Use of Hybrid Public-Key Encryption (HPKE) with Javascript Object Signing and Encryption (JOSE)

Abstract

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

HPKE works for any combination of an asymmetric key encapsulation mechanism (KEM), key derivation function (KDF), and authenticated encryption with additional data (AEAD) function. Authentication for HPKE in JOSE is provided by JOSE-native security mechanisms or by one of the authenticated variants of HPKE.

This document defines the use of the HPKE with JOSE.

About This Document

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

Status information for this document may be found at https://datatracker.ietf.org/doc/draft-rha-jose-hpke/.

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/.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.
Table of Contents

1. Introduction
2. Conventions and Definitions
3. Conventions and Terminology
4. HPKE for JOSE
   4.1. Overview
   4.1.1. HPKE Usage in Direct and Key Agreement with Key Wrapping
5. Ciphersuite Registration
6. HPKE Encryption and Decryption
   6.1. HPKE Encryption with SealBase
   6.2. HPKE Decryption with OpenBase
7. Post-Quantum Considerations
   7.1. Example Hybrid Key Agreement Computation
8. Security Considerations
9. IANA Considerations
10. IANA Considerations
   10.1. JOSE Algorithms Registry (Direct Key Agreement)
   10.2. JOSE Algorithms Registry (Key Agreement with Key Wrapping)
   10.3. JOSE Header Parameters
11. References
   11.1. Normative References
   11.2. Informative References
Acknowledgments
Authors' Addresses
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. HPKE utilizes a non-interactive ephemeral-static Diffie-Hellman exchange to establish a shared secret. The motivation for standardizing a public key encryption scheme is explained in the introduction of [RFC9180].

The HPKE specification provides a variant of public key encryption of arbitrary-sized plaintexts for a recipient public key. It also includes three authenticated variants, including one that authenticates possession of a pre-shared key, one that authenticates possession of a key encapsulation mechanism (KEM) private key, and one that authenticates possession of both a pre-shared key and a KEM private key.

This specification utilizes HPKE as a foundational building block and carries the output to JOSE ([RFC7516], [RFC7518]).

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), a term defined in CMS [RFC2630].
*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].
*Key Type (kty), see [RFC7517].
4. **HPKE for JOSE**

4.1. **Overview**

The JSON Web Algorithms (JWA) [RFC7518] in Section 4.6 defines two ways using the key agreement result. When Direct Key Agreement is employed, the shared secret established through the HPKE will be the content encryption key (CEK). When Key Agreement with Key Wrapping is employed, the shared secret established through the HPKE will wrap the CEK. If multiple recipients are needed, then Key Agreement with Key Wrapping mode is used.

In both cases a new JOSE header parameter, called 'encapsulated_key', is used to convey the content of the "enc" structure defined in the HPKE specification. "enc" represents the serialized public key.

When the alg value is set to any of algorithms registered by this specification then the 'encapsulated_key' header parameter **MUST** be present in the unprotected header parameter.

The 'encapsulated_key' parameter contains the encapsulated key, which is output of the HPKE KEM, and is represented as a base64url encoded string. The parameter "kty" **MUST** be present and set to "OKP" defined in Section 2 of [RFC8037].

4.1.1. **HPKE Usage in Direct and Key Agreement with Key Wrapping**

In Direct Key Agreement mode, HPKE is employed to directly encrypt the plaintext, and the resulting ciphertext is included in the JWE ciphertext. In Key Agreement with Key Wrapping mode, HPKE is used to encrypt the Content Encryption Key (CEK), and the resulting ciphertext is included in the JWE ciphertext.

In both modes, the sender **MUST** specify the 'alg' parameter in the protected header to indicate the use of HPKE. Additionally, the sender **MUST** place the 'encapsulated_key' parameter in the unprotected header. Optionally, the unprotected header **MAY** contain the 'kid' parameter used to identify the static recipient public key used by the sender.

5. **Ciphersuite Registration**

This specification registers a number of ciphersuites for use with HPKE. A ciphersuite is thereby a combination of several algorithm configurations:

*HPKE Mode

*KEM algorithm
*KDF algorithm

*AEAD algorithm

The "KEM", "KDF", and "AEAD" values are conceptually taken from the HPKE IANA registry [HPKE-IANA]. Hence, JOSE-HPKE cannot use an algorithm combination that is not already available with HPKE.

