Introduction to Post-Quantum Cryptography

**Quantum Computers**
- First perceived by physicists (Richard Feynman, David Deutsch) in 1980s
- First significant quantum algorithms (capable of running on quantum computers only) developed in 1990s
- First practical realization in 1998 (2 qubits)
- Significant technological advances during the last 20 years
- **One of ten breakthrough technologies of 2017**

**Timeline of Quantum Computing:**

**Major advances during the last 20 years**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Semiconductor quantum dots</td>
<td>Semiconductor-superconductor hybrids</td>
</tr>
<tr>
<td>Superconducting circuits</td>
<td>Impurities in diamond or silicon</td>
</tr>
</tbody>
</table>

**Photos:** https://www.technologyreview.com

**Source:** Vandersypen, PQCrypto 2017

**Progress in Quantum Computing**

**Remaining Challenges in Quantum Computing**

1. **High sensitivity to manufacturing variations**  
   **Solution:** Best industry cleanrooms, e.g., QuTech-Intel collaboration toward quantum-dot arrays made @ Intel 300mm wafers

2. **Scalable control circuits (currently bulky & expensive)**  
   **Solution:** Tailored cryo-CMOS digital control

3. **Multitude of interconnects and external pins**  
   **Solution:** Multiplexing electronics co-integrated with qubits

4. **Non-standard architecture & limited programmability**  
   **Solution:** System layer approach  
   **Likely to be overcome in the next 10-15 years**

**Photos:** https://www.technologyreview.com

**Source:** Vandersypen, PQCrypto 2017
System Layer Approach

Challenges in each layer Layers are highly interrelated

What Quantum Computers Can Do?

Model complex molecules
Model complex materials
Solve complex math problems

Health: Quantum chemistry for medicine
Energy: Room-temperature superconductivity
Security: factoring and code breaking

Nobel 2012 citation: “The quantum computer may change our everyday lives in this century in the same radical way as the classical computer did in the last century.”

Quantum Computers & Cryptography

1994: Shor’s Algorithm, breaks major public key cryptosystems based on

Factoring: RSA
Discrete logarithm problem (DLP): DSA, Diffie-Hellman
Elliptic Curve DLP: Elliptic Curve Cryptosystems
Independently of the key size assuming a sufficiently powerful and reliable quantum computer available

How Real is the Danger?

“There is a 1 in 7 chance that some fundamental public-key crypto will be broken by quantum by 2026, and a 1 in 2 chance of the same by 2031.”

Dr. Michele Mosca
Deputy Director of the Institute for Quantum Computing, University of Waterloo April 2015

Post-Quantum Cryptography (PQC)

- Public-key cryptographic algorithms for which there are no known attacks using quantum computers
- Capable of being implemented using any traditional methods, including software and hardware
- Running efficiently on any modern computing platforms: PCs, tablets, smartphones, servers with FPGA accelerators, etc.

- Term introduced by Dan Bernstein in 2003
- Equivalent terms: quantum-proof, quantum-safe or quantum-resistant
- Based entirely on traditional semiconductor VLSI technology

Underlying Mathematical Problem - RSA

\[ N = P \cdot Q \ (P, Q \text{ random primes}) \]

\[ 12301866845301177551304949583849627207728535695 \]
\[ 953347921973224521517264005072639575187452021997 \]
\[ 86649389956474942774663845925192573269303453731 \]
\[ 548268507091702612214291341667042291431160221240 \]
\[ 479247377949080665351419597459856902143413 \]
\[ 43087737814467999489 \]
\[ 33478071698956898786044169848212690877047949837 \]
\[ 1376856891243138996286379378780022876147116525317 \]
\[ 43087737814467999489 \]
\[ 33478071698956898786044169848212690877047949837 \]

Record Using Classical Computers, 232 digits, 768 bits
**Underlying Mathematical Problem – PQC**

**Closest Vector Problem**

Lattice in dimension \( n = 2 \):

Set of points given by

\[
\mathbf{p} = i \cdot \mathbf{b}_1 + j \cdot \mathbf{b}_2
\]

where \( i \) and \( j \) are arbitrary integers

Imagine \( n \) in the range of 825

Problem:

Find the point of lattice given by the base vectors \( \mathbf{b}_1 \) and \( \mathbf{b}_2 \) closest to the arbitrary point of an \( n \)-dimensional space \( \mathbf{t} \)

