ACE Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 8, 2017

G. Selander
J. Mattsson
F. Palombini
Ericsson AB

July 07, 2016

Ephemeral Diffie-Hellman Over COSE (EDHOC)
draft-selander-ace-cose-ecdhe-02

Abstract

This document specifies authenticated Diffie-Hellman key exchange with ephemeral keys, embedded in messages encoded with the CBOR Object Signing and Encryption (COSE) format.

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 http://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 January 8, 2017.

Table of Contents

1. Introduction

Security at the application layer provides an attractive option for protecting Internet of Things (IoT) deployments, for example where transport layer security is not sufficient [I-D.hartke-core-e2e-security-reqs]. IoT devices may be constrained in various ways, including memory, storage, processing capacity, and energy [RFC7228]. A method for protecting individual messages at application layer, suitable for constrained devices, is provided by COSE [I-D.ietf-cose-msg]).

In order for a communication session to provide forward secrecy, the communicating parties can run a Diffie-Hellman (DH) key exchange protocol with ephemeral keys, from which shared key material can be derived. This document specifies authenticated DH protocols using COSE objects for integrity protecting the transport of ephemeral public keys. The DH key exchange messages may be authenticated using either pre-shared keys, raw public keys or X.509 certificates. Authentication is based on credentials established out of band, or from a trusted third party, such as an Authorization Server as specified by [I-D.ietf-ace-oauth-authz]. This document also specifies the derivation of the shared key material.

The DH exchange and the key derivation follow [SP-800-56a] and HKDF [RFC5869], and make use of the data structures of COSE which are aligned with these standards.

1.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119]. These words may also appear in this document in lowercase, absent their normative meanings.

The parties exchanging messages are called "party U" and "party V", and the ECDH ephemeral public keys of U and V are denoted "E_U" and "E_V", respectively, see Figure 2. The messages in the authenticated message exchange are called "message_1" and "message_2", see Figure 3.

The keys used to authenticate the key exchange are either symmetric or asymmetric. In case of symmetric, the pre-shared key is denoted "PSK". In case of asymmetric, the public keys of U and V are denoted "S_U" and "S_V", respectively.

Most keys used in this document have an associated identifier. The identifiers used in the document are placeholders for values of the

identifiers. The following key identifiers/value representations are used in the draft:

- kid_eu and kid_ev represent the values of the key identifiers of the ephemeral public keys of U and V, respectively.
  - kid_eu is a sequence number used for replay protection of message_1.
  - kid_ev is used to identify the resulting traffic key, as a means for party V to ensure that different U establishing traffic keys using this method have different identifiers.
- kid_psk represents the value of the key identifier of the pre-shared key between U and V (Section 3.4).
- kid_su and kid_sv represent the values of the key identifiers of the static public keys of U and V, respectively (Section 3.5).

The key notation is summarized in Figure 1.

Key Identifier	Key	Use
kid_eu	E_U	ECDH ephemeral public key of U
kid_ev	E_V	ECDH ephemeral public key of V
kid_psk	PSK	Pre-shared static symmetric key (Section 3)
kid_su	S_U	Static public key of U (Section 4)
kid_sv	S_V	Static public key of V (Section 4)

Figure 1: Notation of keys and key identifiers.

2. Protocol Overview

EDHOC is a 2-pass message exchange of COSE objects. This section gives an overview of the protocol, together with section Section 4, which explains how the messages are processed, while section Section 3 focuses on the detailed message formats embedded as COSE objects.

The underlying scheme is the Elliptic Curve Cofactor Diffie-Hellman with two ephemeral keys as specified in Section 6.1.2.2 of [SP-800-56a], see Figure 2. U and V exchange their ephemeral public keys E_U, E_V, computes the shared secret and derives the keying material as described in [SP-800-56a].