For better readability of the algorithm combination ciphersuites, labels are built according to the following scheme:

HPKE-<Mode>-<KEM>-<KDF>-<AEAD>

The "Mode" indicator may be populated with the following values from Table 1 of [RFC9180]:

**"Base"** refers to "mode_base" described in Section 5.1.1 of [RFC9180], which only enables encryption to the holder of a given KEM private key.

**"PSK"** refers to "mode_psk", described in Section 5.1.2 of [RFC9180], which authenticates using a pre-shared key.

**"Auth"** refers to "mode_auth", described in Section 5.1.3 of [RFC9180], which authenticates using an asymmetric key.

**"Auth_Psk"** refers to "mode_auth_psk", described in Section 5.1.4 of [RFC9180], which authenticates using both a PSK and an asymmetric key.

For a list of ciphersuite registrations, please see Section 10.

6. HPKE Encryption and Decryption

6.1. HPKE Encryption with SealBase

The SealBase(pkR, info, aad, pt) function is used to encrypt a plaintext pt to a recipient's public key (pkR).

Two cases of plaintext need to be distinguished:

*In Direct Key Agreement mode, the plaintext "pt" passed into SealBase is the content to be encrypted. Hence, there is no intermediate layer utilizing a CEK.

*In Key Agreement with Key Wrapping mode, the plaintext "pt" passed into SealBase is the CEK. The CEK is a random byte sequence of length appropriate for the encryption algorithm. For example, AES-128-GCM requires a 16 byte key and the CEK would therefore be 16 bytes long.
The "aad" parameter in SealBase function will take the JWE AAD value as input when the JWE AAD value is non-empty; otherwise, it will take an empty aad.

The HPKE specification defines the "info" parameter as a context information structure that is used to ensure that the derived keying material is bound to the context of the transaction. The "info" parameter in SealBase function will take the JOSE context specific data defined in Section 4.6.2 of [RFC7518] as input.

The SealBase function internally creates the sending HPKE context by invoking SetupBaseS() (Section 5.1.1 of [RFC9180]) with "pkR" and "info". This yields the context "sctxt" and an encapsulation key "enc". The SealBase function then invokes the Seal() method on "sctxt" (Section 5.2 of [RFC9180]) with "aad", yielding ciphertext "ct".

In summary, if SealBase() is successful, it will output a ciphertext "ct" and an encapsulated key "enc". In both JWE Compact Serialization and the JWE JSON Serialization, "ct" and "enc" will be base64url encoded, since JSON lacks a way to directly represent arbitrary octet sequences.

6.2. HPKE Decryption with OpenBase

The recipient will use the OpenBase(enc, skR, info, aad, ct) function with the base64url decoded "encapsulated_key" and the "ciphertext" parameters received from the sender. The "aad" and the "info" parameters are constructed from JWE AAD and JOSE context, respectively.

The OpenBase internally creates the receiving HPKE context by invoking SetupBaseR() (Section 5.1.1 of [RFC9180]) with "skR", "enc", and "info". This yields the context "rctxt". The OpenBase function then decrypts "ct" by invoking the Open() method on "rctxt" (Section 5.2 of [RFC9180]) with "aad", yielding "pt" or an error on failure.

The OpenBase function will, if successful, decrypts "ct". When decrypted, the result will be either the CEK (when Key Agreement with Key Wrapping mode is used), or the content (if Direct Key Agreement mode is used). The CEK is the symmetric key used to decrypt the ciphertext.

7. Post-Quantum Considerations

The migration to Post-Quantum Cryptography (PQC) is unique in the history of modern digital cryptography in that neither the traditional algorithms nor the post-quantum algorithms are fully trusted to protect data for the required data lifetimes. The
traditional algorithms, such as RSA and elliptic curve, will fall to quantum cryptanalysis, while the post-quantum algorithms face uncertainty about the underlying mathematics, compliance issues, unknown vulnerabilities, hardware and software implementations that have not had sufficient maturing time to rule out classical cryptanalytic attacks and implementation bugs.

