Workgroup: CoRE Working Group
Internet-Draft:
draft-ietf-core-oscore-edhoc-02
Published: 25 October 2021
Intended Status: Standards Track
Expires: 28 April 2022
Authors: F. Palombini M. Tiloca R. Hoeglund
Ericsson RISE AB RISE AB
S. Hristozov G. Selander
Fraunhofer AISEC Ericsson

Profiling EDHOC for CoAP and OSCORE

Abstract

The lightweight authenticated key exchange protocol EDHOC can be run over CoAP and used by two peers to establish an OSCORE Security Context. This document further profiles this use of the EDHOC protocol, by specifying a number of additional and optional mechanisms. These especially include an optimization approach for combining the execution of EDHOC with the first subsequent OSCORE transaction. This combination reduces the number of round trips required to set up an OSCORE Security Context and to complete an OSCORE transaction using that Security Context.

Discussion Venues

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

Discussion of this document takes place on the Constrained RESTful Environments Working Group mailing list (core@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/core/.

Source for this draft and an issue tracker can be found at https://github.com/core-wg/oscore-edhoc.

Status of This Memo

This Internet-Draft will expire on 28 April 2022.

Copyright Notice

Table of Contents

1. Introduction

Ephemeral Diffie-Hellman Over COSE (EDHOC) [I-D.ietf-lake-edhoc] is a lightweight authenticated key exchange protocol, especially intended for use in constrained scenarios. In particular, EDHOC messages can be transported over the Constrained Application Protocol (CoAP) [RFC7252] and used for establishing a Security Context for Object Security for Constrained RESTful Environments (OSCORE) [RFC8613].

This document profiles this use of the EDHOC protocol, and specifies a number of additional and optional mechanisms. These especially include an optimization approach, that combines the EDHOC execution with the first subsequent OSCORE transaction (see Section 3). This allows for a minimum number of round trips necessary to setup the OSCORE Security Context and complete an OSCORE transaction, e.g., when an IoT device gets configured in a network for the first time.

This optimization is desirable, since the number of protocol round trips impacts on the minimum number of flights, which in turn can have a substantial impact on the latency of conveying the first OSCORE request, when using certain radio technologies.

Without this optimization, it is not possible, not even in theory, to achieve the minimum number of flights. This optimization makes it possible also in practice, since the last message of the EDHOC protocol can be made relatively small (see Section 1.3 of [I-D.ietf-lake-edhoc]), thus allowing additional OSCORE protected CoAP data within target MTU sizes.

Furthermore, this document defines:

* A method for deterministically converting an OSCORE Sender/Recipient ID to a corresponding EDHOC connection identifier (see Section 4). While this method is required to be used when using the optimization above, it is recommended in general, since it ensures that an OSCORE Sender/Recipient ID is always converted to the EDHOC identifier with the smallest size.

* A number of parameters corresponding to different information elements of an EDHOC applicability statement (see Section 6). These can be specified as target attributes in the link to an EDHOC resource associated to that applicability statement, thus enabling an enhanced discovery of such resource for CoAP clients.

1.1. 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.

The reader is expected to be familiar with terms and concepts defined in CoAP [RFC7252], CBOR [RFC8949], CBOR sequences [RFC8742], OSCORE [RFC8613] and EDHOC [I-D.ietf-lake-edhoc].

2. EDHOC Overview

The EDHOC protocol allows two peers to agree on a cryptographic secret, in a mutually-authenticated way and by using Diffie-Hellman ephemeral keys to achieve perfect forward secrecy. The two peers are denoted as Initiator and Responder, as the one sending or receiving the initial EDHOC message_1, respectively.

After successful processing of EDHOC message_3, both peers agree on a cryptographic secret that can be used to derive further security material, and especially to establish an OSCORE Security Context [RFC8613]. The Responder can also send an optional EDHOC message_4 to achieve key confirmation, e.g., in deployments where no protected application message is sent from the Responder to the Initiator.

Appendix A.3 of [I-D.ietf-lake-edhoc] specifies how to transfer EDHOC over CoAP. That is, the EDHOC data (referred to as "EDHOC messages") are transported in the payload of CoAP requests and responses. The default message flow consists in the CoAP Client acting as Initiator and the CoAP Server acting as Responder. Alternatively, the two roles can be reversed. In the rest of this document, EDHOC messages are considered to be transferred over CoAP.