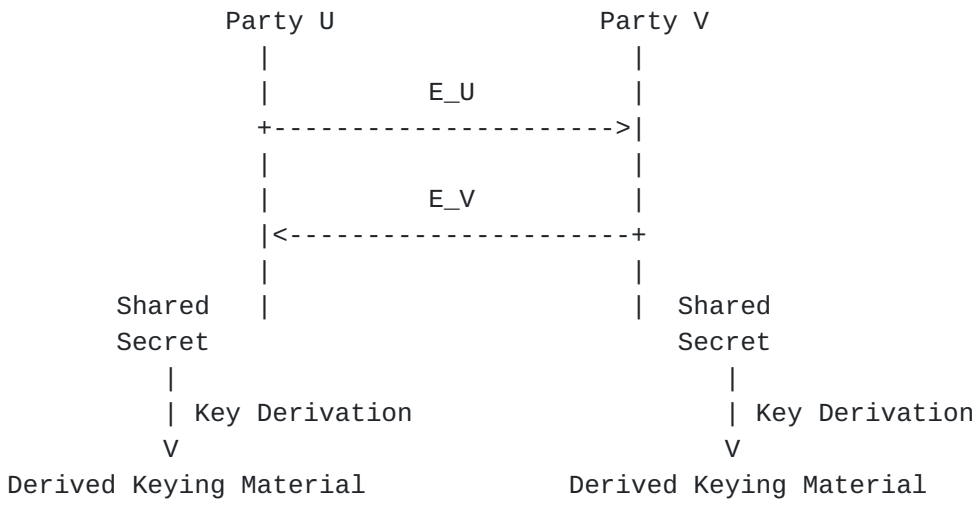

Figure 2: Diffie-Hellman key exchange and key derivation

EDHOC makes the following additions to this scheme (see Figure 3):

- Negotiation of hash function used with HKDF in the key derivation:
  - U proposes one or more hash functions (expressed as HKDF(s) with different hash algorithms).
  - V decides and responds with one function (expressed as HKDF with a particular hash algorithm).
- Optional nonces (N_U, N_V) contributing to the salt used in the extract phase of HKDF [RFC5869].
- Negotiation of subsequent traffic crypto algorithm (TCA) to be used between party U and V:
  - U proposes one or more algorithms (TCA(s)).
  - V decides and responds with one algorithm (TCA).
- Authentication, integrity and replay protection of protocol messages
  - Authentication and integrity protection using the COSE format [I-D.ietf-cose-msg].
    + The MAC/Signature is calculated over the message as defined by COSE.

+ Information about keys, message protection algorithm and replay parameter is included in the COSE Header, as specified in the sections below.

* Authentication may be based on pre-shared keys, raw public keys or X.509 certificates. In the latter case the message contains the public key certificate of the sending endpoints (Cert_U, Cert_V).

The key exchange messages are called "message_1" and "message_2", see Figure 3.

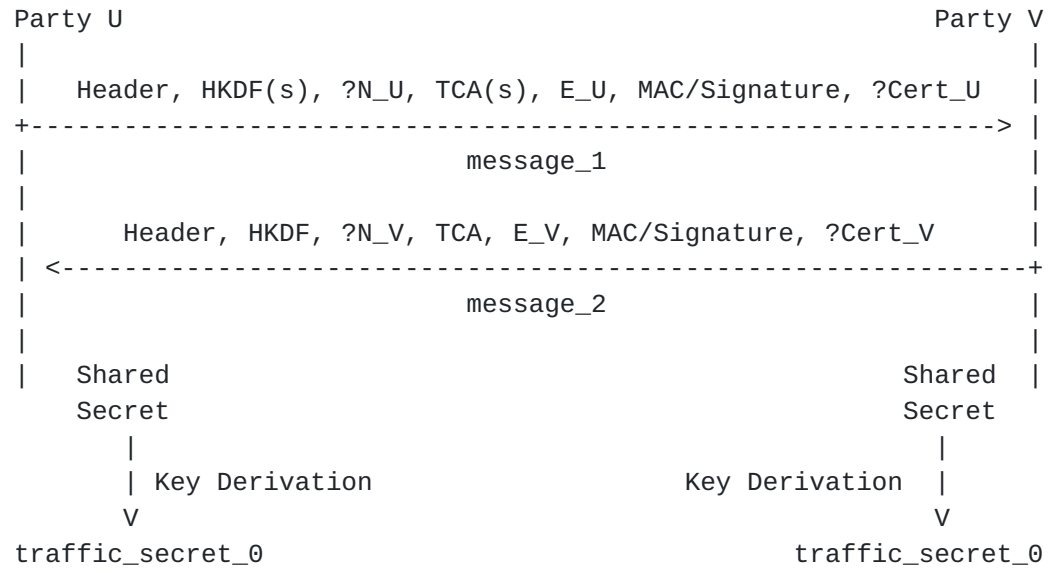

Figure 3: EDHOC Overview. (Optionality indicated by '?'.)

2.1. Authentication methods

The EDHOC protocol messages are authenticated based on credentials pre-established between U and V. The parties may have acquired such a credential from the other party out of band or from a trusted third party, such as an Authorization Server as specified in [I-D.ietf-ace-oauth-authz]. The pre-established credentials are either symmetric secret keys or public keys. The public keys may be raw public keys (RPK), or public keys of a Certificate Authority (CA) used as trust anchor for verification of received certificates.

o Pre-shared symmetric key (PSK). Each message contains a MAC over the message generated by the sending party using PSK.

o Raw Public Keys (RPK). The pre-established credentials may be static raw public keys of the other party (S_U and S_V, of party U

and V, respectively). Each message contain a signature over the message generated by the sending party.

- X.509 Certificates (Cert). The pre-established credentials may also be CA public keys, used to verify received public key certificates. Each message contain the signature over the message, excluding the certificate, generated by the sending party.

