IETF LAMPS

PQ TRANSITION MECHANISMS

Mike Ounsworth

Jan 28, 2021

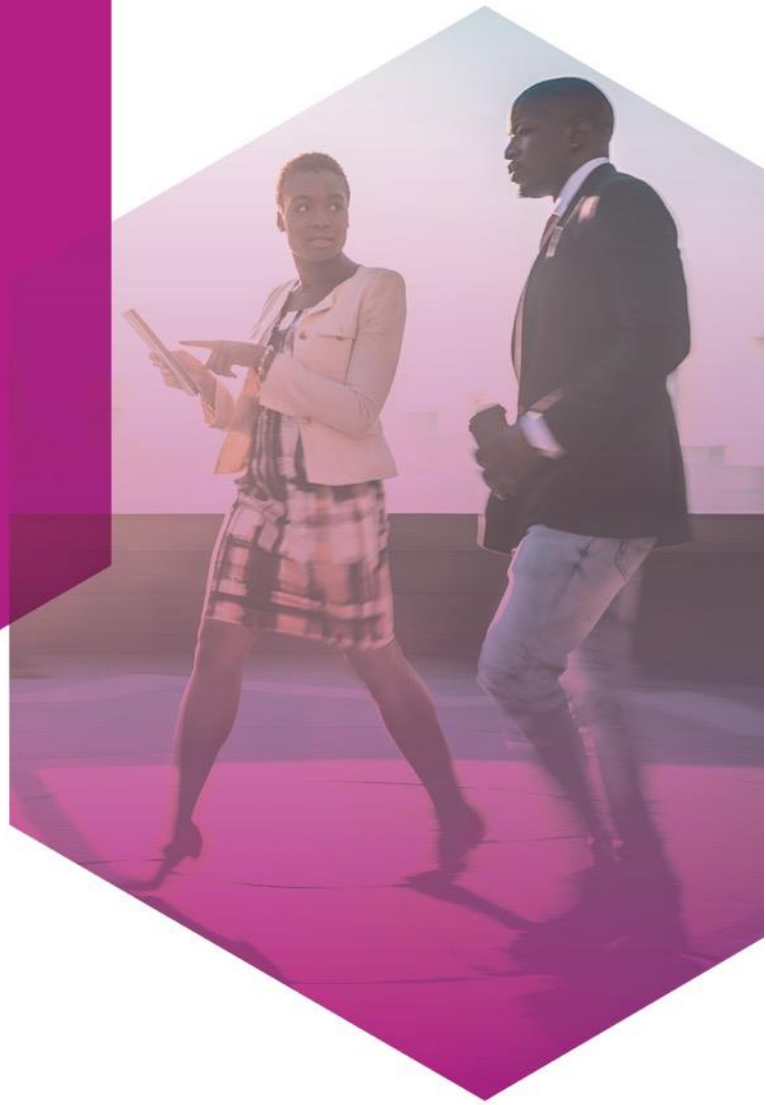
Outline

- LAMPS re-charter suggestion:
  - Hybrid key establishment and Dual signatures
  - Suggested nomenclature
- Rationale for a Composite Signatures standard
- Rationale for a Composite Certificates standard

- Bonus topics:
  - Multiple SignerInfos does not give you Dual Signatures in CMS
  - Supporting KEM Certificates in PKIX
  - Potential changes to Composite draft

