Results of PQCRYPTO (ICT-645622)
Post-quantum cryptography for long-term security

Tanja Lange

19 September 2018
FutureTPM Workshop
Post-Quantum Cryptography for Long-term Security

- Project funded by EU in Horizon 2020.
- Starting date 1 March 2015, project duration 3 years.
- 11 partners from academia and industry, TU/e is coordinator

 TU/e  
BUNDLES DRUCKEREI  
INRIA  
KU LEUVEN  
NXP  
Radboud Universiteit  
University of Haifa

Tanja Lange  
https://pqcrypto.eu.org  
Results of PQCRIPTO (ICT-645622)
Work packages

Technical work packages

▸ WP1: Post-quantum cryptography for small devices
   Leader: Tim Güneysu, co-leader: Peter Schwabe

▸ WP2: Post-quantum cryptography for the Internet
   Leader: Daniel J. Bernstein, co-leader: Wouter Castryck

▸ WP3: Post-quantum cryptography for the cloud
   Leader: Nicolas Sendrier, co-leader: Christian Rechberger

Non-technical work packages

▸ WP4: Management and dissemination
   Leader: Tanja Lange

▸ WP5: Standardization
   Leader: Walter Fumy and Frank Morgner
General PQCRYPTO achievements

- Lots of papers: more than 130 publications, including one Nature paper.
- Widely used “Initial Recommendations” document giving PQCRYPTO recommendations of conservative systems.
- Many presentations (slides are online, some talks have videos).
  - 9 presentations of PQCRYPTO.
  - 32 general presentations of post-quantum cryptography.
  - 31 presentations at (summer) schools.
  - 77 focused presentations of scientific results.
- 17 deliverables from the scientific WPs.
- 22 fully worked-out submissions to NIST.
- Event highlights:
  - PQCRYPTO summer school (126 students);
  - PQCrypto conference 220 participants.
- Active twitter feed https://twitter.com/pqc_eu.
Initial recommendations

- **Symmetric encryption** Thoroughly analyzed, 256-bit keys:
  - AES-256
  - Salsa20 with a 256-bit key

  Evaluating: Serpent-256, ...

- **Symmetric authentication** Information-theoretic MACs:
  - GCM using a 96-bit nonce and a 128-bit authenticator
  - Poly1305

- **Public-key encryption** McEliece with binary Goppa codes:
  - length $n = 6960$, dimension $k = 5413$, $t = 119$ errors

  Evaluating: QC-MDPC, Stehlé-Steinfeld NTRU, ...

- **Public-key signatures** Hash-based (minimal assumptions):
  - XMSS with any of the parameters specified in CFRG draft
  - SPHINCS-256

  Evaluating: HFEv-, ...
WP1: Post-quantum cryptography for small devices

- Find post-quantum secure cryptosystems suitable for small devices in power and memory requirements (e.g. smart cards with 8-bit or 16-bit or 32-bit architectures, with different amounts of RAM, with or without coprocessors).
- Develop efficient implementations of these systems.
- Investigate and improve their security against implementation attacks.
- Deliverables include reference implementations and optimized implementations for software for platforms ranging from small 8-bit microcontrollers to more powerful 32-bit ARM processors.
- Deliverables also include FPGA and ASIC designs and physical security analysis.
Primary Target Platforms

Software

- ARM Cortex M4
  - 32-bit microcontroller
  - up to 168 MHz
  - up to 1 MB Flash, 192 KB RAM
  - Cheap development boards ($\approx$ 12 Euros) and open toolchain
Primary Target Platforms

Software

▶ ARM Cortex M4
  ▶ 32-bit microcontroller
  ▶ up to 168 MHz
  ▶ up to 1 MB Flash, 192 KB RAM
  ▶ Cheap development boards (≈ 12 Euros) and open toolchain

Hardware

▶ Xilinx Spartan-6 FPGA
  ▶ 45nm process technology
  ▶ “lowest total cost for high-volume applications”
Integration: pqm4

- Library and testing/benchmarking framework
- Easy to add schemes using NIST API
- Optimized SHA3 shared across primitives
- Focus on PQCRYPTO submissions to NIST, medium security

Run functional tests of all primitives and implementations:
```
python3 test.py
```

Generate testvectors, compare for consistency (also with host):
```
python3 testvectors.py
```

Run speed and stack benchmarks:
```
python3 benchmarks.py
```

Easy to evaluate only subset of schemes, e.g.:
```
python3 test.py newhope1024cca sphincs-shake256-128s
```
Integration: pqm4