3. Message formatting using COSE

This section details the format for the objects used. Examples are provided for each object in Appendix A.

3.1. ECDH Public Keys using COSE_Key

This section defines the formatting of the ephemeral public keys E_U and E_V.

The ECDH ephemeral public key SHALL be formatted as a COSE_Key with the following fields and values (see [I-D.ietf-cose-msg]):

- kty: The value SHALL be 2 (Elliptic Curve Keys)
- kid:
- crv: The value of the Curve used. The value 1 SHALL be supported by party V (NIST P-256 a.k.a. secp256r1 [RFC4492])
- x:
- y: The value SHOULD be boolean.

TODO: Consider replacing P-256 with Curve25519 as mandatory

3.2. Payload 1 formatting

This section defines the formatting of the payload in message_1.

payload_1 is a CBOR array object containing:

- HKDFs: the set of proposed algorithms to indicate the key derivation
- N_U: optional nonce for use in salt with HKDF
- TCAs: the set of proposed algorithms to use with the derived secret

- E_U: the ephemeral public key (with 'kid' = kid_eu, which is a sequence number)

```
payload_1 = [
    HKDFs : AlgID / [ + AlgID ],
    N_U : nil / bstr,
    TCAs : AlgID / [ + AlgID ],
    E_U : COSE_Key
]

AlgID : int / tstr
```

3.3. Payload 2 formatting

This section defines the formatting of the payload in message_2.

payload_2 is a CBOR array object containing:

- HKDF: the agreed key derivation algorithm
- N_V: optional nonce for use in salt with HKDF
- TCA: the agreed traffic crypto algorithm
- E_V: the ephemeral public key (with 'kid' = kid_ev)

```
payload_2 = [
    HKDF : int / tstr,
    N_V : nil / bstr,
    TCA : int / tstr,
    E_V : COSE_Key
]
```

3.4. Message Formatting with Symmetric Keys

Parties U and V are assumed to have a pre-shared key, PSK. The value of the key identifier kid_psk SHALL be unique for U and V.

3.4.1. Message 1 with PSK

In case of PSK, message_1 SHALL have the COSE_Mac0_Tagged structure [I-D.ietf-cose-msg] with the following fields and values:

- Header
  - Protected
    - Alg: 4 (HMAC 256/64)

- + Kid: kid_psk
- * Unprotected: Empty

- o Payload: payload_1 as defined in Section 3.2
- o MAC: As in section 6.3 of [I-D.ietf-cose-msg]

3.4.2. Message 2 with PSK

In case of PSK, message_2 SHALL have the COSE_Mac0_Tagged structure [I-D.ietf-cose-msg] with the following fields and values:

- o Header
  - * Protected
    - + Alg: 4 (HMAC 256/64)
    - + Kid: kid_psk
  - * Unprotected: empty
- o Payload: payload_2 as defined in Section 3.3
- o Tag: As in section 6.3 of [I-D.ietf-cose-msg], including the external_aad in the MAC_structure.

The external authenticated data to use in the MAC_structure of Section 6.3 of [I-D.ietf-cose-msg] is the MAC of message_1.

- o external_aad: MAC

3.4.3. KDF Context with Symmetric Keys

The key derivation is specified in Section 5 using the following context information COSE_KDF_Context for symmetric keys:

```
COSE_KDF_Context = [
    AlgorithmID : int / tstr,      ; AlgID
    SuppPubInfo : [
        keyDataLength : uint,      ; length
        protected : bstr,          ; zero length bstr
        other : bstr               ; MAC message_1 || MAC message_2
    ],
]
```

3.5. Message Formatting with Asymmetric Keys

Parties U and V are assumed to have access to each other's public key.

- Party U's public key, S_U, SHALL be uniquely identified at V by kid_su.
- Party V's public key, S_V, SHALL be uniquely identified at U by kid_sv.

3.5.1. Message 1 with RPK

In case of RPK message_1 SHALL have the COSE_Sign1_Tagged structure [I-D.ietf-cose-msg], with the following fields and values:

- Header
  - protected:
    - alg: -7 (ECDSA 256)
    - kid: kid_su
  - unprotected: empty
- payload: payload_1 as defined in Section 3.2
- signature: computed as in Section 4.4 of {{I-D.ietf-cose-msg}

3.5.2. Message 2 with RPK

In case of RPK, message_2 SHALL have the COSE_Sign1_Tagged structure [I-D.ietf-cose-msg] with the following fields and values:

- Header
  - Protected
    - Alg: -7 (ECDSA 256)
    - Kid: kid_sv
  - Unprotected: empty
- payload: payload_2 as defined in Section 3.3

- signature: computed as in Section 4.4 of {{I-D.ietf-cose-msg}, including the external_aad in the Sig_structure.

The external authenticated data to use in the Sig_structure of Section 4.4 of [I-D.ietf-cose-msg] is the signature in message_1.

