Workgroup: LAKE Working Group
Internet-Draft: draft-selander-lake-authz-02
Published: 21 April 2023
Intended Status: Standards Track
Expires: 23 October 2023
Authors: G. Selander J. Preuß Mattsson M. Vučinić
Ericsson AB Ericsson AB INRIA
M. Richardson A. Schellenbaum
Sandelman Software Works ZHAW

Lightweight Authorization for EDHOC

Abstract

This document describes a procedure for augmenting the lightweight authenticated Diffie-Hellman key exchange protocol EDHOC with third party assisted authorization, targeting constrained IoT deployments (RFC 7228).

About This Document

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

The latest revision of this draft can be found at https://ericssonresearch.github.io/ace-ake-authz/draft-selander-lake-authz.html. Status information for this document may be found at https://datatracker.ietf.org/doc/draft-selander-lake-authz/.

Discussion of this document takes place on the Lightweight Authenticated Key Exchange Working Group mailing list (mailto:lake@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/lake/. Subscribe at https://www.ietf.org/mailman/listinfo/lake/.

Source for this draft and an issue tracker can be found at https://github.com/EricssonResearch/ace-ake-authz.

Status of This Memo

at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 23 October 2023.

Table of Contents

1. Introduction

For constrained IoT deployments [RFC7228] the overhead and processing contributed by security protocols may be significant which motivates the specification of lightweight protocols that are optimizing, in particular, message overhead (see [I-D.ietf-lake-reqs]). This document describes a procedure for augmenting the lightweight authenticated Diffie-Hellman key exchange EDHOC [I-D.ietf-lake-edhoc] with third party-assisted authorization.

The procedure involves a device, a domain authenticator, and an authorization server. The device and authenticator perform mutual authentication and authorization, assisted by the authorization server which provides relevant authorization information to the device (a "voucher") and to the authenticator.

The protocol assumes that authentication between device and authenticator is performed with EDHOC, and defines the integration of a lightweight authorization procedure using the External Authorization Data (EAD) field defined in EDHOC.

In this document we consider the target interaction for which authorization is needed to be "enrollment", for example joining a network for the first time (e.g., [RFC9031]), but it can be applied to authorize other target interactions.

The protocol enables a low message count by performing authorization and enrollment in parallel with authentication, instead of in sequence which is common for network access. It further reuses protocol elements from EDHOC leading to reduced message sizes on constrained links.

This protocol is applicable to a wide variety of settings, and can be mapped to different authorization architectures.

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.

Readers are expected to have an understanding of CBOR [RFC8949] and EDHOC [I-D.ietf-lake-edhoc]. Appendix C.1 of [I-D.ietf-lake-edhoc] contains some basic info about CBOR.

2. Problem Description

The (potentially constrained) device (U) wants to enroll into a domain over a constrained link. The device authenticates and enforces authorization of the (non-constrained) domain authenticator (V) with the help of a voucher, and makes the enrollment request. The domain authenticator, in turn, authenticates the device and authorizes its enrollment. Authentication between device and domain authenticator is made with the lightweight authenticated Diffie-Hellman key exchange protocol EDHOC [I-D.ietf-lake-edhoc]. The procedure is assisted by a (non-constrained) authorization server (W) located in a non-constrained network behind the domain authenticator providing information to the device and to the domain authenticator as part of the protocol.

The objective of this document is to specify such a protocol which is lightweight over the constrained link by reusing elements of EDHOC. See illustration in Figure 1.

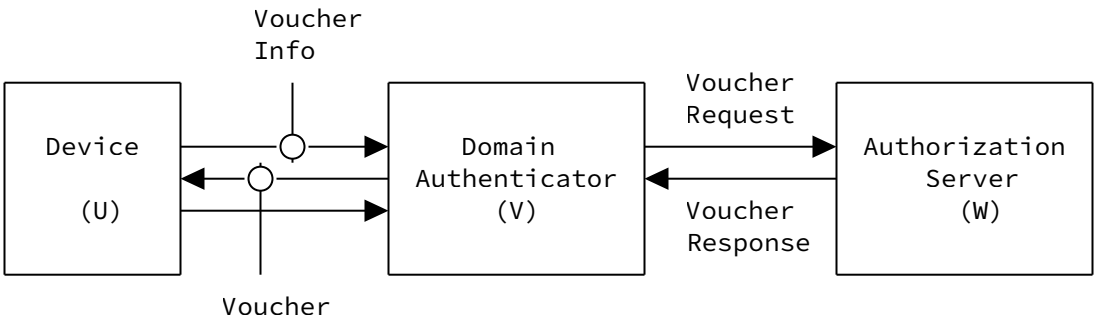


Figure 1: Overview of message flow. EDHOC is used on the constrained link between U and V. Voucher_Info and Voucher are sent in EDHOC External Authorization Data. The link between V and W is not constrained.

