Stream: Internet Engineering Task Force (IETF)
RFC: 9668
Category: Standards Track
Published: October 2024
ISSN: 2070-1721
Authors: F. Palombini *Ericsson AB*; M. Tiloca *RISE AB*; R. Höglund *RISE AB*; S. Hristozov *Fraunhofer AISEC*; G. Selander *Ericsson*

RFC 9668
Using Ephemeral Diffie-Hellman Over COSE (EDHOC) with the Constrained Application Protocol (CoAP) and Object Security for Constrained RESTful Environments (OSCORE)

Abstract

The lightweight authenticated key exchange protocol Ephemeral Diffie-Hellman Over COSE (EDHOC) can be run over the Constrained Application Protocol (CoAP) and used by two peers to establish a Security Context for the security protocol Object Security for Constrained RESTful Environments (OSCORE). This document details this use of the EDHOC protocol by specifying a number of additional and optional mechanisms, including an optimization approach for combining the execution of EDHOC with the first 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.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

Ephemeral Diffie-Hellman Over COSE (EDHOC) [RFC9528] is a lightweight authenticated key exchange protocol that is specifically 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 details the use of the EDHOC protocol with CoAP and OSCORE and specifies a number of additional and optional mechanisms. These include an optimization approach that combines the EDHOC execution with the first OSCORE transaction (see Section 3). This allows for a minimum number of two round trips necessary to set up the OSCORE Security Context and complete an OSCORE transaction, e.g., when an Internet of Things (IoT) device gets configured in a network for the first time.

This optimization is desirable since the number of message exchanges 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 to achieve the minimum number of two round trips. This optimization makes it possible since the message_3 of the EDHOC protocol can be made relatively small (see Section 1.2 of [RFC9528]), thus allowing additional OSCORE-protected CoAP data within target MTU sizes.

The minimum number of two round trips can be achieved only if the default forward message flow of EDHOC is used, i.e., when a CoAP client acts as EDHOC Initiator and a CoAP server acts as EDHOC Responder. The performance advantage of using this optimization can be lost when used in combination with Block-wise transfers [RFC7959] that rely on specific parameter values and block sizes.

Furthermore, this document defines a number of parameters corresponding to different information elements of an EDHOC application profile (see Section 6). These parameters can be specified as target attributes in the link to an EDHOC resource associated with that application profile, thus enabling an enhanced discovery of such a 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], Concise Binary Object Representation (CBOR) [RFC8949], OSCORE [RFC8613], and EDHOC [RFC9528].

2. EDHOC Overview

This section is not normative and summarizes what is specified in [RFC9528] (specifically Appendix A.2 of [RFC9528]). Thus, it provides a baseline for the enhancements in the subsequent sections.

The EDHOC protocol specified in [RFC9528] allows two peers to agree on a cryptographic secret in a mutually-authenticated way and achieves forward secrecy by using Diffie-Hellman ephemeral keys. The two peers are denoted as the "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 establish an OSCORE Security Context [RFC8613]. The Responder can also send an optional EDHOC message_4 in order for the Initiator to achieve key confirmation, e.g., in deployments where no protected application message is sent from the Responder to the Initiator.

Appendix A.2 of [RFC9528] specifies how to transfer EDHOC over CoAP. That is, the EDHOC data (i.e., the EDHOC message possibly with a prepended connection identifier) is transported in the payload of CoAP requests and responses. The default forward message flow of EDHOC consists in the CoAP client acting as Initiator and the CoAP server acting as Responder (see Appendix A.2.1 of [RFC9528]). Alternatively, the two roles can be reversed as per the reverse message flow of EDHOC (see Appendix A.2.2 of [RFC9528]). In the rest of this document, EDHOC messages are considered to be transferred over CoAP.

Figure 1 shows a successful execution of EDHOC, with a CoAP client and a CoAP server running EDHOC as Initiator and Responder, respectively. In particular, it extends Figure 10 from Appendix A.2.1 of [RFC9528] by highlighting when the two peers perform EDHOC verification and establish the OSCORE Security Context, and by adding an exchange of OSCORE-protected CoAP messages after completing the EDHOC execution.

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 encoded as per Section 3.3 of [RFC9528]. The request has Content-Format application/cid-edhoc+cbor-seq.

This triggers the EDHOC execution 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 encoded as per Section 3.3 of [RFC9528]. The response has Content-Format application/edhoc+cbor-seq.

Finally, the client sends a POST request to the same EDHOC resource used earlier when it sent EDHOC message_1. The request payload consists of the EDHOC connection identifier C_R encoded as per Section 3.3 of [RFC9528] concatenated with EDHOC message_3. The request has Content-Format application/cid-edhoc+cbor-seq.

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.1 of [RFC9528]. After that, the client and server can use OSCORE to protect their communications as per [RFC8613]. Note that the EDHOC connection identifier C_R is used as the OSCORE Sender ID of the client (see Appendix A.1 of [RFC9528]). Therefore, C_R is transported in the 'kid' field of the OSCORE option of the OSCORE Request (see Section 6.1 of [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 application profile associated with the EDHOC resource addressed (see Section 3.9 of [RFC9528]).

