Composite vs Parallel Signatures

PQUIP 119

Mike Ounsworth, Orie Steele, Britta Hale

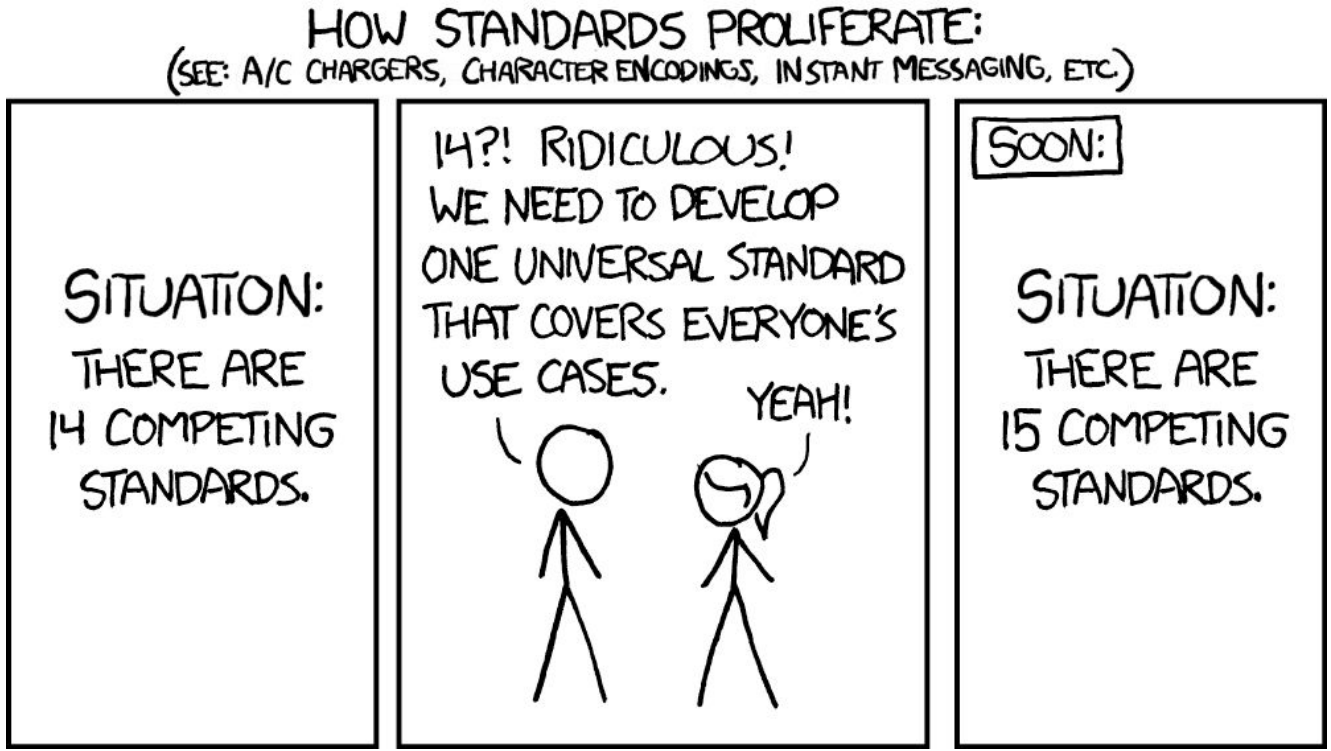
Talk Outline

Composite, Unlinked multi-signatures, Linked multi-signatures, Counter-signatures.

- Definitions
- Instantiations in LAMPS, OpenPGP, JOSE/COSE drafts
- Open questions

Definitions

Mandatory XKCD



We are trying to *NOT* do this; these slides are presenting what already exists, not proposing anything new.