- Library and testing/benchmarking framework
- Easy to add schemes using NIST API
- Optimized SHA3 shared across primitives
- Focus on PQCRYPTO submissions to NIST, medium security
- Run functional tests of all primitives and implementations:
  
  ```python
  python3 test.py
  ```

- Generate testvectors, compare for consistency (also with host):
  
  ```python
  python3 testvectors.py
  ```

- Run speed and stack benchmarks:
  
  ```python
  python3 benchmarks.py
  ```

- Easy to evaluate only subset of schemes, e.g.:
  
  ```python
  python3 test.py newhope1024cca sphincs-shake256-128s
  ```
<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIG QUAKE</td>
<td>?</td>
</tr>
<tr>
<td>BIKE</td>
<td>?</td>
</tr>
<tr>
<td>Classic McEliece</td>
<td>X</td>
</tr>
<tr>
<td>CRYSRALS-Kyber</td>
<td>✓</td>
</tr>
<tr>
<td>DAGS</td>
<td>?</td>
</tr>
<tr>
<td>FrodoKEM</td>
<td>✓</td>
</tr>
<tr>
<td>KINDI</td>
<td>✓</td>
</tr>
<tr>
<td>NewHope</td>
<td>✓</td>
</tr>
<tr>
<td>NTRU-HRSS-KEM</td>
<td>✓</td>
</tr>
<tr>
<td>NTRU Prime</td>
<td>✓</td>
</tr>
<tr>
<td>Post-quantum RSA-Encryption</td>
<td>X</td>
</tr>
<tr>
<td>Ramstake</td>
<td>X(?)</td>
</tr>
<tr>
<td>SABER</td>
<td>✓</td>
</tr>
<tr>
<td>(SIKE)</td>
<td>✓</td>
</tr>
<tr>
<td>Signature Scheme</td>
<td>Result</td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>CRYS'TALS-Dilithium</td>
<td>✓</td>
</tr>
<tr>
<td>GUI</td>
<td>✗</td>
</tr>
<tr>
<td>LUOV</td>
<td>?</td>
</tr>
<tr>
<td>MQDSS</td>
<td>✗(?)</td>
</tr>
<tr>
<td>Picnic</td>
<td>✗</td>
</tr>
<tr>
<td>Post-quantum RSA-Signature</td>
<td>✗</td>
</tr>
<tr>
<td>qTESLA</td>
<td></td>
</tr>
<tr>
<td>Rainbow</td>
<td></td>
</tr>
<tr>
<td>SPHINCS+</td>
<td>✓</td>
</tr>
</tbody>
</table>
WP2: Post-quantum cryptography for the Internet

▶ Find post-quantum secure cryptosystems suitable for busy Internet servers handling many clients simultaneously.
▶ Develop secure and efficient implementations.
▶ Integrate these systems into Internet protocols.
▶ Deliverables include software library for all common Internet platforms, including large server CPUs, smaller desktop and laptop CPUs, netbook CPUs (Atom, Bobcat, etc.), and smartphone CPUs (ARM).
▶ Aim is to get high-security post-quantum crypto ready for the Internet.
WP2 software library: libpqcrypto


77 cryptographic systems from 19 of the 22 PQCrypto submissions:

- BIG QUAKE
- Classic McEliece
- CRYSTALS-DILITHIUM
- CRYSTALS-KYBER
- DAGS
- FrodoKEM
- Gui
- KINDI
- LUOV
- MQDSS
- NewHope
- NTRU-HRSS-KEM
- NTRU Prime
- Picnic
- qTESLA
- Rainbow
- Ramstake
- SABER
- SPHINCS+
50 signature systems

CRYS\-\TALS-\-DILITHIUM: crypto\_sign\_dilithium\{2,3,4\}

Gui: crypto\_sign\_gui\{184,312,448\}

LUOV:
crypto\_sign\_luov\{863256,890351,8117404,4849242,6468330,8086399\}

MQDSS: crypto\_sign\_mqdss\{48,64\}

Picnic: crypto\_sign\_picnicl\{1,3,5\}\{fs,ur\}

qTESLA: crypto\_sign\_qtesla\{128,192,256\}

Rainbow: crypto\_sign\_rainbow\{1a,1b,1c,3b,3c,4a,5c,6a,6b\}

SPHINCS+:
crypto\_sign\_sphincs\{f,s\}\{128,192,256\}\{haraka,sha256,shake256\}
27 encryption systems