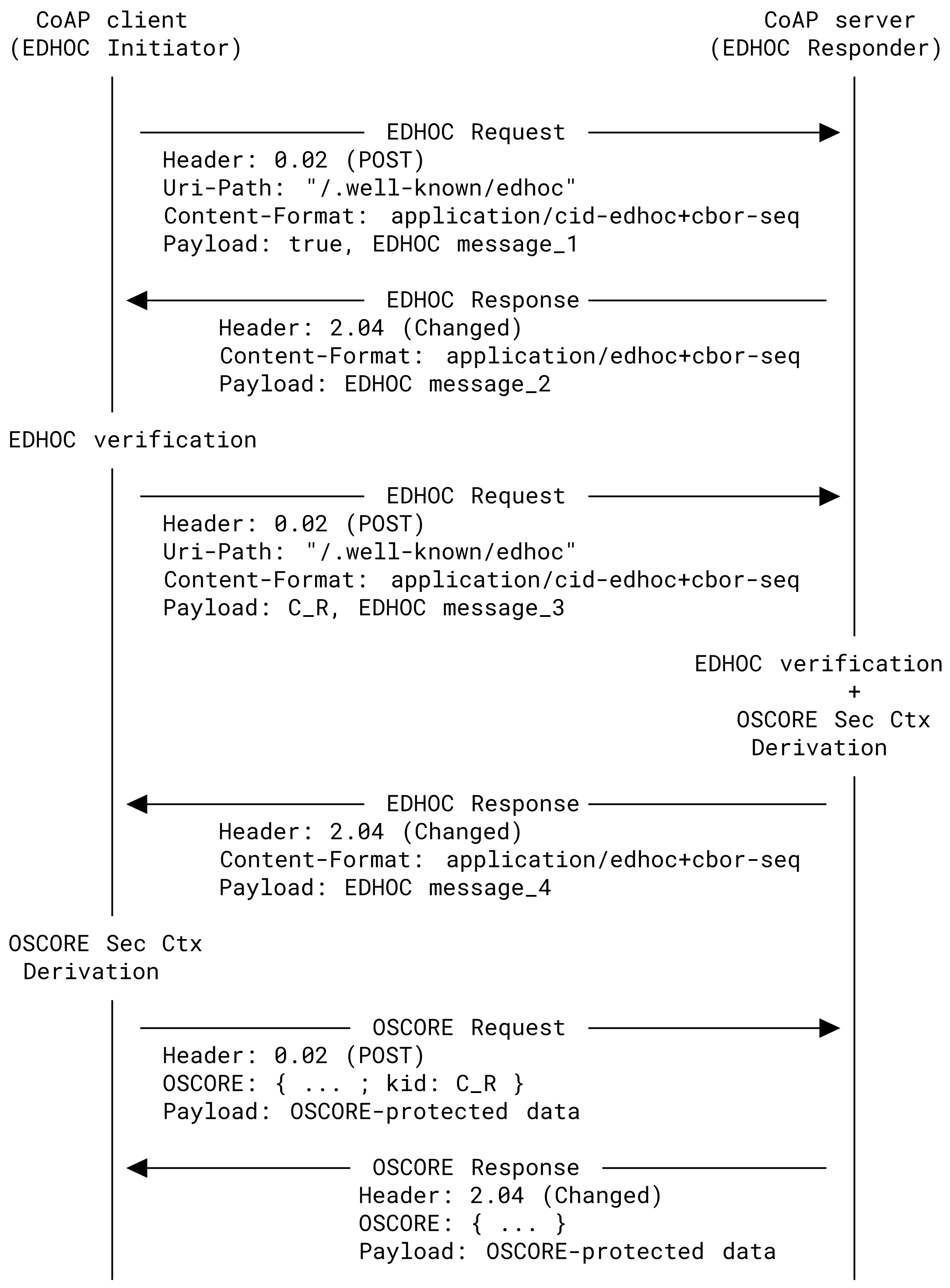

Figure 1: Sequential Flow of EDHOC and OSCORE with the Optional message_4 Included

The sequential flow of EDHOC and OSCORE (where EDHOC runs first and OSCORE is used after) takes three round trips to complete, as shown in Figure 1.

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

3. EDHOC Combined with OSCORE

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

To this end, this approach can be used only if the default forward message flow of EDHOC is used, i.e., when the client acts as Initiator and the server acts as Responder. The same is not possible in the case with reversed roles as per the reverse message flow of EDHOC.

When running the 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 shown in Figure 2.