Definitions

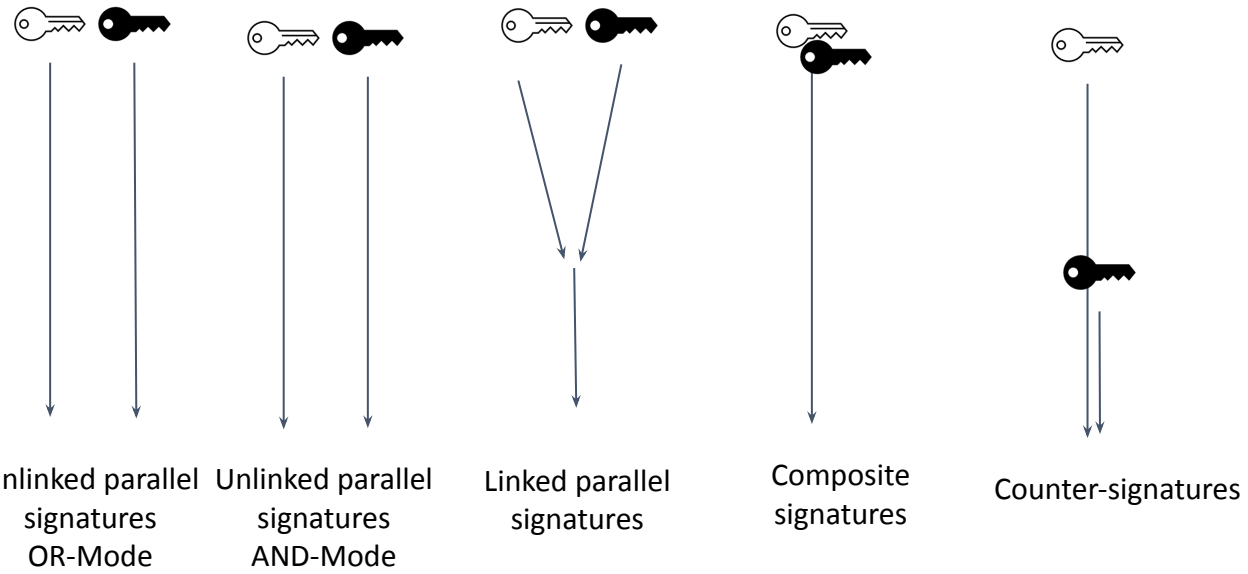
- **Hybrid / Multiple-algorithm scheme**
  - “A cryptographic scheme that incorporates more than one component algorithm, where the component algorithms have the same cryptographic purpose.” (draft-ietf-pquip-pqt-hybrid-terminology)

  - T-T Hybrid (P-256 ECDSA + Ed25519 EdDSA)
  - PQ-T Hybrid (ML-DSA + Ed25519 EdDSA)
  - PQ-PQ Hybrid (ML-DSA + SLH-DSA)

Binding diagrams

Keys / certificates

Protocol



Unlinked parallel signatures OR-Mode

Unlinked parallel signatures AND-Mode

Linked parallel signatures

Composite signatures

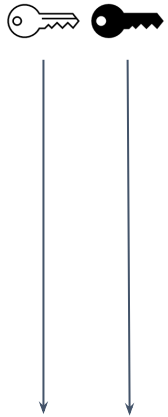
Counter-signatures

Definitions

- **Un-linked parallel signatures**
  - A document contains two detached signatures with no linking information between them.

```
sig1 = ml-dsa.sign(m);
sig2 = ed25519.sign(m);
```

  - Is essentially an OR mode.
  - Does not achieve any Non-Separability as per draft-hale-pquip-hybrid-signature-spectrums.

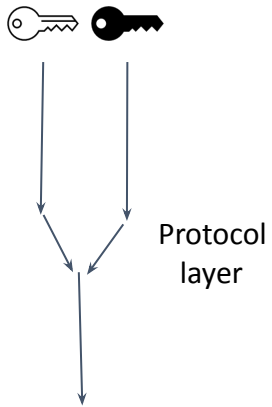


Definitions

- **Protocol-layer Linked parallel signatures**
  - A document contains two detached signatures with some sort of linking information (directed or undirected) between them.

```
sig1 = ml-dsa.sign(m || “also an ed25519 sig exists”);
sig2 = ed25519.sign(m || “also an ml-dsa sig exists”);
```

  - Is an AND mode.
  - Aims to achieve Weak Non-Separability as per draft-hale-pquip-hybrid-signature-spectrums.