3. Assumptions

3.1. Device (U)

U takes the role as EDHOC Initiator with authentication credential CRED_I. CRED_I may for example be an X.509 certificate or a CBOR Web Token (CWT, [RFC8392]). For identification to W, U is provisioned with an identifier ID_U, from which W shall be able to retrieve CRED_I. ID_U is for example a reference to the device authentication credential, or an identifier from a separate name space.

U is also provisioned with information about W:

- A static public DH key of W (G_W) used to protect communication between device and authorization server (see Section 4.3).

- Location information about the authorization server (LOC_W) that can be used by V. This is typically a URI but may be optimized, e.g., only the domain name.

3.2. Domain Authenticator (V)

V takes the role as EDHOC Responder with authentication credential CRED_R. CRED_R is a CWT Claims Set (CCS, [RFC8392]) containing the public authentication key of V, PK_V, see Section 4.4.2.1

V needs to establish secure communication with W based on information in LOC_W. The communication between V and W is assumed to be mutually authenticated and protected; authentication credentials and communication security is out of scope, except for as specified below in this section.

V may in principle use different credentials for authenticating to U and to W (CRED_R is used for the former). However, V MUST prove possession of private key of PK_V to W, since W is asserting (by means of a voucher sent to U) that this credential belongs to V.

In this version of the draft is assumed that V authenticates to W with the public key PK_V using some authentication protocol providing proof of possession of the private key, for example TLS 1.3 [RFC8446]. A future version of this draft may specify explicit proof of possession of the private key of PK_V in VREQ, e.g., by including a signature of the contents of the voucher request made with the private key corresponding to PK_V.

3.3. Authorization Server (W)

W has the private DH key corresponding to G_W, which is used to secure the communication with U (see Section 4.3).

Authentication credentials and communication security used with V is out of scope, except for the need to verify the possession of the private key of PK_V as specified in Section 3.2.

W provides to U the authorization decision for enrollment with V in the form of a voucher, see Section 4.3.2. W may provide V with the authorization credential of U, CRED_I, after V has learnt the identity of U.

W needs to be available during the execution of the protocol between U and V.

4. The Protocol

4.1. Overview

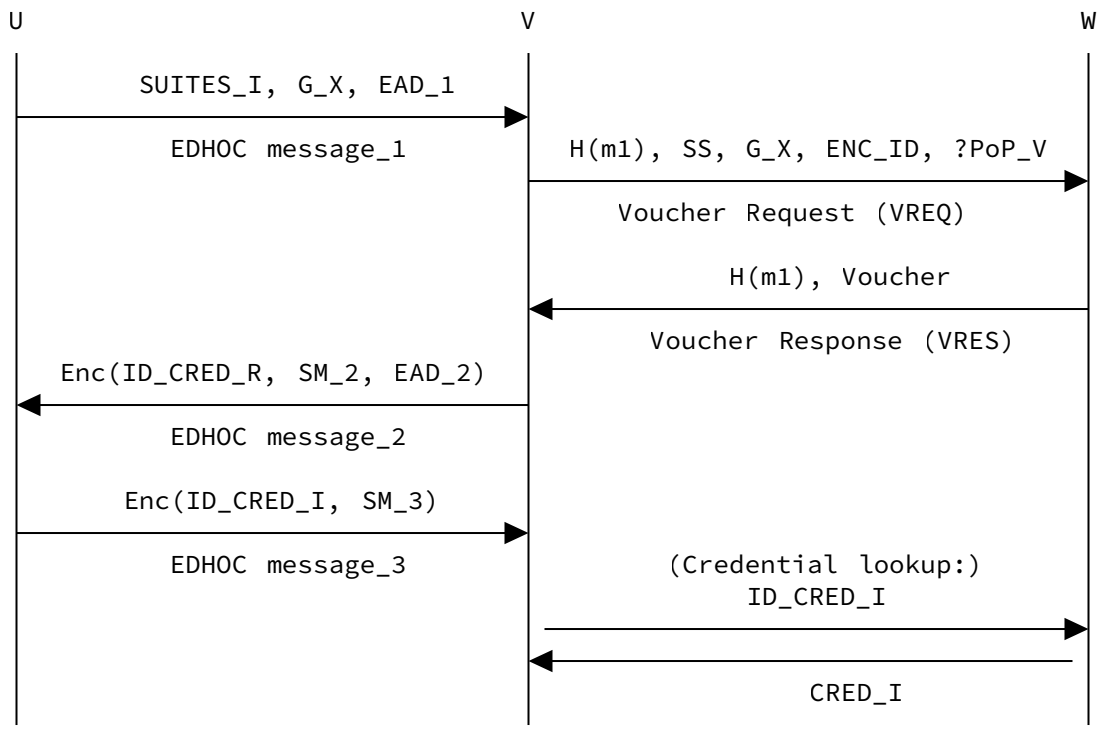
Three security sessions are going on in parallel:

1. EDHOC [I-D.ietf-lake-edhoc] between device (U) and (domain) authenticator (V)

2. Voucher Request/Response between authenticator (V) and authorization server (W)

3. An exchange of voucher-related information, including the voucher itself, between device (U) and authorization server (W), mediated by the authenticator (V).

