Integrating post-quantum crypto into real-life applications

Christian Paquin
@chpaquin
Principal Program Manager
MSR Security & Crypto

Post-quantum cryptography
Microsoft Research
Outline

1. Quantum

2. Post Quantum

3. Integrations
Quantum
“The quantum revolution is coming”

• I’ve been hearing this for 20 years...
  • Studying quantum crypto at UdM with the co-inventor of QKD

• But now, it’s getting serious

• My colleagues are building the full stack: from the chip to an SDK!
  [https://www.microsoft.com/quantum/](https://www.microsoft.com/quantum/)
Quantum computers

• Computers operating using the laws of quantum physics

• A quantum bit, or qubit, can be in superposition of the classical states 0 and 1; i.e. it can be both values simultaneously providing intrinsic parallelism

\[ |\psi\rangle = \alpha |0\rangle + \beta |1\rangle \]

• Measurement of a qubit yields a probabilistic classical value depending on the complex amplitudes \( \alpha \) and \( \beta \)
  • Quantum algorithms must reinforce the desired computational states

• Qubits can be entangled, i.e. be in a shared state across space
  • \[ |\psi\rangle = \frac{1}{\sqrt{2}} |00\rangle + \frac{1}{\sqrt{2}} |11\rangle \] means either both 0 or 1 with equal probability

• Can be built with various physical particles
  • Electron, photon, anyon (topological)

• “Nobody understands quantum mechanics” – Richard Feynman

DC26 CPV – Christian Paquin (MSR)

Charlie Marcus’ lab in Copenhagen
The Quantum Menace

• Quantum computing brings great promises in many fields, but has dire consequences for cryptography

• Shor (1994)
  • Solves the factoring (RSA) and discreet log (DSA, DH, and EC variants) problems in polynomial time
    • Reduce to period finding
  • Affects most of the asymmetric cryptography in use today

• Grover (1996)
  • Speeds up “database search” and “function inversion” in $O(\sqrt{n})$
  • Improves brute force of symmetric cryptography such as hash functions (SHA) and block ciphers (AES)
  • Need to double the size of key/digest: AES128 $\rightarrow$ AES256
Tic toc...

- Michele Mosca (Waterloo):
  “1/7 chance of breaking RSA-2048 by 2026, 1/2 chance by 2031” (2015)
  “1/6 chance within 10 years” (2017)
- Simon Benjamin (Oxford):
  “maybe 6-12 years if someone is willing to go Manhattan project”
- My colleagues estimate 2030

We need quantum-safe alternatives soon: post-quantum cryptography!
Post-quantum
Many reasons to start thinking post-quantum today

- Long-lived secrets/signatures are in danger
  - Capture now, decrypt later
- Need to understand impact on
  - Standards (TLS, SSH, IKE, PKI, S/MIME, …)
  - Products and services
    - Longer key/message/sig sizes
    - Slower running times
    - Code agility
- Early deployment of hybrid scenarios
  - Today’s assurance + safety net against QC
The National Institute of Standards and Technologies (NIST) started the process to specify Post-Quantum Cryptography.

- Looking for signatures, encryption, and key establishments schemes
  - Five levels, corresponding to breaking AES-128/192/256 and SHA-256/384
- 64 submissions remaining (from 69 valid submissions)
  - 19 signature schemes, 45 KEM/encryption schemes

Families of PQC

- **Lattice-based systems (26)**
  - Encryption/signature based on lattices (NTRU in ’96)
  - Learning With Error (LWE, 2005), or its less secure but more efficient Ring version (R-LWE: Peikert $\rightarrow$ BCNS $\rightarrow$ NewHope)

- **Code-based (19)**
  - Encryption/signature based on error-correcting codes (McEliece, Niederreiter)
  - As old as public-key crypto

- **Multivariate-based systems (9)**
  - Encryption/signature based on multivariate polynomials over a finite field
  - Developed in 90’s

- **Hash-based systems (3)**
  - Signatures based on hash functions (Lamport, Merkle)
  - As old as public-key crypto
  - Early standardization candidates: LMS, XMSS

- **Others (7)**
  - SIDH/SIKE: based on isogenies on elliptic curves
  - Picnic: based on symmetric ciphers and ZK proofs
Encryption perf

Data from NIST
Signature perf

Data from NIST
MSR’s collaborations

• FRODO (KEM)
  • Learning With Error (LWE) problem
  • https://frodokem.org/

• SIKE (KEM)
  • Supersingular Isogeny elliptic curves
  • https://sike.org/

• Picnic (sig)
  • Zero-knowledge proofs, hash, and block ciphers
  • https://microsoft.github.io/Picnic/

