

Update on the **OPEN QUANTUM SAFE** project

Christian Paquin

 @chpaquin

Principal Program Manager

 Microsoft Research

#QuantumSafeCryptography

ETSI/IQC Quantum Safe Cryptography- Technical Track
Nov 6-7, 2019

OPEN QUANTUM SAFE

- Goal: support the development and prototyping of quantum-resistant cryptography
- Contributions from



SRI International

Radboud University



- C library integrating multiple PQC algorithms under a common API
 - New v0.2 released in October 2019
 - New and updated (round 2) schemes
- Integrations into boringssl, OpenSSL, OpenSSH, OpenVPN (MSR)
 - **New:** PQ/hybrid KEX/auth in TLS 1.3 and SSH
- C++, C#, Go, and Python wrappers
- Instructions to build nginx, apache, chromium

<https://openquantumsafe.org/>

Prototyping PQC paper

Prototyping post-quantum and hybrid key exchange and authentication in TLS and SSH

Eric Crockett¹, Christian Paquin², and Douglas Stebila³

¹AWS ericcro@amazon.com

²Microsoft Research cpaquin@microsoft.com

³University of Waterloo dstebila@uwaterloo.ca

July 19, 2019

Abstract

Once algorithms for quantum-resistant key exchange and digital signature schemes are selected by standards bodies, adoption of post-quantum cryptography will depend on progress in integrating those algorithms into standards for communication protocols and other parts of the IT infrastructure. In this paper, we explore how two major Internet security protocols, the Transport Layer Security (TLS) and Secure Shell (SSH) protocols, can be adapted to use post-quantum cryptography.

First, we examine various design considerations for integrating post-quantum and hybrid key exchange and authentication into communications protocols generally, and in TLS and SSH specifically. These include issues such as how to negotiate the use of multiple algorithms for hybrid cryptography, how to combine multiple keys, and more. Subsequently, we report on several implementations of post-quantum and hybrid key exchange in TLS 1.2, TLS 1.3, and SSHv2. We also report on work to add hybrid authentication in TLS 1.3 and SSHv2. These integrations are in Amazon s2n and forks of OpenSSL and OpenSSH; the latter two rely on the liboqs library from the Open Quantum Safe project.

- Analyze various options to integrate PQC into TLS and SSH
- Focus on hybrid scenarios
- Lessons learned from OpenSSL, OpenSSH, and s2n integrations

<https://eprint.iacr.org/2019/858>

<https://openquantumsafe.org/papers/NISTPQC-CroPaqSte19.pdf>

Hybrid scenarios



- Early migration should use a hybrid of classical/PQ schemes
 - Security of today + safety net against quantum computer
 - Secure if one of the two is secure
- TLS and SSH negotiate algorithms, but not two at the same time. We need to define either:
 - new combo schemes, e.g. ECDHE-SIKEp503:
 - Easy to implement, backward compatible
 - a new hybrid approach:
 - Flexible negotiation (algs selected separately), need spec/code changes
- Consider backward compatibility, performance, latency, data flow
- Implemented approach: combo schemes and concatenation of keys, ciphertexts, and signatures

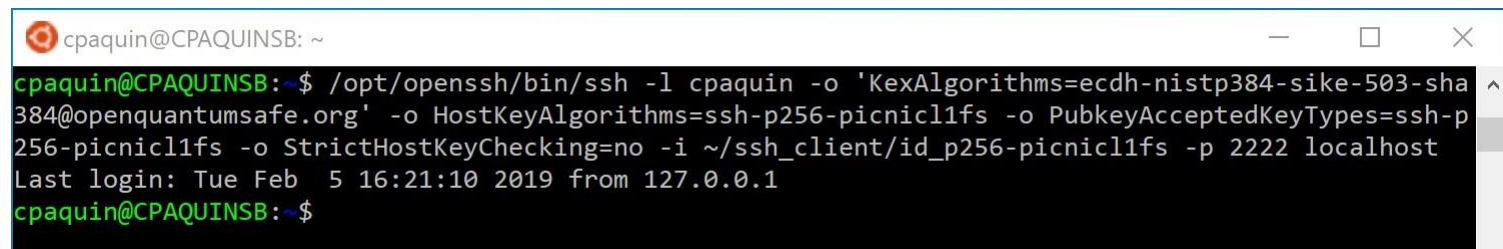
TLS case study

- Added PQ/hybrid KEX & auth
- TLS 1.2 (OpenSSL 1.0.2)
 - Explosion of schemes (specifies KEX, auth, symmetric cipher, hash)
 - Spec message size limit: 2^{24} bytes. OpenSSL limit smaller
 - Tested with OpenSSL tools, apache, OpenVPN
- TLS 1.3 (OpenSSL 1.1.1)
 - PQ algs masquerade as EC curves
 - Concat strategy more secure than 1.2 (KDF hashes transcripts)
 - Spec pub key and sig limit: $2^{16}-1$ bytes, cert limit: $2^{24}-1$ bytes. OpenSSL limit is smaller
 - Tested with OpenSSL tools, nginx
- <https://github.com/open-quantum-safe/openssl>



