CoRE Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 23, 2021

F. Palombini
Ericsson
M. Tiloca
R. Hoeglund
RISE AB
S. Hristozov
Fraunhofer AISEC
G. Selander
Ericsson

February 19, 2021

Combining EDHOC and OSCORE
draft-palombini-core-oscore-edhoc-02

Abstract

This document defines an optimization approach for combining the lightweight authenticated key exchange protocol EDHOC run over CoAP 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.

Table of Contents

1. Introduction

This document defines an optimization approach to combine the lightweight authenticated key exchange protocol EDHOC [I-D.ietf-lake-edhoc], when running over CoAP [RFC7252], with the first subsequent OSCORE [RFC8613] transaction.

This allows for a minimum number of round trips necessary to setup the OSCORE Security Context and complete an OSCORE transaction, for example 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 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 of [I-D.ietf-lake-edhoc]), thus allowing additional OSCORE protected CoAP data within target MTU sizes.

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. Background

EDHOC is a 3-message key exchange protocol. Section 7.2 of [I-D.ietf-lake-edhoc] specifies how to transport EDHOC over CoAP: the EDHOC data (referred to as "EDHOC messages") are transported in the payload of CoAP requests and responses.

This draft deals with the case of the Initiator acting as CoAP Client and the Responder acting as CoAP Server; instead, the case of the Initiator acting as CoAP Server cannot be optimized by using this approach.

That is, the CoAP Client sends a POST request containing EDHOC message_1 to a reserved resource at the CoAP Server. This triggers the EDHOC exchange on the CoAP Server, which replies with a 2.04 (Changed) Response containing EDHOC message_2. Finally, the CoAP Client sends EDHOC message_3, as a CoAP POST request to the same resource used for EDHOC message_1. The Content-Format of these CoAP messages may be set to "application/edhoc".

After this exchange takes place, and after successful verifications specified in the EDHOC protocol, the Client and Server derive the OSCORE Security Context, as specified in Section 7.2.1 of [I-D.ietf-lake-edhoc]. Then, they are ready to use OSCORE.

This sequential way of running EDHOC and then OSCORE is specified in Figure 1. As shown in the figure, this mechanism takes 3 round trips to complete.

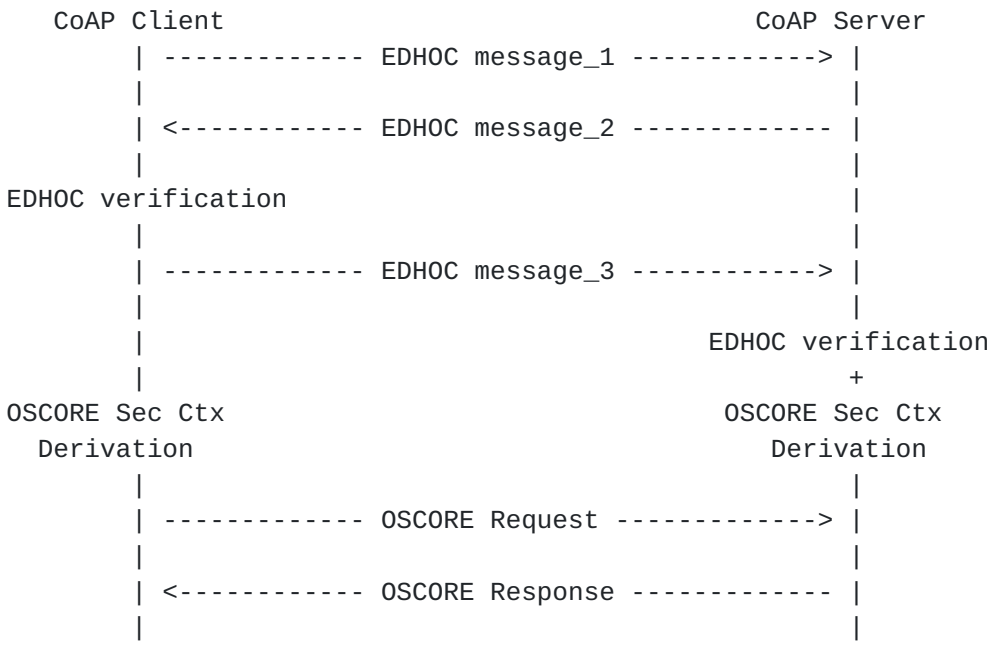

Figure 1: EDHOC and OSCORE run sequentially

The number of roundtrips can be minimized as follows. Already after receiving EDHOC message_2 and before sending EDHOC message_3, the CoAP Client has all the information needed to derive the OSCORE Security Context.

This means that the Client can potentially send at the same time both EDHOC message_3 and the subsequent OSCORE Request. On a semantic level, this approach practically requires to send two separate REST requests at the same time.

The high level message flow of running EDHOC and OSCORE combined is shown in Figure 2.

Defining the specific details of how to transport the data and of their processing order is the goal of this specification, as defined in Section 4.

