Lightweight Authorization for Authenticated Key Exchange

Göran Selander, John Mattsson, Mališa Vučinić, Michael Richardson, Aurelio Schellenbaum

▶ **To cite this version:**

Göran Selander, John Mattsson, Mališa Vučinić, Michael Richardson, Aurelio Schellenbaum. Lightweight Authorization for Authenticated Key Exchange: IETF Internet Draft. IETF Internet Draft, 2021. hal-03119937v2

HAL Id: hal-03119937
https://hal.inria.fr/hal-03119937v2

Submitted on 21 Jan 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Network Working Group
Internet-Draft
Intended status: Informational
Expires: 25 April 2022

G. Selander
J. Preuß Mattsson
Ericsson AB
M. Vučinić
INRIA
M. Richardson
Sandelman Software Works
A. Schellenbaum
ZHAW
22 October 2021

Lightweight Authorization for Authenticated Key Exchange. draft-selander-ace-ake-authz-04

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).

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 25 April 2022.

Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (https://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights

and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

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]), or certificate enrollment (such as [I-D.selander-ace-coap-est-oscore]), 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. This document specifies a profile of the ACE framework [I-D.ietf-ace-oauth-authz]. Other settings such as EAP [RFC3748] are out of scope for this specification.

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 some 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 wants to enroll into a domain over a constrained link. The device authenticates and enforces authorization of the (non-constrained) domain authenticator with the help of a voucher, and makes the enrollment request. The domain authenticator 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 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 and reuses elements of EDHOC. See illustration in Figure 1.

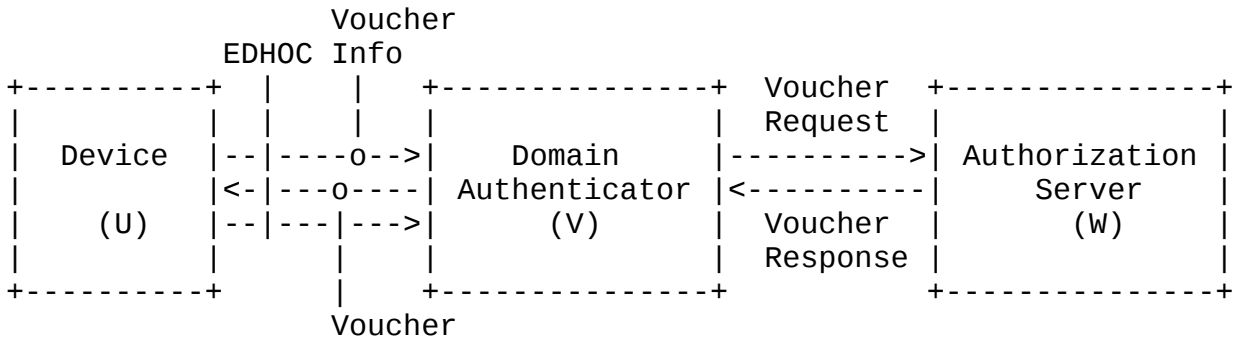


Figure 1: Overview of message flow. Link between U anv V is constrained but link between V and W is not. Voucher Info and Voucher are sent in EDHOC External Authorization Data.

3. Assumptions

3.1. Device

The device is pre-provisioned with an identity ID_U and asymmetric key credentials: a private key, a public key (PK_U), and optionally a public key certificate (Cert_PK_U), issued by a trusted third party such as e.g. the device manufacturer, used to authenticate to the domain authenticator. ID_U may be a reference or pointer to the certificate.

The device is also provisioned with information about its authorization server:

* At least one static public DH key of the authorization server (G_W) used to ensure secure communication with the device (see Section 4.3).

* Location information about the authorization server (LOC_W), e.g. its domain name. This information may be available in the device certificate Cert_PK_U.

3.2. Domain Authenticator

The domain authenticator has a private key and a corresponding public key PK_V used to authenticate to the device.

The domain authenticator needs to be able to locate the authorization server of the device for which LOC_W is expected to be sufficient. The communication between domain authenticator and authorization server 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.

The domain authenticator may in principle use differents credentials for authenticating to the authorization server and to the device, for which PK_V is used. However, the domain authenticator MUST prove possession of private key of PK_V to the authorization server since the authorization server is asserting (by means of the voucher to the device) that this credential belongs to the domain authenticator.

In this version of the draft it is assumed that the domain authenticator authenticates to the authorization server with 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 the voucher request, e.g., by including a signature of the voucher request with the private key corresponding to PK_V.

3.3. Authorization Server

The authorization server has the private DH key corresponding to G_W, which is used to secure the communication with the device (see Section 4.3).