Hybrid key exchange refers to using multiple key exchange algorithms simultaneously and combining the result with the goal of providing security even if all but one of the component algorithms is broken. It is motivated by transition to post-quantum cryptography. HPKE can be extended to support hybrid post-quantum Key Encapsulation Mechanisms (KEMs) as defined in [I-D.westerbaan-cfrg-hpke-xyber768d00-02]. Kyber, which is a KEM does not support the static-ephemeral key exchange that allows HPKE based on DH based KEMs and its optional authenticated modes as discussed in Section 1.2 of [I-D.westerbaan-cfrg-hpke-xyber768d00-02].

The JOSE HPKE PQ/T hybrid ciphersuites are defined in Section 10.

7.1. Example Hybrid Key Agreement Computation

This example uses HPKE-Base-P256-SHA256-AES128GCM which corresponds to the following HPKE algorithm combination:

*KEM: DHKEM(P-256, HKDF-SHA256)

*KDF: HKDF-SHA256

*AEAD: AES-128-GCM

*Mode: Base

*payload: "This is the content"

*aad: ""

```
{
    "header": {
        "alg": "HPKE-Base-P256-SHA256-AES128GCM",
        "kid": "7"
    },
    "encapsulated_key": "BIxvdeRjp3MILzyw06cBNIpXjGeAq6ZYZGaCqa9ykd/Cd+yTw9WHB4GChsEJeCVFczjcPcr/Nn4pUTQunbMNw0c=",
    "ciphertext": "TODO"
}
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. In this case JOSE constructs like JWS and JSON Web Tokens (JWTs) can be used to add authentication. HPKE also offers modes that offer authentication.

HPKE relies on a source of randomness to be available on the device. In Key Agreement with Key Wrapping mode, CEK has to be randomly generated and it MUST be ensured that the guidelines in [RFC8937] for random number generations are followed.

9. IANA Considerations

10. IANA Considerations

This document requests IANA to add new values to the 'JOSE Algorithms' and to the 'JOSE Header Parameters' registries in the 'Standards Action With Expert Review category'.

10.1. JOSE Algorithms Registry (Direct Key Agreement)

*Algorithm Name: HPKE-Base-P256-SHA256-AES128GCM

*Algorithm Description: Cipher suite for JOSE-HPKE version 1 in Base Mode that uses 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: IESG

*Specification Document(s): [[TBD: This RFC]]

*Algorithm Analysis Documents(s): TODO

*Algorithm Name: HPKE-Base-P256-SHA256-ChaCha20Poly1305
Algorithm Description: Cipher suite for JOSE-HPKE version 1 in Base Mode that uses the DHKEM(P-256, HKDF-SHA256) KEM, the HKDF-SHA256 KDF and the ChaCha20Poly1305 AEAD.

Algorithm Usage Location(s): "alg"

JOSE Implementation Requirements: Optional

Change Controller: IESG

Specification Document(s): [[TBD: This RFC]]

Algorithm Analysis Documents(s): TODO

Algorithm Name: HPKE-Base-P384-SHA384-AES256GCM

Algorithm Description: Cipher suite for JOSE-HPKE version 1 in Base Mode that uses 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: IESG

Specification Document(s): [[TBD: This RFC]]

Algorithm Analysis Documents(s): TODO

Algorithm Name: HPKE-Base-P384-SHA384-ChaCha20Poly1305

Algorithm Description: Cipher suite for JOSE-HPKE version 1 in Base Mode that uses the DHKEM(P-384, HKDF-SHA384) KEM, the HKDF-SHA384 KDF, and the ChaCha20Poly1305 AEAD.

Algorithm Usage Location(s): "alg"

JOSE Implementation Requirements: Optional

Change Controller: IESG

Specification Document(s): [[TBD: This RFC]]

Algorithm Analysis Documents(s): TODO