Figure 1 shows a CoAP Client and Server running EDHOC as Initiator and Responder, respectively. That is, the Client sends a POST request to a reserved EDHOC resource at the Server, by default at the Uri-Path "/.well-known/edhoc". The request payload consists of the CBOR simple value "true" (0xf5) concatenated with EDHOC message_1, which also includes the EDHOC connection identifier C_I of the Client.

This triggers the EDHOC exchange at the Server, which replies with a 2.04 (Changed) response. The response payload consists of EDHOC message_2, which also includes the EDHOC connection identifier C_R of the Server. The Content-Format of the response may be set to "application/edhoc".

Finally, the Client sends a POST request to the same EDHOC resource used earlier to send EDHOC message_1. The request payload consists of the EDHOC connection identifier C_R, concatenated with EDHOC message_3.

After this exchange takes place, and after successful verifications as specified in the EDHOC protocol, the Client and Server can derive an OSCORE Security Context, as defined in Appendix A.2 of [I-D.ietf-lake-edhoc]. After that, they can use OSCORE to protect their communications as per [RFC8613].

The Client and Server are required to agree in advance on certain information and parameters describing how they should use EDHOC. These are specified in an applicability statement see Section 3.9 of [I-D.ietf-lake-edhoc], associated to the used EDHOC resource.

```
   CoAP Client                                          CoAP Server
(EDHOC Initiator)                                    (EDHOC Responder)
        |                                                   |
        |                                                   |
        | ----------------- EDHOC Request ----------------> |
        |   Header: 0.02 (POST)                             |
        |   Uri-Path: "/.well-known/edhoc"                  |
        |   Payload: true, EDHOC message_1                  |
        |                                                   |
        | <---------------- EDHOC Response----------------- |
        |              Header: 2.04 (Changed)               |
        |              Content-Format: application/edhoc    |
        |              Payload: EDHOC message_2             |
        |                                                   |
EDHOC verification                                          |
        |                                                   |
        | ----------------- EDHOC Request ----------------> |
        |   Header: 0.02 (POST)                             |
        |   Uri-Path: "/.well-known/edhoc"                  |
        |   Payload: C_R, EDHOC message_3                   |
        |                                                   |
        |                                           EDHOC verification
        |                                                   +
OSCORE Sec Ctx                                        OSCORE Sec Ctx
  Derivation                                            Derivation
        |                                                   |
        | ---------------- OSCORE Request ----------------> |
        |   Header: 0.02 (POST)                             |
        |                                                   |
        | <--------------- OSCORE Response ---------------- |
        |                         Header: 2.04 (Changed)    |
        |                                                   |
```

Figure 1: EDHOC and OSCORE run sequentially

As shown in Figure 1, this purely-sequential flow where EDHOC is run first and then OSCORE is used takes three round trips to complete.

Section 3 defines an optimization for combining EDHOC with the first subsequent OSCORE transaction. This reduces the number of round trips required to set up an OSCORE Security Context and to complete an OSCORE transaction using that Security Context.

3. EDHOC Combined with OSCORE

This section defines an optimization for combining the EDHOC exchange with the first subsequent OSCORE transaction, thus minimizing the number of round trips between the two peers.

This approach can be used only if the default EDHOC message flow is used, i.e., when the Client acts as Initiator and the Server acts as Responder, while it cannot be used in the case with reversed roles.

When running the purely-sequential flow of Section 2, the Client has all the information to derive the OSCORE Security Context already after receiving EDHOC message_2 and before sending EDHOC message_3.

Hence, the Client can potentially send both EDHOC message_3 and the subsequent OSCORE Request at the same time. On a semantic level, this requires sending two REST requests at once, as in Figure 2.

```
   CoAP Client                                       CoAP Server
(EDHOC Initiator)                                 (EDHOC Responder)
        |                                                  |
        | ----------------- EDHOC Request ---------------> |
        |   Header: 0.02 (POST)                            |
        |   Uri-Path: "/.well-known/edhoc"                 |
        |   Payload: true, EDHOC message_1                 |
        |                                                  |
        | <---------------- EDHOC Response---------------- |
        |              Header: Changed (2.04)              |
        |              Content-Format: application/edhoc   |
        |              Payload: EDHOC message_2            |
        |                                                  |
EDHOC verification                                         |
        +                                                  |
  OSCORE Sec Ctx                                           |
    Derivation                                             |
        |                                                  |
        | ------------ EDHOC + OSCORE Request -----------> |
        |   Header: 0.02 (POST)                            |
        |                                                  |
        |                                         EDHOC verification
        |                                                  +
        |                                          OSCORE Sec Ctx
        |                                             Derivation
        |                                                  |
        | <-------------- OSCORE Response ---------------- |
        |                          Header: 2.04 (Changed)  |
        |                                                  |
```

Figure 2: EDHOC and OSCORE combined

To this end, the specific approach defined in this section consists of sending a single EDHOC + OSCORE request, which conveys the pair (C_R, EDHOC message_3) within an OSCORE protected CoAP message.

