

COSE HPKE

IETF 125, Shenzhen, March 2026

Hannes Tschofenig, Ori Steele,
Daisuke Ajitomi, Laurence Lundblade, Mike Jones

Overview

- Draft-18 was [presented](#) at IETF-124
- Since then:
 - Multiple draft versions (19–24) were published
 - Draft -24 was published 15th March 2026
 - Three implementations used to validate examples

High Level Summary of Updates

- Uses **Separate algorithm identifiers** for Integrated Encryption vs. Key Encryption
- **Recipient_structure** instead of **COSE_KDF_Context**
- **next_layer_alg** cryptographic binding to next COSE layer
- Clearer handling of HPKE **aad** and **info** parameters
- Clearer handling of **application-provided aad** and **context**
- Tightened **protected vs. unprotected header parameters**
- Better-defined **key representation**
- Updated **examples** and added **test vectors**
- **Mike Jones** added as **co-author**

- *Kept aligned with JOSE HPKE spec*

Algorithm Identifiers

- Introduction of **separate algorithm identifiers** for
 - HPKE Integrated Encryption
 - HPKE Key Encryption
 - *Use fully-specified algorithm identifiers*
- The integrated and key-encryption uses are separated at the algorithm-registration level
- Addition of **HPKE-7 & HPKE-7-KE algorithms** using
 - DHKEM(P-256, HKDF-SHA256), HKDF-SHA256, & AES-256-GCM

Recipient_structure for Key Encryption Mode

Dedicated **Recipient_structure** used as HPKE **info**, instead of reusing **COSE_KDF_Context**

This structure contains

- Context string "HPKE Recipient"
- **next_layer_alg** - Cryptographically binds to the lower COSE layer
- **recipient_protected_header**
- **recipient_extra_info**

Use of **COSE_KDF_Context** now explicitly prohibited

- Security Design Rationale section explains why

Deterministic Encoding Requirements

- **Recipient_structure must be serialized deterministically** using CBOR [RFC 8949] deterministic encoding rules
- Clarifies that this requirement applies to **Recipient_structure** and its four members, but **does not apply to the byte-string-wrapped protected header content**
 - Important implementation detail

AAD Handling

For **Integrated Encryption**

- HPKE `aad` must contain the serialized COSE `Enc_structure`
- Context string is "`Encrypt0`"
- External AAD must in the COSE `external_aad` field

For **Key Encryption**

- HPKE `info` is deterministic encoding of `Recipient_structure`
- External context goes into `recipient_extra_info`
- Recipient aad is generally **not necessary**
- Large external data should instead be protected at layer 0 using `external_aad`

Header Processing Rules

Specification sharpens the rules around header parameters

- **ek** must carry the encapsulated key and be in the **unprotected headers**
- If **alg** is present in a COSE_Recipient, it **must** be a COSE-HPKE ciphersuite and **must be protected**
- Use of **kid** is explicitly recommended to identify the recipient's static public key
- Placing **kid** in the protected header ensures it is covered by HPKE's key derivation context

Key Representation

If `kty = AKP`, then the key material must be the **raw HPKE key** for the corresponding KEM

Validity rules for `kty/crv` combinations

- Private keys may only use `key_ops = derive bits`
- Public keys must have an empty `key_ops` set, if present

Examples and Test Vectors

- All examples updated due to the normative changes
- Example inputs have been simplified, for example, using empty `external aad` and empty `external info`
- Recipient identifiers such as "`bob`" are made explicit
- New Appendix C contains test vectors
- Implementations by [Orie](#), [Daisuke](#) and [Hannes](#) used for test vector
 - **We would be happy to test with your implementation as well!**

Specification is ready for publication!

Note that JOSE HPKE publication was requested last week