Workgroup: COSE
Internet-Draft: draft-ietf-cose-hpke-01
Published: 7 March 2022
Intended Status: Standards Track
Expires: 8 September 2022
Authors: H. Tschofenig (Arm Limited)
R. Housley (Vigil Security)
B. Moran (Arm Limited)

Use of Hybrid Public-Key Encryption (HPKE) with CBOR Object Signing and Encryption (COSE)

Abstract

This specification defines hybrid public-key encryption (HPKE) for use with CBOR Object Signing and Encryption (COSE). 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) encryption function. Authentication for HPKE in COSE is provided by COSE-native security mechanisms.

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/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 8 September 2022.

Copyright Notice

Table of Contents

1. Introduction

Hybrid public-key encryption (HPKE) [I-D.irtf-cfrg-hpke] 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 [I-D.irtf-cfrg-hpke].

The HPKE specification defines several features for use with public key encryption and a subset of those features is applied to COSE

[RFC8152]. Since COSE provides constructs for authentication, those are not re-used from the HPKE specification. This specification uses the "base" mode, as it is called in HPKE specification language.

2. Conventions and Terminology

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.

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 [I-D.irtf-cfrg-hpke]. - pkR is the public key of the recipient, as defined in [I-D.irtf-cfrg-hpke]. - skR is the private key of the recipient, as defined in [I-D.irtf-cfrg-hpke].

3. HPKE for COSE

3.1. Overview

The CDDL for the COSE_Encrypt structure, as used with this specification, is shown in Figure 1.

HPKE, when used with COSE, follows a two layer structure:

* Layer 0 (corresponding to the COSE_Encrypt structure) contains content (plaintext) encrypted with the CEK. This ciphertext may be detached. If not detached, then it is included in the COSE_Encrypt structure.

* Layer 1 (corresponding to a recipient structure) contains parameters needed for HPKE to generate a shared secret used to encrypt the CEK. This layer includes the encrypted CEK.

This two-layer structure is used to encrypt content that can also be shared with multiple parties at the expense of a single additional encryption operation. As stated above, the specification uses a CEK to encrypt the content at layer 0.

For example, the content encrypted at layer 0 is a firmware image. The same ciphertext firmware image is processed by all of the recipients; however, each recipient uses their own private key to obtain the CEK.

```
COSE_Encrypt_Tagged = #6.96(COSE_Encrypt)

HPKE_Encryption_Info = COSE_Encrypt_Tagged

; Layer 0
COSE_Encrypt = [
  Headers,
  ciphertext : bstr / nil,
  recipients : + COSE_recipient
]

; Layer 1
COSE_recipient = [
  protected   : bstr .cbor header_map, ; must contain alg parameter
  unprotected : header_map, ; must contain kid and ephemeral public key
  encCEK      : bstr, ; CEK encrypted with HPKE-derived shared secret
]

header_map = {
  Generic_Headers,
  * label =values,
}
```

Figure 1: CDDL for HPKE-based COSE_Encrypt Structure

The COSE_recipient structure shown in Figure 1 is repeated for each recipient, and it includes the encrypted CEK as well as the sender-generated ephemeral public key in the unprotected header structure.

3.2. 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).

IMPORTANT: For use in this specification, the plaintext "pt" passed into the SealBase is the CEK. The CEK is a random byte sequence of length appropriate for the encryption algorithm selected in layer 0. For example, AES-128-GCM requires a 16 byte key and the CEK would therefore be 16 bytes long.

The "info" parameter can be used to influence the generation of keys and the "aad" parameter provides additional authenticated data to the AEAD algorithm in use. This specification does not mandate the use of the info and the aad parameters.

If SealBase() is successful, it will output a ciphertext "ct" and an encapsulated key "enc". The content of enc is the ephemeral public key.

The content of the info parameter is based on the 'COSE_KDF_Context' structure, which is detailed in Figure 2.

3.3. HPKE Decryption with OpenBase

The recipient will use the OpenBase(enc, skR, info, aad, ct) function with the enc and ct parameters received from the sender. The "aad" and the "info" parameters are obtained via the context of the usage.

The OpenBase function will, if successful, decrypt "ct". When decrypted, the result will be the CEK. The CK is the symmetric key used to decrypt the ciphertext in layer 0 of the COSE_Encrypt structure.

3.4. Info Structure

This section provides a suggestion for constructing the info structure, when used with SealBase() and OpenBase(). Note that the use of the aad and the info structures for these two functions is optional. Profiles of this specification may require their use and may define different info structure.

This specification re-uses the context information structure defined in [RFC8152] as a foundation for the info structure. This payload becomes the content of the info parameter for the HPKE functions, when utilized. For better readability of this specification the COSE_KDF_Context structure is repeated in Figure 2.

```
PartyInfo = (
    identity : bstr / nil,
    nonce : bstr / int / nil,
    other : bstr / nil
)

COSE_KDF_Context = [
    AlgorithmID : int / tstr,
    PartyUInfo : [ PartyInfo ],
    PartyVInfo : [ PartyInfo ],
    SuppPubInfo : [
        keyDataLength : uint,
        protected : empty_or_serialized_map,
        ? other : bstr
    ],
    ? SuppPrivInfo : bstr
]
```

Figure 2: COSE_KDF_Context Data Structure for info parameter

The fields in Figure 2 are populated as follows:

* PartyUInfo.identity corresponds to the kid found in the COSE_Sign_Tagged or COSE_Sign1_Tagged structure (when a digital signature is used). When utilizing a MAC, then the kid is found in the COSE_Mac_Tagged or COSE_Mac0_Tagged structure.