Figure 2 provides an overview of the message flow detailed in this section, for more details see Section 3.1 of [I-D.ietf-lake-edhoc].



where
H(m1) = H(message_1)
EAD_1 contains Voucher_Info: LOC_W, ENC_ID
EAD_2 contains Voucher: MAC(H(message_1), CRED_R)

Figure 2: W-assisted authorization of U and V to each other: EDHOC between U and V (only selected message fields shown for simplicity), and Voucher Request/Response between V and W.

4.2. Reuse of EDHOC

The protocol illustrated in Figure 2 reuses several components of EDHOC:

* G_X, the 'x' parameter of the ephemeral public Diffie-Hellman key of party U, is also used in the protocol between U and W.

* SUITES_I, the cipher suites relevant to U, which includes the selected cipher suite - here denoted SS, also defines the algorithms used between U and W. In particular SS contains information about (see Section 3.6 of [I-D.ietf-lake-edhoc]):

  - EDHOC AEAD algorithm: used to encrypt the identity of U

  - EDHOC hash algorithm: used for key derivation and to calculate the voucher

  - EDHOC MAC length in bytes: length of the voucher

  - EDHOC key exchange algorithm: used to calculate the shared secret between U and W

* EAD_1, EAD_2 are the External Authorization Data message fields of message_1 and message_2, respectively, see Section 3.8 of [I-D.ietf-lake-edhoc]. This document specifies EAD items with ead_label = TBD1, see Section 7.1).

* ID_CRED_I and ID_CRED_R are used to identify the authentication credentials of U and V. As shown at the bottom of Figure 2, V may use W to obtain CRED_I, the authentication credential of U. The authentication credential of V, CRED_R, is transported in ID_CRED_R in message_2, see Section 4.4.2.1.

* Signature_or_MAC_2 and Signature_or_MAC_3 (abbreviated SM_2 and SM_3 in Figure 2), containing data generated using the private key of V and U, respectively, are shown here just to be able to reason about the use of credentials. The definition of these fields depend on EDHOC method, see Section 5 of [I-D.ietf-lake-edhoc]).

The protocol also reuses the Extract and Expand key derivation from EDHOC (Section 4 of [I-D.ietf-lake-edhoc]).

*The intermediate pseudo-random key PRK is derived using Extract():

-PRK = Extract(salt, IKM)

owhere salt = 0x (the zero-length byte string)

oIKM is the ECDH shared secret G_XW (calculated from G_X and W or G_W and X) as defined in Section 6.3.1 of [RFC9053].

The shared secret is derived using Expand() which is defined in terms of the EDHOC hash algorithm of the selected cipher suite, see Section 4.2. of [I-D.ietf-lake-edhoc]:

*shared secret = Expand(PRK, info, length)

where

```
info = (
   label : int,
   context : bstr,
   length : uint,
)
```

4.3. Device <-> Authorization Server (U <-> W)

The protocol between U and W is carried out via V with certain data protected between the endpoints using the equivalent of a hybrid public key encryption scheme such as [RFC9180]. U uses the public DH key of the W, G_W, together with the private DH key corresponding to ephemeral key G_X in EDHOC message_1, and vice versa for W. The endpoints calculate a shared secret G_XW (see Section 4.2), which is used to derive secret keys to protect data between U and W, as detailed in this section.

The data exchanged between U and W is carried between U and V in message_1 and message_2 (Section 4.4), and between V and W in the Voucher Request/Response (Section 4.5).

4.3.1. Voucher Info

The external authorization data EAD_1 contains an EAD item with ead_label = TBD1 and ead_value = Voucher_Info, which is a CBOR byte string:

```
Voucher_Info = bstr .cbor Voucher_Info_Seq
```

```
Voucher_Info_Seq = (
    LOC_W:      tstr,
    ENC_ID:     bstr
)
```

where

*LOC_W is location information of W, used by V

*ENC_ID is the encrypted blob carrying an identifier of U passed on from V to W, calculated as follows:

ENC_ID is 'ciphertext' of COSE_Encrypt0 (Section 5.2-5.3 of [RFC9052]) computed from the following:

*The encryption key K_1 and nonce IV_1 are derived as specified below.