Definitions

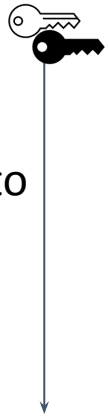
- **Counter-signature (aka “nested signatures”)**
  - A second signature signs the entire original document, including the original signature.

```
sig1 = ml-dsa.sign(m);
sig2 = ed25519.sign(m || sig1);
```

  - Is a half-AND mode – only in one direction since sig2 can be trivially stripped without leaving any evidence.
  - Does not, in general, achieve any Non-Separability as per draft-hale-pquip-hybrid-signature-spectrums (if it does, it’s only in one direction)

Definitions

- **Crypto-layer Linked Parallel Signatures (aka Composite)**
  - The goal is to bundle two things to look like one thing for “protocol backwards compatibility”.
  - And they have the security property that composite signature values are bound to the composite algorithm ID:
    - `private_key = composite_key(private_key_mldsa, private_key_ed25519);`
    - `signature = composite_sig(composite_alg || message,private_key);`

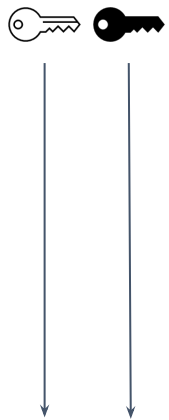


  - So we are considering composites that aim to achieve Weak Non-Separability as per draft-hale-pquip-hybrid-signature-spectrums.

Instantiations in LAMPS CMS, OpenPGP, JOSE/COSE drafts

Un-linked parallel signatures

- LAMPS – CMS
  - RFC 5652 s. 5.1
    ```
    SignerInfos ::= SET OF SignerInfo
    ```
  - Explicitly an OR mode (the client only needs to successfully verify one).
- OpenPGP
  - RFC 4880 (OpenPGP) allows for multiple signature packets.
  - draft-ietf-openpgp-pqc s. 4.4.



Un-linked parallel signatures

RFC 7515 (JSON Web Signatures)

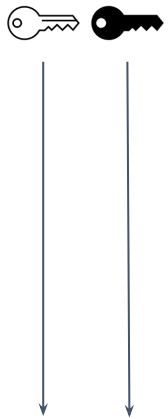
```
{
  "payload":"<payload contents>",
  "signatures":[
  {"protected":"<integrity-protected header 1 contents>",
   "header":<non-integrity-protected header 1 contents>,
   "signature":"<signature 1 contents>"},
...
  {"protected":"<integrity-protected header N contents>",
   "header":<non-integrity-protected header N contents>,
   "signature":"<signature N contents>"}]
}
```

RFC 9052 (CBOR Object Signatures)

```
98([
  / protected / h'',
  / unprotected / {},
  / payload / 'This is the content.',
  / signatures / [
   [
    / protected h'a10126' / <<{/ alg / 1:-7 / ES256 /}>>,
    / unprotected / { / kid / 4:'key-11' },
    / signature / h'e2aeaf…0f30a'
   ],
…
   [
    / protected h'a1013823' / <<{ / alg / 1:-36 / ES512 /}>>,
    / unprotected / { / kid / 4:'key-42' },
    / signature / h'00a2d…7a0297'
   ]
  ]
])
```

Un-linked parallel signatures

- Pro: Mechanism already exists (no new drafts needed).
- Pro: Single algorithm communicates each signature type.
- Pro: Enables keys to be restricted to a single algorithm.
- Pro: Allows easily adding signatures over time.
- Pro: Verifier policy (AND vs OR) can be strengthened over time.



- Con: requires multiple keys
- Con: allows stripping attacks
- Con: Verifier policy AND vs OR complexity

Protocol-layer Linked parallel signatures

- LAMPS – CMS
  - RFC 5752 “Multiple Signatures in CMS”

  “The multiple-signatures attribute type specifies a pointer to a signer's other multiple-signatures attribute(s).”
    - Con: No mandatory verifier policy
- OpenPGP
  - None ?
- JOSE
  - None
- COSE
  - None

Protocol-layer Linked Parallel Signatures

- Mechanism already exists (no new drafts needed).
- Pro: Can be used with single algorithm keys.
- Pro: Allows clients to continue using existing traditional certificates, and add a PQ certificate to it.
- Pro: Provides Weak Non-Separability (Bindel-Hale).

- Con: Multiple keys / certs
- Con: Hybrid security is enforced at protocol level rather than crypto level.