LAMPS RE-CHARTER SUGGESTION

Hybrid key exchanges and Dual signatures



Post-quantum algorithm uncertainty

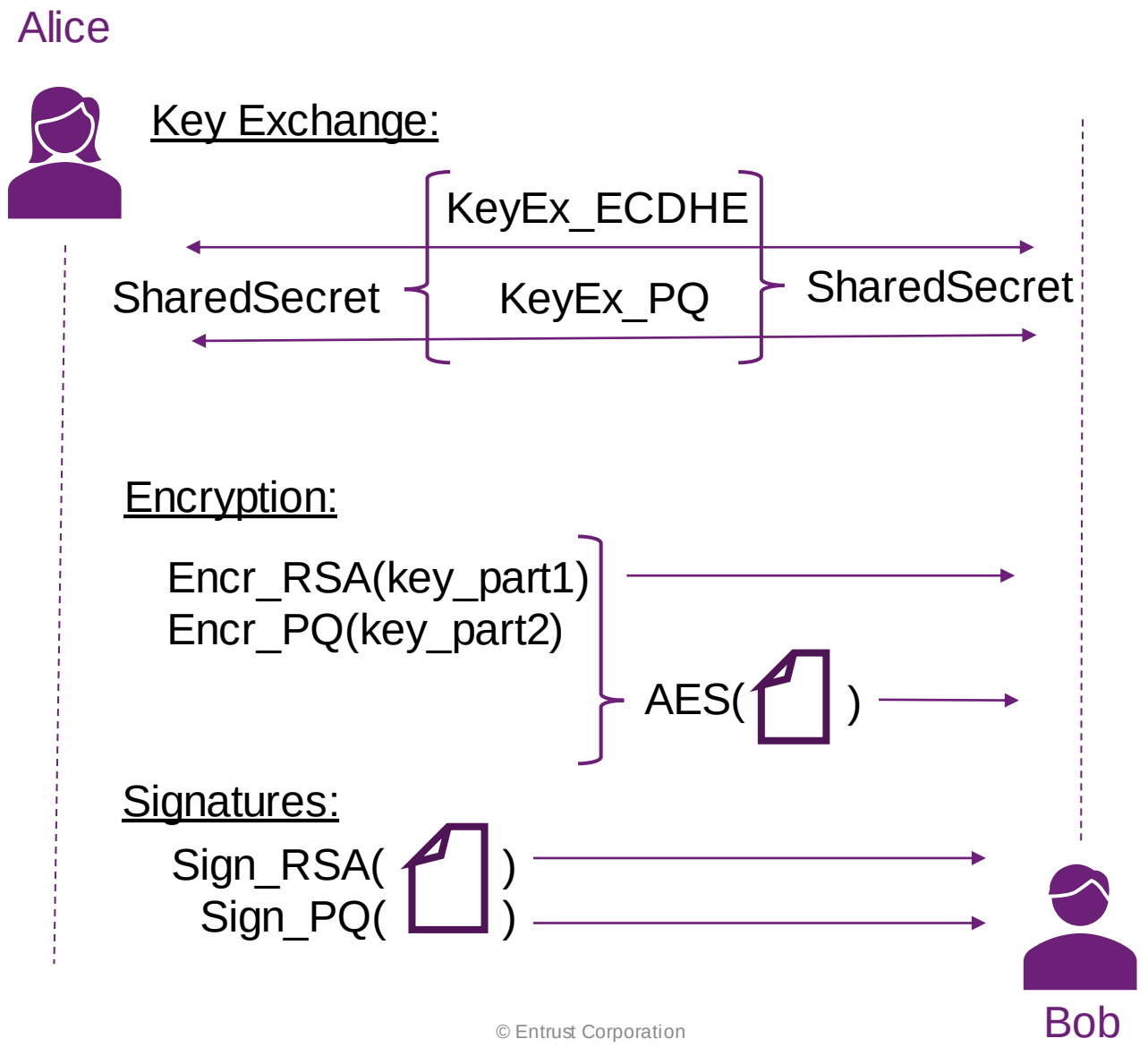
- On 21-Jan-21 Dustin Moody (NIST) said:

  "NIST notes that in the third round there are 3 signature finalists, and 3 signature alternates. Recent cryptanalysis has impacted the analysis of both Rainbow and GeMSS.

- We are late in the NIST PQC process, and still new cryptanalytic attacks are coming out.[1]

1: 21-Jan-21 thread "Diversity of signature schemes" on NIST PQC Forum

Hybrid and Dual crypto modes



Motivation: NIST PQC FAQ

Transition and Migration

Is it possible for a hybrid key-establishment mode to be performed in a FIPS 140 approved mode of operation? (added 1/28/20)

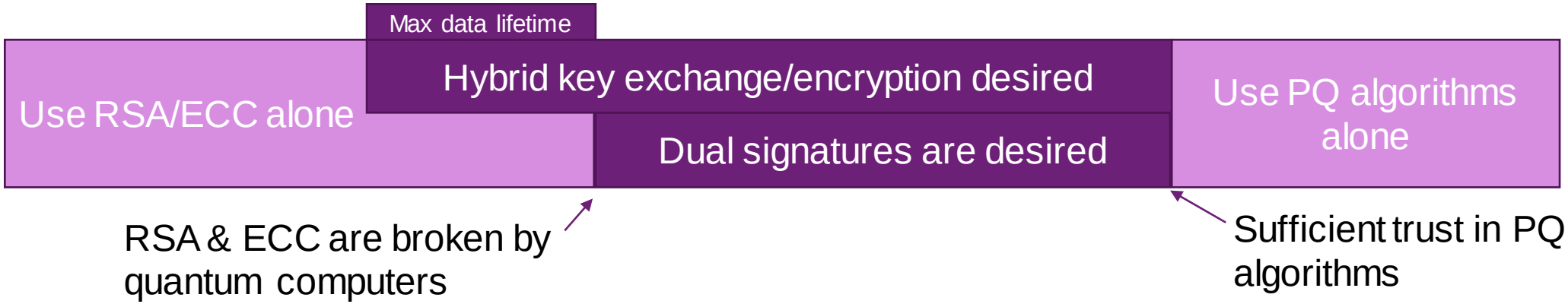
Is it possible for dual signature generation or verification to be performed in a FIPS 140 approved mode of operation? (added 1/28/20)

