td>2 KB</td>
<td>2 KB</td>
<td></td>
</tr>
<tr>
<td>Goppa-based McEliece</td>
<td>Code-based</td>
<td>1 MB</td>
<td>11.5 KB</td>
<td></td>
</tr>
<tr>
<td>Random Linear Code based encryption</td>
<td>RLCE</td>
<td>115 KB</td>
<td>3 KB</td>
<td></td>
</tr>
<tr>
<td>Quasi-cyclic MDPC-based McEliece</td>
<td>Code-based</td>
<td>1232 B</td>
<td>2464 B</td>
<td></td>
</tr>
<tr>
<td>SIDH</td>
<td>Isogeny</td>
<td>751 B</td>
<td>48 B</td>
<td></td>
</tr>
<tr>
<td>SIDH (compressed keys)</td>
<td>Isogeny</td>
<td>564 B</td>
<td>48 B</td>
<td></td>
</tr>
<tr>
<td>3072-bit Discrete Log</td>
<td>not PQC</td>
<td>384 B</td>
<td>32 B</td>
<td>96 B</td>
</tr>
<tr>
<td>256-bit Elliptic Curve</td>
<td>not PQC</td>
<td>32 B</td>
<td>32 B</td>
<td>65 B</td>
</tr>
</tbody>
</table>
Where to get implementations

liboqs: open-source C implementation
  - Wrappers for C#, C++, Python, Go
  - Common API for key encapsulation and signatures
  - OpenSSL and OpenSSH forks
  - Uses of liboqs:
    - Microsoft OpenVPN fork, Utimaco HSM, Mullvad VPN client

https://openquantumsafe.org/
QUANTUM KEY DISTRIBUTION
Quantum Key Distribution

<table>
<thead>
<tr>
<th>QKD</th>
<th>PQC</th>
</tr>
</thead>
<tbody>
<tr>
<td>p2p fiber optic &amp; single-photon</td>
<td>Existing infrastructure</td>
</tr>
<tr>
<td>Unconditional security</td>
<td>Hard math problems</td>
</tr>
<tr>
<td>Authenticated classical channel</td>
<td>Public key distribution</td>
</tr>
</tbody>
</table>

QUANTUM INTERWEBS
RECAP

• Public key crypto based on hard math problems.

• Easy for quantum computers, but they don’t exist just yet.

• Still important to think of replacing RSA and DH.

• Lots of quantum-resistant alternatives (standards soon).

• QKD also exists.
THANK YOU

Philippe Lamontagne • Research Officer
Philippe.Lamontagne2@cnrc-nrc.gc.ca