Counter-signatures

- LAMPS – CMS
  - RFC 5652 s. 11.4
    "Countersignature values have the same meaning as SignerInfo values for ordinary signatures, except that:
    3. The input to the message-digesting process is the contents octets of the DER encoding of the signatureValue field of the SignerInfo value with which the attribute is associated."
  - Also, SignedData objects can be nested.
- OpenPGP
  - Nested Signatures & One Pass Signature Packets RFC 4880 Section 5.1
  - You can of course fully nest two OpenPGP MIME messages.
  - **Details? Does a spec or package exist?**
- JOSE / COSE
  - Embedded tokens (OAuth) - draft-yusef-oauth-nested-jwt
  - Counter signatures (COSE) - RFC 9338
  - Receipts - draft-ietf-cose-merkle-tree-proofs
- WIMSE
  - Biscuits? Macaroons?

Counter-signatures

- Mechanism already exists (no new drafts needed).
- Pro: Can be used with single algorithm keys.
- Pro: Allows clients to continue using existing traditional certificates, and add a PQ certificate to it.
- Pro: Allows easily adding signatures over time – ie you can re-sign existing content even without access to the original signing key.

- Con: Requires multiple keys
- Con: Allows stripping attacks (in one direction)
- Con: Does not provide Weak Non-Separability (Bindel-Hale)

Crypto-layer Linked Parallel Signatures (aka composite)

- Can accomplish the “strongest” form of Weak Non-Separability (Bindel-Hale) when signatures bind to the set of algorithms used such that individual signatures will not verify under the individual component algorithms in isolation.
- Nice “protocol-level backwards compatibility” because it’s just a new Alg; implemented the same way across protocols; the crypto lib takes care of it and the protocol layer does not need to implement any hybrid logic.
  - Puts responsibility on crypto lib developers, not protocol or application implementers.
  - Provides a mandatory validation policy for non-repudiation.
- Improve interoperability through normal IETF registry process for (composite) algorithms.
- Can (should!) be done in such a way that the composite keys and signature algorithm primitives can be re-used across protocols.

Protocol Linked Parallel Signatures (aka composite)

- LAMPS – CMS
  - draft-ounsworth-pq-composite-sigs Section 5
    - id-MLDSA65-ECDSA-P256-SHA512
    - (and more)
- OpenPGP
  - draft-ietf-openpgp-pqc Section 3.1
    - ML-DSA-65 + Ed25519
    - (and more)

Summary of existing mechanisms

	LAMPS CMS	OpenPGP	JOSE / COSE
Unlinked Parallel Signatures	✓ RFC5652	✓ RFC 4880	✓ RFC 7518 RFC 9052
Counter-signatures	✓ RFC5652	✓	✓ RFC 9338
Linked Parallel Signatures	✓ RFC5752		
Composite Signatures	I-D exists	I-D exists	

Open Questions
and
Biased Position Statement.
…

Position Statement

- We know that people are starting to deploy lattice schemes, and we know that people want to do that in hybrid.
- We know that hybrids are more complicated to get right than they appear
  - (see: KEM Combiners, Hybrid Signature Spectrums)
- The IETF should provide sound hybrid mechanisms to reduce the amount of “roll-your-own” hybrid mechanisms.

Open questions

- What security properties to various IETF protocols need from hybrid schemes?
- Are we satisfied with the hybrid mechanisms that already exist in CMS, OpenPGP, and JOSE/COSE, or do we need more (ie registering ML-DSA+ECDSA, ML-DSA+RSA composite algorithms)?