*'protected' is a byte string of size 0

*'plaintext and 'external_aad' as below:

```
plaintext = (
    ID_U:            bstr,
 )

external_aad = (
    SS:              int,
 )
```

where

*ID_U is an identity of the device, for example a reference to the device authentication credential, see Section 3.1.

*SS is the selected cipher suite in SUITES_I.

The derivation of K_1 = Expand(PRK, info, length) uses the following input to the info struct (Section 4.2):

*label = TBD1

*context = h''

*length is length of key of the EDHOC AEAD algorithm in bytes

The derivation of IV_1 = Expand(PRK, info, length) uses the following input to the info struct (Section 4.2):

*label = TBD1

*context = h'00'

*length is length of nonce of the EDHOC AEAD algorithm in bytes

4.3.2. Voucher

The voucher is an assertion for U that W has performed the relevant verifications and that U is authorized to continue the protocol with V. The voucher is essentially a message authentication code which binds the authentication credential of V to message_1 of EDHOC, integrity protected with the shared secret context between U and W.

The external authorization data EAD_2 contains an EAD item with ead_label = TBD1 and ead_value = Voucher, which is a CBOR byte string:

```
Voucher = bstr .cbor Expand(PRK, info, length)
```

calculated with the following input to the info struct (Section 4.2):

*label is TBD1

*context = bstr .cbor voucher_input

*length is EDHOC MAC length in bytes

where context is a CBOR bstr wrapping of the following CBOR sequence:

```
voucher_input = (
    H(message_1):  bstr,
    CRED_R:        bstr,
)
```

where

*H(message_1) is copied from the associated voucher request.

*CRED_R is a CWT Claims Set (CCS, [RFC8392]) containing the public authentication key of V, PK_V, see Section 4.4.2.1

4.4. Device <-> Authenticator (U <-> V)

This section describes the processing in U and V, which execute the EDHOC protocol using their respective authentication credentials, see Figure 2. Normal EDHOC processing is omitted here.

4.4.1. Message 1

4.4.1.1. Processing in U

U composes EDHOC message_1 using authentication method, identifiers, etc. according to an agreed application profile, see Section 3.9 of [I-D.ietf-lake-edhoc]. The selected cipher suite, in this document denoted SS, applies also to the interaction with W as detailed in Section 4.2, in particular, to the key agreement algorithm which is used with the static public DH key G_W of W. As part of the normal EDHOC processing, U generates the ephemeral public key G_X which is reused in the interaction with W, see Section 4.3.

The device sends EDHOC message_1 with EAD item (-TBD1, Voucher_Info) included in EAD_1, where Voucher_Info is specified in Section 4.3. The negative sign indicates that the EAD item is critical, see Section 3.8 in [I-D.ietf-lake-edhoc].

4.4.1.2. Processing in V

V receives EDHOC message_1 from U and processes it as specified in Section 5.2.3 of [I-D.ietf-lake-edhoc], with the additional step of processing the EAD item in EAD_1. Since the EAD item is critical, if V does not recognize it or it contains information that V cannot process, then V MUST discontinue EDHOC, see Section 3.8 in [I-D.ietf-lake-edhoc]. Otherwise, the ead_label = TBD1, triggers the voucher request to W as described in Section 4.5. The exchange between V and W needs to be completed successfully for the EDHOC exchange to be continued.

4.4.2. Message 2

4.4.2.1. Processing in V

V receives the voucher response from W as described in Section 4.5.

V sends EDHOC message_2 to U with the critical EAD item (-TBD1, Voucher) included in EAD_2, where the Voucher is specified in Section 4.3.

CRED_R is a CWT Claims Set (CCS, [RFC8392]) containing the public authentication key of the authenticator PK_V encoded as a COSE_Key in the 'cnf' claim, see Section 3.5.2 of [I-D.ietf-lake-edhoc].

ID_CRED_R contains the CCS with 'kccs' as COSE header_map, see Section 9.6 of [I-D.ietf-lake-edhoc]. The Signature_or_MAC_2 field calculated using the private key corresponding to PK_V is either a signature or a MAC depending on EDHOC method.

4.4.2.2. Processing in U

U receives EDHOC message_2 from V and processes it as specified in Section 5.3.2 of [I-D.ietf-lake-edhoc], with the additional step of processing the EAD item in EAD_2.

If U does not recognize the EAD item or the EAD item contains information that U cannot process, then U MUST discontinue EDHOC, see Section 3.8 in [I-D.ietf-lake-edhoc]. Otherwise U MUST verify the Voucher by performing the same calculation as in Section 4.3.2 using H(message_1) and CRED_R received in ID_CRED_R of message_2. If the voucher calculated in this way is not identical to what was received in message_2, then U MUST discontinue the protocol.