Algorithm Name: HPKE-Base-P521-SHA512-AES256GCM

Algorithm Description: Cipher suite for JOSE-HPKE version 1 in Base Mode that uses 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: IESG

*Specification Document(s): [[TBD: This RFC]]

*Algorithm Analysis Documents(s): TODO

*Algorithm Name: HPKE-Base-P521-SHA512-ChaCha20Poly1305

*Algorithm Description: Cipher suite for JOSE-HPKE version 1 in Base Mode that uses the DHKEM(P-521, HKDF-SHA512) KEM, the HKDF-SHA512 KDF, and the ChaCha20Poly1305 AEAD.

*Algorithm Usage Location(s): "alg"

*JOSE Implementation Requirements: Optional

*Change Controller: IESG

*Specification Document(s): [[TBD: This RFC]]

*Algorithm Analysis Documents(s): TODO

*Algorithm Name: HPKE-Base-X25519-SHA256-AES128GCM

*Algorithm Description: Cipher suite for JOSE-HPKE version 1 in Base Mode that uses 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: IESG

*Specification Document(s): [[TBD: This RFC]]

*Algorithm Analysis Documents(s): TODO

*Algorithm Name: HPKE-Base-X25519-SHA256-ChaCha20Poly1305

*Algorithm Description: Cipher suite for JOSE-HPKE version 1 in Base Mode that uses 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: IESG

*Specification Document(s): [[TBD: This RFC]]

*Algorithm Analysis Documents(s): TODO

*Algorithm Name: HPKE-Base-X448-SHA512-AES256GCM

*Algorithm Description: Cipher suite for JOSE-HPKE version 1 in Base Mode that uses 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: IESG

*Specification Document(s): [[TBD: This RFC]]

*Algorithm Analysis Documents(s): TODO

*Algorithm Name: HPKE-Base-X448-SHA512-ChaCha20Poly1305

*Algorithm Description: Cipher suite for JOSE-HPKE version 1 in Base Mode that uses 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: IESG

*Specification Document(s): [[TBD: This RFC]]

*Algorithm Analysis Documents(s): TODO

*Algorithm Name: HPKE-Base-X25519Kyber768-SHA256-AES256GCM

*Algorithm Description: Cipher suite for JOSE-HPKE version 1 in Base Mode that uses the X25519Kyber768Draft00 Hybrid KEM, the HKDF-SHA256 KDF, and the AES-256-GCM AEAD.

*Algorithm Usage Location(s): "alg"

*JOSE Implementation Requirements: Optional

*Change Controller: IESG

*Specification Document(s): [[TBD: This RFC]]
*Algorithm Name: HPKE-Base-X25519Kyber768-SHA256-ChaCha20Poly1305

*Algorithm Description: Cipher suite for JOSE-HPKE version 1 in Base Mode that uses the X25519Kyber768Draft00 Hybrid KEM, the HKDF-SHA256 KDF, and the ChaCha20Poly1305 AEAD.

*Algorithm Usage Location(s): "alg"

*JOSE Implementation Requirements: Optional

*Change Controller: IESG

*Specification Document(s): [[TBD: This RFC]]

*Algorithm Analysis Documents(s): TODO

*Algorithm Name: HPKE-Base-CP256-SHA256-ChaCha20Poly1305

*Algorithm Description: Cipher suite for JOSE-HPKE version 1 in Base Mode that uses the DHKEM(CP-256, HKDF-SHA256) KEM, the HKDF-SHA256 KDF, and the ChaCha20Poly1305 AEAD.

*Algorithm Usage Location(s): "alg"

*JOSE Implementation Requirements: Optional

*Change Controller: IESG

*Specification Document(s): [[TBD: This RFC]]

*Algorithm Analysis Documents(s): TODO

*Algorithm Name: HPKE-Base-CP384-SHA512-ChaCha20Poly1305

*Algorithm Description: Cipher suite for JOSE-HPKE in Base Mode that uses the DHKEM(CP384, HKDF-SHA512) KEM, the HKDF-SHA512 KDF, and the AES256GCM AEAD.

*Algorithm Usage Location(s): "alg"

*JOSE Implementation Requirements: Optional