Position Statement

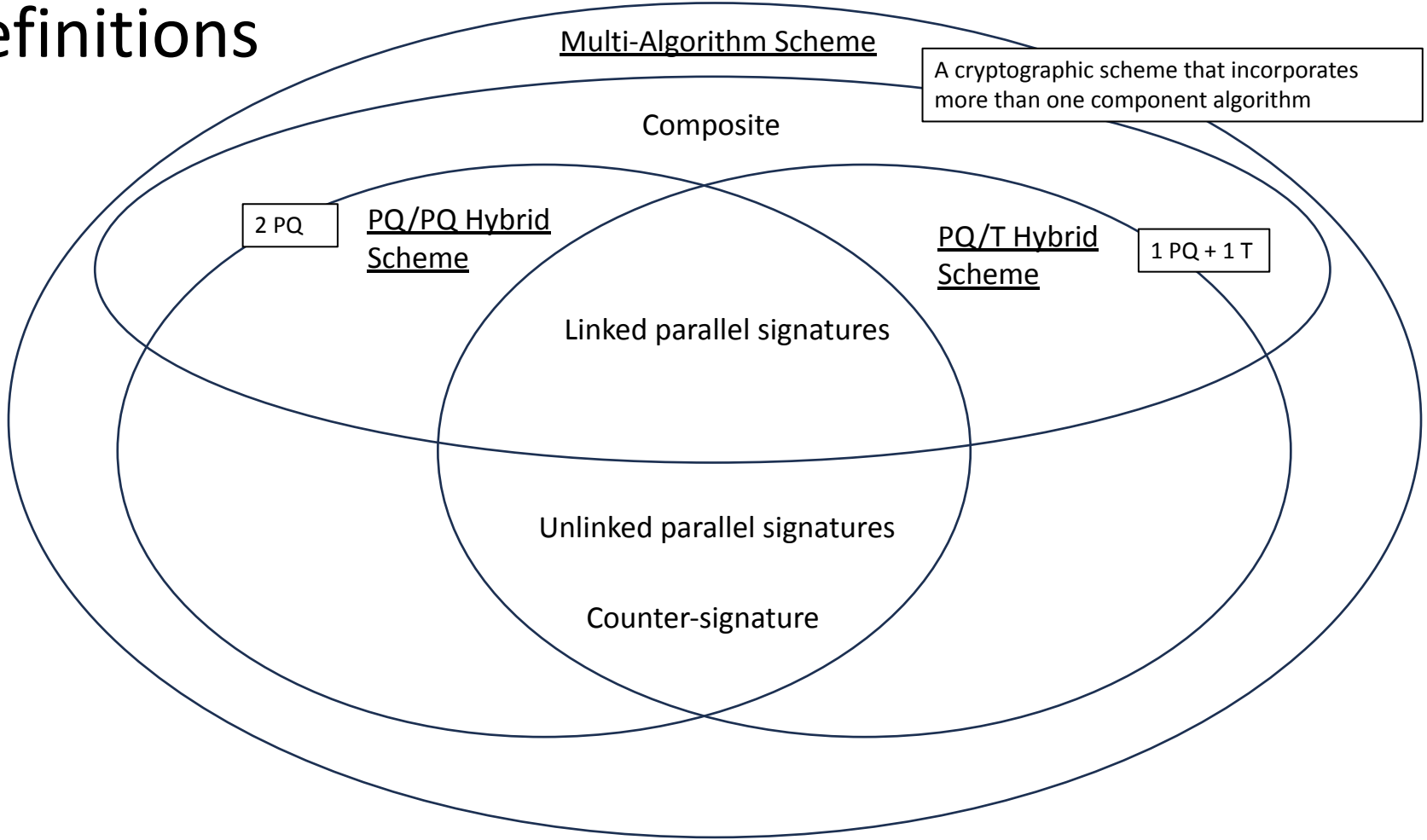
- If hybrids SHOULD or SHOULD NOT happen, CFRG should help answer the “why” and the “with what security properties”.
- If hybrids are happening, protocols SHOULD be able to share crypto libraries, cryptographic module verifications, and security proofs.