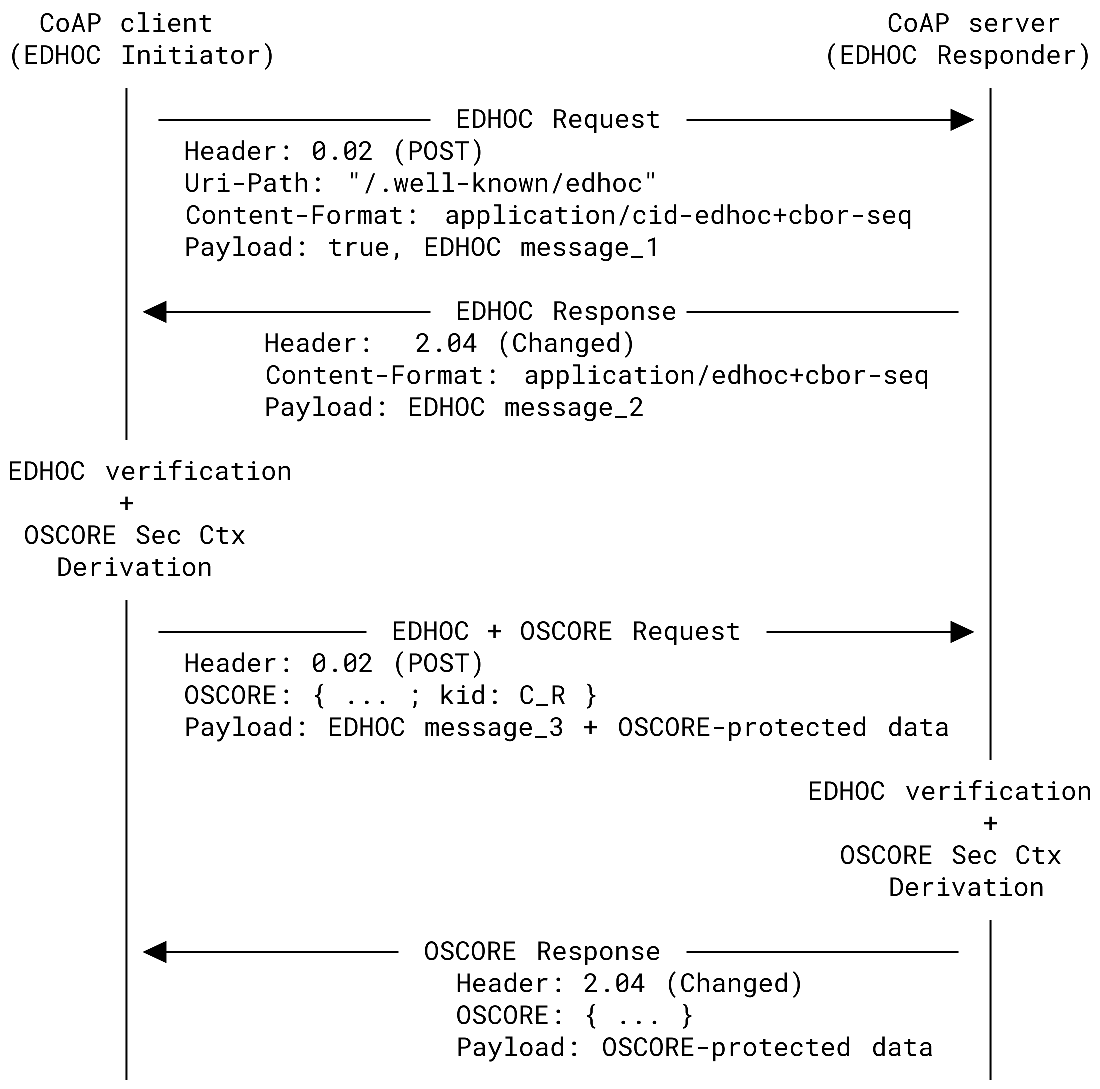

Figure 2: EDHOC and OSCORE Combined

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 composed of the following two parts combined together in a single CoAP message. The steps for processing the EDHOC + OSCORE request and the two parts combined in the request itself are defined in Sections 3.2.1 and 3.3.1.

- The OSCORE Request from Figure 1, which, in this case, is also sent to a protected resource with the correct CoAP method and options intended for accessing that resource.

- EDHOC data consisting of the pair (C_R, EDHOC message_3) required for completing the EDHOC session transported as follows:
  - C_R is the OSCORE Sender ID of the client; hence, it is transported in the 'kid' field of the OSCORE option (see Section 6.1 of [RFC8613]). Unlike the sequential workflow shown in Figure 1, C_R is not transported in the payload of the EDHOC + OSCORE request.
  - EDHOC message_3 is transported in the payload of the EDHOC + OSCORE request and prepended to the payload of the OSCORE Request. This is because EDHOC message_3 may be too large to be included in a CoAP option, e.g., when conveying a large public key certificate chain in the ID_CRED_I field (see Section 3.5.3 of [RFC9528]), or when conveying large External Authorization Data in the EAD_3 field (see Section 3.8 of [RFC9528]).

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 (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. This 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 Table 1, 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 **MUST** be empty. If any value is sent, the recipient **MUST** ignore it. (Future documents may update the definition of the option by expanding its semantics and specifying admitted values.) 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
21	x				EDHOC	Empty	0	(none)

Table 1: The EDHOC Option. C=Critical, U=Unsafe, N=NoCacheKey, R=Repeatable

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

Figure 3 shows an example of a CoAP message that is transported over UDP and that contains 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) ...
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 | Observe Option| OSCORE Option ...
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 | EDHOC Option  | Other Options (if any) ...
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |1 1 1 1 1 1 1 1| Payload ...
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 3: Example of a CoAP Message Containing the Combined EDHOC and OSCORE Data, Signalled by the EDHOC Option and Transported over UDP

3.2. Client Processing

This section describes the processing on the client side.

