JOSE
Internet-Draft
Updates: 7516 (if approved)
Intended status: Standards Track
Expires: 5 April 2026

T. Reddy
Nokia
H. Tschofenig
H-BRS
A. Banerjee
Nokia
O. Steele
Tradeverifyd
M. Jones
Self-Issued Consulting

2 October 2025

Use of Hybrid Public Key Encryption (HPKE) with JSON Object Signing and Encryption (JOSE)
draft-ietf-jose-hpke-encrypt-12

Abstract

This specification defines Hybrid Public Key Encryption (HPKE) for use with JSON Object Signing and Encryption (JOSE). HPKE offers a variant of public key encryption of arbitrary-sized plaintexts for a recipient public key.

HPKE is a general encryption framework utilizing an asymmetric key encapsulation mechanism (KEM), a key derivation function (KDF), and an Authenticated Encryption with Associated Data (AEAD) algorithm.

This document defines the use of HPKE with JOSE. The specification chooses a specific subset of the HPKE features to use with JOSE.

About This Document

This note is to be removed before publishing as an RFC.

The latest revision of this draft can be found at https://ietf-wg-jose.github.io/draft-ietf-jose-hpke-encrypt/draft-ietf-jose-hpke-encrypt.html. Status information for this document may be found at https://datatracker.ietf.org/doc/draft-ietf-jose-hpke-encrypt/.

Discussion of this document takes place on the jose Working Group mailing list (mailto:jose@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/jose/. Subscribe at https://www.ietf.org/mailman/listinfo/jose/.

Source for this draft and an issue tracker can be found at https://github.com/ietf-wg-jose/draft-ietf-jose-hpke-encrypt.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

Hybrid Public Key Encryption (HPKE) [RFC9180] is a scheme that provides public key encryption of arbitrary-sized plaintexts given a recipient's public key.

This specification enables JSON Web Encryption (JWE) to leverage HPKE, bringing support for KEMs and the possibility of Post Quantum or Hybrid KEMs to JWE.

2. Conventions and Definitions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. Conventions and Terminology

This specification uses the following abbreviations and terms:

* Content Encryption Key (CEK), is defined in [RFC7517].

* Hybrid Public Key Encryption (HPKE) is defined in [RFC9180].

* pkR is the public key of the recipient, as defined in [RFC9180].

* skR is the private key of the recipient, as defined in [RFC9180].

* Key Encapsulation Mechanism (KEM), see [RFC9180].

* Key Derivation Function (KDF), see [RFC9180].

* Authenticated Encryption with Associated Data (AEAD), see [RFC9180].

* Additional Authenticated Data (AAD), see [RFC9180].

4. Overview

This specification defines two modes of use for HPKE in JWE:

* HPKE JWE Integrated Encryption, where HPKE is used to encrypt the plaintext.

* HPKE JWE Key Encryption, where HPKE is used to encrypt a content encryption key (CEK) and the CEK is subsequently used to encrypt the plaintext.

When "alg" is a JOSE-HPKE algorithm:

* If "enc" is "int", HPKE JWE Integrated Encryption is used.

* If "enc" is an AEAD algorithm, the recipient Key Management mode is Key Encryption.

The HPKE KEM, KDF, and AEAD used depend on the JOSE-HPKE algorithm used.

HPKE supports several modes, which are described in Table 1 of [RFC9180].

In JOSE-HPKE, only "mode_base" and "mode_psk" are supported. When "psk_id" JOSE Header parameter is present the mode is "mode_psk", otherwise the mode is "mode_base".

JWE supports different serializations, including Compact JWE Serialization as described in Section 3.1 of [RFC7516], General JWE JSON Serialization as described in Section 3.2 of [RFC7516].

Certain JWE features are only supported in specific serializations.

For example Compact JWE Serialization does not support the following:

* additional authenticated data

* multiple recipients

* unprotected headers

HPKE JWE Key Encryption can be used with "aad" but only when not expressed with Compact JWE Serialization.