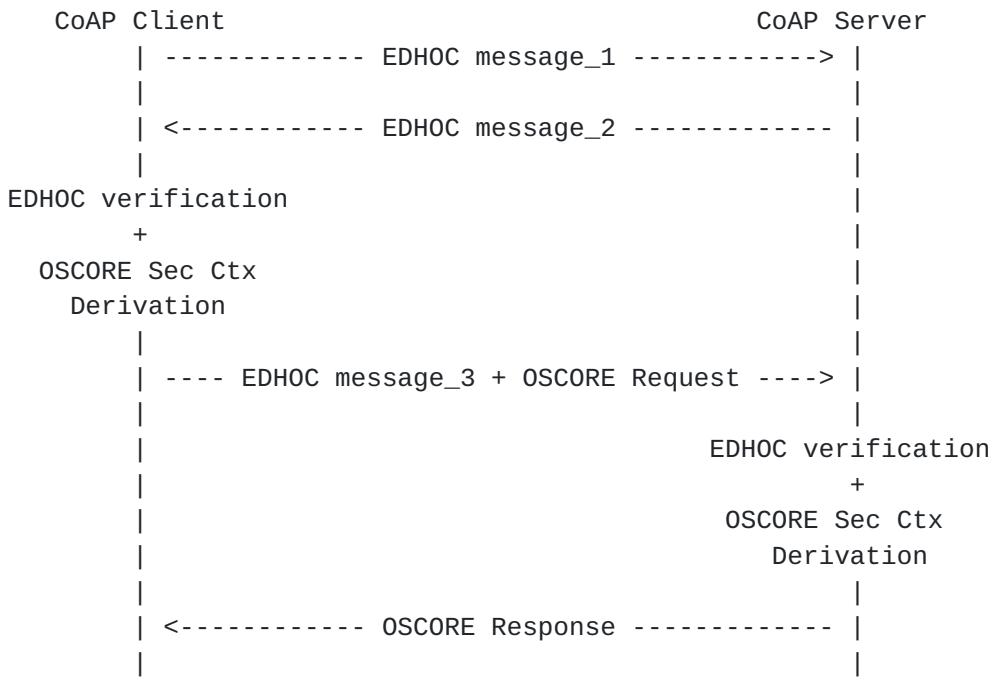

Figure 2: EDHOC and OSCORE combined

3. EDHOC Option

This section defines the EDHOC Option, used in a CoAP request to signal that the request combines EDHOC message_3 and OSCORE protected data.

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
TBD13	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 4.2, 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.

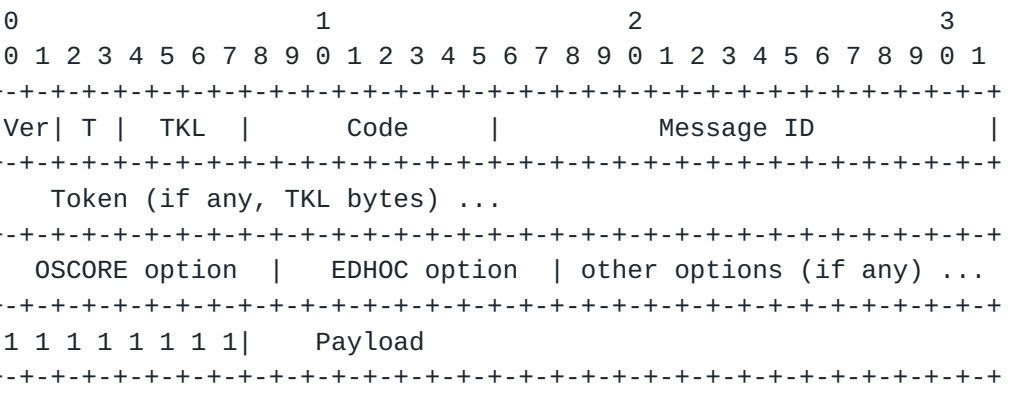

Figure 4: CoAP message for EDHOC and OSCORE combined - signalled with the EDHOC Option

4. EDHOC Combined with OSCORE

The approach defined in this specification consists of sending EDHOC message_3 inside an OSCORE protected CoAP message.

The resulting EDHOC + OSCORE request is in practice the OSCORE Request from Figure 1, sent to a protected resource and with the correct CoAP method and options, with the addition that it also transports EDHOC message_3.

Since 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 through the CoAP payload.

The use of this approach is explicitly signalled by including an EDHOC Option (see Section 3) in the EDHOC + OSCORE request.

4.1. 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].

   Since the Client is the EDHOC Initiator and the used Correlation Method is 1 (see Section 3.2.4 of [I-D.ietf-lake-edhoc]), the EDHOC message_3 always includes the Connection Identifier C_R and CIPHERTEXT_3. Note that C_R is the OSCORE Sender ID of the

Client, encoded as a bstr_identifier (see Section 5.1 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 CIPHERTEXT_3 of the EDHOC message_3 resulting from step 3.

   * 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.

5. Signal the usage of this approach within the EDHOC + OSCORE request, by including the new EDHOC Option defined in Section 3.