Imagine \( m \) and \( n \) in the range of 70 and above

---

**Solving a system of \( m \) quadratic equations with \( n \) unknowns**

\[
p^{(1)}(x_1, \ldots, x_n) = \sum_{i=1}^{n} \sum_{j=1}^{n} p_{ij}^{(1)} \cdot x_i \cdot x_j + \sum_{i=1}^{n} p_{i}^{(1)} \cdot x_i + p_{0}^{(1)}
\]

\[
p^{(2)}(x_1, \ldots, x_n) = \sum_{i=1}^{n} \sum_{j=1}^{n} p_{ij}^{(2)} \cdot x_i \cdot x_j + \sum_{i=1}^{n} p_{i}^{(2)} \cdot x_i + p_{0}^{(2)}
\]

\[
p^{(m)}(x_1, \ldots, x_n) = \sum_{i=1}^{n} \sum_{j=1}^{n} p_{ij}^{(m)} \cdot x_i \cdot x_j + \sum_{i=1}^{n} p_{i}^{(m)} \cdot x_i + p_{0}^{(m)}
\]

---

**Theorem** by Mosca

If \( z < y + x \), then worry!

\( y \) – Time to Develop & Deploy PQC Standards

\( x \) – Time Information Must Remain Protected

\( z \) – Time to Build Quantum Computers

Encrypted Data Stored by Powerful Adversaries

No Announcement when Quantum Computer Available to NSA, Foreign Governments, or Organized Crime

---

**NIST PQC Standardization Process**

- **Feb. 2016**: NIST announcement of standardization plans at PQCrypto 2016, Fukuoka, Japan,
- **Dec. 2016**: NIST Call for Proposals and Request for Nominations for Public-Key Post-Quantum Cryptographic Algorithms:
- **Nov. 30, 2017**: Deadline for submitting candidates
- **Dec. 2017**: Announcement of the First Round Candidates
- **Apr. 2018**: The First NIST PQC Standardization Conference

---

**Three Types of PQC Schemes**

1. **Public Key Encryption**

2. **Digital Signature**

3. **Key Encapsulation Mechanism (KEM)**

---

**Five Security Categories**

<table>
<thead>
<tr>
<th>Level</th>
<th>Security Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>At least as hard to break as AES-128 using exhaustive key search</td>
</tr>
<tr>
<td>II</td>
<td>At least as hard to break as SHA-256 using collision search</td>
</tr>
<tr>
<td>III</td>
<td>At least as hard to break as AES-192 using exhaustive key search</td>
</tr>
<tr>
<td>IV</td>
<td>At least as hard to break as SHA-384 using collision search</td>
</tr>
<tr>
<td>V</td>
<td>At least as hard to break as AES-256 using exhaustive key search</td>
</tr>
</tbody>
</table>
Leading PQC Families

<table>
<thead>
<tr>
<th>Family</th>
<th>Encryption/KEM</th>
<th>Signature</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hash-based</td>
<td>XX</td>
<td>X</td>
</tr>
<tr>
<td>Code-based</td>
<td>XX</td>
<td>X</td>
</tr>
<tr>
<td>Lattice-based</td>
<td>XX</td>
<td>X</td>
</tr>
<tr>
<td>Multivariate</td>
<td>X</td>
<td>XX</td>
</tr>
<tr>
<td>Isogeny-based</td>
<td>X</td>
<td></td>
</tr>
</tbody>
</table>

XX – high-confidence candidates, X – medium-confidence candidates

Round 1 Candidates

<table>
<thead>
<tr>
<th>Family</th>
<th>Signature</th>
<th>Encryption/KEM</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lattice-based</td>
<td>5</td>
<td>22</td>
<td>27</td>
</tr>
<tr>
<td>Code-based</td>
<td>2</td>
<td>16</td>
<td>18</td>
</tr>
<tr>
<td>Multivariate</td>
<td>7</td>
<td>2</td>
<td>9</td>
</tr>
<tr>
<td>Hash-based</td>
<td>2</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>Isogeny-based</td>
<td>1</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Other</td>
<td>3</td>
<td>4</td>
<td>7</td>
</tr>
<tr>
<td>Total</td>
<td>19</td>
<td>45</td>
<td>64</td>
</tr>
</tbody>
</table>

69 accepted as complete, 5 since withdrawn

Round 1 Submissions

69 Submissions, 26 Countries, 260 co-authors

GMU Team Implementation Developed in Fall 2017-Spring 2018. Preliminary Results Presented at the Code-Based Cryptography Workshop in April 2018.