4.4.3. Message 3

4.4.3.1. Processing in U

If all verifications are passed, then U sends EDHOC message_3.

The Signature_or_MAC_3 field calculated using the private key corresponding to PK_U is either a signature or a MAC depending on EDHOC method.

EAD_3 MAY contain a certificate enrollment request, see e.g., CSR specified in [I-D.ietf-cose-cbor-encoded-cert], or other request which the device is now authorized to make.

EDHOC message_3 may be combined with an OSCORE request, see [I-D.ietf-core-oscore-edhoc].

4.4.3.2. Processing in V

V performs the normal EDHOC verifications of message_3. V may retrieve CRED_I from W, after V learnt ID_CRED_I from U.

4.5. Authenticator <-> Authorization Server (V <-> W)

V and W are assumed to be able to authenticate and set up a secure connection, out of scope for this specification, for example TLS 1.3 authenticated with certificates. V is assumed to authenticate with the public key PK_V, see Section 3.2.

This secure connection protects the Voucher Request/Response Protocol (see protocol between V and W in Figure 2).

The hash of EDHOC message_1, H(message_1), acts as session identifier of the Voucher Request/Response protocol, and binds together instances of the two protocols (U<->V and V<->W).

4.5.1. Voucher Request

4.5.1.1. Processing in V

Unless already in place, V and W establish a secure connection. V uses H(message_1) as a session identifier associated to this connection with W. If the same value of H(message_1) is already used for a connection with this or other W, the protocol SHALL be discontinued.

V sends the voucher request to W. The Voucher Request SHALL be a CBOR array as defined below:

```
Voucher_Request = [
    H(message_1):    bstr,
    SS:              int,
    G_X:             bstr,
    ENC_ID:          bstr,
  ? PoP_V:           bstr,
]
```

where the parameters are defined in Section 4.3, except:

- PoP_V is a proof-of-possession of public key PK_V using the corresponding private key. PoP_V is optional.

Editor's note: Define PoP_V (include G_X, ENC_ID in the calculation for binding to this EDHOC session). One case to study is when V authenticates to U with static DH and to W with signature.

4.5.1.2. Processing in W

W receives the voucher request, verifies and decrypts ENC_ID, and associates the session identifier H(message_1) to ID_U. If H(message_1) is not unique among session identifiers associated to this identity, the protocol SHALL be discontinued.

W uses the identity of the device, ID_U, to look up and verify the associated authorization policies for U. This is out of scope for the specification.

4.5.2. Voucher Response

4.5.2.1. Processing in W

W retrieves the public key of V, PK_V, used to authenticate the secure connection with V, and constructs the CCS (see Section 4.4.2.1) and the Voucher (see Section 4.3.2).

Editor's note: Make sure the CCS is defined to allow W to generate it uniquely from PK_V.

W generates the voucher response and sends it to V over the secure connection. The Voucher_Response SHALL be a CBOR array as defined below:

```
Voucher_Response = [
    H(message_1):   bstr,
    Voucher:        bstr
]
```

where

- *H(message_1) is copied from the associated voucher request.

- *The Voucher is defined in Section 4.3.2.

4.5.2.2. Processing in V

V receives the voucher response from W over the secure connection. If the received session identifier does not match the session identifier H(message_1) associated to the secure connection, the protocol SHALL be discontinued.

5. REST Interface at W

The interaction between V and W is enabled through a RESTful interface exposed by W. V SHOULD access the resources exposed by W through the protocol indicated by the scheme in LOC_W URI. In case the scheme indicates "https", V SHOULD perform a TLS handshake with W and use HTTP. In case the scheme indicates "coaps", V SHOULD perform a DTLS handshake with W and access the same resources using CoAP. In both cases, V MUST perform client authentication to authenticate to W, using a certificate containing the PK_V public key.

5.1. HTTP URIs

W MUST support the use of the path-prefix "/.well-known/", as defined in [RFC8615], and the registered name "lake-authz". A valid URI thus begins with "https://www.example.com/.well-known/lake-authz". Each operation specified in the following is indicated by a path-suffix.

5.2. Voucher Request (/voucherrequest)

To request a voucher, V MUST issue an HTTP request:

- *Method is POST

*Payload is the serialization of the Voucher Request object, as specified in Section 4.5.1.

In case of successful processing at W, W MUST issue a 200 OK response with payload containing the serialized Voucher Response object, as specified in Section 4.5.2.

5.3. Certificate Request (/certrequest)

V requests the public key certificate of U from W through the "/certrequest" path-suffix. To request U's authentication credential, V MUST issue an HTTP request:

*Method is POST

*Payload is the serialization of the ID_CRED_I object, as received in EDHOC message_3.