Does NIST consider the hybrid key establishment modes and dual signatures to be long-term solutions? (added 1/28/20)

NIST leaves the decision to each specific application as to whether it can afford the implementation cost, performance reduction, and engineering complexity (including proper and independent security review) of a hybrid mode for key establishment or the use of dual signatures. Future experience will help to decide on whether they can be a useful long-term solution. To assist external parties who desire such a mechanism, NIST will accommodate the use of a hybrid key-establishment mode and dual signatures in FIPS 140 validation when suitably combined with a NIST-approved scheme.

Post-quantum algorithm uncertainty

- Hybrid KeyEx / DualSigs (RSA/ECC + PQ) provide protection against further cryptanalytic breakthroughs until we have confidence in lattice and multivariate-based crypto.



- Hybrid / dual modes provide protection in two senses:
  - Time between publication of NIST standards, and full alg confidence.
  - Time between publication of new attack, and patching.

Proposed LAMPS Charter Text

Post-quantum cryptography (PQC) will require a transition period in some ways similar to previous crypto migrations, but unique in that timelines require deployment of PQC before cryptographers have full confidence in the replacement algorithms. NIST has called for transition mechanisms that “layer” traditional and PQ crypto together, referred to as “hybrid key exchange” and “dual signatures”. The LAMPS working group will update documents produced by the PKIX and S/MIME WG to specify hybrid key establishment, encryption, and dual signature mechanisms.

Suggested nomenclature

Concepts:

- **Hybrid Key Establishment** := Any “key-establishment scheme that is a combination of two or more components that are themselves cryptographic key-establishment schemes. ” [NIST PQC FAQ; and SP 800-56C]
- **Dual Signature** := Any signature scheme that “consists of two (or more) signatures on a common message.” [NIST PQC FAQ]

Instantiations:

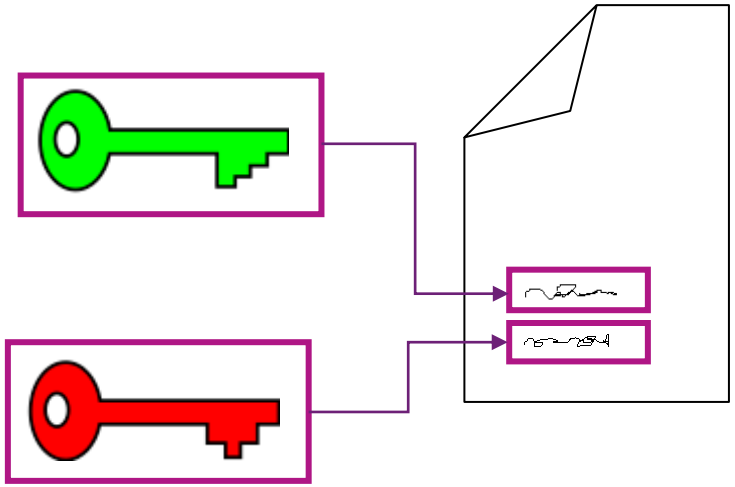
- **Composite Signature** := A type of dual signature that combines multiple “*component”* keys / signatures into a single object. *draft-ounsworth-pq-composite-sigs* provides a proposal for achieving this.

RATIONALE FOR A COMPOSITE SIGNATURES STANDARD

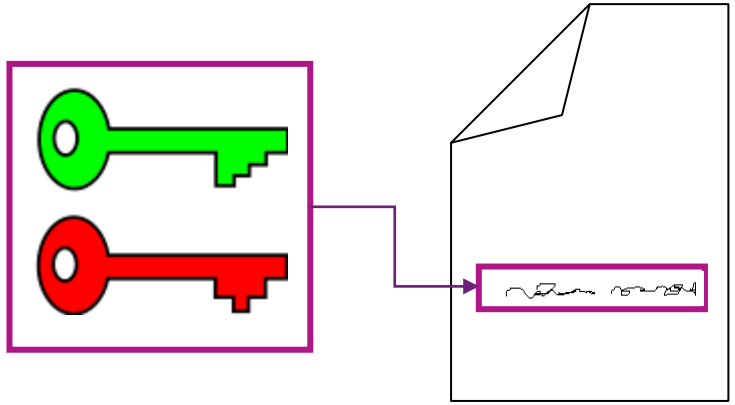
Alternative models for dual signatures

- At a high level, you have two design options:

(1) Independent keys and signature objects



(2) Combined key and signature object



- Russ has advocated for (1), we are advocating for (2).
  - *… but maybe not mutually-exclusively? There are likely valid use-cases for both…*

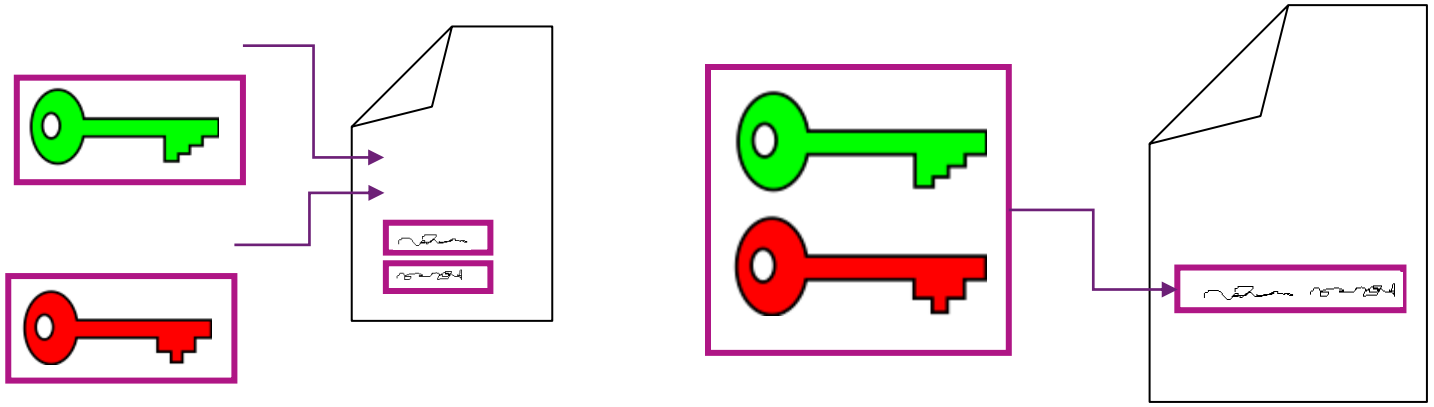
Rationale for composite: protocol integration simplicity

- “Just another public key and signature OID", fits into existing crypto agility mechanisms.
- Alternative: each protocol defines how multiple signatures and keys are carried, but independent signatures bring up considerations that do not arise if a single composite signature is used:
  - Where in the protocol can the second signature or public key be placed?
  - Stripping attacks: can the attacker drop one of the signatures and have the message accepted?

Rationale for composite: Do the security analysis once

- Most of the effort spent on draft-ounsworth-pq-sigs was getting the composite signing and verifying algorithms right; lessons learned:
  - No recursion!
  - Preventing **stripping attacks** (attacker entirely removes either the RSA or PQ part). Requires:
    - All algorithms MUST sign over the algIds of all algorithms, ie independent detached signatures are vulnerable to stripping / downgrade attacks.
    - All referenced pub keys / algIds MUST produce a signature (ie no "subset signatures") as verifier cannot distinguish this from a stripping attack.
- We believe there is enough complexity here to warrant a standard for composite signatures.