Open questions

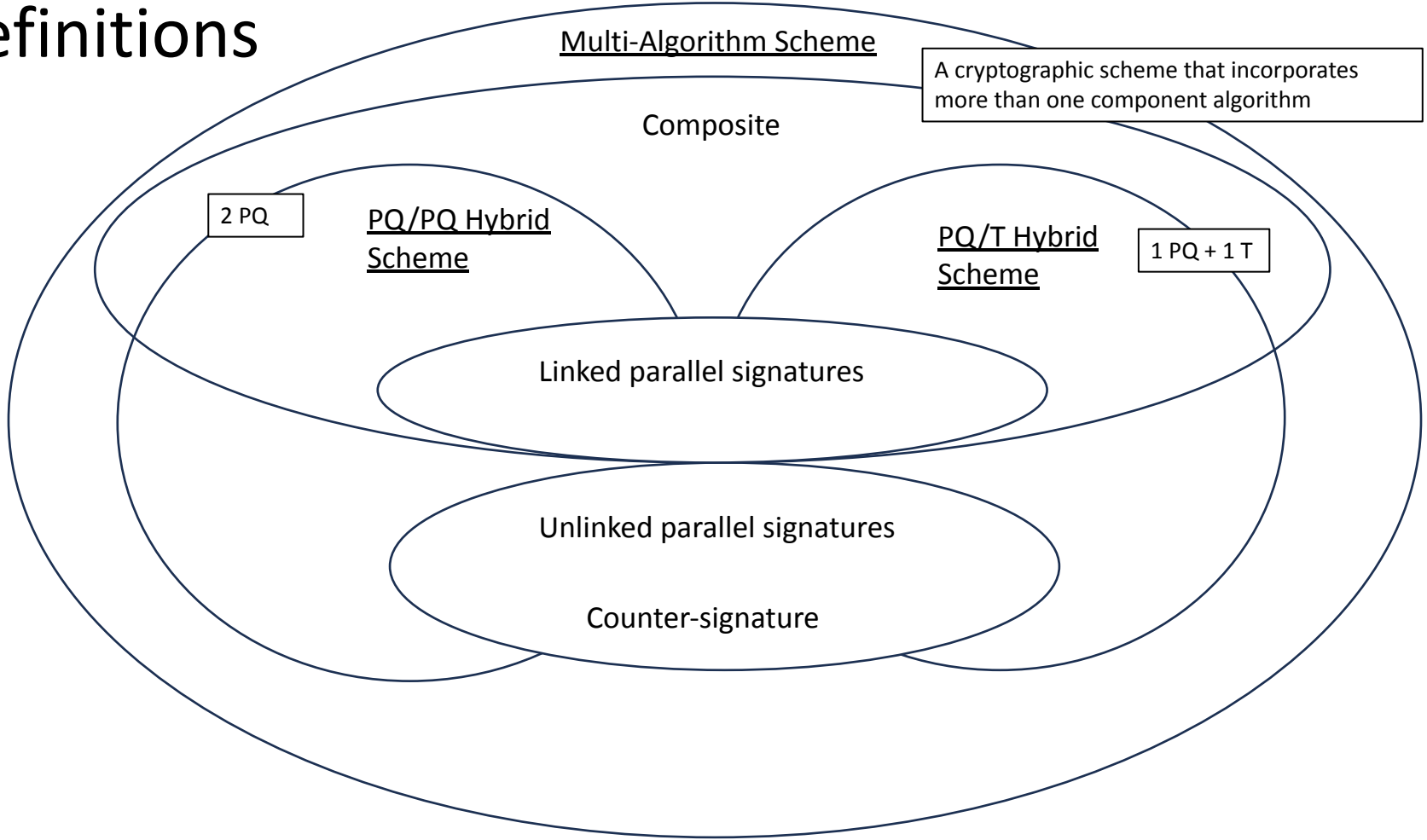
- Should CFRG create a registry for composite signatures that meet minimum security thresholds?

Recycling bin starts here…

Definitions



Definitions



Definitions

- **Composite Linked Parallel Signatures**
  - “A cryptographic element that incorporates multiple component cryptographic elements of the same type in a multi-algorithm scheme.” (draft-ietf-pquip-pqt-hybrid-terminology)
  - Basically, you have one key / algorithm that is actually two keys inside, but we will treat it as if it is one key / algorithm.
  - `private_key = composite(private_key_mldsa, private_key_ed25519);`
  - `public_key = public_from_private(private_key);`
  - `signature = linked_parallel_sign(message, public_key);`
  - AND-mode composites aims to achieve Weak Non-Separability as per draft-hale-pquip-hybrid-signature-spectrums.
  - Instantiations: draft-ietf-lamps-composite-sigs, draft-ietf-openpgp-pqc

Use order dependent signatures?

Yes when…	No when…
● Order is important ● Each signer uses the same TSA ● Commitment to the exact inner signature (not just alg) is desired	● Each signer uses a different TSA ● Verifiers require all signatures

Should also ensure:

- Algorithm NIST bit strength equivalency is the same or
- varies from strong to weak.

Use order independent signatures?

Yes when…	No when…
● Order is not important ● Each signer uses the same TSA ● Algorithms strength is unknown	● Each signer uses a different TSA ● Stripping signatures is a feature

Use independent signatures?