BIG QUAKE: crypto_kem_bigquake{1,3,5}
Classic McEliece: crypto_kem_mceliece{6960119,8192128}
CRYSTALS-KYBER: crypto_kem_kyber{512,768,1024}
DAGS: crypto_kem_dags{3,5}
FrodoKEM: crypto_kem_frodokem{640,976}
KINDI: crypto_kem_kindi{256342,256522,512222,512241,512321}
NewHope: crypto_kem_newhope{512,1024}cca
NTRU-HRSS-KEM: crypto_kem_ntruhrss701
NTRU Prime: crypto_kem_{ntrulpr,sntrup}4591761
Ramstake: crypto_kem_ramstakers{216091,756839}
SABER: crypto_kem_{lightsaber,saber,firesaber}
NIST submissions vs. libpqcrypto

Each NIST submission includes software:

- a reference C implementation;
- often additional implementations providing better performance.

libpqcrypto collects this software into an integrated library, with

- a unified compilation framework;
NIST submissions vs. libpqcrypto

Each NIST submission includes software:
▶ a reference C implementation;
▶ often additional implementations providing better performance.

libpqcrypto collects this software into an integrated library, with
▶ a unified compilation framework;
▶ an automatic test framework;
NIST submissions vs. libpqcrypto

Each NIST submission includes software:

- a reference C implementation;
- often additional implementations providing better performance.

Libpqcrypto collects this software into an integrated library, with

- a unified compilation framework;
- an automatic test framework;
- automatic selection of the fastest implementation of each system;
NIST submissions vs. libpqcrypto

Each NIST submission includes software:
- a reference C implementation;
- often additional implementations providing better performance.

libpqcrypto collects this software into an integrated library, with
- a unified compilation framework;
- an automatic test framework;
- automatic selection of the fastest implementation of each system;
- a unified C interface following the NaCl/TweetNaCl/SUPERCOP/libuuid API;
NIST submissions vs. libpqcrypto

Each NIST submission includes software:

- a reference C implementation;
- often additional implementations providing better performance.

libpqcrypto collects this software into an integrated library, with

- a unified compilation framework;
- an automatic test framework;
- automatic selection of the fastest implementation of each system;
- a unified C interface following the NaCl/TweetNaCl/SUPERCOP/libsodium API;
- a unified Python interface;
NIST submissions vs. libpqcrypto

Each NIST submission includes software:

- a reference C implementation;
- often additional implementations providing better performance.

libpqcrypto collects this software into an integrated library, with

- a unified compilation framework;
- an automatic test framework;
- automatic selection of the fastest implementation of each system;
- a unified C interface following the NaCl/TweetNaCl/SUPERCOP/libsodium API;
- a unified Python interface;
- command-line signature/verification/encryption/decryption tools;
NIST submissions vs. libpqcrypto

Each NIST submission includes software:
- a reference C implementation;
- often additional implementations providing better performance.

libpqcrypto collects this software into an integrated library, with
- a unified compilation framework;
- an automatic test framework;
- automatic selection of the fastest implementation of each system;
- a unified C interface following the NaCl/TweetNaCl/SUPERCOP/libsodium API;
- a unified Python interface;
- command-line signature/verification/encryption/decryption tools;
- command-line benchmarking tools.
Signature size (vertical) vs. public-key size (horizontal)
WP3: Post-quantum cryptography for the cloud

▶ Provide 50 years of protection for files that users store in the cloud, even if the cloud service providers are not trustworthy.
▶ Allow sharing and editing of cloud data under user-specified security policies.
▶ Support advanced cloud applications such as privacy-preserving keyword search.
▶ Work includes public-key and symmetric-key cryptography.
▶ Prioritize high security and speed over key size.
WP3 – Post-quantum cryptography for the cloud

- Encrypt at home
  - Main target: authenticated encryption.
  - Quantum cryptanalysis of symmetric schemes.
  - Grover/quantum random walk gives subtle advantages in linear and differential cryptanalysis.
  - Some modes of operation for authenticated encryption can get broken in some quantum models.

- Share files
  - Main targets: digital signature, public-key encryption.
  - Quantum cryptanalysis of public-key schemes.
  - Provide foundations and support for the design of post-quantum cryptosystems.