Authentication credentials and communication security used with the domain authenticator 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.

The authorization server provides to the device the authorization decision for enrollment with the domain authenticator in the form of a voucher. The authorization server provides information to the domain authenticator about the device, such as the device's certificate Cert_PK_U.

The authorization server needs to be available during the execution of the protocol.

4. The Protocol

4.1. Overview

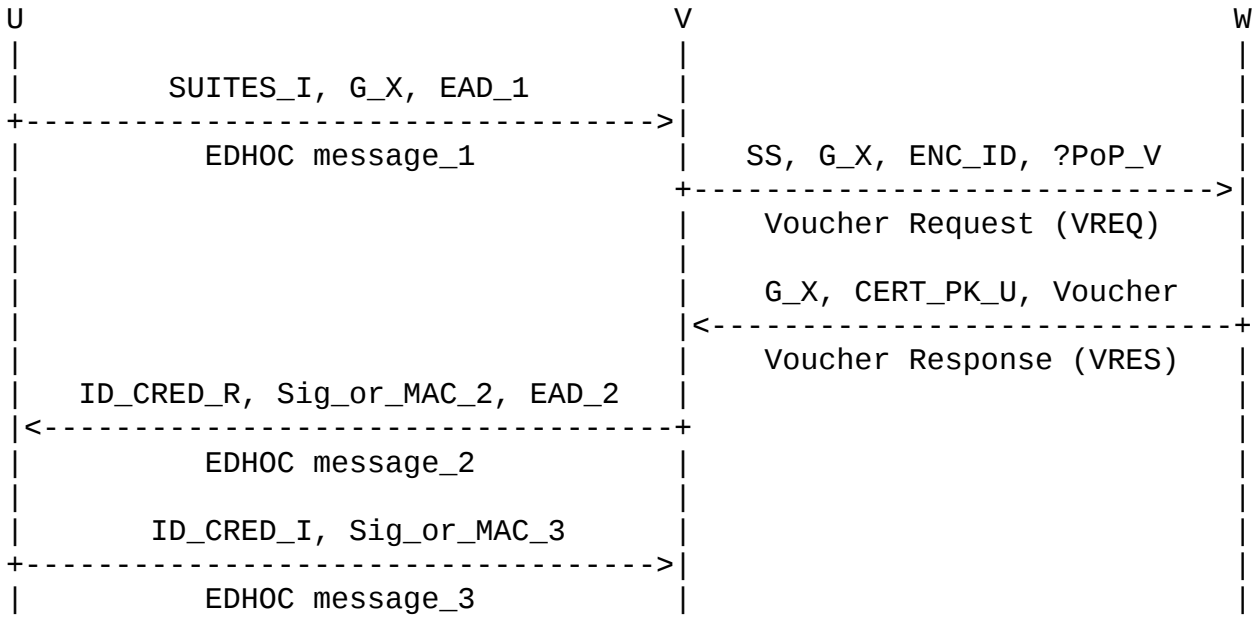
Three security sessions are going on in parallel:

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.

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



where
EAD_1 = (L, Voucher_Info)
Voucher_Info = [LOC_W, ENC_ID]
EAD_2 = (L, Voucher)
Voucher = MAC(V_TYPE, SS, G_X, ID_U, PK_V)

Figure 2: W-assisted authorization of AKE between U and V: EDHOC between U and V (only selected message fields shown), 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, as ephemeral key and nonce.

* 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 of message_1 and message_2, respectively, for which dedicated content is defined in this document.

* ID_CRED_I and ID_CRED_R are used to identify the public authentication keys of U and V. In this protocol ID_CRED_I can be empty since V obtains the certificate of U from W, whereas ID_CRED_R contains the public authentication key of V.

* Signature_or_MAC_2 and Signature_or_MAC_3 (abbreviated 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 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)

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

    o IKM 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 [I-D.ietf-cose-rfc8152bis-algs].

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 = (
   transcript_hash : bstr,
   label : tstr,
   context : bstr,
   length : uint,
)
```

4.3. Device <-> Authorization Server

The protocol between device and authorization server (U and W in Figure 2) is carried out via the authenticator (V) with certain data protected between the endpoints using the equivalent of a hybrid encryption scheme (see, e.g., [I-D.irtf-cfrg-hpke]). The device uses the public DH key of the authorization server G_W together with the private DH key corresponding to ephemeral key G_X in EDHOC message_1, and vice versa for the authorization server. 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 betweeen U and W is carried between U and V in EAD_1 and EAD_2 (Section 4.4), and between V and W in Voucher Request/Response (Section 4.5).

4.3.1. Voucher Info

The external authorization data EAD_1 of EDHOC message_1 includes Voucher Info, which is the following CBOR sequence:

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

where

* LOC_W is location information about the authorization server, used by the authenticator

* ENC_ID is the encrypted blob carrying the identity of the device and an optional identity of the authenticator, passed on from the authenticator to the authorization server, calculated as follows:

ENC_ID is 'ciphertext' of COSE_Encrypt0 (Section 5.2-5.3 of [RFC8152]) 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,
  ? ID_V:           bstr,
 )

external_aad = (
    SS:             int,
 )
```

where

* ID_U is the identity of the device, for example a reference or pointer to the device certificate