Attack against the published parameter set announced on May 16

Risks of Early Hardware Implementations

An efficient structural attack on NIST submission DARGS

Why Cryptographic Contests?

Avoid back-door theories

- Speed-up the acceptance of a new standard

- Stimulate non-classified research on methods of designing a specific cryptographic transformation

- Focus the effort of a relatively small cryptographic community

Previous Cryptographic Contests

<table>
<thead>
<tr>
<th>IX.1997</th>
<th>X.2000</th>
</tr>
</thead>
<tbody>
<tr>
<td>AES</td>
<td>15 block ciphers → 1 winner</td>
</tr>
<tr>
<td>NESSIE</td>
<td></td>
</tr>
<tr>
<td>X.2002</td>
<td></td>
</tr>
<tr>
<td>CRYPTREC</td>
<td></td>
</tr>
<tr>
<td>X.2004</td>
<td></td>
</tr>
<tr>
<td>IV.2008</td>
<td></td>
</tr>
<tr>
<td>34 stream ciphers → 4 HW winners</td>
<td></td>
</tr>
<tr>
<td>51 hash functions → 1 winner</td>
<td></td>
</tr>
<tr>
<td>57 authenticated ciphers → multiple winners</td>
<td></td>
</tr>
</tbody>
</table>

2018 CAESAR
**Evaluation Criteria**

- Security
- Software Efficiency
  - µProcessors
  - µControllers
- Hardware Efficiency
  - FPGAs
  - ASICs
- Flexibility
- Simplicity
- Licensing

**AES Contest, 1997-2000: Block Ciphers**

GMU FPGA Results & Straw Poll @ NIST AES 3 conference

- Rijndael second best in FPGAs, selected as a winner due to much better performance in software & ASICs

**SHA-3 Round 2: 14 candidates**

Throughput vs. Area: Normalized to Results for SHA-2 and Averaged over 7 FPGA Families

- Best: Fast & Small
- Worst: Slow & Big

**SHA-3 Contest, 2007-2012: Hash Functions**

Keccak – an early leader in hardware and winner of the competition

**CAESAR, 2013-2018: Authenticated Ciphers**

Relative (w.r.t. AES-GCM) Throughput in Virtex 6

- 14-16x better than the current standard
- in red – algorithms qualified to Round 3

**Similarties with Previous Contests**

- Evaluation to be performed in rounds (12-18 months each)
- A pool of candidates narrowed down after each round
- Small tweaks allowed at the beginning of each round
- Optimized software implementation developed for various platforms
- No immediate plans for obligatory hardware implementations
Differences from Previous Contests

- Candidates not qualified to the next round (and not withdrawn by the authors) may be considered at a later date
- Taking into account quantum attacks, possible only on the platforms that do not exist at the time of the standard development
- Security analysis much more challenging and often controversial

Adi Shamir’s Proposal

After the initial evaluation period (e.g., 3 years) the division of all schemes into the following categories:

- **2 Productions Schemes**: recommended for actual wide-scale deployment, Highly Trusted
- **4 Development Schemes**: Time-Tested, Trusted; at least 15 years of analysis behind them; Intended for initial R&D by industry.
- **8 Research Schemes**: Promising Properties, Good Performance. May contain some high-risk candidates. Main Goal: Concentrate the effort of the research community.

PQCRYPTO Consortium

11 universities and companies
Funded by European Commission under the H2020 program

Initial Recommendations published in 2015
Co-authors of 22 Submissions

The Most Trusted Schemes – Encryption/KEM

**Classical McEliece**
- Proposed 40 years ago as an alternative to RSA
- Code-based family
- Based on binary Goppa codes
- No patents
- Conservative parameters (Category 5, 256-bit security):
  - a) length n=6960, dimension k= 5413, errors=119
  - b) length n=8192, dimension k= 6528, errors=128
- Complexity of the best attack identical after 40 years of analysis, and more than 30 papers devoted to thorough cryptanalysis
- Sizes:
  - Public key: a) 1,047,319 bytes, b) 1,357,824 bytes
  - Private key: a) 13,908 bytes, b) 14,080 bytes
  - Ciphertext: a) 226 bytes, b) 240 bytes
- Efficient Software (Haswell, larger parameter set)
  - 295,930 for encryption, 355,152 for decryption
- Constant time
- Efficient Hardware (Yale)