3.2.1. Processing of the EDHOC + OSCORE Request

The client prepares an EDHOC + OSCORE request as follows.

Step 1. Compose EDHOC message_3 into EDHOC_MSG_3 as per Section 5.4.2 of [RFC9528].

Step 2. Establish the new OSCORE Security Context and use it to encrypt the original CoAP request as per Section 8.1 of [RFC8613].

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.

Step 3. Build COMB_PAYLOAD as the concatenation of EDHOC_MSG_3 and OSCORE_PAYLOAD in the order of COMB_PAYLOAD = EDHOC_MSG_3 | OSCORE_PAYLOAD, where | denotes byte string concatenation and:

- EDHOC_MSG_3 is the binary encoding of EDHOC message_3 resulting from Step 1. As per Section 5.4.1 of [RFC9528], EDHOC message_3 consists of one CBOR data item CIPHERTEXT_3, which is a CBOR byte string. Therefore, EDHOC_MSG_3 is the binary encoding of CIPHERTEXT_3.
- OSCORE_PAYLOAD is the OSCORE ciphertext of the OSCORE-protected CoAP request resulting from Step 2.

Step 4. Compose the EDHOC + OSCORE request, as the OSCORE-protected CoAP request resulting from Step 2, where the payload is replaced with COMB_PAYLOAD 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 is the OSCORE Sender ID of the client, which is already specified as the value of the 'kid' field in the OSCORE option of the request from Step 2; hence, C_R is specified as the value of the 'kid' field of the EDHOC + OSCORE request.

Step 5. Include the new EDHOC option defined in Section 3.1 into the EDHOC + OSCORE request.

The application/cid-edhoc+cbor-seq media type does not apply to this message, whose media type is unnamed.

Step 6. Send the EDHOC + OSCORE request to the server.

With the same server, the client **SHOULD NOT** have multiple simultaneous outstanding interactions (see Section 4.7 of [RFC7252]), such that they consist of an EDHOC + OSCORE request and their EDHOC data pertains to the EDHOC session with the same connection identifier C_R.

An exception might apply for clients that operate under particular time constraints over particularly unreliable networks, thus raising the chances to promptly complete the EDHOC execution with the server through multiple simultaneous EDHOC + OSCORE requests. As discussed in Section 7, this does not have any impact in terms of security.

3.2.2. Supporting Block-Wise Transfers

If Block-wise transfers [RFC7959] are supported, the client may fragment the first CoAP application request before protecting it as an original message with OSCORE as defined in Section 4.1.3.4.1 of [RFC8613].

In such a case, the OSCORE processing in Step 2 of Section 3.2.1 is performed on each inner block of the first CoAP application request. The following also applies.

- The client takes the following additional step between Steps 2 and 3 of Section 3.2.1.

  Step 2.1. If the OSCORE-protected request from Step 2 conveys a non-first inner block of the first CoAP application request (i.e., the Block1 option processed at Step 2 had NUM different than 0), then the client skips the following steps and sends the OSCORE-protected request to the server. In particular, the client **MUST NOT** include the EDHOC option in the OSCORE-protected request.
- The client takes the following additional step between Steps 3 and 4 of Section 3.2.1.

  Step 3.1. If the size of COMB_PAYLOAD exceeds MAX_UNFRAGMENTED_SIZE (see Section 4.1.3.4.2 of [RFC8613]), the client **MUST** stop processing the request and **MUST** abandon the Block-wise transfer. Then, the client can continue by switching to the sequential workflow shown in Figure 1. That is, the client first sends EDHOC message_3 prepended by the EDHOC connection identifier C_R encoded as per Section 3.3 of [RFC9528]. Then, the client sends the OSCORE-protected CoAP request once the EDHOC execution is completed.

The performance advantage of using the EDHOC + OSCORE request can be lost when used in combination with Block-wise transfers that rely on specific parameter values and block sizes. Application policies at the CoAP client can define when and how to detect whether the performance advantage is lost. If that is the case, they can also define whether to appropriately adjust the parameter values and block sizes or to fall back on the sequential workflow of EDHOC.

3.3. Server Processing

This section describes the processing on the server side.

3.3.1. Processing of the EDHOC + OSCORE Request

In order to process a request containing the EDHOC option, i.e., an EDHOC + OSCORE request, the server **MUST** perform the following steps.

Step 1. Check that the EDHOC + OSCORE request includes the OSCORE option and that the request payload has the format defined at Step 3 of Section 3.2.1 for COMB_PAYLOAD. If this is not the case, the server **MUST** stop processing the request and **MUST** reply with a 4.00 (Bad Request) error response.

Step 2. Extract EDHOC message_3 from the payload COMB_PAYLOAD of the EDHOC + OSCORE request as the first element EDHOC_MSG_3 (see Step 3 of Section 3.2.1).

Step 3. Take the value of the 'kid' field from the OSCORE option of the EDHOC + OSCORE request (i.e., the OSCORE Sender ID of the client), and use it as the EDHOC connection identifier C_R.