That is, the EDHOC + OSCORE request is in practice the OSCORE Request from Figure 1, as still sent to a protected resource and with the correct CoAP method and options intended for accessing that resource. At the same time, the EDHOC + OSCORE request also transports the pair (C_R, EDHOC message_3) required for completing the EDHOC exchange.

As EDHOC message_3 may be too large to be included in a CoAP Option, e.g., if containing a large public key certificate chain, it has to be transported in the CoAP payload of the EDHOC + OSCORE request.

The rest of this section specifies how to transport the data in the EDHOC + OSCORE request and their processing order. In particular, the use of this approach is explicitly signalled by including an EDHOC Option (see Section 3.1) in the EDHOC + OSCORE request. The processing of the EDHOC + OSCORE request is specified in Section 3.2 for the Client side and in Section 3.3 for the Server side.

3.1. EDHOC Option

This section defines the EDHOC Option. The option is used in a CoAP request, to signal that the request payload conveys both an EDHOC message_3 and OSCORE protected data, combined together.

The EDHOC Option has the properties summarized in Figure 3, which extends Table 4 of [RFC7252]. The option is Critical, Safe-to-Forward, and part of the Cache-Key. The option MUST occur at most once and is always empty. If any value is sent, the value is simply ignored. The option is intended only for CoAP requests and is of Class U for OSCORE [RFC8613].

```
+-------+---+---+---+---+-------+--------+--------+---------+
| No.   | C | U | N | R | Name  | Format | Length | Default |
+-------+---+---+---+---+-------+--------+--------+---------+
| TBD21 | x |   |   |   | EDHOC | Empty  |   0    | (none)  |
+-------+---+---+---+---+-------+--------+--------+---------+
       C=Critical, U=Unsafe, N=NoCacheKey, R=Repeatable
```

Figure 3: The EDHOC Option.

The presence of this option means that the message payload contains also EDHOC data, that must be extracted and processed as defined in Section 3.3, before the rest of the message can be processed.

Figure 4 shows the format of a CoAP message containing both the EDHOC data and the OSCORE ciphertext, using the newly defined EDHOC option for signalling.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Ver| T |  TKL  |      Code     |          Message ID           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Token (if any, TKL bytes) ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  OSCORE option  |   EDHOC option  | Other options (if any) ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|1 1 1 1 1 1 1 1|    Payload
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 4: CoAP message for EDHOC and OSCORE combined - signalled with the EDHOC Option

3.2. Client Processing

The Client prepares an EDHOC + OSCORE request as follows.

1. Compose EDHOC message_3 as per Section 5.4.2 of [I-D.ietf-lake-edhoc].

2. Encrypt the original CoAP request as per Section 8.1 of [RFC8613], using the new OSCORE Security Context established after receiving EDHOC message_2.

   Note that the OSCORE ciphertext is not computed over EDHOC message_3, which is not protected by OSCORE. That is, the result of this step is the OSCORE Request as in Figure 1.

3. Build a CBOR sequence [RFC8742] composed of two CBOR byte strings in the following order.

   * The first CBOR byte string is the EDHOC message_3 resulting from step 1.

   * The second CBOR byte string has as value the OSCORE ciphertext of the OSCORE protected CoAP request resulting from step 2.

4. Compose the EDHOC + OSCORE request, as the OSCORE protected CoAP request resulting from step 2, where the payload is replaced with the CBOR sequence built at step 3.

   Note that the new payload includes EDHOC message_3, but it does not include the EDHOC connection identifier C_R. As the Client is the EDHOC Initiator, C_R encodes the OSCORE Sender ID of the Client, which is already specified as 'kid' in the OSCORE Option of the request from step 2, hence of the EDHOC + OSCORE request.

5. Signal the usage of this approach, by including the new EDHOC Option defined in Section 3.1 into the EDHOC + OSCORE request.

3.3. Server Processing

When receiving a request containing the EDHOC option, i.e., an EDHOC + OSCORE request, the Server MUST perform the following steps.

1. Check that the payload of the EDHOC + OSCORE request is a CBOR sequence composed of two CBOR byte strings. If this is not the

case, the Server MUST stop processing the request and MUST reply with a 4.00 (Bad Request) error response.