The Most Trusted Schemes – Signatures

**Hash-based Schemes**:
- Security based on the security of a single underlying primitive hash function

**Representatives**:
- SPHINCS-256 => SPHINCS+

**Features**:
- Efficient signature generation and verification
- Relatively large keys (~ tens of kilobytes)

Likely Development Schemes – Lattice-based

- Efficient encryption & decryption
- Relatively small key sizes (kilobytes)
- Suitable for constrained environments

- No proof of security
Likely Development Schemes – Lattice-based

- New lattice-based schemes with extended security proof, smaller key sizes, and better efficiency

  **Ring-LWE (Learning with Errors)**
  **Binary RLWE**
  **NewHope**
  **CRYSTALS-KYBER**
  **CRYSTALS-DILITHIUM**

Pilot Hardware Implementations:

- Ruhr University of Bochum, Germany
- Technical University Munich, Germany
- ESAT/COSIC KU Leuven, Belgium
- The University of Texas at Austin, USA
- George Mason University, USA, etc.

Major Implementation Challenges

- Mathematical complexity
- Large amount of man-power required
- Large keys and internal states
- Hardware resources required for full parallelization
- New types of basic operations
- Need for Random Sampling not only from uniform but also from Discrete Gaussian distributions
- Constant-time implementations
- Need for new SCA (Side-Channel Attack) countermeasures against power and electromagnetic analysis
- Plug-and-play replacement for classical public-key cryptography units
- Intermediate use of Hybrid Systems

PQC Major Challenges

<table>
<thead>
<tr>
<th>Fairness</th>
<th>Number of Candidates</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Percentage of Candidates In Hardware</th>
</tr>
</thead>
<tbody>
<tr>
<td>AES</td>
</tr>
<tr>
<td>eSTREAM</td>
</tr>
<tr>
<td>SHA-3</td>
</tr>
<tr>
<td>CAESAR</td>
</tr>
</tbody>
</table>

PQC Hardware API proposed by GMU

1. Minimum Compliance Criteria
   - Encryption & decryption
   - Signature generation & verification
   - Maximum message size
   - Padding
   - Permitted data port widths, etc.

2. Interface

3. Communication Protocol

4. Timing Characteristics

PQC Development Package

- Universal Testbench (HDL)
- Pre- and PostProcessors Common for All Candidates (HDL)
- Test Vector Generator (Python)
- Library of Most Common Operations (HDL)
Open-Source Repositories & Databases of Results

High-Level Synthesis (HLS)

- High Level Language (C, C++, Java, Python, etc.)
- High-Level Synthesis
- Hardware Description Language (VHDL or Verilog)

Popular HLS Tools

Commercial (FPGA-oriented):
- Vivado HLS: Xilinx

Academic:
- Bambu: Politecnico di Milano, Italy
- DWARV: Delft University of Technology, The Netherlands
- GAUT: Universite de Bretagne-Sud, France
- LegUp: University of Toronto, Canada

Case for HLS in Crypto Competitions

- All submissions include reference implementations in C
- Development time potentially decreased several times
- All candidates can be implemented by the same group, and even the same designer, reducing the bias
- Results from High-Level Synthesis could have a large impact in early stages of the competitions and help narrow down the search (saving thousands of man-hours of cryptanalysis)
- Potential for quickly detecting suboptimal code written manually

GMU Case Studies

- 5 Final SHA_3 Candidates + SHA-2
  Applied Reconfigurable Computing, ARC 2015, Bochum, Apr. 2015

- 16 Round 3 CAESAR Candidates + AES-GCM
  Field Programmable Technology Conference, Melbourne, Dec. 2017

HLS & Crypto: State-of-the-Art before GMU

<table>
<thead>
<tr>
<th>Cryptographic Benchmarks</th>
<th>aes-encrypt</th>
<th>aes-decrypt</th>
<th>sha</th>
<th>blowfish</th>
</tr>
</thead>
<tbody>
<tr>
<td>Best HLS</td>
<td>1,191</td>
<td>2,579</td>
<td>51,399</td>
<td>57,590</td>
</tr>
<tr>
<td>Manual</td>
<td>20</td>
<td>20</td>
<td>20,480</td>
<td>18,736</td>
</tr>
<tr>
<td>Best HLS /Manual</td>
<td>60</td>
<td>129</td>
<td>2.5</td>
<td>3.1</td>
</tr>
</tbody>
</table>

Best HLS: Best result (minimum number of clock cycles) from among those generated by Vivado HLS, Bambu, DWARV, and LegUp