Step 4. Retrieve the correct EDHOC session by using the connection identifier C_R from Step 3.

If the application profile used in the EDHOC session specifies that EDHOC message_4 shall be sent, the server **MUST** stop the EDHOC processing and consider it failed 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 [RFC9528] based on the protocol state of the retrieved EDHOC session.

The application profile used in the EDHOC session is the same one associated with 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 that may generally refer to different application profiles.

Step 5. Establish a new OSCORE Security Context associated with the client as per Appendix A.1 of [RFC9528] using the EDHOC output from Step 4.

Step 6. Extract the OSCORE ciphertext from the payload COMB_PAYLOAD of the EDHOC + OSCORE request as the second element OSCORE_PAYLOAD (see Step 3 of Section 3.2.1).

Step 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. Then, remove the EDHOC option.

Step 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.

When the decrypted request is checked for any critical CoAP options (as it is during regular CoAP processing), the presence of an EDHOC option **MUST** be regarded as an unprocessed critical option unless it is processed by some further mechanism.

Step 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 (see Section 5.4.3 of [RFC9528]). The usage of EDHOC message_4 as defined in Section 5.5 of [RFC9528] is not applicable to the approach defined in this document.

If Step 4 (EDHOC processing) fails, the server aborts the session as per Section 5.4.3 of [RFC9528] and responds with an EDHOC error message with error code 1, which is formatted as defined in Section 6.2 of [RFC9528]. The server **MUST NOT** establish a new OSCORE Security Context from the present EDHOC session with the client. The CoAP response conveying the EDHOC error message is not protected with OSCORE. As per Section 9.5 of [RFC9528], the server has to make sure that the error message does not reveal sensitive information. The CoAP response conveying the EDHOC error message **MUST** have Content-Format set to application/edhoc+cbor-seq registered in Section 10.9 of [RFC9528].

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.

3.3.2. Supporting Block-Wise Transfers

If Block-wise transfers [RFC7959] are supported, the server takes the additional following step before any other in Section 3.3.1.

Step 0. If a Block option is present in the request, then process the Outer Block options according to [RFC7959] until all blocks of the request have been received (see Section 4.1.3.4 of [RFC8613]).

3.4. Example of the EDHOC + OSCORE Request

Figure 4 shows an example of an EDHOC + OSCORE request transported over UDP. In particular, the example assumes that:

- The OSCORE Partial IV in use 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 3.3.3 of [RFC9528], this straightforwardly corresponds to the EDHOC connection identifier C_R 0x01.

> As per Section 3.3.2 of [RFC9528], when using the sequential flow shown in Figure 1, the same C_R with a value of 0x01 would be encoded on the wire as the CBOR integer 1 (0x01 in CBOR encoding) and prepended to EDHOC message_3 in the payload of the second EDHOC request.

This results in the following components shown in Figure 4:

OSCORE option value: 0x090001 (3 bytes)

EDHOC option value: - (0 bytes)

EDHOC message_3: 0x52d5535f3147e85f1cfacd9e78abf9e0a81bbf (19 bytes)

OSCORE ciphertext: 0x612f1092f1776f1c1668b3825e (13 bytes)

```
    0x44025d1f                ; CoAP 4-byte Header
      00003974                ; Token
      93 090001               ; OSCORE Option
      c0                      ; EDHOC Option
      ff 52d5535f3147e85f1cfacd9e78abf9e0a81bbf
         612f1092f1776f1c1668b3825e
    (46 bytes)
```

Figure 4: Example of a Protected CoAP Request Combining EDHOC and OSCORE Data

4. Use of EDHOC Connection Identifiers with OSCORE

The OSCORE Sender/Recipient IDs are the EDHOC connection identifiers (see Section 3.3.3 of [RFC9528]). This applies also to the optimized workflow defined in Section 3 of this document.

Note that the value of the 'kid' field in the OSCORE option of the EDHOC + OSCORE request is both the server's Recipient ID (i.e., the client's Sender ID) and the EDHOC connection identifier C_R of the server at Step 3 of Section 3.3.1.

4.1. Additional Processing of EDHOC Messages

When using EDHOC to establish an OSCORE Security Context, the client and server **MUST** perform the following additional steps during an EDHOC execution, thus extending Section 5 of [RFC9528].

4.1.1. Initiator Processing of Message 1

The Initiator selects an EDHOC connection identifier C_I as follows.

The Initiator **MUST** choose a C_I that is neither used in any current EDHOC session as this peer's EDHOC connection identifier nor the Recipient ID in a current OSCORE Security Context where the ID Context is not present.

The chosen C_I **SHOULD NOT** be the Recipient ID of any current OSCORE Security Context. Note that, unless the two peers concurrently use alternative methods to establish OSCORE Security Contexts, this allows the Responder to always omit the 'kid context' in the OSCORE option of its messages sent to the Initiator when protecting those with an OSCORE Security Context where C_I is the Responder's OSCORE Sender ID (see Section 6.1 of [RFC8613]).

4.1.2. Responder Processing of Message 2

The Responder selects an EDHOC connection identifier C_R as follows.