In case of a successful lookup of the authentication credential at W, W MUST issue 200 OK response with payload containing the serialized CRED_I.

6. Security Considerations

This specification builds on and reuses many of the security constructions of EDHOC, e.g., shared secret calculation and key derivation. The security considerations of EDHOC [I-D.ietf-lake-edhoc] apply with modifications discussed here.

EDHOC provides identity protection of the Initiator, here the device. The encryption of the device identity in the first message should consider potential information leaking from the length of the identifier ID_U, either by making all identifiers having the same length or the use of a padding scheme.

Although W learns about the identity of U after receiving VREQ, this information must not be disclosed to V, until U has revealed its identity to V with ID_CRED_I in message_3. W may be used for lookup of CRED_I from ID_CRED_I, or this credential lookup function may be separate from the authorization function of W. The trust model used here is that U decides to which V it reveals its identity. In an alternative trust model where U trusts W to decide to which V it reveal's U's identity, CRED_I could be sent in Voucher Response.

As noted Section 8.2 of [I-D.ietf-lake-edhoc] an ephemeral key may be used to calculate several ECDH shared secrets. In this specification the ephemeral key G_X is also used to calculate G_XW, the shared secret with the authorization server.

The private ephemeral key is thus used in the device for calculations of key material relating to both the authenticator and the authorization server. There are different options for where to implement these calculations, one option is as an addition to EDHOC, i.e., to extend the EDHOC API in the device with input of public key of W (G_W) and identifier of U (ID_U), and produce the encryption of ID_U which is included in Voucher_Info in EAD_1.

7. IANA Considerations

7.1. EDHOC External Authorization Data Registry

IANA has registered the following entry in the "EDHOC External Authorization Data" registry under the group name "Ephemeral Diffie-Hellman Over COSE (EDHOC)". The ead_label = TBD_1 corresponds to the ead_value Voucher_Info in EAD_1, and Voucher in EAD_2 with processing specified in Section 4.4.1 and Section 4.4.2, respectively, of this document.

Label	Value Type	Description
TBD1	bstr	Voucher related information

Table 1: Addition to the EDHOC EAD registry

7.2. The Well-Known URI Registry

IANA has registered the following entry in "The Well-Known URI Registry", using the template from [RFC8615]:

*URI suffix: lake-authz

*Change controller: IETF

*Specification document: [[this document]]

*Related information: None

7.3. Well-Known Name Under ".arpa" Name Space

This document allocates a well-known name under the .arpa name space according to the rules given in [RFC3172] and [RFC6761]. The name "lake-authz.arpa" is requested. No subdomains are expected, and addition of any such subdomains requires the publication of an IETF Standards Track RFC. No A, AAAA, or PTR record is requested.

8. References

8.1. Normative References

[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-19, 3 February 2023, <https://datatracker.ietf.org/doc/html/draft-ietf-lake-edhoc-19>.

[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>.

[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>.

[RFC9052] Schaad, J., "CBOR Object Signing and Encryption (COSE): Structures and Process", STD 96, RFC 9052, DOI 10.17487/RFC9052, August 2022, <https://www.rfc-editor.org/info/rfc9052>.

[RFC9053] Schaad, J., "CBOR Object Signing and Encryption (COSE): Initial Algorithms", RFC 9053, DOI 10.17487/RFC9053, August 2022, <https://www.rfc-editor.org/info/rfc9053>.

8.2. Informative References

[I-D.ietf-core-oscore-edhoc] Palombini, F., Tiloca, M., Höglund, R., Hristozov, S., and G. Selander, "Using EDHOC with CoAP and OSCORE", Work in Progress, Internet-Draft, draft-ietf-core-oscore-edhoc-07, 13 March 2023, <https://datatracker.ietf.org/doc/html/draft-ietf-core-oscore-edhoc-07>.

[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-05, 10 January 2023, <https://datatracker.ietf.org/doc/html/draft-ietf-cose-cbor-encoded-cert-05>.

[I-D.ietf-lake-reqs] Vučinić, M., Selander, G., Mattsson, J. P., and D. Garcia-Carillo, "Requirements for a Lightweight AKE for OSCORE", Work in Progress, Internet-Draft, draft-ietf-lake-reqs-04, 8 June 2020, <https://datatracker.ietf.org/doc/html/draft-ietf-lake-reqs-04>.

[IEEE802.15.4] IEEE standard for Information Technology, "IEEE Std 802.15.4 Standard for Low-Rate Wireless Networks", n.d..

[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>.

[RFC3172] Huston, G., Ed., "Management Guidelines & Operational Requirements for the Address and Routing Parameter Area Domain ("arpa")", BCP 52, RFC 3172, DOI 10.17487/RFC3172, September 2001, <https://www.rfc-editor.org/info/rfc3172>.