*Change Controller: IESG

*Specification Document(s): [[TBD: This RFC]]

*Algorithm Analysis Documents(s): TODO

*Algorithm Name: HPKE-Base-CP521-SHA512-ChaCha20Poly1305
Algorithm Description: Cipher suite for JOSE-HPKE version 1 in Base Mode that uses the DHKEM(CP-521, HKDF-SHA512) KEM, the HKDF-SHA512 KDF, and the ChaCha20Poly1305 AEAD.

Algorithm Usage Location(s): "alg"

JOSE Implementation Requirements: Optional

Change Controller: IESG

Specification Document(s): [[TBD: This RFC]]

Algorithm Analysis Documents(s): TODO

Algorithm Name: HPKE-Base-CP256-SHA256-AES128GCM

Algorithm Description: Cipher suite for JOSE-HPKE in Base Mode that uses the DHKEM(CP-256, HKDF-SHA256) KEM, the HKDF-SHA256 KDF, and the AES128GCM AEAD.

Algorithm Usage Location(s): "alg"

JOSE Implementation Requirements: Optional

Change Controller: IESG

Specification Document(s): [[TBD: This RFC]]

Algorithm Analysis Documents(s): TODO

Algorithm Name: HPKE-Base-CP384-SHA512-AES256GCM

Algorithm Description: Cipher suite for JOSE-HPKE in Base Mode that uses the DHKEM(CP384, HKDF-SHA512) KEM, the HKDF-SHA512 KDF, and the AES256GCM AEAD.

Algorithm Usage Location(s): "alg"

JOSE Implementation Requirements: Optional

Change Controller: IESG

Specification Document(s): [[TBD: This RFC]]

Algorithm Analysis Documents(s): TODO

Algorithm Name: HPKE-Base-CP521-SHA512-AES256GCM

Algorithm Description: Cipher suite for JOSE-HPKE in Base Mode that uses the DHKEM(CP-521, HKDF-SHA512) KEM, the HKDF-SHA512 KDF, and the AES256GCM AEAD.
10.2. JOSE Algorithms Registry (Key Agreement with Key Wrapping)

*Algorithm Name: HPKE-Base-P256-SHA256-AES128GCMKW

*Algorithm Description: Cipher suite for JOSE-HPKE version 1 in Base Mode that uses the DHKEM(P-256, HKDF-SHA256) KEM, the HKDF-SHA256 KDF and Key wrapping with the AES-128-GCM AEAD.

*Algorithm Usage Location(s): "alg"

*JOSE Implementation Requirements: Optional

*Change Controller: IESG

*Specification Document(s): [[TBD: This RFC]]

*Algorithm Analysis Documents(s): TODO

*Algorithm Name: HPKE-Base-P256-SHA256-ChaCha20Poly1305KW

*Algorithm Description: Cipher suite for JOSE-HPKE version 1 in Base Mode that uses the DHKEM(P-256, HKDF-SHA256) KEM, the HKDF-SHA256 KDF and Key wrapping with the ChaCha20Poly1305 AEAD.

*Algorithm Usage Location(s): "alg"

*JOSE Implementation Requirements: Optional

*Change Controller: IESG

*Specification Document(s): [[TBD: This RFC]]

*Algorithm Analysis Documents(s): TODO

*Algorithm Name: HPKE-Base-P384-SHA384-AES256GCMKW

*Algorithm Description: Cipher suite for JOSE-HPKE version 1 in Base Mode that uses the DHKEM(P-384, HKDF-SHA384) KEM, the HKDF-SHA384 KDF, and Key wrapping with the AES-256-GCM AEAD.

*Algorithm Usage Location(s): "alg"
*JOSE Implementation Requirements: Optional

*Change Controller: IESG

*Specification Document(s): [[TBD: This RFC]]

*Algorithm Analysis Documents(s): TODO

*Algorithm Name: HPKE-Base-P384-SHA384-ChaCha20Poly1305KW

*Algorithm Description: Cipher suite for JOSE-HPKE version 1 in Base Mode that uses the DHKEM(P-384, HKDF-SHA384) KEM, the HKDF-SHA384 KDF, and Key wrapping with the ChaCha20Poly1305 AEAD.

*Algorithm Usage Location(s): "alg"

*JOSE Implementation Requirements: Optional

*Change Controller: IESG

*Specification Document(s): [[TBD: This RFC]]

*Algorithm Analysis Documents(s): TODO

*Algorithm Name: HPKE-Base-P521-SHA512-AES256GCMKW