The Responder **MUST** choose a C_R that is none of the following:

- used in any current EDHOC session as this peer's EDHOC connection identifier,
- equal to the EDHOC connection identifier C_I specified in the EDHOC message_1 of the present EDHOC session, or
- the Recipient ID in a current OSCORE Security Context where the ID Context is not present.

The chosen C_R **SHOULD NOT** be the Recipient ID of any current OSCORE Security Context. Note that, for a reason analogous to the one given in Section 4.1.1 with C_I, this allows the Initiator to always omit the 'kid context' in the OSCORE option of its messages sent to the Responder when protecting those with an OSCORE Security Context where C_R is the Initiator's OSCORE Sender ID (see Section 6.1 of [RFC8613]).

4.1.3. Initiator Processing of Message 2

If the EDHOC connection identifier C_I is equal to the EDHOC connection identifier C_R specified in EDHOC message_2, then the Initiator **MUST** abort the session and reply with an EDHOC error message with error code 1 formatted as defined in Section 6.2 of [RFC9528].

5. Extension and Consistency of Application Profiles

It is possible to include the information below in the application profile referred by the client and server according to the specified consistency rules.

If the server supports the EDHOC + OSCORE request within an EDHOC execution started at a certain EDHOC resource, then the application profile associated with that resource **SHOULD** explicitly specify support for the EDHOC + OSCORE request.

In the case where the application profile indicates that the server supports the optional EDHOC message_4 (see Section 5.5 of [RFC9528]), it is still possible to use the optimized workflow based on the EDHOC + OSCORE request. However, this means that the server is not going to send EDHOC message_4 since it is not applicable to the optimized workflow (see Section 3.3.1).

Also, in the case where the application profile indicates that the server shall send EDHOC message_4, the application profile **MUST NOT** specify support for the EDHOC + OSCORE request. There is no point for the client to use the optimized workflow that is bound to fail (see Section 3.3.1).

6. Web Linking

Section 10.10 of [RFC9528] 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 a filter criterion.

At the same time, the application profile associated with an EDHOC resource provides information describing how the EDHOC protocol can be used through that resource. A client may become aware of the application profile, e.g., by obtaining its information elements upon discovering the EDHOC resources at the server. This allows the client to discover the EDHOC resources whose associated application profile denotes a way of using EDHOC that is most suitable to the client, e.g., with EDHOC cipher suites or authentication methods that the client also supports or prefers.

That is, while discovering an EDHOC resource, a client can contextually obtain relevant pieces of information from the application profile associated with that resource. The resource discovery can occur by means of a direct interaction with the server or by means of the CoRE Resource Directory [RFC9176] 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 criterion 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 application profile **MUST** be followed.

The following parameters are defined.

'ed-i': If present, specifies that the server supports the EDHOC Initiator role, hence the reverse message flow of EDHOC. A value **MUST NOT** be given to this parameter and any present value **MUST** be ignored by the recipient.

'ed-r': If present, specifies that the server supports the EDHOC Responder role, hence the forward message flow of EDHOC. A value **MUST NOT** be given to this parameter and any present value **MUST** be ignored by the recipient.

'ed-method': Specifies 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 10.3 of [RFC9528]. This parameter **MAY** occur multiple times, with each occurrence specifying an authentication method.

'ed-csuite': Specifies an EDHOC cipher suite 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 10.2 of [RFC9528]. This parameter **MAY** occur multiple times, with each occurrence specifying a cipher suite.

'ed-cred-t': Specifies a type of authentication credential supported by the server. This parameter **MUST** specify a single value, which is taken from the 'Value' column of the "EDHOC Authentication Credential Types" Registry defined in Section 8.3 of this document. This parameter **MAY** occur multiple times, with each occurrence specifying a type of authentication credential.

'ed-idcred-t': Specifies a type of identifier 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 Header Parameters" registry [COSE.Header.Parameters]. This parameter **MAY** occur multiple times, with each occurrence specifying a type of identifier for authentication credentials.

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

'ed-ead': Specifies the support of the server for an External Authorization Data (EAD) item (see Section 3.8 of [RFC9528]). This parameter **MUST** specify a single value, which is taken from the 'Label' column of the "EDHOC External Authorization Data" registry defined in Section 10.5 of [RFC9528]. This parameter **MAY** occur multiple times, with each occurrence specifying the ead_label of an EAD item that the server supports.

'ed-comb-req': If present, specifies that the server supports the EDHOC + OSCORE request defined in Section 3. A value **MUST NOT** be given to this parameter and any present value **MUST** be ignored by the recipient.

Future documents may update the definition of the parameters 'ed-i', 'ed-r', and 'ed-comb-req' by expanding their semantics and specifying what they can take as value.