2. Extract EDHOC message_3 from the payload of the EDHOC + OSCORE request, as the first CBOR byte string in the CBOR sequence.

3. Take the value of 'kid' from the OSCORE option of the EDHOC + OSCORE request (i.e., the OSCORE Sender ID of the Client), and use it to rebuild the EDHOC connection identifier C_R, as per Section 4.1.

4. Retrieve the correct EDHOC session by using the connection identifier C_R rebuilt at step 3.

   If the applicability statement used in the EDHOC session specifies that EDHOC message_4 shall be sent, the Server MUST stop the EDHOC processing and consider it failed, as due to a client error.

   Otherwise, perform the EDHOC processing on the EDHOC message_3 extracted at step 2 as per Section 5.4.3 of [I-D.ietf-lake-edhoc], based on the protocol state of the retrieved EDHOC session.

   The applicability statement used in the EDHOC session is the same one associated to the EDHOC resource where the server received the request conveying EDHOC message_1 that started the session. This is relevant in case the server provides multiple EDHOC resources, which may generally refer to different applicability statements.

5. Establish a new OSCORE Security Context associated to the client as per Appendix A.2 of [I-D.ietf-lake-edhoc], using the EDHOC output from step 4.

6. Extract the OSCORE ciphertext from the payload of the EDHOC + OSCORE request, as the value of the second CBOR byte string in the CBOR sequence.

7. Rebuild the OSCORE protected CoAP request as the EDHOC + OSCORE request, where the payload is replaced with the OSCORE ciphertext extracted at step 6.

8. Decrypt and verify the OSCORE protected CoAP request rebuilt at step 7, as per Section 8.2 of [RFC8613], by using the OSCORE Security Context established at step 5.

9. Deliver the CoAP request resulting from step 8 to the application.

If steps 4 (EDHOC processing) and 8 (OSCORE processing) are both successfully completed, the Server MUST reply with an OSCORE protected response, in order for the Client to achieve key confirmation (see Section 5.4.2 of [I-D.ietf-lake-edhoc]). The usage of EDHOC message_4 as defined in Section 5.5 of [I-D.ietf-lake-edhoc] is not applicable to the approach defined in this document.

If step 4 (EDHOC processing) fails, the server discontinues the protocol as per Section 5.4.3 of [I-D.ietf-lake-edhoc] and responds with an EDHOC error message with error code 1, formatted as defined in Section 6.2 of [I-D.ietf-lake-edhoc]. In particular, the CoAP response conveying the EDHOC error message MUST have Content-Format set to application/edhoc defined in Section 9.12 of [I-D.ietf-lake-edhoc].

If step 4 (EDHOC processing) is successfully completed but step 8 (OSCORE processing) fails, the same OSCORE error handling as defined in Section 8.2 of [RFC8613] applies.

3.4. Example of EDHOC + OSCORE Request

Figure 5 shows an example of EDHOC + OSCORE Request. In particular, the example assumes that:

* The used OSCORE Partial IV is 0, consistently with the first request protected with the new OSCORE Security Context.

* The OSCORE Sender ID of the Client is 0x01.

  As per Section 4.1, this corresponds to the numeric EDHOC connection identifier C_R with value 1. When using the purely-sequential flow shown in Figure 1, this would be prepended to EDHOC message_3 as the CBOR integer 1 (0x01 in CBOR encoding), in the payload of the second EDHOC request.

* The EDHOC option is registered with CoAP option number 21.

o OSCORE option value: 0x090001 (3 bytes)

o EDHOC option value: - (0 bytes)

o EDHOC message_3: 0x52d5535f3147e85f1cfacd9e78abf9e0a81bbf (19 bytes)

o OSCORE ciphertext: 0x612f1092f1776f1c1668b3825e (13 bytes)

From there:

o Protected CoAP request (OSCORE message):

```
0x44025d1f               ; CoAP 4-byte header
  00003974               ; Token
  39 6c6f63616c686f7374  ; Uri-Host Option: "localhost"
  63 090001              ; OSCORE Option
  c0                     ; EDHOC Option
  ff 52d5535f3147e85f1cfacd9e78abf9e0a81bbf
     4d612f1092f1776f1c1668b3825e
(57 bytes)
```

Figure 5: Example of CoAP message with EDHOC and OSCORE combined

4. Conversion from OSCORE to EDHOC Identifiers

Appendix A.1 of [I-D.ietf-lake-edhoc] defines how an EDHOC connection identifier is converted to an OSCORE Sender/Recipient ID.

In the following, Section 4.1 defines a method for converting from OSCORE Sender/Recipient ID to EDHOC connection identifier.