*Algorithm Description: Cipher suite for JOSE-HPKE version 1 in Base Mode that uses the DHKEM(P-521, HKDF-SHA512) KEM, the HKDF-SHA512 KDF, and Key wrapping with the AES-256-GCM AEAD.

*Algorithm Usage Location(s): "alg"

*JOSE Implementation Requirements: Optional

*Change Controller: IESG

*Specification Document(s): [[TBD: This RFC]]

*Algorithm Analysis Documents(s): TODO

*Algorithm Name: HPKE-Base-P521-SHA512-ChaCha20Poly1305KW

*Algorithm Description: Cipher suite for JOSE-HPKE version 1 in Base Mode that uses the DHKEM(P-521, HKDF-SHA512) KEM, the HKDF-SHA512 KDF, and Key wrapping with the ChaCha20Poly1305 AEAD.

*Algorithm Usage Location(s): "alg"

*JOSE Implementation Requirements: Optional

*Change Controller: IESG
*Specification Document(s): [[TBD: This RFC]]

*Algorithm Analysis Documents(s): TODO

*Algorithm Name: HPKE-Base-X25519-SHA256-AES128GCMKW

*Algorithm Description: Cipher suite for JOSE-HPKE version 1 in Base Mode that uses the DHKEM(X25519, HKDF-SHA256) KEM, the HKDF-SHA256 KDF, and Key wrapping with the AES-128-GCM AEAD.

*Algorithm Usage Location(s): "alg"

*JOSE Implementation Requirements: Optional

*Change Controller: IESG

*Specification Document(s): [[TBD: This RFC]]

*Algorithm Analysis Documents(s): TODO

*Algorithm Name: HPKE-Base-X25519-SHA256-ChaCha20Poly1305KW

*Algorithm Description: Cipher suite for JOSE-HPKE version 1 in Base Mode that uses the DHKEM(X25519, HKDF-SHA256) KEM, the HKDF-SHA256 KDF, and Key wrapping with the ChaCha20Poly1305 AEAD.

*Algorithm Usage Location(s): "alg"

*JOSE Implementation Requirements: Optional

*Change Controller: IESG

*Specification Document(s): [[TBD: This RFC]]

*Algorithm Analysis Documents(s): TODO

*Algorithm Name: HPKE-Base-X448-SHA512-AES256GCMKW

*Algorithm Description: Cipher suite for JOSE-HPKE version 1 in Base Mode that uses the DHKEM(X448, HKDF-SHA512) KEM, the HKDF-SHA512 KDF, and Key wrapping with the AES-256-GCM AEAD.

*Algorithm Usage Location(s): "alg"

*JOSE Implementation Requirements: Optional

*Change Controller: IESG

*Specification Document(s): [[TBD: This RFC]]

*Algorithm Analysis Documents(s): TODO
*Algorithm Name: HPKE-Base-X448-SHA512-ChaCha20Poly1305KW

*Algorithm Description: Cipher suite for JOSE-HPKE version 1 in Base Mode that uses the DHKEM(X448, HKDF-SHA512) KEM, the HKDF-SHA512 KDF, and Key wrapping with the ChaCha20Poly1305 AEAD.

*Algorithm Usage Location(s): "alg"

*JOSE Implementation Requirements: Optional

*Change Controller: IESG

*Specification Document(s): [[TBD: This RFC]]

*Algorithm Analysis Documents(s): TODO

*Algorithm Name: HPKE-Base-X25519Kyber768-SHA256-AES256GCMKW

*Algorithm Description: Cipher suite for JOSE-HPKE version 1 in Base Mode that uses the X25519Kyber768Draft00 Hybrid KEM, the HKDF-SHA256 KDF, and Key wrapping with the AES-256-GCM AEAD.

*Algorithm Usage Location(s): "alg"

*JOSE Implementation Requirements: Optional

*Change Controller: IESG

*Specification Document(s): [[TBD: This RFC]]

*Algorithm Analysis Documents(s): TODO

*Algorithm Name: HPKE-Base-X25519Kyber768-SHA256-ChaCha20Poly1305KW

*Algorithm Description: Cipher suite for JOSE-HPKE version 1 in Base Mode that uses the X25519Kyber768Draft00 Hybrid KEM, the HKDF-SHA256 KDF, and Key wrapping with the ChaCha20Poly1305 AEAD.

*Algorithm Usage Location(s): "alg"

*JOSE Implementation Requirements: Optional