- external_aad: signature

3.5.3. Message 1 with Cert

The case of Certificates is similar to RPK. message_1 SHALL have the COSE_Sign1_Tagged structure [I-D.ietf-cose-msg], with the same fields and values as Section 3.5.1 with the addition of the unprotected header field "x5c" containing the X.509 certificate of S_U as a byte string.

- Header
  - protected:
    - alg: -7 (ECDSA 256)
    - kid: kid_su
  - unprotected:
    - x5c: bstr
- payload: payload_1 as defined in Section 3.2
- signature: computed as in Section 4.4 of {{I-D.ietf-cose-msg}

3.5.4. Message 2 with Cert

message_2 is analogous to message_1. message_2 SHALL have the COSE_Sign1_Tagged structure [I-D.ietf-cose-msg], with the same fields and values as Section 3.5.2 and with the addition of the unprotected header field "x5c" containing the X.509 certificate of S_V as a byte string.

- Header
  - Protected
    - Alg: -7 (ECDSA 256)
    - Kid: kid_sv

* Unprotected:

  + x5c: bstr

- payload: payload_2 as defined in Section 3.3

- signature: computed as in Section 4.4 of {{I-D.ietf-cose-msg}, including the external_aad in the Sig_structure.

The external authenticated data to use in the Sig_structure of Section 4.4 of [I-D.ietf-cose-msg] is the signature in message_1.

- external_aad: signature

3.5.5. KDF Context with Asymmetric Keys

The key derivation is specified in Section 5 using the following context information COSE_KDF_Context for asymmetric keys:

```
COSE_KDF_Context = [
    AlgorithmID : int / tstr,      ; AlgID
    SuppPubInfo : [
        keyDataLength : uint,      ; length
        protected : bstr,          ; zero length bstr
        other : bstr               ; signature of message_1 ||
                                     signature of message_2
    ]
]
```

4. Message Processing

Party U and V are assumed to have pre-established credentials as described in Section 2.1.

4.1. U -> message_1

Party U processes message_1 for party V as follows:

- Party U SHALL generate a fresh ephemeral ECDH key pair as specified in Section 5 of [SP-800-56a] using ECC domain parameters of a curve complying with security policies for communicating with party V.

- The ephemeral public key, E_U, SHALL be formatted as a COSE_key as specified in Section 3.1.

- The key identifier kid_eu of the ephemeral public key E_U SHALL be a sequence number initiated to 1 and increased by 1 for each message_1 associated to a particular party V.

- Party U SHALL define the parameters and format payload_1 as specified in Section 3.2 complying with the security policies for communicating with party V.

- message_1 SHALL be formatted as a COSE message according to [I-D.ietf-cose-msg] with parameters specified in Section 3.4.1 (PSK) , Section 3.5.1 (RPK) or Section 3.5.3 (Cert). Party U SHALL cache the MAC/Signature value of the request.

- Party U sends message_1 to party V.

4.2. message_1 -> V

Party V processes the received message_1 as follows:

- If the message contains a certificate, party V SHALL verify the certificate using the pre-established trust anchor and the revocation verification policies relevant for party U. If the verification fails the message is discarded.

- Party V SHALL verify that kid_eu is greater than a counter tracking latest message associated with party U, identified with the sending party key. (The counter is initialized to 0 at first contact with party U.) If kid_eu is less than or equal than the counter, the message is discarded.

- Party V SHALL verify the COSE message as specified in [I-D.ietf-cose-msg] using the key associated to party U, identified by the key identifier kid_psk/kid_su in the message header, or in the verified received certificate. If the MAC/ Signature of the received request can be verified, then the counter associated to party U is updated with kid-eu, else the message is discarded.

- Party V SHALL verify that the ECDH curve is compliant with its security policy for communicating with U, or else respond with an error.

- V SHALL select a preferred pair of (HKDF, TCA) out of those proposed by U, compliant with the security policy relevant for party U. If such a pair does not exist, V SHALL stop processing the message and MAY respond with an error, indicating that no common algorithm could be found.

4.3. message_2 <- V

Party V composes message_2 for party U as follows:

- Party V SHALL generate a fresh ephemeral ECDH key pair as specified in Section 5 of [SP-800-56a] using same curve/ECC domain parameters as used by party U.

- The ephemeral public key, E_V, SHALL be formatted as a COSE_key as specified in Section 3.1. The key identifier kid_ev SHALL be unique among key identifiers used for traffic keys by party V.

- Party SHALL define the parameters and format payload_2 as specified in Section 3.3 complying with the security policies for communicating with party V.

- message_2 SHALL be formatted as a COSE message according to [I-D.ietf-cose-msg] with parameters specified in Section 3.4.2 (PSK) , Section 3.5.2 (RPK) or Section 3.5.4 (Cert) using the same authentication scheme as in message_1. Note that the MAC/ Signature value of the request is included as additional authenticated data of the response.