[RFC6761] Cheshire, S. and M. Krochmal, "Special-Use Domain Names", RFC 6761, DOI 10.17487/RFC6761, February 2013, <https://www.rfc-editor.org/info/rfc6761>.

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

[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>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

[RFC8615] Nottingham, M., "Well-Known Uniform Resource Identifiers (URIs)", RFC 8615, DOI 10.17487/RFC8615, May 2019, <https://www.rfc-editor.org/info/rfc8615>.

[RFC9031] Vučinić, M., Ed., Simon, J., Pister, K., and M. Richardson, "Constrained Join Protocol (CoJP) for 6TiSCH", RFC 9031, DOI 10.17487/RFC9031, May 2021, <https://www.rfc-editor.org/info/rfc9031>.

[RFC9180] Barnes, R., Bhargavan, K., Lipp, B., and C. Wood, "Hybrid Public Key Encryption", RFC 9180, DOI 10.17487/RFC9180, February 2022, <https://www.rfc-editor.org/info/rfc9180>.

Appendix A. Use with Constrained Join Protocol (CoJP)

This section outlines how the protocol is used for network enrollment and parameter provisioning. An IEEE 802.15.4 network is used as an example of how a new device (U) can be enrolled into the domain managed by the domain authenticator (V).

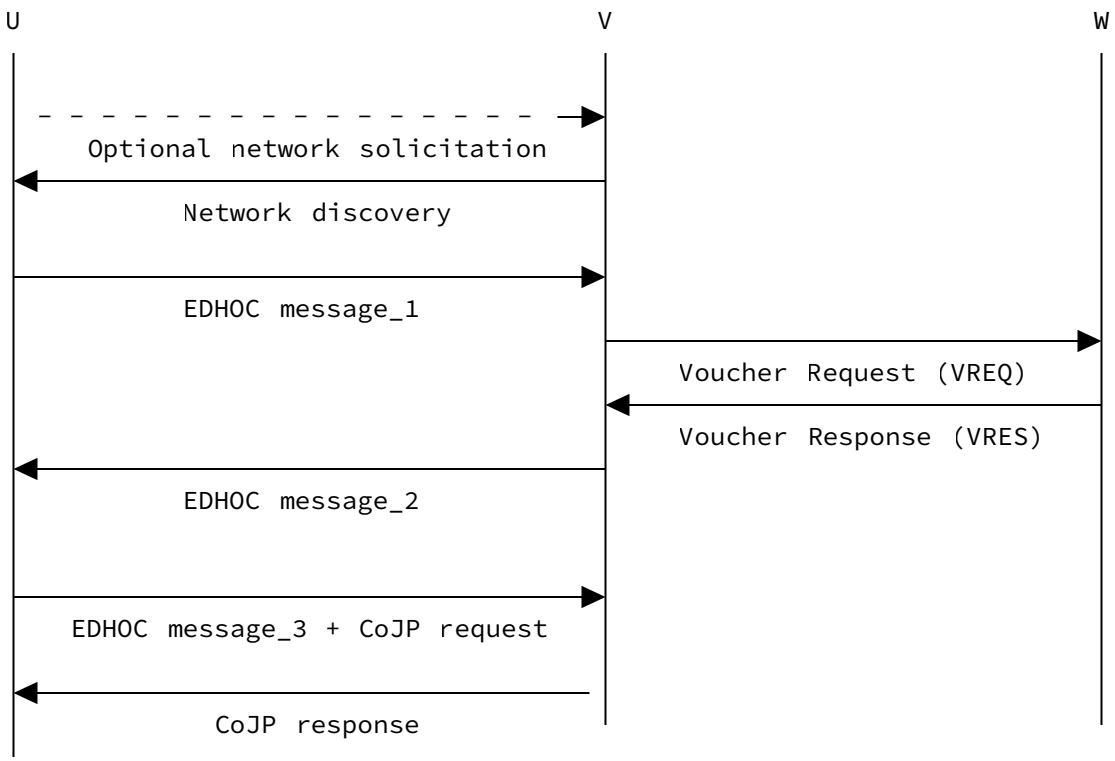


Figure 3: Use of draft-selander-lake-authz with CoJP.