Single recipient HPKE JWE Key Encryption with no "aad" can be expressed in Compact JWE Serialization, so long as the recipient and sender use the same HPKE Setup process as described in Section 5 of [RFC9180].

This specification updates the "enc" definition in Section 4.1.2 of [RFC7516] by allowing the "enc" value "int" when the "alg" value is a JOSE-HPKE algorithm. When "alg" is not a JOSE-HPKE algorithm and the "enc" value is "int", the input MUST be rejected.

4.1. Auxiliary Authenticated Application Information

The HPKE "aad parameter" for Open() and Seal() specified in Section 8.1 of [RFC9180] is used with both HPKE JWE Integrated Encryption and HPKE JWE Key Encryption. Its value is the Additional Authenticated Data encryption parameter value computed in Step 14 of Section 5.1 of [RFC7518] (Message Encryption).

Despite similarities to ECDH-ES, this specification does not use the apu and apv header parameters, which are described in Section 4.6.1 of [RFC7518].

4.2. Encapsulated Keys

HPKE encapsulated key is defined in Section 5.1.1 of [RFC9180].

In HPKE JWE Integrated Encryption, the JWE Encrypted Key of the sole recipient is the HPKE encapsulated key.

In HPKE JWE Key Encryption, each recipient JWE Encrypted Key is the encrypted content encryption key, and the value of JOSE Header parameter "ek" is base64url-encoded HPKE encapsulated key.

5. Integrated Encryption

In HPKE JWE Integrated Encryption:

* The protected header MUST contain an "alg" that is JOSE-HPKE algorithm.

* The protected header MUST contain an "enc" with value "int". This is an explicit exception to requirement in Section 4.1.2 of [RFC7516] that "enc" must be an AEAD algorithm. This is appropriate, as HPKE will perform plaintext encryption.

* The protected header parameters "psk_id" MAY be present.

* The protected header parameter "ek" MUST NOT be present.

* There MUST be exactly one recipient.

* The JWE Encrypted Key MUST be encapsulated key, as defined in Section 5.1.1 of [RFC9180].

* The JWE Initialization Vector and JWE Authentication Tag MUST be the empty octet sequence.

* The JWE AAD MAY be present when using the JWE JSON Serialization.

* The JWE Ciphertext is the ciphertext defined in Section 5.2 of [RFC9180].

* The HPKE info parameter defaults to the empty string; mutually known private information MAY be used instead. The concept of mutually known private information is defined in [NIST.SP.800-56Ar3] as an input to the key derivation function.

* The HPKE aad parameter MUST be set to the "Additional Authenticated Data encryption parameter", as specified in Step 14 of Section 5.1 of [RFC7516].

* Then follow Steps 11-19 of Section 5.1 of [RFC7516] (Message Encryption).

When decrypting, the checks in Section 5.2 of [RFC7516], Steps 1 through 5 MUST be performed. The JWE Encrypted Key in Step 2 is the base64url-encoded encapsulated key.

5.1. Compact Example

Below is an example of a Compact JWE using HPKE integrated encryption:


```
=============== NOTE: '\' line wrapping per RFC 8792 ================

eyJhbGciOiAiSFBLRS0wIiwgImVuYyI6ICJpbnQiLCAia2lkIjogIkc1Tl9fQ3FNdl9r\
SkdpZUdTRnVBdWd2bDBqclFKQ1ozeUt3Vks2c1VNNG8ifQ.BIh6I40uiBbK8-\
UK7nHdo3ISEfgwJ_MF3zWjQzLt00GhFF2-\
1VgWKHSYLXdeVeRV7AinyocYiCYmISvW0yqiDmc..Ov-\
                                        llz6VUyiw8nZL0OPGLGZckLTm5UcTZFg.
```


The keys used for this example are in Appendix A.

6. Key Encryption

When using the JWE JSON Serialization, recipients using JOSE-HPKE can be added alongside other recipients (e.g., those using ECDH-ES+A128KW or RSA-OAEP-256), since HPKE is used to encrypt the Content Encryption Key, which is then processed as specified in JWE.