Party V sends message_2 to party U. Then party V derives the traffic_secret_0 key as specified Section 5, and labels it with kid_ev.

4.4. U <- message_2

Party U processes the received message_2 as follows:

- If the message contains a certificate, party V SHALL verify the certificate using the pre-established trust anchor and the revocation verification policies relevant for party U. If the verification fails the message is discarded.

- Party U SHALL verify the received COSE message as defined in [I-D.ietf-cose-msg] using the key associated to the key identifier (kid_psk/kid_su) in the message header, or in the verified received certificate. Note the use of the cached MAC/Signature of the request. If the COSE message cannot be verified, the message is discarded.

- U SHALL verify that the received pair (HKDF, TCA) is one element of those proposed in the request, else stop processing the message.

- If the response is verified, U derives the traffic_secret_0 as specified Section 5.

5. Key Derivation

The key derivation is identical to Section 11 of [I-D.ietf-cose-msg], using HKDF [RFC5869] agreed during the message exchange.

- the secret SHALL be the ECDH shared secret as defined in Section 12.4.1 of [I-D.ietf-cose-msg], where the computed secret is specified in section 5.7.1.2 of [SP-800-56a]

- the salt SHALL be B1 XOR B2, where

  * B1 = N_U || 00 .. 0, i.e. the nonce N_U, if present, appended with padded zeros to the size of the hash function; or else just an octet string of zeros

  * B2 = 00... 0 || N_V, i.e. the nonce N_V, if present, prepended with padded zeros to the size of the hash function; or else just an octet string of zeros

  * This corresponds to salt = N_U || N_V in the case of each party contributing a nonce of half the size of the salt, but also accommodates for nonces of different sizes.

- the length SHALL be the length of the key in TCA.

- the context information SHALL be the serialized COSE_KDF_Context defined in the next paragraph.

- the PRF SHALL be the one indicated in HKDF using the Table 18 of [I-D.ietf-cose-msg] (in our examples, -27 corresponds to HMAC with SHA-256)

The context information COSE_KDF_Context is defined as follows:

- AlgorithmID SHALL be the algorithm for which the key material will be derived. It's value is AlgID

- PartyUInfo SHALL be empty

- PartyVInfo SHALL be empty

- SuppPubInfo SHALL contain:

  * KeyDataLength SHALL be equal to 'length'

* protected SHALL be a zero length bstr

* other SHALL be the concatenation of the MAC/Signature of message_1 and the MAC/Signature of message_2

o SuppPrivInfo SHALL be empty

The tags are either MACs (PSK) or the Signatures (RPK, Cert) of the COSE messages.

```
COSE\_KDF\_Context = [
    AlgorithmID : int / tstr,      ; HKDF
    SuppPubInfo : [
        keyDataLength : uint,      ; length
        protected : bstr,          ; zero length bstr
        other : bstr               ; MAC/Signature message_1
                                     || MAC/Signature message_2
    ],
]
```

The output from the key derivation is denoted "traffic_secret_0".

6. Security Considerations

For unauthenticated Diffie-Hellman it is recommended that public information about parties U and V, such as their identifiers, is included in the context information used in the key derivation. In the present case the assumption is that the parties authenticate each other with pre-established credentials, and the tag (MAC/Signature) created with the pre-established credentials is included in the key derivation context.

The referenced processing instructions in [SP-800-56a] must be complied with, including deleting the intermediate computed values along with any ephemeral ECDH secrets after the key derivation is completed.

The choice of key length used in the different algorithms needs to be harmonized, so that right security level is maintained throughout the calculations.

The identifier of the ephemeral key of party U is used for replay protection of U's requests.

With the current protocol, key confirmation of the Diffie-Hellman shared secret/traffic keys is performed when the keys are successfully used. The addition of key confirmation to the protocol is for further study.

TODO: Expand on the security considerations in a future version of the draft

7. Privacy Considerations

TODO

8. IANA Considerations

9. Acknowledgments

The authors wants to thank Ilari Liusvaara, Jim Schaad and Ludwig Seitz for timely review and helpful comments.

10. References

10.1. Normative References

[I-D.ietf-cose-msg]
Schaad, J., "CBOR Object Signing and Encryption (COSE)", draft-ietf-cose-msg-14 (work in progress), June 2016.

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

[SP-800-56a]
Barker, E., Chen, L., Roginsky, A., and M. Smid, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography", NIST Special Publication 800-56A, May 2013, <http://dx.doi.org/10.6028/NIST.SP.800-56Ar2>.

10.2. Informative References

[I-D.hartke-core-e2e-security-reqs]
Selander, G., Palombini, F., and K. Hartke, "Requirements for CoAP End-To-End Security", draft-hartke-core-e2e-security-reqs-01 (work in progress), July 2016.