*Change Controller: IESG

*Specification Document(s): [[TBD: This RFC]]

*Algorithm Analysis Documents(s): TODO

*Algorithm Name: HPKE-Base-CP256-SHA256-ChaCha20Poly1305KW
Algorithm Description: Cipher suite for JOSE-HPKE version 1 in Base Mode that uses the DHKEM(CP-256, HKDF-SHA256) KEM, the HKDF-SHA256 KDF, and Key wrapping with the ChaCha20Poly1305 AEAD.

Algorithm Usage Location(s): "alg"

JOSE Implementation Requirements: Optional

Change Controller: IESG

Specification Document(s): [[TBD: This RFC]]

Algorithm Analysis Documents(s): TODO

Algorithm Name: HPKE-Base-CP521-SHA512-ChaCha20Poly1305KW

Algorithm Description: Cipher suite for JOSE-HPKE version 1 in Base Mode that uses the DHKEM(CP-521, HKDF-SHA512) KEM, the HKDF-SHA512 KDF, and Key wrapping with the ChaCha20Poly1305 AEAD.

Algorithm Usage Location(s): "alg"

JOSE Implementation Requirements: Optional

Change Controller: IESG

Specification Document(s): [[TBD: This RFC]]

Algorithm Analysis Documents(s): TODO

Algorithm Name: HPKE-Base-CP256-SHA256-AES128GCMKW

Algorithm Description: Cipher suite for JOSE-HPKE in Base Mode that uses the DHKEM(CP-256, HKDF-SHA256) KEM, the HKDF-SHA256 KDF and Key wrapping with the AES128GCM AEAD.

Algorithm Usage Location(s): "alg"

JOSE Implementation Requirements: Optional

Change Controller: IESG

Specification Document(s): [[TBD: This RFC]]

Algorithm Analysis Documents(s): TODO

Algorithm Name: HPKE-Base-CP384-SHA512-AES256GCMKW

Algorithm Description: Cipher suite for JOSE-HPKE in Base Mode that uses the DHKEM(CP384, HKDF-SHA512) KEM, the HKDF-SHA512 KDF, and Key wrapping with the AES256GCM AEAD.
*Algorithm Usage Location(s): "alg"
*JOSE Implementation Requirements: Optional
*Change Controller: IESG
*Specification Document(s): [[TBD: This RFC]]
*Algorithm Analysis Documents(s): TODO
*Algorithm Name: HPKE-Base-CP521-SHA512-AES256GCMKW
*Algorithm Description: Cipher suite for JOSE-HPKE in Base Mode that uses the DHKEM(CP-521, HKDF-SHA512) KEM, the HKDF-SHA512 KDF, and Key wrapping with the AES256GCM AEAD.
*Algorithm Usage Location(s): "alg"
*JOSE Implementation Requirements: Optional
*Change Controller: IESG
*Specification Document(s): [[TBD: This RFC]]
*Algorithm Analysis Documents(s): TODO

10.3. JOSE Header Parameters

*Parameter Name: "encapsulated_key"
*Parameter Description: HPKE encapsulated key
*Parameter Information Class: Public
*Used with "kty" Value(s): "OKP"
*Change Controller: IESG
*Specification Document(s): [[This specification]]

11. References

11.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/
11.2. Informative References


[I-D.westerbaan-cfrg-hpke-xyber768d00-02] Westerbaan, B. and C. A. Wood, "X25519Kyber768Draft00 hybrid post-quantum KEM for HPKE", Work in Progress, Internet-Draft, draft-westerbaan-cfrg-hpke-xyber768d00-02, 4 May 2023,


Acknowledgments

This specification leverages text from [I-D.ietf-cose-hpke]. We would like to thank Matt Chanda and Aaron Parecki for their feedback.

Authors' Addresses

Tirumaleswar Reddy
Nokia
Bangalore
Karnataka
India

Email: kondtir@gmail.com

Hannes Tschofenig
Austria

Email: hannes.tschofenig@gmx.net

Aritra Banerjee
Nokia
Munich
Germany

Email: aritra.banerjee@nokia.com

Orie Steele
Transmute
United States

Email: orie@transmute.industries

Michael B. Jones
independent
United States
Email: michael_b_jones@hotmail.com