When running EDHOC through a certain EDHOC resource, the Client and Server MUST both use the conversion method defined in Section 4.1 and MUST perform the additional message processing specified in Section 4.2, if at least one of the following conditions hold.

* The applicability statement associated to the EDHOC resource indicates that the server supports the EDHOC + OSCORE request defined in Section 3.

* The applicability statement associated to the EDHOC resource indicates that the conversion method defined in Section 4.1 is the one to use.

Instead, if none of the above conditions hold, the Client and the Server can independently use any consistent conversion method, such as the one defined in Section 4.1 or different ones defined in separate specifications. In particular, the Client and Server are not required to use the same conversion method. In fact, as per Appendix A.1 of [I-D.ietf-lake-edhoc], it is sufficient that the two connection identifiers C_I and C_R exchanged during an EDHOC execution are different and not "equivalent", hence not convertible to the same OSCORE Sender/Recipient ID.

Even in case none of the above conditions hold, it is RECOMMENDED for the Client and Server to use the conversion method defined in Section 4.1, since it ensures that an OSCORE Sender/Recipient ID is always converted to the EDHOC identifier with the smallest size among the two equivalent ones.

4.1. Conversion Method

The process defined in this section ensures that every OSCORE Sender/Recipient ID is converted to only one of the two corresponding, equivalent EDHOC connection identifiers, see Appendix A.1 of [I-D.ietf-lake-edhoc].

An OSCORE Sender/Recipient ID, OSCORE_ID, is converted to an EDHOC connection identifier, EDHOC_ID, as follows.

* If OSCORE_ID is 0 bytes in size, it is converted to the empty byte string EDHOC_ID (0x40 in CBOR encoding).

* If OSCORE_ID has a size in bytes different than 0, 1, 2, 3, 5 or 9, it is converted to a byte-valued EDHOC_ID, i.e., a CBOR byte string with value OSCORE_ID.

  For example, the OSCORE_ID 0x001122334455 is converted to the byte-valued EDHOC_ID 0x001122334455 (0x46001122334455 in CBOR encoding).

* If OSCORE_ID has a size in bytes equal to 1, 2, 3, 5 or 9 the following applies.

  - If OSCORE_ID is a valid CBOR encoding for an integer value (i.e., it can be correctly decoded as a CBOR integer), then it is converted to a numeric EDHOC_ID having OSCORE_ID as its CBOR encoded form.

    For example, the OSCORE_ID 0x01 is converted to the numeric EDHOC_ID 1 (0x01 in CBOR encoding), while the OSCORE_ID 0x2B is converted to the numeric EDHOC_ID -12 (0x2B in CBOR encoding).

  - If OSCORE_ID is *not* a valid CBOR encoding for an integer value (i.e., it *cannot* be correctly decoded as a CBOR integer), then it is converted to a byte-valued EDHOC_ID having OSCORE_ID as its value.

For example, the OSCORE_ID 0xFF is converted to the byte-valued EDHOC_ID 0xFF (0x41FF in CBOR encoding).

Implementations can easily determine which case holds for a given OSCORE_ID with no need to try to actually CBOR-decode it, e.g., by using the approach in Appendix A.

When performing the conversions above, the two peers MUST always refer to Deterministically Encoded CBOR as specified in Section 4.2.1 of [RFC8949], consistently with what is required by the EDHOC protocol [I-D.ietf-lake-edhoc].

4.2. Additional Processing of EDHOC Messages

This section specifies additional EDHOC message processing compared to what is specified in Section 5 of [I-D.ietf-lake-edhoc].

4.2.1. Initiator Processing of Message 1

The Initiator selects C_I as follows.

1. The Initiator initializes a set ID_SET as the empty set.

2. The Initiator selects an available OSCORE Recipient ID, ID*, which is not included in ID_SET.

3. The Initiator converts ID* to the EDHOC connection identifier C_I, as per Section 4.1.

4. If the resulting C_I is already used, the Initiator adds ID* to ID_SET and moves back to step 2. Otherwise, it uses C_I as its EDHOC connection identifier.

4.2.2. Responder Processing of Message 1

The Responder MUST discontinue the protocol and reply with an EDHOC error message with error code 1, formatted as defined in Section 6.2 of [I-D.ietf-lake-edhoc], if C_I is a CBOR byte string and it has as value a valid CBOR encoding of an integer value (e.g., C_I is CBOR encoded as 0x4100).

In fact, this would mean that the Initiator has not followed the conversion rule in Section 4.1 when converting its (to be) OSCORE Recipient ID to C_I.