[I-D.ietf-ace-oauth-authz]
Seitz, L., Selander, G., Wahlstroem, E., Erdtman, S., and H. Tschofenig, "Authentication and Authorization for Constrained Environments (ACE)", draft-ietf-ace-oauth-authz-02 (work in progress), June 2016.

[RFC4492] Blake-Wilson, S., Bolyard, N., Gupta, V., Hawk, C., and B. Moeller, "Elliptic Curve Cryptography (ECC) Cipher Suites for Transport Layer Security (TLS)", RFC 4492, DOI 10.17487/RFC4492, May 2006, <http://www.rfc-editor.org/info/rfc4492>.

[RFC5869] Krawczyk, H. and P. Eronen, "HMAC-based Extract-and-Expand Key Derivation Function (HKDF)", RFC 5869, DOI 10.17487/RFC5869, May 2010, <http://www.rfc-editor.org/info/rfc5869>.

[RFC7228] Bormann, C., Ersue, M., and A. Keranen, "Terminology for Constrained-Node Networks", RFC 7228, DOI 10.17487/RFC7228, May 2014, <http://www.rfc-editor.org/info/rfc7228>.

[RFC7252] Shelby, Z., Hartke, K., and C. Bormann, "The Constrained Application Protocol (CoAP)", RFC 7252, DOI 10.17487/RFC7252, June 2014, <http://www.rfc-editor.org/info/rfc7252>.

Appendix A. Examples

A.1. ECDH Public Key

An example of COSE_Key structure, representing an ECDH public key, is given in Figure 4, using CBOR's diagnostic notation. In this example, the ephemeral key is identified by a 4 bytes 'kid'.

```
/ ephemeral / -1:{
            / kty / 1:2,
            / kid / 2:h'78f67901',
            / crv / -1:1,
            / x / -2:h'98f50a4ff6c05861c8860d13a638ea56c3f5ad7590b
            bfbf054e1c7b4d91d6280',
            / y / -3:true
          }
```

Figure 4: Example of an ECDH public key formatted as a COSE_Key

The equivalent CBOR encoding is: h'a120a50102024478f67901200121582098 f50a4ff6c05861c8860d13a638ea56c3f5ad7590bbfbf054e1c7b4d91d628022f5', which has a size of 51 bytes.

A.2. Payload 1

An example of COSE encoding for payload_1 is given in Figure 5, using CBOR's diagnostic notation. In this example, the size of the identifier of U's ephemeral key, kid_eu, is 1 byte.

The payload_1 is:

```
[
  -27, / HKDFs /
  null, / N_U /
  12, / TCAs /
  h'a120a501020241032001215820 98f50a4ff6c05861c8860d13a638ea56c3f5a
  d7590bbfbf054e1c7b4d91d628022f5' / COSE_Key E_U { /
    / ephemeral -1:{ /
    / kty 1:2, /
    / kid 2:h'03', kid_eu /
    / crv -1:1, /
    / x -2:h'98f50a4ff6c05861c8860d13a638ea56c3f5ad7590bbfb /
    / f054e1c7b4d91d6280', /
    / y -3:true /
    / } /
  / } /
]
```

Figure 5: Example of payload of message_1

The equivalent CBOR encoding of the payload is: h'84381af60c582fa120a501020241032001215820 98f50a4ff6c05861c8860d13a638ea56c3f5ad7590bbfbf054e1c7b4d91d628022f5', which has a size of 54 bytes.

A.3. Payload 2

An example of COSE encoding for message_2 is given in Figure 6 using CBOR's diagnostic notation. In this example, kid_ev, the identifier of V's ephemeral public key, is 4 bytes.

The payload is:

```
[
  -27, / HKDF /
  null, / N_V /
  12, / TCA /
  h'a120a5010202442edb61f92001215820acbee6672a28340affce41c721901eb
  d7868231bd1d86e41888a07822214050022f5' / COSE_Key E_V { /
    / ephemeral -1:{ /
    / kty 1:2, /
    / kid 2:h'2edb61f9', kid_ev /
    / crv -1:1, /
    / x -2:h'acbee6672a28340affce41c721901ebd7868231bd1d /
    / 86e41888a078222140500', /
    / y -3:true /
    / } /
  / } /
]
```

Figure 6: Example of payload of message_2

The equivalent CBOR encoding of the payload is: h'84381af60c5832a120a5010202442edb61f92001215820acbee6672a28340affce41c721901ebd7868231bd1d86e41888a07822214050022f5', which has a size of 57 bytes.

A.4. Message 1 with PSK

An example of COSE encoding for message_1 is given in Figure 7 using CBOR's diagnostic notation. In this example, kid_psk, the identifier of PSK is 4 bytes, and the payload is as in Appendix A.2.