The encoding of the protected header remains consistent with existing JWE rules.

In HPKE JWE Key Encryption:

* The Key Management Mode is Key Encryption.

* When all recipients use the same HPKE algorithm to secure the Content Encryption Key, the JWE Protected Header SHOULD contain "alg". Otherwise, the JWE Protected Header (and JWE Shared Unprotected Header) MUST NOT contain "alg".

* JOSE Header parameter "alg" MUST be a JOSE-HPKE algorithm.

* JOSE Header parameter "psk_id" MAY be present.

* JOSE Header parameter "ek" MUST be present and contain the base64url-encoded HPKE encapsulated key.

* Recipient JWE Encrypted Key MUST be the ciphertext from HPKE Encryption.

* The HPKE info parameter contains the encoding of the Recipient_structure, which is described in Section 6.1.

* The HPKE AAD parameter defaults to the empty string; externally provided information MAY be used instead.

* THE HPKE plaintext MUST be set to the CEK.

The processing of "enc", "iv", "tag", "aad", and "ciphertext" is as already defined in [RFC7516]. Implementations process these parameters as defined in [RFC7516]; no additional processing requirements are introduced by HPKE-based key encryption.

6.1. Recipient_structure

The Recipient_structure is a JSON object with the following members:

* context (string): This member MUST include the constant string value "JOSE HPKE Recipient".

* next_layer_alg (string): Identifies the algorithm with which the HPKE-encrypted key MUST be used. Its value MUST match the "enc" (encryption algorithm) header parameter in the JWE protected header. This field is included for alignment with the COSE HPKE [I-D.ietf-cose-hpke] specification. Currently, there are no known attacks that allow a downgrade attack of the content encryption algorithm.

* recipient_protected_header (object): This member contains the base64url-encoded JWE Per-Recipient Unprotected Header (see JWE JSON Serialization in Section 7.1 of [RFC7156] of the recipients member. To serialize this header member the procedure from Section 3.3 of RFC 7638 MUST be used. Unlike with RFC 7638, all members from this member are included except for the "ek" member. The inclusion of this data in the Recipient_structure allows context information to be included in the key derivation.

* recipient_extra_info (string): Contains additional context information that the application includes in the key derivation via the HPKE info parameter. Mutually known private information, which is defined in [NIST.SP.800-56Ar3], MAY be used in this input parameter. If no additional context is provided, this value MUST be the empty string "".

6.1.1. Deterministic Serialization for HPKE info

JSON texts that are semantically identical can serialize differently (e.g., member order, whitespace), which would lead to divergent info values and failed key agreement.

To produce the HPKE info byte string from a Recipient_structure, both sides MUST produce the deterministic JSON representation using the JSON Web Key (JWK) Thumbprint serialization rules [RFC7638]:

1. Construct the Recipient_structure JSON object exactly as defined Section 6.1.

2. Prepare the JSON structure based on Section 3.3 of RFC 7638.

3. Use the resulting JSON structure, base64url-encode it and use the octets as the HPKE info value.

6.1.1.1. Example

The example below shows a pretty-printed JSON object with an empty recipient_extra_info member.

{{::include-fold examples/recipient_structure_example.txt}}

The serialized JSON leads to:

{{::include-fold examples/serialization_example1.txt}}

The base64url-encoded JSON structure above is used as the HPKE info bytes.

6.2. JSON Example

Below is an example of a JWE using the JSON Serialization and HPKE key encryption:

```
=============== NOTE: '\' line wrapping per RFC 8792 ================

{
  "protected": "eyJlbmMiOiAiQTEyOEdDTSJ9",
  "ciphertext": "9AxOd65ROJY1cQ",
  "iv": "2u3NRi3CSr-x7Wuj",
  "tag": "1NKYSWVV4pw5thsq7t6m6Q",
  "recipients": [
    {
      "encrypted_key": "l9VRW1K5CA037fY2ZqVF4bDej413TaAtfjoe3k89-eI",
      "header": {
        "alg": "HPKE-0",
        "kid": "G5N__CqMv_kJGieGSFuAugvl0jrQJCZ3yKwVK6sUM4o",
        "ek": "BJl0V6KLl3HOAZbzFwiAL9eaYbFQPg7-\
             ROmIJpluIQjNS5zultZsC4rGhGzmW1GUWG8bzJUWLQtxFF9oze0AKhU"
      }
    }
  ]
}
```

The keys used for this example are in Appendix A.