4.2.3. Responder Processing of Message 2

The Responder selects C_R as follows.

1. The Responder initializes a set ID_SET as the empty set.

2. The Responder selects an available OSCORE Recipient ID, ID*, which is not included in ID_SET.

3. The Responder converts ID* to the EDHOC connection identifier C_R, as per Section 4.1.

4. If the resulting C_R is already used or it is equal byte-by-byte to the C_I specified in EDHOC message_1, the Initiator adds ID* to ID_SET and moves back to step 2. Otherwise, it uses C_R as its EDHOC connection identifier.

4.2.4. Initiator Processing of Message 2

If any of the following conditions holds, the Initiator MUST discontinue the protocol and reply with an EDHOC error message with error code 1, formatted as defined in Section 6.2 of [I-D.ietf-lake-edhoc].

* C_R is equal byte-by-byte to the C_I that was specified in EDHOC message_1.

* C_R is a CBOR byte string and it has as value a valid CBOR encoding of an integer value (e.g., C_R is CBOR encoded as 0x4100).

  In fact, this would mean that the Responder has not followed the conversion rule in Section 4.1 when converting its (to be) OSCORE Recipient ID to C_R.

5. Extension and Consistency of Applicability Statement

The applicability statement referred by the Client and Server can include the information elements introduced below, in accordance with the specified consistency rules.

If the Server supports the EDHOC + OSCORE request within an EDHOC execution started at a certain EDHOC resource, then the applicability statement associated to that resource:

* MUST NOT specify that EDHOC message_4 shall be sent.

* SHOULD explicitly specify support for the EDHOC + OSCORE request.

* SHOULD explicitly specify that the method to convert from EDHOC to OSCORE identifiers is the one defined in Section 4 and MUST NOT specify any other method than that.

  If the support for the EDHOC + OSCORE request is explicitly specified and the method defined in Section 4 is not explicitly specified, then the Client and Server MUST use it as conversion method.

If the Server does not support the EDHOC + OSCORE request within an EDHOC execution started at a certain EDHOC resource, then the applicability statement associated to that resource MAY specify a method to convert from EDHOC to OSCORE identifiers. In such a case, the Client and Server MUST use the specified conversion method, which MAY be the one defined in Section 4.

6. Web Linking

Section 9.13 of [I-D.ietf-lake-edhoc] registers the resource type "core.edhoc", which can be used as target attribute in a web link [RFC8288] to an EDHOC resource, e.g., using a link-format document [RFC6690]. This enables Clients to discover the presence of EDHOC resources at a Server, possibly using the resource type as filter criterion.

At the same time, the applicability statement associated to an EDHOC resource provides a number of information describing how the EDHOC protocol can be used through that resource. While a Client may become aware of the applicability statement through several means, it would be convenient to obtain its information elements upon discovering the EDHOC resources at the Server. This might aim at discovering especially the EDHOC resources whose associated applicability statement denotes a way of using EDHOC which is most suitable to the Client, e.g., with EDHOC cipher suites or authentication methods that the Client also supports or prefers.

That is, it would be convenient that a Client discovering an EDHOC resource contextually obtains relevant pieces of information from the applicability statement associated to that resource. The resource discovery can occur by means of a direct interaction with the Server, or instead by means of the CoRE Resource Directory [I-D.ietf-core-resource-directory], where the Server may have registered the links to its resources.

In order to enable the above, this section defines a number of parameters, each of which can be optionally specified as a target attribute with the same name in the link to the respective EDHOC resource, or as filter criteria in a discovery request from the Client. When specifying these parameters in a link to an EDHOC resource, the target attribute rt="core.edhoc" MUST be included, and the same consistency rules defined in Section 5 for the corresponding information elements of an applicability statement MUST be followed.

The following parameters are defined.

* 'method', specifying an authentication method supported by the Server. This parameter MUST specify a single value, which is taken from the 'Value' column of the "EDHOC Method Type" registry defined in Section 9.3 of [I-D.ietf-lake-edhoc]. This parameter MAY occur multiple times, with each occurrence specifying a different authentication method.

* 'csuite', specifying an EDHOC ciphersuite supported by the Server. This parameter MUST specify a single value, which is taken from the 'Value' column of the "EDHOC Cipher Suites" registry defined in Section 9.2 of [I-D.ietf-lake-edhoc]. This parameter MAY occur multiple times, with each occurrence specifying a different authentication method.