Yes when…	No when…
● Order is not important ● Each signer uses the same TSA ● Algorithms strength is consistent ● Signers represent the same entity	● Each signer uses a different TSA ● Signers represent distinct entities ● Verifiers require all signatures

Do we even need hybrid signatures?

- Ref: https://datatracker.ietf.org/meeting/118/materials/slides-118-pquip-hybrid-signature-spectrums-00, slide 2

Open questions

- Do we actually need hybrid signatures in IETF protocols?
  - We think the answer is yes, but it should be asked.
- Do they need (at least) the Weak Non-Separability property?
  - We think the answer is yes, but it should be asked.
- What about TLS?
  - Should someone write drafts showing what Composite and Multi-sigs would look like for TLS?
  - You probably get linking for free thanks to transcript.

Naming is hard

Ordered Signature Lists	Unordered Signature Sets
● Embedded Signatures - RFC 5126 ● Hierarchical Signatures ● Counter signatures - RFC 9338 and RFC 5652 ● “Receipts” - SCITT / COSE ● “Presentations” - SD-CWT/SD-JWT ● “Proof Chains” - W3C Verifiable Credentials ● “Verifiable Presentations”- ^	● Independent Signatures - RFC 5126 ● Parallel Signatures ● Unlinked multi-sig ● Linked multi-sig ● “Proof Sets” - W3C Verifiable Credentials ● “Verifiable Presentations”- ^

Distinguishing Multitudinous Signatures

“Independent signatures are parallel signatures where the ordering of the signatures is not important. The capability to have more than one independent signature over the same data shall be provided.

Embedded signatures are applied one after the other and are used where the order in which the signatures are applied is important. The capability to sign over signed data shall be provided.”

- Section C.5 of RFC 5126 (2008)

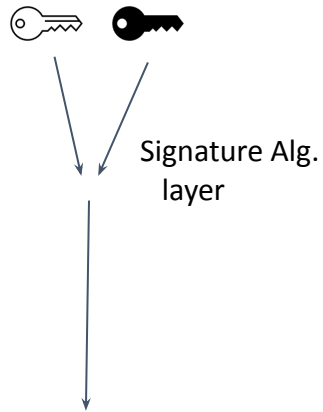
“Where signatures are hierarchical, the order of signing is preserved and each signature covers the preceding signatures. Where signatures are parallel, the signatures attest to the same content, but not the signatures themselves”

- Section 6.3 of Common Format for Information that is Digitally Signed (2001)

TLDR: ordered sets != unordered sets

Definitions

- **Composite Signature from Non-Composite Keys**
  - “A cryptographic element that incorporates multiple component cryptographic elements of the same type in a multi-algorithm scheme.” (draft-ietf-pquip-pqt-hybrid-terminology)
  - Basically, you have one key / algorithm that is actually two keys inside, but we will treat it as if it is one key / algorithm.
  - `signature = composite_sign(message, public_key1, public_key2);`
    - Ex.: an ML-DSA and an RSA key together produce a signature of type ML-DSA_RSA-PSS.
  - Instantiations: draft-ietf-lamps-composite-sigs, draft-ietf-openpgp-pqc