7. Mapping HPKE Keys to JWK for JOSE

JWKs can be used to represent JOSE-HPKE private or public keys. For the algorithms defined in this document, the valid combinations of the JWE Algorithm, "kty", and "crv" are shown in Figure 1.

JWE Algorithm	JWK kty	JWK crv
HPKE-0	EC	P-256
HPKE-1	EC	P-384
HPKE-2	EC	P-521
HPKE-3, HPKE-4	OKP	X25519
HPKE-5, HPKE-6	OKP	X448

Figure 1: JWK Types and Curves for JOSE-HPKE Ciphersuites

When the "kty" field is "AKP" (Algorithm Key Pair [I-D.ietf-cose-dilithium]) and "alg" is a JOSE-HPKE algorithm, the public and private keys MUST be raw HPKE public and private keys (respectively) for the KEM used by HPKE.

7.1. JWK Representation of a JOSE-HPKE Key with HPKE Ciphersuite

The example below is a JWK representation of a JOSE-HPKE public and private key:

```
{
  "kty": "OKP",
  "crv": "X25519",
  "x": "3pPHgcHYVYpOpB6ISwHdoPRB6jNgd8mM4nRyyj4H3aE",
  "d": "nWGxne0tAiV8Hk6kcy4rN0wMskjl9yND0N3Xeho9n6g",
  "kid": "recipient-key-1",
  "alg": "HPKE-3",
  "key_ops": "encrypt"
}
```

It uses the "key_ops" value of "encrypt", which is appropriate when using integrated encryption.

8. Security Considerations

This specification is based on HPKE and the security considerations of [RFC9180] are therefore applicable also to this specification.

HPKE assumes the sender is in possession of the public key of the recipient and HPKE JOSE makes the same assumptions. Hence, some form of public key distribution mechanism is assumed to exist but outside the scope of this document.

HPKE in Base mode does not offer authentication as part of the HPKE KEM.

HPKE relies on a source of randomness being available on the device. In Key Agreement with Key Wrapping mode, the CEK has to be randomly generated. The guidance on randomness in [RFC4086] applies.

8.1. Key Management

A single KEM key should not be used with multiple algorithms. Each key and its associated algorithm suite, comprising the KEM, KDF, and AEAD, should be managed independently. This separation prevents unintended interactions or vulnerabilities between algorithms, ensuring the integrity and security guarantees of each algorithm are preserved. Additionally, the same key should not be used for both key encryption and integrated encryption, as it may introduce security risks. It creates algorithm confusion, increases the potential for key leakage, cross-suite attacks, and improper handling of the key.

8.2. Static Asymmetric Authentication in HPKE

Authenticated KEMs based on static asymmetric key authentication are not supported in JOSE HPKE for the following reasons:

* The security implications vary depending on whether they are applied to Key Encryption or Integrated Encryption. When used for Key Encryption, authenticated KEMs offer little meaningful security benefit and may give a false impression of data origin authentication.

* Authenticated KEMs are susceptible to Key-Compromise Impersonation (KCI) attacks. If the sender’s static private key is compromised, an attacker can generate ciphertexts that the recipient will accept as authentic, compromising message integrity.

8.3. Review JWT Best Current Practices

The guidance in [RFC8725] about encryption is also pertinent to this specification.

9. Ciphersuite Registration

This specification registers a number of ciphersuites for use with HPKE. A ciphersuite is a group of algorithms, often sharing component algorithms such as hash functions, targeting a security level. A JOSE-HPKE algorithm makes choices for the following HPKE parameters:

* KEM Algorithm

* KDF Algorithm

* AEAD Algorithm

The "KEM", "KDF", and "AEAD" values are chosen from the IANA HPKE registry [IANA.HPKE].