4.2. Server Processing

When receiving an EDHOC + OSCORE request, the Server performs the following steps.

1. Check the presence of the EDHOC option defined in Section 3, to determine that the received request is an EDHOC + OSCORE request. If this is the case, the Server continues with the steps defined below.

2. Extract CIPHERTEXT_3 from the payload of the EDHOC + OSCORE request, as the first CBOR byte string in the CBOR sequence.

3. Rebuild EDHOC message_3, as a CBOR sequence composed of two CBOR byte strings in the following order.

   * The first CBOR byte string is the 'kid' of the Client indicated in the OSCORE option of the EDHOC + OSCORE request, encoded as a bstr_identifier (see Section 5.1 of [I-D.ietf-lake-edhoc]).

* The second CBOR byte string is the CIPHERTEXT_3 retrieved at step 2.

4. Perform the EDHOC processing on the EDHOC message_3 rebuilt at step 3, including verifications, and the OSCORE Security Context derivation, as per Section 5.4.3 and Section 7.2.1 of [I-D.ietf-lake-edhoc], respectively.

5. 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.

6. Rebuild the OSCORE protected CoAP request as the EDHOC + OSCORE request, where the payload is replaced with the OSCORE ciphertext resulting from step 5.

7. Decrypt and verify the OSCORE protected CoAP request resulting from step 6, as per Section 8.2 of [RFC8613], by using the new OSCORE Security Context established at step 4.

8. Process the CoAP request resulting from step 7.

If steps 4 (EDHOC processing) and 7 (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 7.1 of [I-D.ietf-lake-edhoc] is not applicable to the approach defined in this specification.

If step 4 (EDHOC processing) fails, the server discontinues the protocol as per Section 5.4.3 of [I-D.ietf-lake-edhoc] and sends an EDHOC error message, formatted as defined in Section 6.1 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 9.5 of [I-D.ietf-lake-edhoc].

- MUST specify a CoAP error response code, i.e. 4.00 (Bad Request) in case of client error (e.g. due to a malformed EDHOC message_3), or 5.00 (Internal Server Error) in case of server error (e.g. due to failure in deriving EDHOC key material).

If step 4 (EDHOC processing) is successfully completed but step 7 (OSCORE processing) fails, the same OSCORE error handling applies as defined in Section 8.2 of [RFC8613].

5. Example of EDHOC + OSCORE Request

An example based on the OSCORE test vector from Appendix C.4 of [RFC8613] and the EDHOC test vector from Appendix B.2 of [I-D.ietf-lake-edhoc] is given in Figure 5. 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 0x20. This corresponds to the EDHOC Connection Identifier C_R, which is encoded as the bstr_identifier 0x08 in EDHOC message_3.
- The EDHOC option is registered with CoAP option number 13.
    - OSCORE option value: 0x090020 (3 bytes)
    - EDHOC option value: - (0 bytes)
    - C_R: 0x20 (1 byte)
    - CIPHERTEXT_3: 0x5253c3991999a5ffb86921e99b607c067770e0 (19 bytes)
    - EDHOC message_3: 0x08 5253c3991999a5ffb86921e99b607c067770e0 (20 bytes)
    - OSCORE ciphertext: 0x612f1092f1776f1c1668b3825e (13 bytes)

    From there:

    - Protected CoAP request (OSCORE message):

```
0x44025d1f               ; CoAP 4-byte header
  00003974               ; Token
  39 6c6f63616c686f7374  ; Uri-Host Option: "localhost"
  63 090020              ; OSCORE Option
  40                     ; EDHOC Option
  ff 5253c3991999a5ffb86921e99b607c067770e0
     4d612f1092f1776f1c1668b3825e
(57 bytes)
```

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

6. Security Considerations

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

TODO (more considerations)

7. IANA Considerations

This document has the following actions for IANA.

7.1. CoAP Option Numbers Registry

IANA is asked to enter the following option numbers to the "CoAP Option Numbers" registry defined in [RFC7252] within the "CoRE Parameters" registry.

[

The CoAP option numbers 13 and 21 are both consistent with the properties of the EDHOC Option defined in Section 3, 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.

]

Number	Name	Reference
TBD13	EDHOC	[[this document]]

8. Normative References

[I-D.ietf-lake-edhoc]
Selander, G., Mattsson, J., and F. Palombini, "Ephemeral Diffie-Hellman Over COSE (EDHOC)", draft-ietf-lake-edhoc-03 (work in progress), December 2020.

[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>.

[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>.

[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>.

Acknowledgments

The authors sincerely thank Christian Amsuess, Klaus Hartke, Jim Schaad and Malisa Vucinic for their feedback and comments in the discussion leading up to this draft.

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
Kista SE-16440 Stockholm
Sweden

Email: marco.tiloca@ri.se

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

Email: rikard.hoglund@ri.se

Stefan Hristozov
Fraunhofer AISEC

Email: stefan.hristozov@aisec.fraunhofer.de

Goeran Selander
Ericsson

Email: goran.selander@ericsson.com