The message_1 is:

```
996(
  [
    / protected / h'a201040444e19648b5' / { /
        / alg 1:4, HMAC 256//64 /
        / kid 4:h'e19648b5' kid_psk /
    / } / ,
    / unprotected / {},
    / payload / h'84381af60c582fa120a501020241032001215820
    98f50a4ff6c05861c8860d13a638ea56c3f5ad7590bbfbf054e1c7b4
    d91d628022f5', / payload_1 /
    / tag / h'e77fe81c66c3b5c0'
  ]
)
```

Figure 7: Example of message_1 authenticated with PSK

The equivalent CBOR encoding is: h'd903e48449a201040444e19648b5a05836
84381af60c582fa120a501020241032001215820 98f50a4ff6c05861c8860d13a638e
a56c3f5ad7590bbfbf054e1c7b4d91d628022f548e77fe81c66c3b5c0', which has a size of 80 bytes.

A.5. Message 2 with PSK

An example of COSE encoding for message_2 is given in Figure 8 using CBOR's diagnostic notation. In this example, kid_psk, the identifier of PSK, is 4 bytes, and the payload is as in Appendix A.3.

The message_2 is:

```
996(
  [
    / protected / h'a201040444e19648b5' / { /
        / alg 1:4, HMAC 256//64 /
        / kid 4:h'e19648b5' kid_psk /
      / } / ,
    / unprotected / {},
    / payload / h'84381af60c5832a120a5010202442edb61f92001215820
    acbee6672a28340affce41c721901ebd7868231bd1d86e41888a07822214
    050022f5', / payload_2 /
    / tag / h'6113268ad246f2c9'
  ]
)
```

Figure 8: Example of message_2 authenticated with PSK

The equivalent CBOR encoding is: h'd903e48449a201040444e19648b5a05839
84381af60c5832a120a5010202442edb61f92001215820acbee6672a28340affce41c
721901ebd7868231bd1d86e41888a07822214050022f5486113268ad246f2c9',
which has a size of 83 bytes.

A.6. Message 1 with RPK

An example of COSE encoding for message_1 is given in Figure 9, using CBOR's diagnostic notation. In this example, the size of the identifier of the static public key of U, kid_su, is 4 bytes, and the payload is as in Appendix A.2.

The message_1 is:

```
997(
  [
    / protected / h'a201260444c150d41c' / { /
      / alg  1:-7,  ECDSA 256 /
      / kid  4:h'c150d41c',  kid_c /
      / } / ,
    / unprotected / {},
    / payload / h'84381af60c582fa120a501020241032001215820
    98f50a4ff6c05861c8860d13a638ea56c3f5ad7590bbfbf054e1c7b4
    d91d628022f5', / payload_1 /
    / signature / h'eae868ecc1276883766c5dc5ba5b8dca25dab3c2e56a
    51ce5705b793914348e14eea4aee6e0c9f09db4ef3ddeca8f3506cd1a98a8fb64
    327be470355c9657ce0'
  ]
)
```

Figure 9: Example of message_1 authenticated with RPK

The equivalent CBOR encoding is: h'd903e58449a201260444c150d41ca05836
84381af60c582fa120a501020241032001215820 98f50a4ff6c05861c8860d13a638e
a56c3f5ad7590bbfbf054e1c7b4d91d628022f55840eae868ecc1276883766c5dc5ba
5b8dca25dab3c2e56a51ce5705b793914348e14eea4aee6e0c9f09db4ef3ddeca8f35
06cd1a98a8fb64327be470355c9657ce0', which has a size of 137 bytes.

A.7. Message 2 with RPK

An example of COSE encoding for Message 2 is given in Figure 11, using CBOR's diagnostic notation. In this example, the size of the identifier of the public key of V, kid_sv, is 4 bytes, and the payload is as in Appendix A.3.

The external_aad is the signature of message_1:

```
  / external\_aad / h'eae868ecc1276883766c5dc5ba5b8dca25dab3c2e56a
 51ce5705b793914348e14eea4aee6e0c9f09db4ef3ddeca8f3506cd1a98a8fb64327
 be470355c9657ce0'
```

Figure 10: Example of external additional authenticated data

The message_2 is:

```
997(
  [
    / protected / h'a2012604447a2af164' / { /
      / alg 1:-7, ECDSA 256 /
      / kid 4:h'7a2af164', kid_sv /
    / } / ,
    / unprotected / {},
    / payload, / h'84381af60c5832a120a5010202442edb61f92001215820acb
    ee6672a28340affce41c721901ebd7868231bd1d86e41888a078222140
    50022f5', / payload_2 /
    / signature / h'2374e27a3d9eeb4f66c5dc5ba5b8dca25dab3c2e56a551ce
    5705b793914348e14eea4aee6e0c9f09db4ef3ddeca8f3506cd1a98a8fb64327
    be470355c9657ce0'
  ]
)
```

Figure 11: Example of message_2 authenticated with RPK.