All JOSE-HPKE algorithm identifiers registered by this specification begin with the string "HPKE-". Future JOSE-HPKE ciphersuite names registered MUST also follow this convention.

10. IANA Considerations

10.1. JSON Web Signature and Encryption Algorithms

The following entries are added to the IANA "JSON Web Signature and Encryption Algorithms" registry [IANA.JOSE]:

10.1.1. HPKE-0

- Algorithm Name: HPKE-0
- Algorithm Description: Cipher suite for JOSE-HPKE using the DHKEM(P-256, HKDF-SHA256) KEM, the HKDF-SHA256 KDF and the AES-128-GCM AEAD
- Algorithm Usage Location(s): "alg"
- JOSE Implementation Requirements: Optional
- Change Controller: IETF
- Specification Document(s): Section 7 of this specification
- Algorithm Analysis Documents(s): [RFC9180]

10.1.2. HPKE-1

- Algorithm Name: HPKE-1
- Algorithm Description: Cipher suite for JOSE-HPKE using the DHKEM(P-384, HKDF-SHA384) KEM, the HKDF-SHA384 KDF, and the AES-256-GCM AEAD
- Algorithm Usage Location(s): "alg"
- JOSE Implementation Requirements: Optional
- Change Controller: IETF
- Specification Document(s): Section 7 of this specification
- Algorithm Analysis Documents(s): [RFC9180]

10.1.3. HPKE-2

- Algorithm Name: HPKE-2
- Algorithm Description: Cipher suite for JOSE-HPKE using the DHKEM(P-521, HKDF-SHA512) KEM, the HKDF-SHA512 KDF, and the AES-256-GCM AEAD
- Algorithm Usage Location(s): "alg"
- JOSE Implementation Requirements: Optional

* Change Controller: IETF

* Specification Document(s): Section 7 of this specification

* Algorithm Analysis Documents(s): [RFC9180]

10.1.4. HPKE-3

* Algorithm Name: HPKE-3

* Algorithm Description: Cipher suite for JOSE-HPKE using the DHKEM(X25519, HKDF-SHA256) KEM, the HKDF-SHA256 KDF, and the AES-128-GCM AEAD

* Algorithm Usage Location(s): "alg"

* JOSE Implementation Requirements: Optional

* Change Controller: IETF

* Specification Document(s): Section 7 of this specification

* Algorithm Analysis Documents(s): [RFC9180]

10.1.5. HPKE-4

* Algorithm Name: HPKE-4

* Algorithm Description: Cipher suite for JOSE-HPKE using the DHKEM(X25519, HKDF-SHA256) KEM, the HKDF-SHA256 KDF, and the ChaCha20Poly1305 AEAD

* Algorithm Usage Location(s): "alg"

* JOSE Implementation Requirements: Optional

* Change Controller: IETF

* Specification Document(s): Section 7 of this specification

* Algorithm Analysis Documents(s): [RFC9180]

10.1.6. HPKE-5

* Algorithm Name: HPKE-5

* Algorithm Description: Cipher suite for JOSE-HPKE using the DHKEM(X448, HKDF-SHA512) KEM, the HKDF-SHA512 KDF, and the AES-256-GCM AEAD

* Algorithm Usage Location(s): "alg"

* JOSE Implementation Requirements: Optional

* Change Controller: IETF

* Specification Document(s): Section 7 of this specification

* Algorithm Analysis Documents(s): [RFC9180]

10.1.7. HPKE-6

* Algorithm Name: HPKE-6

* Algorithm Description: Cipher suite for JOSE-HPKE using the DHKEM(X448, HKDF-SHA512) KEM, the HKDF-SHA512 KDF, and the ChaCha20Poly1305 AEAD

* Algorithm Usage Location(s): "alg"

* JOSE Implementation Requirements: Optional

* Change Controller: IETF

* Specification Document(s): Section 7 of this specification

* Algorithm Analysis Documents(s): [RFC9180]