A.1. Network Discovery

When a device first boots, it needs to discover the network it attempts to join. The network discovery procedure is defined by the link-layer technology in use. In case of Time-slotted Channel Hopping (TSCH) networks, a mode of [IEEE802.15.4], the device scans the radio channels for Enhanced Beacon (EB) frames, a procedure known as passive scan. EBs carry the information about the network, and particularly the network identifier. Based on the EB, the network identifier, the information pre-configured into the device, the device makes the decision on whether it should join the network advertised by the received EB frame. This process is described in Section 4.1. of [RFC9031]. In case of other, non-TSCH modes of IEEE 802.15.4 it is possible to use the active scan procedure and send solicitation frames. These solicitation frames trigger the nearest network coordinator to respond by emitting a beacon frame. The network coordinator emitting beacons may be multiple link-layer hops away from the domain authenticator (V), in which case it plays the role of a Join Proxy (see [RFC9031]). Join Proxy does not participate in the protocol and acts as a transparent router between the device and the domain authenticator. For simplicity, Figure 3 illustrates the case when the device and the domain authenticator are a single hop away and can communicate directly.

A.2. The Enrollment Protocol with Parameter Provisioning

A.2.1. Flight 1

Once the device has discovered the network it wants to join, it constructs the EDHOC message_1, as described in Section 4.4. The device SHALL map the message to a CoAP request:

- *The request method is POST.
- *The type is Confirmable (CON).
- *The Proxy-Scheme option is set to "coap".
- *The Uri-Host option is set to "lake-authz.arpa". This is an anycast type of identifier of the domain authenticator (V) that is resolved to its IPv6 address by the Join Proxy.
- *The Uri-Path option is set to ".well-known/edhoc".
- *The Content-Format option is set to "application/cid-edhoc+cbor-seq"
- *The payload is the (true, EDHOC message_1) CBOR sequence, where EDHOC message_1 is constructed as defined in Section 4.4.

A.2.2. Flight 2

The domain authenticator receives message_1 and processes it as described in Section 4.4. The message triggers the exchange with the authorization server, as described in Section 4.5. If the exchange between V and W completes successfully, the domain authenticator prepares EDHOC message_2, as described in Section 4.4. The authenticator SHALL map the message to a CoAP response:

- *The response code is 2.04 Changed.
- *The Content-Format option is set to "application/edhoc+cbor-seq"
- *The payload is the EDHOC message_2, as defined in Section 4.4.

A.2.3. Flight 3

The device receives EDHOC message_2 and processes it as described in Section 4.4}. Upon successful processing of message_2, the device prepares flight 3, which is an OSCORE-protected CoJP request containing an EDHOC message_3, as described in [I-D.ietf-core-oscore-edhoc]. EDHOC message_3 is prepared as described in Section 4.4. The OSCORE-protected payload is the CoJP Join Request object specified in Section 8.4.1. of [RFC9031]. OSCORE

protection leverages the OSCORE Security Context derived from the EDHOC exchange, as specified in Appendix A of [I-D.ietf-lake-edhoc]. To that end, [I-D.ietf-core-oscore-edhoc] specifies that the Sender ID of the client (device) must be set to the connection identifier selected by the domain authenticator, C_R. OSCORE includes the Sender ID as the kid in the OSCORE option. The network identifier in the CoJP Join Request object is set to the network identifier obtained from the network discovery phase. In case of IEEE 802.15.4 networks, this is the PAN ID.

The device SHALL map the message to a CoAP request:

- The request method is POST.
- The type is Confirmable (CON).
- The Proxy-Scheme option is set to "coap".
- The Uri-Host option is set to "lake-authz.arpa".
- The Uri-Path option is set to ".well-known/edhoc".
- The EDHOC option [I-D.ietf-core-oscore-edhoc] is set and is empty.
- The payload is prepared as described in Section 3.2. of [I-D.ietf-core-oscore-edhoc], with EDHOC message_3 and the CoJP Join Request object as the OSCORE-protected payload.

Note that the OSCORE Sender IDs are derived from the connection identifiers of the EDHOC exchange. This is in contrast with [RFC9031] where ID Context of the OSCORE Security Context is set to the device identifier (pledge identifier). Since the device identity is exchanged during the EDHOC handshake, and the certificate of the device is communicated to the authenticator as part of the Voucher Response message, there is no need to transport the device identity in OSCORE messages. The authenticator playing the role of the [RFC9031] JRC obtains the device identity from the execution of the authorization protocol.

A.2.4. Flight 4

Flight 4 is the OSCORE response carrying CoJP response message. The message is processed as specified in Section 8.4.2. of [RFC9031].

Authors' Addresses

Göran Selander
Ericsson AB
Sweden

Email: goran.selander@ericsson.com

John Preuß Mattsson
Ericsson AB
Sweden

Email: john.mattsson@ericsson.com

Mališa Vučinić
INRIA
France

Email: malisa.vucinic@inria.fr

Michael Richardson
Sandelman Software Works
Canada

Email: mcr+ietf@sandelman.ca

Aurelio Schellenbaum
Institute of Embedded Systems, ZHAW
Switzerland

Email: aureliorubendario.schellenbaum@zhaw.ch