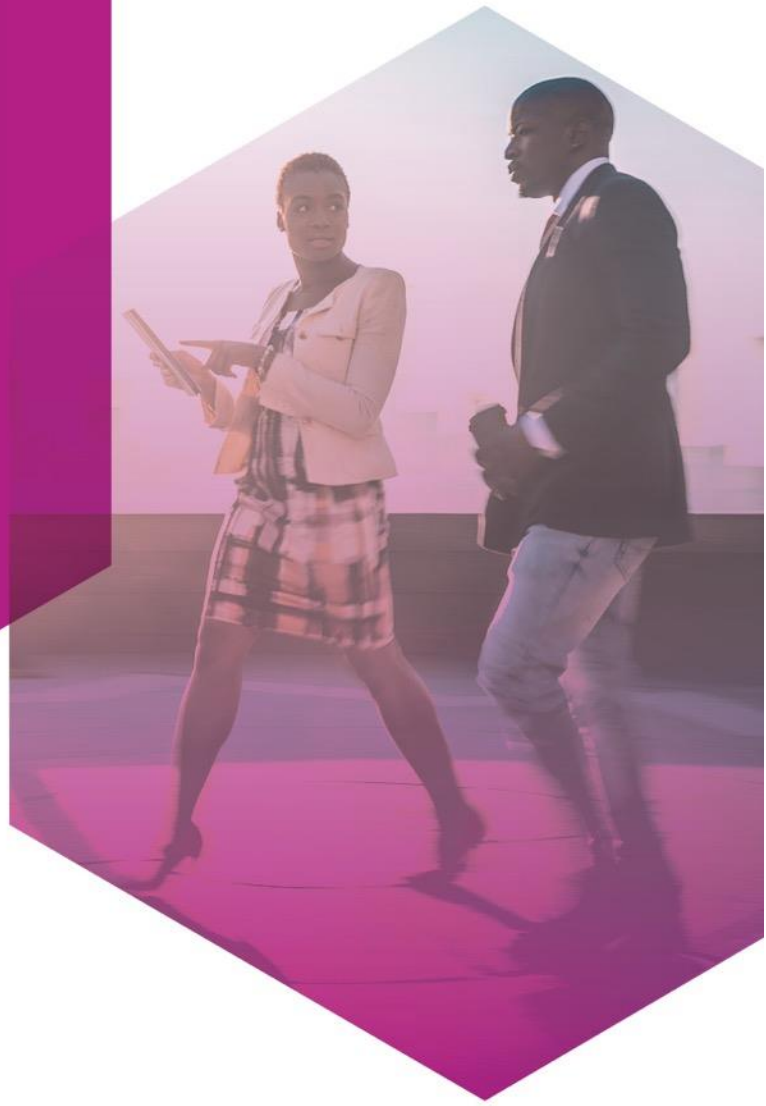
A composite signatures standard is needed



For reasons of protocol integration simplicity, and doing the security analysis in one central place, we believe at least some protocols would make use of a centrally-defined standard for a composite signature algorithm, rather than leaving each protocol to independently figure out how to use multiple certificates.

See bonus material section: we believe that CMS will need some sort of composite mechanism; that multiple SignerInfos is not sufficient.

RATIONALE FOR A COMPOSITE CERTIFICATES STANDARD



Rationale for composite certs: stronger cryptographic binding

- Bind both keys together by the CA.
  - Stronger protection against stripping / downgrade attacks because attacker does not have RSA-only and PQ-only certs to manipulate independently.
- Failure mode: user forgets to load second cert into application (thereby silently loses dual crypto protection).
- For protocols that make authorization decisions based on DN of client cert, need to check that both certs presented match.

Rationale against multi-cert: management complexity

- Environment fleet management: doubles the number of certs for admins to distribute, track, renew, etc.
- "Joe Admin" needs to do two CSR flows, load two certs into application (without mixing up the private keys).
  - I know, not a standards body concern, but multi-cert sounds like a customer support headache.

Responses to Composite counter-arguments

- Having a signature primitive that combines arbitrary pairs (or more) of signature schemes increases risk.
  - If preferred, we can alter the composite-sigs draft to take pairwise algIds (ex.: sa-compositeRSAandDilithium) in such a way that defining a new pair is just instantiating the composite ASN.1 Information Object Class with a pair of algorithms.
- "Jumbo" certs.
  - In discussions we're aware of, criticism pertained to mixing KeyUsages (signing + keyEx) within the same cert.
  - Size: If you already have a PQ cert, adding RSA or ECC is small.
    - *(the "one-more-shovel-to-a-dumptruck" argument)*

Summary

- LAMPS re-charter suggestion:
  - Hybrid key establishment and Dual signatures
  - Suggested nomenclature
- Rationale for a Composite Signatures standard
- Rationale for a Composite Certificates standard

