Addressing Public Key Algorithms Uncertainties with Composite Crypto

{ Cable**Labs**® ◦ Open**CA** ◦ EntrustDatacard ◦ Cisco Systems ◦ ISARA ◦ DigiCert }

Massimiliano Pala <m.pala@cablelabs.com>

Mike Ounsworth <mike.ounsworth@entrustdatacard.com>

Problem Statement

- With the advancements in cryptography and the level of investment in Quantum Computing technologies, we live in a period of uncertainty
  - Will RSA and ECC fall ?
  - Are Lattice schemes as strong as we believe ?
- A possible workaround is to use hybrid solutions where multiple algorithms can be used together (e.g., Certificates)
- Some initial experiment for the deployment of new QR algorithms have been published for TLS
  - More work needs to be done to address the Size/Latency of PQ algorithms

Current Proposals

- Some industry verticals are already looking at how to address the deployment of these new algorithms
  - Is it possible to protect today's infrastructure and transition to the new algorithms in the future (deferred algorithm agility) ?
  - Are your PKIs going to be secure in 20 years ? What about 30 ?
- There are currently different categories of solutions, none of which seems to fit all use cases
  - Multi-Chain (Multiple Certificates)
  - ISARA Catalyst (draft-truskovsky-lamps-pq-hybrid-x509)
  - ***Composite Crypto (draft-ounsworth-pq-composite-sigs)***

Solution #1: Multiple Certificates

- Pros
  - Great for Negotiated Protocols – Server can choose which certificate(s) to sign with based on the, for example, TLS' Client Hello
  - Protocol Designers have control over how/when to transmit the large PQ certs
- Cons
  - Require changes to all protocols to be able to validate multiple signatures
  - It is not clear how this would work with non-negotiated protocols like S/MIME
    - Does the receiving client support PQ algorithms ?
    - CMS (and S/MIME) support multiple `signerInfos` – should the signatures be treated as an AND or OR when validating ? How to link the identities in the two certificates ?
  - Difficult linking identities across different infrastructures / CAs
    - Complicates PKI operations (dependencies across different CAs)

Solution #2: Hybrid Certificates

- ITU-T added new extensions and rules for alternative signatures and keys. Their use-case is migration, i.e. verify one signature or the other, although the idea could be extended to composite crypto.
- Pros
  - Backwards Compatible – non-critical extensions can be ignored
- Cons
  - Backwards Compatible – because the signature is included in a non-critical extension, applications might not process it but still accept the certificate (false sense of security)
  - Requires changes to all protocols to append and validate two signatures
  - Large PQ keys and signatures are always sent (even when they are not used)[1]

[1] This could be mitigated with a (more complicated) modification to send the alt values separately

Solution #3: Composite Cryptography

- Composite Cryptography is defined as new Public Key Algorithm
- The data structures defined to hold keys and signatures use, internally, SEQUENCES of signatures and public keys to carry the authentication data in a single “container”

```
CompositePublicKey ::= SEQUENCE SIZE (1..MAX) OF SubjectPublicKeyInfo

CompositePrivateKey ::= SEQUENCE SIZE (1..MAX) OF OneAsymmetricKey

CompositeSignatureValue ::= SEQUENCE SIZE (1..MAX) OF BIT STRING

CompositeParams ::= SEQUENCE SIZE (1..MAX) OF AlgorithmIdentifier
```

- Provides compatibility with existing certificate processing rules
  - One Composite Key generates One Composite Signature

Solution #3: Composite Cryptography[1]

- Pros
  - “Drop-In” for any existing OID-based crypto protocols
  - Can be extended to combine { 2, 3, ... , N } algorithms and public keys
  - Complexity moved to the crypto layer – normal PKI operations
  - Can be used in all aspects of PKIX cycle management (e.g., revocation, storage, etc.)
  - Can be used with any protocols that make use of X.509 certificates today
- Cons
  - Backward Compatibility requires the support of the new algorithm identifier.
  - Large PQ keys and signatures are always sent (even when they are not used)

[1] ITU X.510-dis uses a similar approach (different validation rules) – should avoid competing standards

Current Status of Discussion

- Discussion has been going on for a while on IETF SecDispatch, LAMPS, and PKIX
- Mixed Reaction to the problem statement I-D and proposal for a solution
  - There has been agreement that this is a problem that needs to be solved
  - Large PKI providers (mostly private PKIs) would like to see a solution that they can start deploying in the “near-ish” future
  - Other proposals have emerged in the discussion for completely different approaches (e.g., MathMesh)
  - Largely we have not heard from the TLS community on this ...

Conclusions

- Discussion around deploying quantum-resistant Trust Infrastructures has been going on for a while, and there is agreement that this is a problem that we would like to have a solution for
  - Industry would like this tool to plan for the next 30 or 40 years (!!!)
- Some solutions have been proposed that address different aspects of the problem
  - None of which provides a solution that would not require protocol changes
- We think that Composite Crypto provides the missing piece/tool for securing all aspects of PKIs – from certificates to revocation information

Future Work

- What is the best path forward ?
  - BoF ? WG Adoption (which WG) ? Individually Sponsored ?

- ***References***

  - *Problem Statement and Overview of Solution Space*
    https://datatracker.ietf.org/doc/draft-pq-pkix-problem-statement/

  - *Composite Keys and Signatures*
    https://datatracker.ietf.org/doc/draft-ounsworth-pq-composite-sigs/