**Transformation to HLS-ready C/C++ Code**

1. Addition of HLS Tool directives (pragmas)
2. Hardware-driven code refactoring
3. Mapping software to hardware API

---

**Adding HLS Tool Directives - Pragmas**

- **Unrolling of loops:**
  ```c
  for (i = 0; i < 4; i++)
  #pragma HLS UNROLL
  for (j = 0; j < 4; j++)
  #pragma HLS UNROLL
  b[i][j] = s[i][j];
  ```

- **Flattening function's hierarchy:**
  ```c
  void KeyUpdate (word8 k[4][4], word8 round)
  {
    #pragma HLS INLINE
    ...
  }
  ```

- **Change array shapes:**
  ```c
  void AES_encrypt (word8 a[4][4], word8 k[4][4], word8 b[4][4])
  {
    #pragma HLS ARRAY_RESHAPE variable=a[0] complete dim=1 reshape
    #pragma HLS ARRAY_RESHAPE variable=a[1] complete dim=1 reshape
    #pragma HLS ARRAY_RESHAPE variable=a[2] complete dim=1 reshape
    #pragma HLS ARRAY_RESHAPE variable=a[3] complete dim=1 reshape
    #pragma HLS ARRAY_RESHAPE variable=a complete dim = 1 reshape
  }
  ```
Code Refactoring – High-Level

Reference C

Encryption

Decryption

HLS-ready C/C++

Encryption/Decryption

Use of pragmas possible but unreliable

Code Refactoring: Low-Level

Single vs. Multiple Function Calls:

```c
// (a) Before modification
for (round++; round<N_ROUNDS; ++round) {
    if (round == N_ROUNDS-1) {
        single_round(state, i);
    } else {
        single_round(state, 0);
    }
}

// (b) After modification
for (round++; round<N_ROUNDS; ++round) {
    if (round == N_ROUNDS-1) {
        x = 1;
        if (x == 0) {
            single_round(state, x);
        }
    } else {
        single_round(state, 0);
    }
}
```

HLS Benchmarking: Conclusions

Accuracy:
- Good correlation between algorithm rankings using manual and HLS approaches

Efficiency:
- 3-10 shorter development time
- Designer can focus on functionality: control logic inferred
- Much easier verification:
  - C++ testbenches
  - A single designer can produce implementations of multiple (and even all) candidates

However:
- Manual design approach still predominant
- HLS design approach at the experimental stage – more research needed

Software/Hardware Implementations

Software

Hardware

Most time-critical operation

Possible Environments:
- Xilinx SDSoC
- Intel SoC Embedded Development Suite

Major Optimization Targets

High-Speed

- Parallel processing
- Constant-time
- Parametric code

Lightweight

- Small area, power, energy per bit
- Resistance to power & electromagnetic analysis

Timeline of the NIST Standardization Effort

- By Nov. 30, 2018: Allowing/encouraging similar submissions to merge
- Early 2018: Beginning of Round 2
  - Candidates withdrawn or judged unsuitable by NIST
  - Candidates qualified to Round 2
  - Candidates left for future consideration
- Aug. 2019: Second NIST PQC Conference
- Early 2020: Beginning of Round 3
  - Candidates selected for standardization
  - Candidates withdrawn or judged unsuitable by NIST
  - Candidates qualified to Round 3
  - Candidates left for future consideration
- 2021-2024: Possible future rounds
- Possible parallel efforts by IETF, IEEE, ANSI, ETSI, ISO/IEC
PQC Opportunities & Challenges

- The biggest revolution in cryptography, since the invention of public-key cryptography in 1970s
- Efficient hardware implementations in FPGAs and ASICs desperately needed to prove the candidates suitability for high-performance applications and constrained environments. **Collaboration sought by submission teams!**
- Likely extensions to Instruction Set Architectures of multiple major microprocessors
- Start-up & new-product opportunities
- Once in the lifetime opportunity!

Conclusions

- Contests for cryptographic standards are important
  - Stimulate progress in design and analysis of cryptographic algorithms
  - Determine future of cryptography for the next decades
  - Promote cryptology: Are easy to understand by general audience
  - Provide immediate recognition and visibility worldwide
- Computer Scientists, Digital System Designers, System Developers can play an important role in these contests
  - Co-designers of new cryptographic algorithms
  - Evaluators
  - Tool developers
  - Early adopters of new standards
- Get involved! It is fun!

Q&A

Questions? Comments?

CERG: http://cryptography.gmu.edu
ATHENA: http://cryptography.gmu.edu/athena