* 'cred_t', specifying type of authentication credentials supported by the Server. This parameter MAY occur multiple times, with each occurrence specifying a different authentication credential type. Possible values are: "x509", for X.509 certificate [RFC5280]; "c509", for C509 certificate [I-D.ietf-cose-cbor-encoded-cert]; "cwt" for CWT [RFC8392]; "ccs" for CWT Claims Set (CCS) [RFC8392].

* 'idcred_t', specifying the type of identifiers supported by the Server for identifying authentication credentials. This parameter MUST specify a single value, which is taken from the 'Label' column of the "COSE Headers Parameters" registry [COSE.Header.Parameters]. This parameter MAY occur multiple times, with each occurrence specifying a different type of identifier for authentication credentials.

  Note that the values in the 'Label' column of the "COSE Headers Parameters" registry are strongly typed. On the contrary, Link Format is weakly typed and thus does not distinguish between, for instance, the string value "-10" and the integer value -10. Thus, if responses in Link Format are returned, string values which look like an integer are not supported. Therefore, such values MUST NOT be used in the 'idcred_t' parameter.

* 'ead_1', 'ead_2', 'ead_3' and 'ead_4', specifying if the Server supports the use of external authorization data EAD_1, EAD_2, EAD_3 and EAD_4, respectively (see Section 3.8 of [I-D.ietf-lake-edhoc]). For each of these parameters, the following applies.

  - It MUST occur at most once, with its presence denoting support from the server for the respective external authorization data.

- It MUST specify a single value, which is taken from the 'Label' column of the "EDHOC External Authorization Data" registry defined in Section 9.5 of [I-D.ietf-lake-edhoc].

* 'comb_req', specifying whether the server supports the EDHOC + OSCORE request defined in Section 3, with its presence denoting support from the server. A value MUST NOT be given to this parameter and any present value MUST be ignored by parsers.

* 'conv_osc_id', specifying the method used to convert from OSCORE to EDHOC identifiers. If such a method is the one defined in Section 4, this parameter MUST take value 0.

The example in Figure 6 shows how a Client discovers one EDHOC resource at a Server, obtaining information elements from the applicability statement. The Link Format notation from Section 5 of [RFC6690] is used.

```
REQ: GET /.well-known/core

RES: 2.05 Content
    </sensors/temp>;osc,
    </sensors/light>;if="sensor",
    </.well-known/edhoc>;rt="core.edhoc";csuite="0";csuite="2";
    method="0";cred_t="c509";cred_t="ccs";idcred_t="4";comb_req
```

Figure 6: The Web Link

7. Security Considerations

The same security considerations from OSCORE [RFC8613] and EDHOC [I-D.ietf-lake-edhoc] hold for this document.

TODO: more considerations

8. IANA Considerations

RFC Editor: Please replace "[[this document]]" with the RFC number of this document and delete this paragraph.

This document has the following actions for IANA.

8.1. CoAP Option Numbers Registry

IANA is asked to enter the following option number to the "CoAP Option Numbers" registry within the "CoRE Parameters" registry group.

[

The CoAP option numbers 13 and 21 are both consistent with the properties of the EDHOC Option defined in Section 3.1, and they both allow the EDHOC Option to always result in an overall size of 1 byte. This is because:

* The EDHOC option is always empty, i.e., with zero-length value; and

* Since the OSCORE option with option number 9 is always present in the CoAP request, the EDHOC option would be encoded with a maximum delta of 4 or 12, depending on its option number being 13 or 21.

At the time of writing, the CoAP option numbers 13 and 21 are both unassigned in the "CoAP Option Numbers" registry, as first available and consistent option numbers for the EDHOC option.

This document suggests 21 (TBD21) as option number to be assigned to the new EDHOC option, since both 13 and 21 are consistent for the use case in question, but different use cases or protocols may make better use of the option number 13.

]

Number	Name	Reference
TBD21	EDHOC	[[this document]]

9. References

9.1. Normative References

[COSE.Header.Parameters] IANA, "COSE Header Parameters", <https://www.iana.org/assignments/cose/cose.xhtml#header-parameters>.

[I-D.ietf-core-resource-directory] Amsüss, C., Shelby, Z., Koster, M., Bormann, C., and P. V. D. Stok, "CoRE Resource Directory", Work in Progress, Internet-Draft, draft-ietf-core-resource-directory-28, 7 March 2021, <https://www.ietf.org/archive/id/draft-ietf-core-resource-directory-28.txt>.