• qTesla (sig)
  • Ring Learning with Error problem
  • https://qtesla.org
Integrations
• Created to simplify integration of PQC into applications
• Multi-org dev team
• Master branch (for integration) and NIST branch (for experimentation)
• Shipped integrations with OpenSSL, OpenSSH
  • More in the pipeline
• https://openquantumsafe.org/
TLS 1.2 integration

- Added OQS key exchange (KEX) and authentication algs to OpenSSL 1.0.2
  - libcrypto: modified signature and X.509
  - libssl: modified TLS handler
- Defined new cipher suites
  - PQ or hybrid Key Exchange (KEX), e.g.
    - OQSKEX-SIDH-PICNIC-AES256-GCM-SHA384, OQSKEX-SIDH-ECDHE-PICNIC-AES256-GCM-SHA384
  - Pre-master secret := ECDH secret || PQ secret
  - Classical or PQ auth (Picnic)
    - Challenge: sig size limit of $2^{32} - 1$ bytes
- Tested with Apache 2.4.25
- [https://github.com/open-quantum-safe/openssl](https://github.com/open-quantum-safe/openssl)
  - Branch: OpenSSL_1_0_2-stable
TLS 1.2 KEX performance

- Measurements from OQS-enabled Apache server and test client
- Using pre-NIST submissions build
TLS 1.2 Auth (Picnic) performance

- Fetch time for various pages on slow/fast network
TLS 1.3 integration

• Added OQS key exchange (KEX) and auth to OpenSSL 1.1.1 (beta4)
• Defined new “curves” for TLS 1.3
  • PQ or hybrid Key Exchange (KEX)
• Tested with nginx 1.5.0
• We need extensions to enable PQC in TLS 1.3
• Details on OQS’s page
  • https://github.com/open-quantum-safe/openssl/wiki/PQC-integration-into-TLS-1.3
  • Branch: OQS-master
Hybrid scenarios

• TLS 1.3 KEX, two approaches
  • Naïve: define combo schemes and concatenate the data (currently implemented)
  • Multiple key shares (classical and PQC) both updating the master secret
    • State machine already supports hybrid keys, for PSK + ECDHE
    • PQC proposals: draft-whyte-qsh-tls13-06, draft-schanck-tls-additional-keyshare-00

• PKI, need to convey a classical and PQC signature
  • Hybrid signature scheme
  • Convey two certs
  • TLS PQC cert extension
  • X.509 extension for an extra PQC key
  • Bindel, Herath, McKague, Stebila; Transitioning to QR PKI
TLS 1.3 Perf

Measurements with client/server on localhost (no network delay)

#fetch/sec for L1 KEX (w/ ECDSA P256)

- P256-Newhope
- Newhope
- P256-NTRU
- NTRU
- P256-Sike503
- Sike503
- P256-Frodo
- Frodo
- ECDHE P256

#fetch/sec for L1 sigs (w/ Frodo KEX)

- qtestf1
- PicnicL1FS
- P256-ECDSA+SHA256
- RSA3072-PSS+SHA256

July 15th built of OQS/OpenSSL
Azure Standard D4s v3 VM, Ubuntu OS

DC26 CPV – Christian Paquin (MSR)
SSH integration

- Integrated OQS in OpenSSH 7.7
  - KEX algs from master branch
- Supports PQC and hybrid modes
  - Shared secret = concatenation of classical & PQC shared secrets

https://github.com/open-quantum-safe.openssh-portable

DC26 CPV – Christian Paquin (MSR)
OpenVPN

- Integration in OpenVPN 2.4.4
  - Uses OQS-OpenSSL to protect TLS key establishment
  - Uses RSA or Picnic auth
- Easy way to achieve PQC tunnel to the cloud even if applications haven’t been updated
  - Good for backward compatibility
- Tested with Raspberry Pi and Windows clients, and Azure Linux VM service
- [https://github.com/Microsoft/PQCrypto-VPN](https://github.com/Microsoft/PQCrypto-VPN)
HSM integration

• Integrated Picnic into an Utimaco HSM (Security Server Se50 LAN v4)
• Experiment consisted of
  1. Picnic key generation and signing in HSM (using reference implementation)
  2. Generated self-signed root Picnic cert
  3. Issued end-user RSA certs using the Picnic cert
• https://microsoft.github.io/Picnic/

DC26 CPV – Christian Paquin (MSR)
The road ahead

- Start planning transition to PQC

- Make sure your apps/services are crypto agile

- Consider deploying hybrid solutions for long-lived, high-value data

- Consider wrapping long-tail apps/services in a PQC-VPN tunnel
Questions?

cpaquin@microsoft.com