10.1.8. int

* Algorithm Name: int

* Algorithm Description: Indicates that HPKE Integrated Encryption is being used

* Algorithm Usage Location(s): "enc"

* JOSE Implementation Requirements: Required

* Change Controller: IETF

* Specification Document(s): Section 4 of this specification

* Algorithm Analysis Documents(s): [RFC9180]

10.2. JSON Web Signature and Encryption Header Parameters

The following entries are added to the IANA "JSON Web Key Parameters" registry [IANA.JOSE]:

10.2.1. ek

* Header Parameter Name: "ek"
* Header Parameter Description: A base64url-encoded encapsulated key, as defined in Section 5.1.1 of [RFC9180]
* Header Parameter Usage Location(s): JWE
* Change Controller: IETF
* Specification Document(s): Section 4.2 of this specification

10.2.2. psk_id

* Header Parameter Name: "psk_id"
* Header Parameter Description: A base64url-encoded key identifier (kid) for the pre-shared key, as defined in Section 5.1.2 of [RFC9180]
* Header Parameter Usage Location(s): JWE
* Change Controller: IETF
* Specification Document(s): Section 4 of this specification

11. References

11.1. Normative References

[IANA.JOSE] IANA, "JSON Web Signature and Encryption Algorithms", n.d., <https://www.iana.org/assignments/jose>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/rfc/rfc2119>.

[RFC7156] Zorn, G., Wu, Q., and J. Korhonen, "Diameter Support for Proxy Mobile IPv6 Localized Routing", RFC 7156, DOI 10.17487/RFC7156, April 2014, <https://www.rfc-editor.org/rfc/rfc7156>.

[RFC7516] Jones, M. and J. Hildebrand, "JSON Web Encryption (JWE)", RFC 7516, DOI 10.17487/RFC7516, May 2015, <https://www.rfc-editor.org/rfc/rfc7516>.

[RFC7517] Jones, M., "JSON Web Key (JWK)", RFC 7517, DOI 10.17487/RFC7517, May 2015, <https://www.rfc-editor.org/rfc/rfc7517>.

[RFC7518] Jones, M., "JSON Web Algorithms (JWA)", RFC 7518, DOI 10.17487/RFC7518, May 2015, <https://www.rfc-editor.org/rfc/rfc7518>.

[RFC7638] Jones, M. and N. Sakimura, "JSON Web Key (JWK) Thumbprint", RFC 7638, DOI 10.17487/RFC7638, September 2015, <https://www.rfc-editor.org/rfc/rfc7638>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/rfc/rfc8174>.

[RFC8725] Sheffer, Y., Hardt, D., and M. Jones, "JSON Web Token Best Current Practices", BCP 225, RFC 8725, DOI 10.17487/RFC8725, February 2020, <https://www.rfc-editor.org/rfc/rfc8725>.

[RFC9180] Barnes, R., Bhargavan, K., Lipp, B., and C. Wood, "Hybrid Public Key Encryption", RFC 9180, DOI 10.17487/RFC9180, February 2022, <https://www.rfc-editor.org/rfc/rfc9180>.

11.2. Informative References

[I-D.ietf-cose-dilithium] Prorock, M. and O. Steele, "ML-DSA for JOSE and COSE", Work in Progress, Internet-Draft, draft-ietf-cose-dilithium-09, 12 September 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-cose-dilithium-09>.

[I-D.ietf-cose-hpke]
Tschofenig, H., Steele, O., Daisuke, A., and L. Lundblade, "Use of Hybrid Public-Key Encryption (HPKE) with CBOR Object Signing and Encryption (COSE)", Work in Progress, Internet-Draft, draft-ietf-cose-hpke-16, 19 September 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-cose-hpke-16>.

[IANA.HPKE]
IANA, "Hybrid Public Key Encryption (HPKE)", n.d., <https://www.iana.org/assignments/hpke>.