* PartyVInfo.identity corresponds to the kid used for the respective recipient from the inner-most recipients array.

* The value in the AlgorithmID field corresponds to the alg parameter in the unprotected header structure of the recipient structure.

* keyDataLength is set to the number of bits of the desired output value.

* protected refers to the protected structure of the inner-most array.

4. Example

An example of the COSE_Encrypt structure using the HPKE scheme is shown in Figure 3. Line breaks and comments have been inserted for better readability. It uses the following algorithm combination:

* AES-GCM-128 for encryption of detached ciphertext in layer 0.

* AES-GCM-128 for encryption of the CEK in layer 1 as well as ECDH with NIST P-256 and HKDF-SHA256 as a Key Encapsulation Mechanism (KEM).

The algorithm selection is based on the registry of the values offered by the alg parameters.

```
96_0([
    / protected header with alg=AES-GCM-128 /
    h'a10101',
    / unprotected header with nonce /
    {5: h'938b528516193cc7123ff037809f4c2a'},
    / detached ciphertext /
    null,
    / recipient structure /
    [
        / protected field with alg for HPKE /
        h'a1013863',
        / unprotected header /
        {
            / ephemeral public key with x / y coodinate /
            -1: h'a401022001215820a596f2ca8d159c04942308ca90
                  cfbfca65b108ca127df8fe191a063d00d7c5172258
                  20aef47a45d6d6c572e7bd1b9f3e69b50ad3875c68
                  f6da0caaa90c675df4162c39',
             /  kid for recipient static ECDH public key /
             4: h'6b69642d32',
        },
        / encrypted CEK /
        h'9aba6fa44e9b2cef9d646614dcda670dbdb31a3b9d37c7a
          65b099a8152533062',
    ],
])
```

Figure 3: COSE_Encrypt Example for HPKE

Note that the COSE_Sign1 wrapper outside the COSE_Encrypt structure is not shown in the example above.

5. Security Considerations

This specification is based on HPKE and the security considerations of HPKE [I-D.irtf-cfrg-hpke] are therefore applicable also to this specification.

HPKE assumes the sender is in possession of the public key of the recipient and HPKE COSE makes the same assumptions. Some form of public key distribution mechanism is assumed to exist.

Since the CEK is randomly generated it must be ensured that the guidelines for random number generations are followed, see [RFC8937].

The COSE_Encrypt structure must be authenticated using COSE constructs like COSE_Sign, or COSE_Sign1.

6. IANA Considerations

This document requests IANA to add new values to the COSE Algorithms registry defined in [RFC8152] (in the Standards Action With Expert Review category):

6.1. HPKE/P-256+HKDF-256 and AES-128-GCM

*Name: HPKE_P256_HKDF256_AES128_GCM

*Value: TBD1

*Description: HPKE/P-256+HKDF-256 and AES-128-GCM

*Capabilities: [kty]

*Change Controller: IESG

*Reference: [[TBD: This RFC]]

*Recommended: Yes

6.2. HPKE/P-512+HKDF-512 and AES-256-GCM

*Name: HPKE_P521_HKDF512_AES256_GCM

*Value: TBD2

*Description: HPKE/P-512+HKDF-512 and AES-256-GCM

*Capabilities: [kty]

*Change Controller: IESG

*Reference: [[TBD: This RFC]]

*Recommended: Yes

TBD: More values to be added.

7. References

7.1. Normative References

[I-D.irtf-cfrg-hpke] Barnes, R. L., Bhargavan, K., Lipp, B., and C. A. Wood, "Hybrid Public Key Encryption", Work in Progress, Internet-Draft, draft-irtf-cfrg-hpke-12, 2

September 2021, <https://www.ietf.org/archive/id/draft-irtf-cfrg-hpke-12.txt>.

[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/info/rfc2119>.

[RFC8152] Schaad, J., "CBOR Object Signing and Encryption (COSE)", RFC 8152, DOI 10.17487/RFC8152, July 2017, <https://www.rfc-editor.org/info/rfc8152>.

[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/info/rfc8174>.

7.2. Informative References

[RFC2630] Housley, R., "Cryptographic Message Syntax", RFC 2630, DOI 10.17487/RFC2630, June 1999, <https://www.rfc-editor.org/info/rfc2630>.

[RFC8937] Cremers, C., Garratt, L., Smyshlyaev, S., Sullivan, N., and C. Wood, "Randomness Improvements for Security Protocols", RFC 8937, DOI 10.17487/RFC8937, October 2020, <https://www.rfc-editor.org/info/rfc8937>.

Appendix A. Acknowledgements

We would like to thank Goeran Selander, John Mattsson and Ilari Liusvaara for their review feedback.

Authors' Addresses

Hannes Tschofenig
Arm Limited

Email: hannes.tschofenig@arm.com

Russ Housley
Vigil Security, LLC

Email: housley@vigilsec.com

Brendan Moran
Arm Limited

Email: Brendan.Moran@arm.com