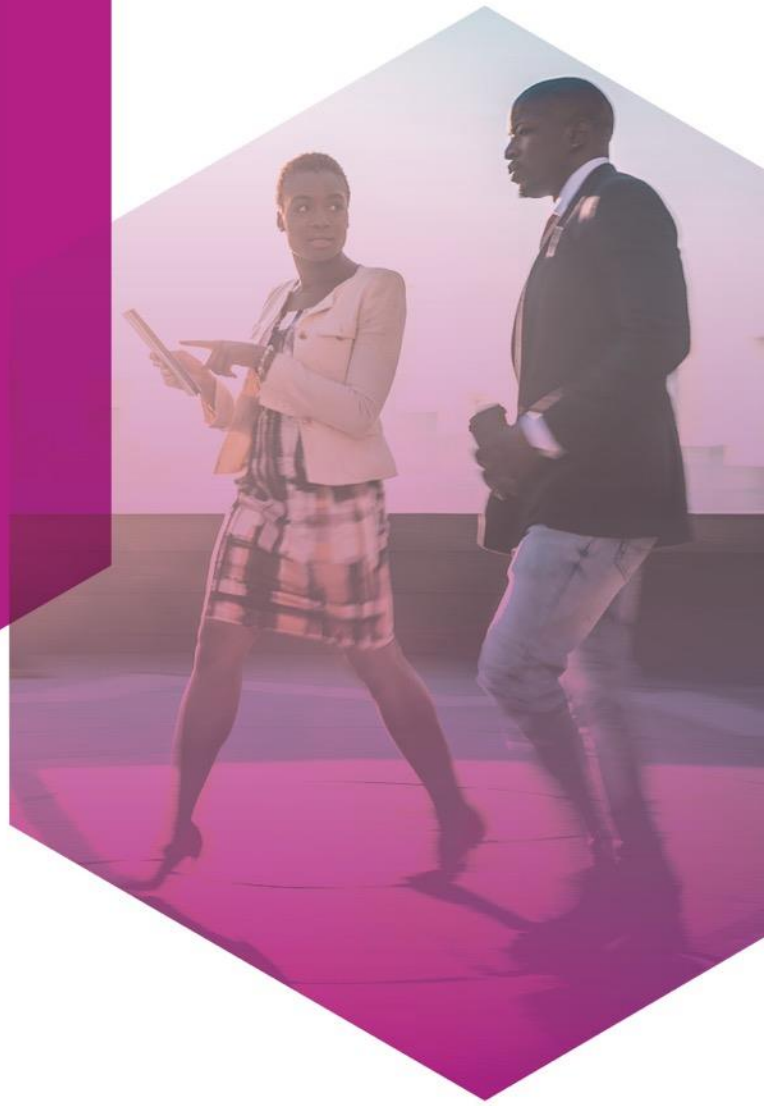
draft-ounsworth-pq-composite-sigs

mike.ounsworth@entrust.com

- Bonus topics:
  - Multiple SignerInfos does not give you Dual Signatures in CMS
  - Supporting KEM Certificates in PKIX
  - Potential changes to Composite draft


ENTRUST
SECURING A WORLD IN MOTION
BONUS TOPICS

MULTIPLE SIGNERINFOS DOES NOT GIVE YOU DUAL SIGNATURES IN CMS



Ex use case: CMS (RFC 5652) using multi-SignerInfo

```
SignedData ::= SEQUENCE {
        ...
        signerInfos SignerInfos }

    SignerInfos ::= SET OF SignerInfo
```

> *When more than one signature is present, the successful validation of one signature associated with a given signer is usually treated as a successful signature by that signer. However, there are some application environments where other rules are needed.*

So we can't mandate using multiple SignerInfos to accomplish PQ dual signatures without getting into hideous backwards compatibility messes with implementations that already use multiple SignerInfos for some useful purpose.

Also, a naïve implementation is vulnerable to stripping attacks since attackers can remove a SignerInfo without invalidating the signatures in other SignerInfos, and thus is not a cryptographically-strong dual signature.

HOWEVER…

Ex use case: CMS (RFC 5652) using Composite

```
SignerInfo ::= SEQUENCE{
        version CMSVersion,
-->     sid SignerIdentifier,
        digestAlgorithm DigestAlgorithmIdentifier,
        signedAttrs [0] IMPLICIT SignedAttributes OPTIONAL,
-->     signatureAlgorithm SignatureAlgorithmIdentifier,
-->     signature SignatureValue,
        unsignedAttrs [1] IMPLICIT UnsignedAttributes OPTIONAL}
```

> *sid specifies the signer's certificate (and thereby the signer's public key).*

If you use composite structures for the things marked with "-->", then you have multiple single-algorithm certificates that combine to create a single composite signature.

(and only this SignerInfo is Composite; it would not interfere with other SignerInfos carrying other signatures.)

