Composite Signature –v10 Updates

Mike Ounsworth, John Gray, Jan Klaussner, Max Pala

Changes affecting Interoperability

- Re-worked the wire format of the composite signature by pre-hashing and concatenating the object identifier (OID) to each component signature. This gives a stronger non-separability property as specified in draft-hale-pquip-hybrid-signature-spectrums-00)
- Removed CompositeSignatureParams, the OIDS now fully specify the component algorithms
  - Removed the redundancy of storing the full subject public key in each composite public key and private key structures.
  - RSA Keys now fixed by the OID (Either 2048 or 3072)
- Added MLDSA44 combinations
- Changed all SEQUENCE OF SIZE (2..MAX) to SEQUENCE OF SIZE (2)

ENTRUST

Composite Signature Generation Process

1. Compute a hash of the Message

   M' = HASH(Message)

2. Generate the n component signatures independently, according to their algorithm specifications.

   $S1 := Sign_{A1}(K1, OID || M')$

   $S2 := Sign_{A2}(K2, OID || M')$

3. Encode each component signature S1 and S2 into a BIT STRING according to its algorithm specification.

   signature ::= SEQUENCE { S1, S2 }

4. Output signature

Where:

K1, K1 are signing private keys for each component algorithm

A1. A2 are the component signing algorithms (for example ML-DSA, RSA, ECDSA or FN-DSA)

ENTRUST

Composite Signature Verification Process

1. Check keys, signatures, and algorithms lists for consistency.

2. Compute a Hash of the Message

    M' = HASH(Message)

3. Check each component signature individually, according to its algorithm specification. If any fail, then the entire signature validation fails.

    if not Verify_{A1}(P1, OID || M', S1) then output "Invalid signature"

    if not Verify_{A2}(P2, OID || M', S2) then output "Invalid signature"

    output "Valid signature"

"applications MUST output "Valid signature" (true) if and only if all component signatures were successfully validated, and "Invalid signature" (false) otherwise."

ENTRUST

Compact Composite Public Key Format

Previous Version:

CompositePublicKey ::= SEQUENCE SIZE (2..MAX) OF SubjectPublicKeyInfo

New Version:

CompositeSignaturePublicKey ::= SEQUENCE SIZE (2) OF BIT STRING

"using BIT STRING allows for easier transcription between CompositeSignaturePublicKey and SubjectPublicKeyInfo"

```
SubjectPublicKeyInfo  ::=  SEQUENCE  {
    algorithm           AlgorithmIdentifier,
    subjectPublicKey    BIT STRING
}
```


ENTRUST

Editorial Changes

The sections of this document align with the adopted Composite KEM draft.

- Made this document standalone by folding in the minimum necessary content from composite-keys
- Added reference to draft-vaira-pquip-pqc-use-cases
- Added a paragraph describing how to reconstitute component SubjectPublkicKeyInfo
- Added a section showing the HEX encoding of the String algorithm names
- Added a section on pre-hashing
- Renamed Dilithium to ML-DSA
- Added an Implementation consideration about FIPS validation when only one component algorithm is FIPS-approved.
- Added a section on Signature APIs (Keygen, Sign, Verify) in introduction

ENTRUST

Recent Comments that will be addressed

- Falko Strenzke's suggested:
  - We make it clear in the text that the component keys in a composite MUST not be used in other keys (either separately or other composites).
  - Regarding this section:

    "**In the absence of an application profile specifying otherwise**, compliant applications MUST output "Valid signature" (true) if and only if all component signatures were successfully validated, and "Invalid signature" (false) otherwise.

    Instead, verification of all signatures should be mandatory without any exception.

ENTRUST

Recent Comments that will be addressed

- During the hackathon we discussed changing the contents of the OID concatenation label which gives the strong non-separability property (draft-hale-pquip-hybrid-signature-spectrums-00)
  - A hash of the DER encoded Object Identifier, or
  - A hash of the DER encoded AlgorithmID.
    - Both ways offer a stronger binding if the composite ObjectID is changed due to tweaks in underlying component algorithms (a signature failure would result if the wrong composite algorithm is used).
    - Encoding the hash of the full composite algorithmID would allow the composite signature algorithm to check if the component keys are reconstructed correctly.
    - Adding a hash of the full composite key (binding to specific keys)
    - The OID values in the table in section 2.4 would become a fixed length
  - Discussion ongoing

ENTRUST

Call for Adoption?

- We have gone through the comments from the failed call for adoption pre-117.
  - We believe we have addressed all objections
    - Clarified scope, collected use-cases and support testimonials which have led to publishing the initial draft of **draft-vaira-pquip-pqc-use-cases-00**
    - Simplified the key format representation and eliminated parameter requirements
    - Strengthened the non-separability property to category 4 concatenation which is discussed further in (**draft-hale-pquip-hybrid-signature-spectrums-00**)

- **Questions:**
  - Are any of the objectors satisfied with these changes and now support adoption?
  - Are any of the objectors still opposed to adoption?

ENTRUST

Thank You

entrust.com

© Entrust Corporation