[NIST.SP.800-56Ar3]
National Institute of Standards and Technology, "Recommendation for Pair-Wise Key-Establishment Schemes Using Discrete Logarithm Cryptography, NIST Special Publication 800-56A Revision 3", April 2018, <https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-56Ar3.pdf>.

[RFC4086] Eastlake 3rd, D., Schiller, J., and S. Crocker, "Randomness Requirements for Security", BCP 106, RFC 4086, DOI 10.17487/RFC4086, June 2005, <https://www.rfc-editor.org/rfc/rfc4086>.

Appendix A. Keys Used in Examples

This private key and its implied public key are used the examples:

```
{
  "kty": "EC",
  "use": "enc",
  "alg": "HPKE-0",
  "kid": "G5N__CqMv_kJGieGSFuAugvl0jrQJCZ3yKwVK6sUM4o",
  "crv": "P-256",
  "x": "gixQJ0qg4Ag-6HSMaIEDL_zbDhoXavMyKlmdn__AQVE",
  "y": "ZxTgRLWaKONCL_GbZKLNPsW9EW6nBsN4AwQGEFAFFbM",
  "d": "g2DXtKapi2oN2zL_RCWX8D4bWURHCKN2-ZNGC05ZaR8"
}
```

Acknowledgments

This specification leverages text from [I-D.ietf-cose-hpke]. We would like to thank Matt Chanda, Ilari Liusvaara, Neil Madden, Aaron Parecki, Filip Skokan, and Sebastian Stenzel for their contributions to the specification.

Document History

-12

* Added the recipient_structure

-11

* Fix too long lines

-10

* Addressed WGLC review comments by Neil Madden and Sebastian Stenzel.

-09

* Corrected examples.

-08

* Use "enc":"int" for integrated encryption.

* Described reasons for excluding authenticated HPKE.

* Stated that mutually known private information MAY be used as the HPKE info value.

-07

* Clarifications

-06

* Remove auth mode and auth_kid from the specification.

* HPKE AAD for JOSE HPKE Key Encryption is now empty.

-05

* Removed incorrect text about HPKE algorithm names.

* Fixed #21: Comply with NIST SP 800-227 Recommendations for Key-Encapsulation Mechanisms.

* Fixed #19: Binding the Application Context.

* Fixed #18: Use of apu and apv in Recipient context.

* Added new Section 7.1 (Authentication using an Asymmetric Key).

* Updated Section 7.2 (Key Management) to prevent cross-protocol attacks.

* Updated HPKE Setup info parameter to be empty.

* Added details on HPKE AEAD AAD, compression and decryption for HPKE Integrated Encryption.

-04

* Fixed #8: Use short algorithm identifiers, per the JOSE naming conventions.

-03

* Added new section 7.1 to discuss Key Management.

* HPKE Setup info parameter is updated to carry JOSE context-specific data for both modes.

-02

* Fixed #4: HPKE Integrated Encryption "enc: dir".

* Updated text on the use of HPKE Setup info parameter.

* Added Examples in Sections 5.1, 5.2 and 6.1.

* Use of registered HPKE "alg" value in the recipient unprotected header for Key Encryption.

-01

* Apply feedback from call for adoption.

* Provide examples of auth and psk modes for JSON and Compact Serializations

* Simplify description of HPKE modes

* Adjust IANA registration requests

* Remove HPKE Mode from named algorithms

* Fix AEAD named algorithms

-00

* Created initial working group version from draft-rha-jose-hpke-encrypt-07

Authors' Addresses

Tirumaleswar Reddy
Nokia
Bangalore
Karnataka
India
Email: kondtir@gmail.com

Hannes Tschofenig
University of Applied Sciences Bonn-Rhein-Sieg
Germany
Email: hannes.tschofenig@gmx.net

Aritra Banerjee
Nokia
Munich
Germany
Email: aritra.banerjee@nokia.com

Orie Steele
Tradeverifyd
United States
Email: orie@or13.io

Michael B. Jones
Self-Issued Consulting
United States
Email: michael_b_jones@hotmail.com
URI: https://self-issued.info/