SSH case study

- Added PQ/hybrid KEX & auth to OpenSSH
- Define new algorithms, e.g.: [ecdh-nistp384-sike-503-sha384@openquantumsafe.org](https://openquantumsafe.org/ecdh-nistp384-sike-503-sha384@openquantumsafe.org)
- Supports both client and server public key authentication
- Spec message size limit: 2^{32} bytes, large enough for all round 2 candidates, but OpenSSH limit is smaller (2^{18})
- <https://github.com/open-quantum-safe/openssh-portable>

A terminal window with a black background and green text. The window title is 'cpaquin@CPAQUINSB: ~'. The command entered is: `/opt/openssh/bin/ssh -l cpaquin -o 'KexAlgorithms=ecdh-nistp384-sike-503-sha384@openquantumsafe.org' -o HostKeyAlgorithms=ssh-p256-picnic1fs -o PubkeyAcceptedKeyTypes=ssh-p256-picnic1fs -o StrictHostKeyChecking=no -i ~/ssh_client/id_p256-picnic1fs -p 2222 localhost`. The output shows the last login time: 'Last login: Tue Feb 5 16:21:10 2019 from 127.0.0.1'. The prompt returns to `cpaquin@CPAQUINSB:~$`.

```
cpaquin@CPAQUINSB:~$ /opt/openssh/bin/ssh -l cpaquin -o 'KexAlgorithms=ecdh-nistp384-sike-503-sha384@openquantumsafe.org' -o HostKeyAlgorithms=ssh-p256-picnic1fs -o PubkeyAcceptedKeyTypes=ssh-p256-picnic1fs -o StrictHostKeyChecking=no -i ~/ssh_client/id_p256-picnic1fs -p 2222 localhost
Last login: Tue Feb 5 16:21:10 2019 from 127.0.0.1
cpaquin@CPAQUINSB:~$
```

Key Encapsulation Mechanisms

KEM scheme	OpenSSL 1.0.2 TLS 1.2	OpenSSL 1.1.1 TLS 1.3	OpenSSH 7.9 SSH2
BIKE 1/2/3 L1/3/5 (round 1)	✓	✓	✓
Frodo KEM 640/976 AES/SHAKE	✓	✓	✓
Frodo KEM 1344 AES/SHAKE	☑	☑	✓
Kyber 512/768/1024	✓	✓	✓
LEDAcrypt KEM LT 12/32/52	✓	✓	✓
NewHope 512/1024 CCA	✓	✓	✓
NTRU HPS (2048-509/677)/(4096-821)	✓	✓	✓
NTRU HRSS 701	✓	✓	✓
NTS KEM (12,64)	✗	✗	✗
LightSaber/Saber/FireSaber KEM	✓	✓	✓
SIKE p434/p503/p610/p751	✓	✓	✓

KEM integrations for both
PQ and hybrid (with ECDHE)

Legend:

✓ Success

☑ Works with code mods

✗ Did not work

Signatures

KEM scheme	OpenSSL 1.1.1 TLS 1.3	OpenSSH 7.9 SSH2
Dilithium 2/3/4	✓	✓
MQDSS 31 48/64	☑	✓
Picnic L1 FS/UR	☑	✓
Picnic L3/L5 FS/UR	✗	✓
Picnic2 L1 FS	✓	✓
Picnic2 L3/L5 FS	☑	✓
qTesla I/III-size/III-speed (round 1)	✓	✓
Rainbow Ia Classic	☑	☑
Rainbow Ia Cyclic/Compressed	✓	✓
Rainbow IIIc/Vc Classic/Cyclic/Compressed	☑	✗
SPHINCS+ * 128s *	✓	✓
SPHINCS+ * 128f/192f/192s/256f/256s *	☑	✓

Signature integrations for both PQ and hybrid (with ECDSA)

Legend:

✓ Success

☑ Works with code mods

✗ Did not work

What's next?

For us

- Test all round 2 schemes
- Performance test
- More protocols



For you

- Start planning migration to PQC
- Start using some tools (SSH, OpenVPN)

cpaquin@microsoft.com

 @chpaquin