The example in Figure 5 shows how a client discovers one EDHOC resource at a server and obtains information elements from the respective application profile. The CoRE 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;ed-csuite=0;ed-csuite=2;
                 ed-method=0;ed-cred-t=0;ed-cred-t=1;ed-idcred-t=4;
                 ed-i;ed-r;ed-comb-req
```

Figure 5: The Web Link

7. Security Considerations

The same security considerations from OSCORE [RFC8613] and EDHOC [RFC9528] hold for this document. In addition, the following considerations apply.

Section 3.2.1 specifies that a client **SHOULD NOT** have multiple outstanding EDHOC + OSCORE requests pertaining to the same EDHOC session. Even if a client did not fulfill this requirement, it would not have any impact in terms of security. That is, the server would still not process different instances of the same EDHOC message_3 more than once in the same EDHOC session (see Section 5.1 of [RFC9528]) and would still enforce replay protection of the OSCORE-protected request (see Sections 7.4 and 8.2 of [RFC8613]).

When using the optimized workflow in Figure 2, a minimum of 128-bit security against online brute-force attacks is achieved after the client receives and successfully verifies the first OSCORE-protected response (see Sections 9.1 and 9.4 of [RFC9528]). As an example, if EDHOC is used with method 3 (see Section 3.2 of [RFC9528]) and cipher suite 2 (see Section 3.6 of [RFC9528]), then the following holds:

- The Initiator is authenticated with 128-bit security against online attacks. As per Section 9.1 of [RFC9528], this results from the combination of the strength of the 64-bit Message Authentication Code (MAC) in EDHOC message_3 and of the 64-bit MAC in the Authenticated Encryption with Associated Data (AEAD) of the first OSCORE-protected CoAP request as rebuilt at Step 7 of Section 3.3.1.
- The Responder is authenticated with 128-bit security against online attacks. As per Section 9.1 of [RFC9528], this results from the combination of the strength of the 64-bit MAC in EDHOC message_2 and of the 64-bit MAC in the AEAD of the first OSCORE-protected CoAP response.

With reference to the sequential workflow in Figure 1, the OSCORE request might have to undergo access-control checks at the server before being actually executed for accessing the target protected resource. The same **MUST** hold when the optimized workflow in Figure 2 is used, i.e., when using the EDHOC + OSCORE request.

That is, the rebuilt OSCORE-protected application request from Step 7 in Section 3.3.1 **MUST** undergo the same access-control checks that would be performed on a traditional OSCORE-protected application request sent individually as shown in Figure 1.

To this end, validated information to perform access-control checks (e.g., an access token issued by a trusted party) has to be available at the server before starting to process the rebuilt OSCORE-protected application request. Such information may have been provided to the server separately before starting the EDHOC execution altogether, or instead as External Authorization Data during the EDHOC execution (see Section 3.8 of [RFC9528]).

Thus, a successful completion of the EDHOC protocol and the following derivation of the OSCORE Security Context at the server do not play a role in determining whether the rebuilt OSCORE-protected request is authorized to access the target protected resource at the server.

8. IANA Considerations

This document has the following actions for IANA.

8.1. CoAP Option Numbers Registry

IANA has registered the following option number in the "CoAP Option Numbers" registry within the "Constrained RESTful Environments (CoRE) Parameters" registry group.

Number	Name	Reference
21	EDHOC	RFC 9668

Table 2: Registration in the "CoAP Option Numbers" Registry

8.2. Target Attributes Registry

IANA has registered the following entries in the "Target Attributes" registry [CORE.Target.Attributes] within the "Constrained RESTful Environments (CoRE) Parameters" registry group as per [RFC9423]. For all entries, the Change Controller is "IETF" and the reference is "[RFC 9668]".

Attribute Name	Brief Description
ed-i	Hint: support for the EDHOC Initiator role
ed-r	Hint: support for the EDHOC Responder role
ed-method	A supported authentication method for EDHOC
ed-csuite	A supported cipher suite for EDHOC
ed-cred-t	A supported type of authentication credential for EDHOC
ed-idcred-t	A supported type of authentication credential identifier for EDHOC
ed-ead	A supported External Authorization Data (EAD) item for EDHOC
ed-comb-req	Hint: support for the EDHOC + OSCORE request

Table 3: Registrations in the "Target Attributes" Registry

8.3. EDHOC Authentication Credential Types Registry

IANA has created the "EDHOC Authentication Credential Types" registry within the "Ephemeral Diffie-Hellman Over COSE (EDHOC)" registry group defined in [RFC9528].

The registration policy is either "Private Use", "Standards Action with Expert Review", or "Specification Required" per [RFC8126]. "Expert Review" guidelines are provided in Section 8.4.

All assignments according to "Standards Action with Expert Review" are made on a "Standards Action" basis per Section 4.9 of [RFC8126] with "Expert Review" additionally required per Section 4.5 of [RFC8126]. The procedure for early IANA allocation of "standards track code points" defined in [RFC7120] also applies. When such a procedure is used, IANA will ask the designated expert(s) to approve the early allocation before registration. In addition, working group chairs are encouraged to consult the expert(s) early during the process outlined in Section 3.1 of [RFC7120].

The columns of this registry are:

Value: This field contains the value used to identify the type of authentication credential. These values **MUST** be unique. The value can be an unsigned integer or a negative integer. Different ranges of values use different registration policies:

- Integer values from -24 to 23 are designated as "Standards Action With Expert Review".
- Integer values from -65536 to -25 and from 24 to 65535 are designated as "Specification Required".
- Integer values smaller than -65536 and greater than 65535 are marked as "Private Use".

Description: This field contains a short description of the type of authentication credential.

Reference: This field contains a pointer to the public specification for the type of authentication credential.

Value	Description	Reference
0	CBOR Web Token (CWT) containing a COSE_Key in a 'cnf' claim and possibly other claims. CWT is defined in RFC 8392.	[RFC8392]
1	CWT Claims Set (CCS) containing a COSE_Key in a 'cnf' claim and possibly other claims. CCS is defined in RFC 8392.	[RFC8392]
2	X.509 certificate	[RFC5280]

Table 4: Initial Entries in the "EDHOC Authentication Credential Types" Registry

8.4. Expert Review Instructions

"Standards Action with Expert Review" and "Specification Required" are two of the registration policies defined for the IANA registry established in Section 8.3. This section gives some general guidelines for what the experts should be looking for; however, they are being designated as experts for a reason, so they should be given substantial latitude.

Expert reviewers should take into consideration the following points:

- Clarity and correctness of registrations. Experts are expected to check the clarity of purpose and use of the requested entries. Experts need to make sure that registered identifiers indicate a type of authentication credential whose format and encoding is clearly defined in the corresponding specification. Identifiers of types of authentication credentials that do not meet these objectives of clarity and completeness must not be registered.
- Point squatting should be discouraged. Reviewers are encouraged to get sufficient information for registration requests to ensure that the usage is not going to duplicate one that is already registered and that the point is likely to be used in deployments. The zones tagged as "Private Use" are intended for testing purposes and closed environments. Code points in other ranges should not be assigned for testing.
- Specifications are required for the "Standards Action With Expert Review" range of point assignment. Specifications should exist for "Specification Required" ranges, but early assignment before a specification is available is considered to be permissible. When specifications are not provided, the description provided needs to have sufficient information to identify what the point is being used for.
- Experts should take into account the expected usage of fields when approving point assignment. Documents published via Standards Action can also register points outside the Standards Action range. The length of the encoded value should be weighed against how many code points of that length are left, the size of device it will be used on, and the number of code points left that encode to that size.

9. References

9.1. Normative References

[CORE.Target.Attributes] IANA, "Target Attributes", <https://www.iana.org/assignments/core-parameters>.

[COSE.Header.Parameters] IANA, "COSE Header Parameters", <https://www.iana.org/assignments/cose>.

[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>.

[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>.

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

[RFC7120] Cotton, M., "Early IANA Allocation of Standards Track Code Points", BCP 100, RFC 7120, DOI 10.17487/RFC7120, January 2014, <https://www.rfc-editor.org/info/rfc7120>.

[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>.

[RFC7959] Bormann, C. and Z. Shelby, Ed., "Block-Wise Transfers in the Constrained Application Protocol (CoAP)", RFC 7959, DOI 10.17487/RFC7959, August 2016, <https://www.rfc-editor.org/info/rfc7959>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/info/rfc8126>.

[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>.

[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>.

[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>.

[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>.

[RFC9176] Amsüss, C., Ed., Shelby, Z., Koster, M., Bormann, C., and P. van der Stok, "Constrained RESTful Environments (CoRE) Resource Directory", RFC 9176, DOI 10.17487/RFC9176, April 2022, <https://www.rfc-editor.org/info/rfc9176>.

[RFC9528] Selander, G., Preuß Mattsson, J., and F. Palombini, "Ephemeral Diffie-Hellman Over COSE (EDHOC)", RFC 9528, DOI 10.17487/RFC9528, March 2024, <https://www.rfc-editor.org/info/rfc9528>.

9.2. Informative References

[RFC9423] Bormann, C., "Constrained RESTful Environments (CoRE) Target Attributes Registry", RFC 9423, DOI 10.17487/RFC9423, April 2024, <https://www.rfc-editor.org/info/rfc9423>.

Acknowledgments

The authors sincerely thank Christian Amsüss, Emmanuel Baccelli, Carsten Bormann, Roman Danyliw, Esko Dijk, Joel Halpern, Wes Hardaker, Klaus Hartke, John Preuß Mattsson, David Navarro, Shuping Peng, Jim Schaad, Jürgen Schönwälder, John Scudder, Orie Steele, Gunter Van de Velde, Mališa Vučinić, and Paul Wouters for their feedback and comments.

The work on this document has been partly supported by the Sweden's Innovation Agency VINNOVA and the Celtic-Next project CRITISEC, and by the H2020 project SIFIS-Home (Grant agreement 952652).

Authors' Addresses

Francesca Palombini
Ericsson AB
Torshamnsgatan 23
SE-164 40 Kista
Sweden
Email: francesca.palombini@ericsson.com

Marco Tiloca
RISE AB
Isafjordsgatan 22
SE-164 40 Kista
Sweden
Email: marco.tiloca@ri.se

Rikard Höglund
RISE AB
Isafjordsgatan 22
SE-16440 Stockholm Kista
Sweden
Email: rikard.hoglund@ri.se

Stefan Hristozov
Fraunhofer AISEC
Email: stefan.hristozov@eriptic.com

Göran Selander
Ericsson
Email: goran.selander@ericsson.com