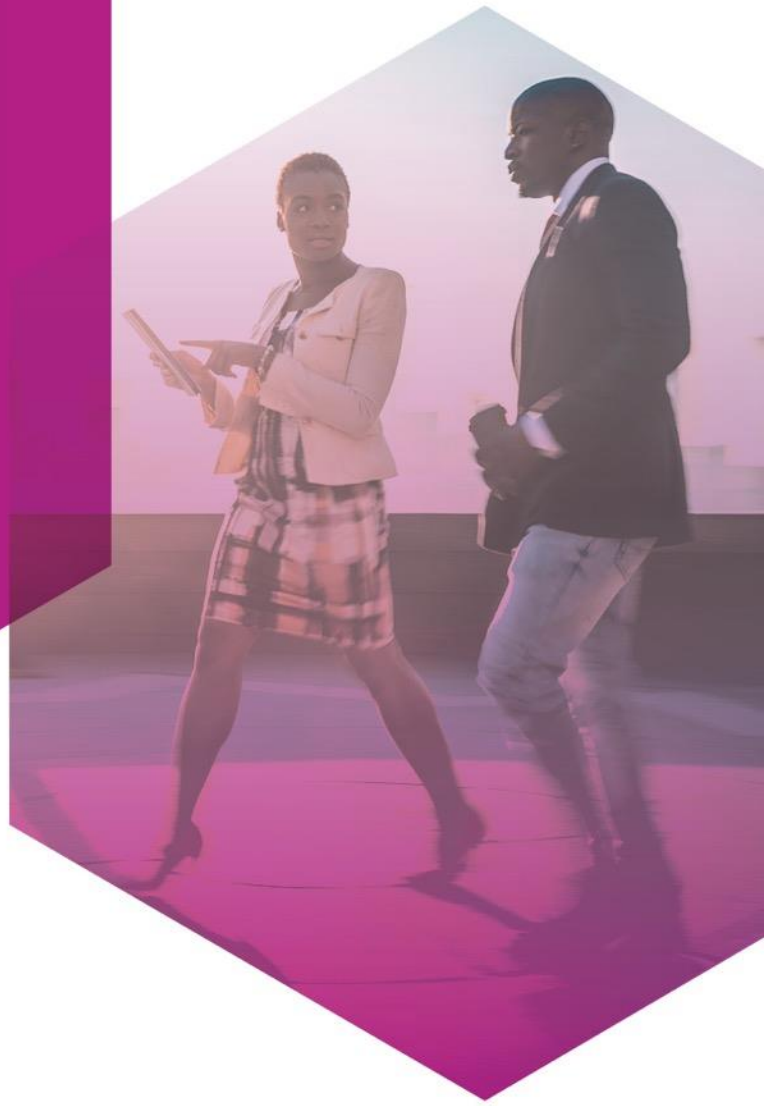
Benefits of a single composite signature spec

```
SignerInfo ::= SEQUENCE{
-->     sid SignerIdentifier,
…
-->     signatureAlgorithm SignatureAlgorithmIdentifier,
-->     signature SignatureValue,
...}
```

The generation and verification of composite signatures have some security-critical complexity, so even though the ASN.1 data structures are straight-forward, there is benefit in a centralized spec for the behaviour.

Therefore, if LAMPS decides to proceed with adopting PQ “dual signature” modes in CMS and PKIX, then draft-ounsworth-pq-composite-sigs, or a draft like it, is necessary to standardize the generation and verification behaviour.

SUPPORTING KEM CERTIFICATES IN PKIX



Supporting KEM Certificates in PKIX

- KEMs are not my main expertise, but:
  - PQ encryption algs are tending towards Key Encapsulation Mechanisms (KEMs – aka “key transport”) rather than DH-style algs.
  - There are some proposals for KEM-only handshake protocols which also use KEMs for authentication (ie replacing signatures), these require certificates containing KEM keys (see: KEMTLS [1, 2]).
  - Therefore, we *may* see increased popularity of `keyUsage :: keyEncipherment` certificates, that don’t necessarily have a matching signing certificate.

- Open Question: are modern PKI deployments equipped to handle enrollment, update, revocation of certs without signing keys?

1: “Post-quantum TLS without handshake signatures”, Swabe, Stebila, Wiggers, https://eprint.iacr.org/2020/534
2: “KEMTLS: Post-quantum TLS without signatures”, Celi, Wiggers, https://blog.cloudflare.com/kemtls-post-quantum-tls-without-signatures/

Supporting KEMs in CMS and PKIX

- Quick PKI KEM-compatibility search; does it have **enrollment / key update** mechanisms for:

Enrollment / key update	CMP (RFC 4210)	Lightweight CMP	EST (RFC 7030)	EST with full CMC (RFCs 7030 + 5272)
digitalSignature	Yes	Yes	Yes	Yes
keyEncipherment / dataEncipherment	Yes	Partial [1]	No [2]	Yes
keyAgreement	Yes	Partial [1]	No [2]	No?

1: EE must also have a signature cert
2: RFC 7030 “section 3.4 Proof-of-Possession” only mentions signed enrollment requests