The equivalent CBOR encoding is: h'd903e58449a2012604447a2af164a05839
84381af60c5832a120a5010202442edb61f92001215820acbee6672a28340affce41c
721901ebd7868231bd1d86e41888a07822214050022f558402374e27a3d9eeb4f66c5
dc5ba5b8dca25dab3c2e56a551ce5705b793914348e14eea4aee6e0c9f09db4ef3dde
ca8f3506cd1a98a8fb64327be470355c9657ce0', which has a size of 140
bytes.

A.8. Message 1 with Cert

An example of COSE encoding for message_1 is given in Figure 12, using CBOR's diagnostic notation. In this example, the size of the identifier of the static public key of U, kid_su, is 4 bytes, and the payload is as in Appendix A.2.

The message_1 is:

```
997(
  [
    / protected / h'a201260444c150d41c' / { /
      / alg 1:-7, ECDSA 256 /
      / kid 4:h'c150d41c', kid_su /
      / } / ,
    / unprotected / {"x5c": certificate-chain},
    / payload / h'84381af60c582fa120a501020241032001215820 98f
    50a4ff6c05861c8860d13a638ea56c3f5ad7590bbfbf054e1c7b4d9
    1d628022f5', / payload_1 /
    / signature / h'eae868ecc1276883766c5dc5ba5b8dca25dab3c2e56a
    51ce5705b793914348e14eea4aee6e0c9f09db4ef3ddeca8f3506cd1a98a8fb64
    327be470355c9657ce0'
  ]
)
```

Figure 12: Example of message_1 authenticated with Certificate

The equivalent CBOR encoding is:

```
h'd903e58449a201260444c150d41ca163783563 40... 583684381af60c582fa120
a501020241032001215820 98f50a4ff6c05861c8860d13a638ea56c3f5ad7590bbfbf
054e1c7b4d91d628022f55840eae868ecc1276883766c5dc5ba5b8dca25dab3c2e56a
51ce5705b793914348e14eea4aee6e0c9f09db4ef3ddeca8f3506cd1a98a8fb64327b
e470355c9657ce0',
```

which has a size of 142 bytes plus the size of the certificate.

A.9. Message 2 with Cert

An example of COSE encoding for message_2 is given in Figure 13, using CBOR's diagnostic notation. In this example, the size of the identifier of the static public key of U, kid_su, is 4 bytes, and the payload is as in Appendix A.3.

The message_2 is:

```
997(
  [
    / protected / h'a2012604447a2af164' / { /
      / alg 1:-7, ECDSA 256 /
      / kid 4:h'7a2af164', kid_sv /
    / } / ,
    / unprotected / {"x5c": certificate-chain},
    / payload / h'84381af60c5832a120a5010202442edb61f92001215820
    acbee6672a28340affce41c721901ebd7868231bd1d86e41888a07822214
    050022f5', / payload_2 /
    / signature / h'2374e27a3d9eeb4f66c5dc5ba5b8dca25dab3c2e56a551ce
    5705b793914348e14eea4aee6e0c9f09db4ef3ddeca8f3506cd1a98a8fb64327
    be470355c9657ce0'
  ]
)
```

Figure 13: Example of message_2 authenticated with Certificate.

The equivalent CBOR encoding is:

```
h'd903e58449a2012604447a2af164a163783563 40... 583984381af60c5832a120
a5010202442edb61f92001215820acbee6672a28340affce41c721901ebd7868231bd
1d86e41888a07822214050022f558402374e27a3d9eeb4f66c5dc5ba5b8dca25dab3c
2e56a551ce5705b793914348e14eea4aee6e0c9f09db4ef3ddeca8f3506cd1a98a8fb
64327be470355c9657ce0',
```

which has a size of 145 bytes plus the size of the certificate.

Appendix B. Implementing EDHOC with CoAP

The DH key exchange specified in this document can be implemented as a CoAP [RFC7252] message exchange with the CoAP client as party U and the CoAP server as party V. A strawman is sketched here.

The CoAP client makes the following request:

- The request method is POST
- Content-Format is "application/cose+cbor"
- The Uri-Path is "edhoc"
- The Payload is message_1

The CoAP server performs the verifications of the COSE object as specified in [I-D.ietf-cose-msg]. If successful, then the server provides the following response:

- The response Code is 2.04 (Changed)

- The Payload is message_2

Authors' Addresses

Goeran Selander
Ericsson AB
Farogatan 6
Kista SE-16480 Stockholm
Sweden

Email: goran.selander@ericsson.com

John Mattsson
Ericsson AB
Farogatan 6
Kista SE-16480 Stockholm
Sweden

Email: john.mattsson@ericsson.com

Francesca Palombini
Ericsson AB
Farogatan 6
Kista SE-16480 Stockholm
Sweden

Email: francesca.palombini@ericsson.com