State of Hybrid and Composite PQC in IETF Certificate Standards

7th ETSI / IQC Quantum Safe Workshop

November 7, 2019
Talk outline

• The “P+Q” approach: why we don’t trust any one algorithm

• Proposals for P+Q certificates:
  • Problem Statement
  • Multi-cert
  • “Catalyst” or “Hybrid” certificates: *IETF draft-truskovksy*
  • “Composite” certificates: *IETF draft-ounsworth*
  • IETF community reaction

• Closing thoughts
The “P+Q” approach

- Term coined by Dan Bernstein
- We don’t trust any one algorithm.
  - When will / did quantum computers come online?
  - When will / did PQ algs’ math break?
- “P+Q” allows us to build resilience and agility into the infrastructure.
- It’s FIPS compliant today 🟢
Proposals for P+Q
Transitional Certificates
PQ Problem Statement draft

- After IETF 105 in July, we stepped back and wrote a “Problem Statement” draft proposing base requirements of a PQ multi-key certificate, and the 3 categories of solution:

Table of Contents

1. Problem Statement ........................................... 2
   1.1. Formal Security Requirements ......................... 2
2. Solution Space ................................................. 3
   2.1. "Composite" concatenated keys and signatures ........ 4
   2.2. Application: X.509 ....................................... 5
       2.2.1. Multiple Certificate Chains ...................... 5
       2.2.2. "Hybrid" v3 extensions ............................ 6

More on community reaction in a few slides…

IETF: draft-pq-pkix-problem-statement
Solution 1: “Multiple certs”

Subject: “Joe”
Issuer: JoeCorpCA
PubKey: RSA2048
v3 Extns:
- joe@corp.com
...
SigAlg: RSA4096
Sig: 10111010100…

Subject: “Joe”
Issuer: JoeCorpCA
PubKey: Dilithium
v3 Extns:
- joe@corp.com
...
SigAlg: SPHINCS+
Sig: 1101000001…

Digital Signature 1: 11101001…
Digital Signature 2: 11101001…
Solution 1: “Multiple certs”

Driving motivation:
- **No new x509 tech**, just CAs on new algorithms.
- **Great for negotiated protocols** (TLS, IPSEC) because servers can choose which / how many certificates to sign with based on ClientHello.
- **Protocol designers have control** of how / when to transmit the large PQ cert.

Cons:
- Requires changes to all protocols to append and validate two signatures.
- Unclear if this is applies to **non-negotiated protocols**. For example CMS (S/MIME):
  - You don’t know if receiving clients can handle PQ algorithms.
  - CMS (and S/MIME) currently support multiple SignerInfos, but whether to treat this as **AND** or **OR** is undefined in the spec; behaviour varies by client. Similar for XMLDSig, DLL code signing, etc …
Solution 2: ISARA “Catalyst” certs (aka “Hybrid certs”)

Subject: “Joe”
Issuer: JoeCorpCA
PubKey: RSA2048
v3 Extns:
  - joe@corp.com
  - AltPubKey: Dilithium
  - AltSigAlg: SPHINCS+
  - AltSig: 01101001…

SigAlg: RSA4096
Sig: 10111010100…

IETF: draft-truskovsky-lamps-pq-hybrid-x509
ITU: Approved October 2019, publication pending
Solution 2: ISARA “Catalyst” certs (aka “Hybrid certs”)

Driving motivation:
- **Highly backwards-compatible** because non-critical v3 extns can be ignored by legacy clients.

Cons:
- Great backwards compatibility **may give a false sense of security** …
- Requires **changes to all protocols** to append and validate two signatures.
- **Large-bandwidth PQ keys / sigs are always sent** even if the client doesn’t use them (*TLS WG doesn’t like this.*)

*IETF:* draft-truskovsky-lamps-pq-hybrid-x509

*ITU:* Approved October 2019, publication pending
Solution 3: “Composite certs”

Subject: “Joe”
Issuer: JoeCorpCA
PubKey: Composite
  {RSA2048, Dilithium}
v3 Extns:
  - joe@corp.com
...
SigAlg: Composite
  {RSA4096, SPHINCS+}
Sig: {10111010100..., 011010011010...}

IETF: draft-ounsworth-pq-composite-sigs
Solution 3: “Composite certs”

Driving motivation:

• “Drop-in” for *any* existing OID-based crypto protocols; just add a new signature algorithm id-alg-composite.
• Trivially extends to 3+ keypairs.
• **Complexity moved to crypto layer**: less to get wrong for protocol designers and implementors.
• ITU X.510-dis proposes a variant of this (*but will not apply to IETF PKIX because of ASN.1 struct divergence between IETF and ITU*).

Cons:

• **Backwards compatibility is tricky.**
• **Large-bandwidth PQ keys / sigs are always sent** even if the client doesn’t use them (*TLS WG doesn’t like this*).
• **Complexity moved to crypto layer**: less control for protocol designers and implementors.

**IETF**: draft-ounsworth-pq-composite-sigs
Reaction to Problem Statement draft

- Discussion is on IETF SecDispatch mailing list.
- Mixed reaction:
  - Agreement this problem needs to be solved. Disagreement about how.
  - Implementors of closed PKI environments want a simple extension of X.509 to use now (Catalyst or Composite would be fine – speed more important than substance).
  - Some want to ditch X.509 entirely for something totally new (MathMesh?)
  - Largely silence from the TLS community.
- IETF 106 Singapore, Nov 16 – 22 will be interesting…
Bonus topic: What about PQ Pub-Key size??

• Million-dollar question: will 5 kb+ pub keys / signatures break everything?
  • Concern: what happens if the ServerCertificate is larger than ethernet MTU (1,500 bytes) or TCP congestion window during the (~ 1.4 kb) 2nd RTT of the TLS handshake?
  • Open question: How significant is this concern?

• Recent experiments by CloudFlare / AWS / Google are starting to shed some light and numbers.
Closing thoughts
Summary

• Watch IETF SecDispatch and LAMPS for public discussion of the way forward for TLS and PKIX.
  • Please weigh in with use-case specific concerns!

• Watch for ITU to publish a Catalyst certificate spec if you need strong backwards-compatibility.

• Think about the implications of backwards compatibility and large certificates in your environments … What’s going to fail?
  • Do you need a staged migration? Is it even possible?
Mike Ounsworth
Email: mike.ounsworth@entrustdatacard.com