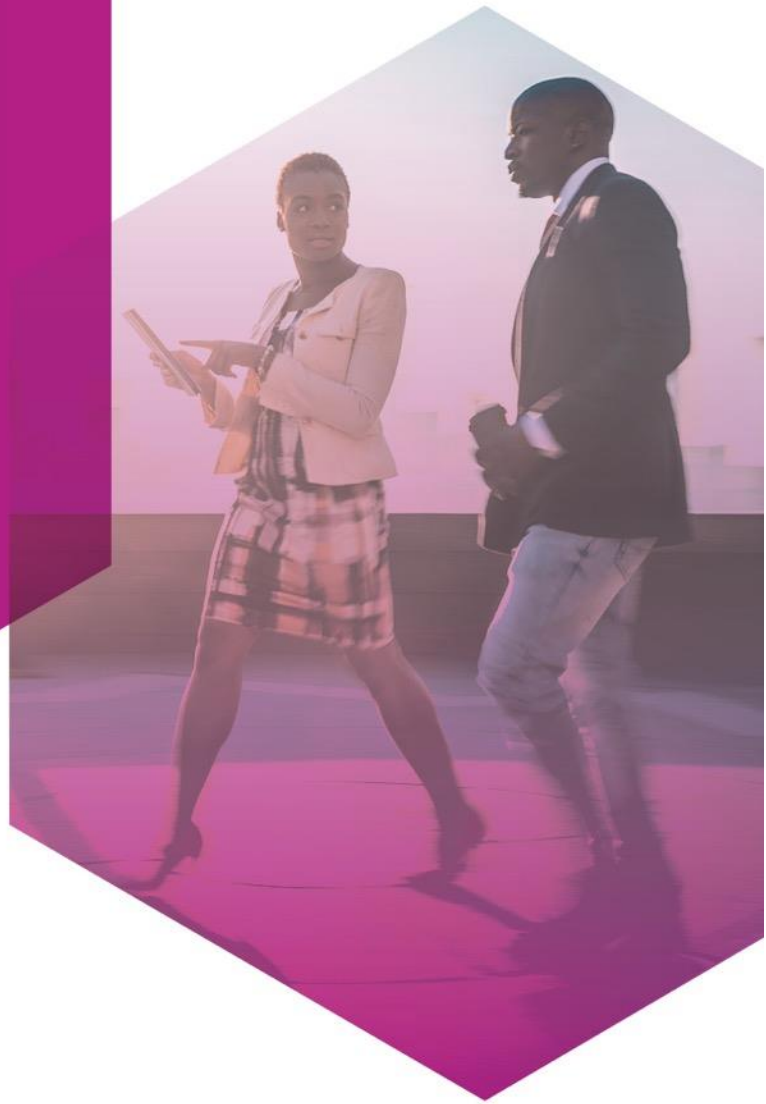
Supporting KEMs in CMS and PKIX

- Quick PKI KEM-compatibility search; does it have **revocation** mechanisms for:

Revocation	CMP (RFC 4210)	Lightweight CMP	EST (RFC 7030)	EST with full CMC (RFCs 7030 + 5272)
digitalSignature	Yes	Yes	Yes	Yes
keyEncipherment / dataEncipherment	No [1]	Partial [2]	No [3]	Yes
keyAgreement	Yes [1]	Partial [2]	No [3]	No?

1: A DH mechanism is provided, but not a keyEncipmerment mechanism.
2: EE must also have a signature cert
3: RFC 7030 “section 3.4 Proof-of-Possession” only mentions signed enrollment requests

CHANGES WE'RE CONSIDERING TO DRAFT-OUNSWORTH-PQ-SIGS



Composite variant: explicit pair-wise OIDs

Currently:

```
CompositePublicKey ::= SEQUENCE SIZE (1..MAX) OF SubjectPublicKeyInfo
sa-CompositeSignature .. CompositeParams ::= SEQUENCE SIZE (1..MAX) OF AlgorithmIdentifier
sa-CompositeSignature .. CompositeSignatureValue ::= SEQUENCE SIZE (1..MAX) OF BIT STRING
```

Allows arbitrary number in arbitrary combinations.

Based on list feedback, we're working on another version of the draft that accepts pairwise algIds (ex.: `sa-compositeRSAandDilithium`) in such a way that defining a new pair is just instantiating the composite ASN.1 Information Object Class with a pair of algorithms.

```
sa-explicitCompositeSignatureAlgorithm {
   OBJECT IDENTIFIER:algId,
   SIGNATURE-ALGORITHM:firstAlg,
   PUBLIC-KEY:firstPublicKey,
   FirstPublicKeyType,
   SIGNATURE-ALGORITHM:secondAlg,
   PUBLIC-KEY:secondPublicKey,
   SecondPublicKeyType} SIGNATURE-ALGORITHM ::= {
      ...
}
```

```
sa-entrust-sha256RSAandECDSASIGNATURE-ALGORITHM ::=
  sa-explicitCompositeSignatureAlgorithm {
     id-sa-entrust-sha256RSAandECDSA,
     sa-sha256WithRSAEncryption,
     pk-rsa,
     RSAPublicKey,
     sa-ecdsaWithSHA256,
     pk-ec,
     ECPoint
}
```