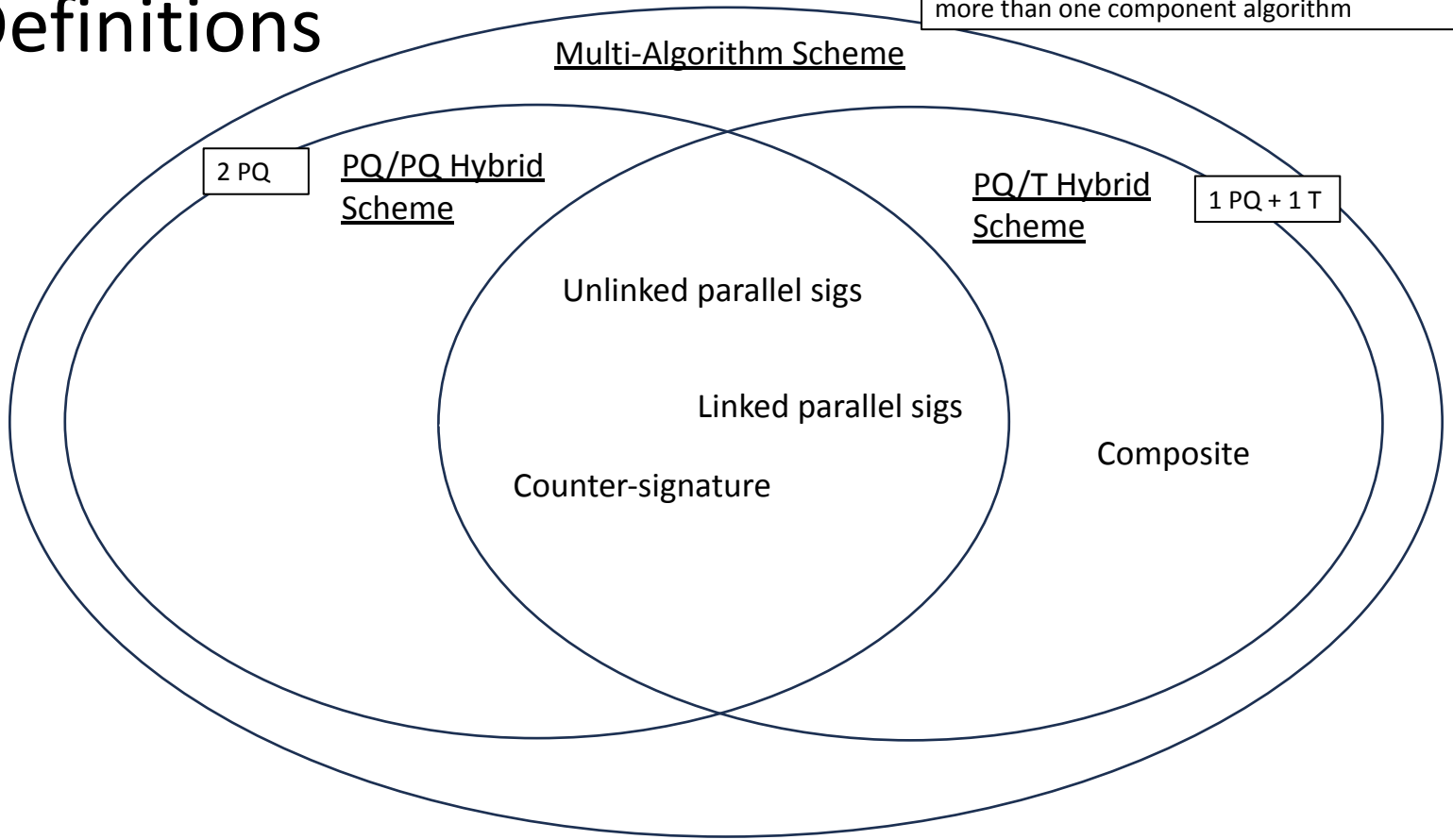


Sequential signatures

- LAMPS – CMS
  - RFC 5652 “CMS”
  - You can arbitrarily nest a SignedData inside another SignedData.
- OpenPGP
  - ??
- JOSE
  - SD-JWT where the directed link is established by “sd_hash” in the “key binding token”.
- COSE
  - “Receipts” where the directed link is established through a “proof type”.
  - Counter signatures, where the directed link is established through nesting.

Definitions

A cryptographic scheme that incorporates more than one component algorithm



Composite Signatures

- JOSE / COSE
  - None that we are aware of.
  - But it could be done like this:

```
98([
  / content type : text/plain; charset=utf-8 /
  h'A10300',
  {},
  h'546869…6E74656E742E',
  [
    [
      / alg: Ed25519 + ML-DSA / h'A10127',
      {/ kid /4: h'31…31'},
      h'77F3EACD11852C4BF9C…3FC9EC106'
    ]
  ]
])
```

* This is not a call for a draft defining Ed25519+ML-DSA for JOSE/COSE.

* This is highlighting that ***composite signatures can be protocol agnostic***.

COSE Order Dependent (Counter Signatures)

```
98([
  / content type : text/plain; charset=utf-8 /
  h'A10300',
  { / counter signature / 9: h'D3AFD…6E0A'},
  h'546869…6E74656E742E',
  [
    [
      / alg: ML-DSA / h'A10127',
      {/ kid /4: h'3131'},
      h'77F3EACD11852C4BF9C…3FC9EC106'
    ]
  ]
])
```