[I-D.ietf-lake-edhoc] Selander, G., Mattsson, J. P., and F. Palombini, "Ephemeral Diffie-Hellman Over COSE (EDHOC)", Work in Progress, Internet-Draft, draft-ietf-lake-edhoc-12, 20 October 2021, <https://www.ietf.org/archive/id/draft-ietf-lake-edhoc-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>.

[RFC6690] Shelby, Z., "Constrained RESTful Environments (CoRE) Link Format", RFC 6690, DOI 10.17487/RFC6690, August 2012, <https://www.rfc-editor.org/info/rfc6690>.

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

[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>.

[RFC8288] Nottingham, M., "Web Linking", RFC 8288, DOI 10.17487/RFC8288, October 2017, <https://www.rfc-editor.org/info/rfc8288>.

[RFC8613] Selander, G., Mattsson, J., Palombini, F., and L. Seitz, "Object Security for Constrained RESTful Environments (OSCORE)", RFC 8613, DOI 10.17487/RFC8613, July 2019, <https://www.rfc-editor.org/info/rfc8613>.

[RFC8742] Bormann, C., "Concise Binary Object Representation (CBOR) Sequences", RFC 8742, DOI 10.17487/RFC8742, February 2020, <https://www.rfc-editor.org/info/rfc8742>.

[RFC8949] Bormann, C. and P. Hoffman, "Concise Binary Object Representation (CBOR)", STD 94, RFC 8949, DOI 10.17487/RFC8949, December 2020, <https://www.rfc-editor.org/info/rfc8949>.

9.2. Informative References

[I-D.ietf-cose-cbor-encoded-cert]
Mattsson, J. P., Selander, G., Raza, S., Höglund, J., and M. Furuhed, "CBOR Encoded X.509 Certificates (C509 Certificates)", Work in Progress, Internet-Draft, draft-ietf-cose-cbor-encoded-cert-02, 12 July 2021, <https://www.ietf.org/archive/id/draft-ietf-cose-cbor-encoded-cert-02.txt>.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation

List (CRL) Profile", RFC 5280, DOI 10.17487/RFC5280, May 2008, <https://www.rfc-editor.org/info/rfc5280>.

[RFC8392] Jones, M., Wahlstroem, E., Erdtman, S., and H. Tschofenig, "CBOR Web Token (CWT)", RFC 8392, DOI 10.17487/RFC8392, May 2018, <https://www.rfc-editor.org/info/rfc8392>.

Appendix A. Checking CBOR Encoding of Numeric Values

A binary string of N bytes in size is a valid CBOR encoding of an integer value if and only if both the following conditions hold, with reference to the table below.

* The size N is one of the values specified in the "Size" column.

* The first byte of the binary string is equal to one of the byte values specified in the "First byte" column, exactly for the row having N as value of the "Size" column.

Size	First byte
1	0x00-0x17 , 0x20-0x37
2	0x18 , 0x38
3	0x19 , 0x39
5	0x1A , 0x3A
9	0x1B , 0x3B

Appendix B. Document Updates

RFC Editor: Please remove this section.

B.1. Version -01 to -02

* New title, abstract and introduction.

* Restructured table of content.

* Alignment with latest format of EDHOC messages.

* Guideline on ID conversions based on applicability statement.

* Clarifications, extension and consistency on applicability statement.

* Section on web-linking.

* RFC8126 terminology in IANA considerations.

* Revised Appendix "Checking CBOR Encoding of Numeric Values".

B.2. Version -00 to -01

* Improved background overview of EDHOC.

* Added explicit rules for converting OSCORE Sender/Recipient IDs to EDHOC connection identifiers following the removal of bstr_identifier from EDHOC.

* Revised section organization.

* Recommended number for EDHOC option changed to 21.

* Editorial improvements.

Acknowledgments

The authors sincerely thank Christian Amsuess, Klaus Hartke, Jim Schaad and Malisa Vucinic for their feedback and comments.

The work on this document has been partly supported by VINNOVA and the Celtic-Next project CRITISEC; and by the H2020 project SIFIS-Home (Grant agreement 952652).

Authors' Addresses

Francesca Palombini
Ericsson

Email: francesca.palombini@ericsson.com

Marco Tiloca
RISE AB
Isafjordsgatan 22
SE-16440 Stockholm Kista
Sweden

Email: marco.tiloca@ri.se

Rikard Hoeglund
RISE AB
Isafjordsgatan 22
SE-16440 Stockholm Kista
Sweden

Email: rikard.hoglund@ri.se

Stefan Hristozov
Fraunhofer AISEC

Email: stefan.hristozov@aisec.fraunhofer.de

Goeran Selander
Ericsson

Email: goran.selander@ericsson.com