- Advanced applications
  - Keywords: searchable encryption, multi-party computation, fully homomorphic encryption.
  - Related topics: secret sharing, oblivious transfer, private information retrieval, locally decodable codes.
WP5: Standardization

Objectives:

Make the project visible [...] by providing dissemination material by which the project’s results will be submitted for consideration by international standardization bodies (SDOs) such as ISO, ETSI or IEEE.

Organize the project’s standardization activities which are relevant both for communicating the project’s outcomes and getting valuable feedback from standardization expert groups.
Major Players for Standardization of Cryptography

- **ISO/IEC JTC 1/SC 27**: Information technology – Security techniques
  Standardization of generic IT security services and techniques
- **ISO/TC 68 SC 2**: Financial Services, security
- **NIST**: National Institute of Standards and Technology
  Issues standards and guidelines as Federal Information Processing Standards (FIPS) for use by the US government
- **IETF and IRTF Crypto Forum Research Group (CFRG)**: Issues RFC’s for cryptography used in Internet protocols
- **OASIS KMIP**: Key Management Interoperability Protocol
- **ETSI SAGE**: Security Experts Group
- **ETSI QSC**: Quantum-Safe Cryptography Industry Specification Group
- **ANSI X9F**: Data and Information Security
  Standards for the financial services industry
- **IEEE P1363**: Standard Specifications for Public-Key Cryptography
OASIS Key Management Interoperability Protocol (KMIP)

KMIP Vendors

Standards by industry for industry.
Profile and system specification for post-quantum level

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Encryption</strong></td>
<td>SHOULD ChaCha20 (with 256-bit key)</td>
</tr>
<tr>
<td></td>
<td>MAY AES-256</td>
</tr>
<tr>
<td><strong>Digital Signature</strong></td>
<td>SHOULD SPHINCS-256 (stateless)</td>
</tr>
<tr>
<td></td>
<td>SHOULD XMSS (statefull)</td>
</tr>
<tr>
<td><strong>Key Exchange</strong></td>
<td>SHALL McEliece (with binary Goppa codes using length n = 6960,</td>
</tr>
<tr>
<td></td>
<td>dimension k = 5413 and adding t = 119 errors).</td>
</tr>
<tr>
<td><strong>Encryption with Authentication</strong></td>
<td>SHOULD ChaCha20Poly1305 (with 256-bit key)</td>
</tr>
<tr>
<td><strong>Hashes</strong></td>
<td>SHOULD SHA3-384 or SHA3-512</td>
</tr>
<tr>
<td></td>
<td>MAY SHA-384 or SHA-512</td>
</tr>
</tbody>
</table>

KMIP uses the PQCRYPTO initial recommendations.
Collaboration with CRYPTSOFT

(Card is preview of demo at RSA Conference 2018).
Selected SDO Activities: NIST

- Workshop on cybersecurity in a post-quantum world, April 2015
  - PQCRIPTO talk for introducing the EU project
  - PQCRIPTO presented several other papers
- Call for “quantum-resistant cryptographic algorithms for new public-key crypto standards”
  - PQCRIPTO was continuously contributing to the discussion of the final evaluation criteria, submitting tens of formal comments
- Submissions to call for “quantum-resistant cryptographic algorithms for new public-key crypto standards”
  - 22 of 69 “complete & proper” submissions by PQCRIPTO
- Evaluation phase
  - PQCRIPTO presented attacks on 15 submissions since then, 10 of which are fatal
  - Only one PQCRIPTO submission successfully attacked (not fatal)
Selected SDO Activities: IETF / IRTF/CFRG

Post-quantum activities

- Several Internet Drafts on post-quantum cryptography
- Ongoing discussion about post-quantum cryptography
- For now: Limitation to stateful hash-based signatures in CFRG (with later adoption by NIST), for everything else wait for NIST

RFC by PQCRYPTO

  draft-irtf-cfrg-xmss-hash-based-signatures-12
Stay tuned for more

- July 2019: Executive summer school in Eindhoven.
  - Many pointers: e.g., PQCrypto conference series.
  - Bibliography for 4 major PQC systems.
- **PQCrypto 2016** with slides and videos from lectures (incl. winter school)
- **PQCrypto 2017**
- [https://pqcrypto.eu.org](https://pqcrypto.eu.org): PQCRYPTO EU project.
  - Expert recommendations.
  - Free software libraries (libpqcrypto, pqm4, pqhw).
  - Lots of reports, scientific papers, (overview) presentations.