* ID_V is the identity of the authenticator as presented to the authorization server. This may be a name in a name space agreed out-of-band and managed by a party trusted by the authorization server, for example a common name of an X.509 certificate signed by a CA trusted by the authorization server. The value may be obtained by the device through out-of-band means, possibly through secure network discovery.

* 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):

* transcript_hash = h''

* label is "EDHOC_ACE_AKE_AUTHZ_K_1"

* context = h''

* length is length of key of the EDHOC AEAD algorithm

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

* transcript_hash = h''

* label is "EDHOC_ACE_AKE_AUTHZ_IV_1"

* context = h''

* length is length of nonce of the EDHOC AEAD algorithm

4.3.2. Voucher

The voucher is an assertion by the authorization server to the device that the authorization server has performed the relevant verifications and that the device is authorized to continue the protocol with the authenticator. The Voucher is essentially a message authentication code which binds the identity of the authenticator to message_1 of EDHOC, integrity protected with the shared secret context between U and W.

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

* transcript_hash = h''

* label is "EDHOC_ACE_AKE_AUTHZ_MAC"

* context = bstr .cbor voucher_input

* length is EDHOC MAC length in bytes

where context is a CBOR bstr wrapping of voucher_input:

```
voucher_input = (
    V_TYPE:        int,
    SS:            int,
    G_X:           bstr,
    ID_U:          bstr,
    PK_V:          bstr,
)
```

where

* V_TYPE indicates the type of voucher used (TBD)

* SS is the selected cipher suite of the EDHOC protocol, see Section 4.2

* PK_V is a CWT Claims Set (CCS, [RFC8392]) containing the public authentication key of the authenticator encoded as a COSE_Key in the 'cnf' claim, see Section 3.5.3 of [I-D.ietf-lake-edhoc].

* G_X is encoded as in EDHOC message_1, see Section 3.7 of [I-D.ietf-lake-edhoc]

* ID_U is defined in Section 4.3

Editor's note: With the current definition of EAD as (ead_label, ead_value), do we need to redefine the voucher to be a CBOR map? Do we even need the V_TYPE?

4.4. Device <-> Authenticator

The device and authenticator run the EDHOC protocol authenticated with their public keys (PK_U and PK_V), see Figure 2. Normal EDHOC processing is omitted here.

4.4.1. Message 1

4.4.1.1. Device processing

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

The device sends EDHOC message_1 with EAD_1 = (L, Voucher_Info) where L is the External Auxiliary Data Label for this protocol (IANA registry created in Section 9.5 of [I-D.ietf-lake-edhoc]), and Voucher_Info is specified in Section 4.3.

4.4.1.2. Authenticator processing

The authenticator receives EDHOC message_1 from the device and processes as specified in Section 5.2.3 of [I-D.ietf-lake-edhoc], with the additional step that the presence of EAD with label L triggers the voucher request to the authorization server as described in Section 4.5. The exchange with V needs to be completed successfully for the EDHOC exchange to be continued.

4.4.2. Message 2

4.4.2.1. Authenticator processing

The authenticator receives the voucher response from the authorization server as described in Section 4.5.

The authenticator sends EDHOC message_2 to the device with EAD_2 = (L, Voucher) where L is the External Auxiliary Data Label for this protocol (IANA registry created in Section 9.5 of [I-D.ietf-lake-edhoc]) and 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 3.5.3 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 Sig_or_MAC_2 field calculated using the private key corresponding to PK_V is either signature or MAC depending on EDHOC method.

4.4.2.2. Device processing

In addition to normal EDHOC verifications, the device MUST verify the Voucher by performing the same calculation as in Section 4.3.2 using the SS, G_X and ID_U carried in message_1 and PK_V received in message_2. If the voucher calculated in this way is not identical to what was received in message_2, then the device MUST discontinue the protocol.

Editor's note: Consider replace SS, G_X, ID_U in Voucher with H(message_1), since that is already required by EDHOC to be cached by the initiator. H(message_1) needs to be added to VREQ message in that case.

4.4.3. Message 3

4.4.3.1. Device processing

If all verifications are passed, then the device sends EDHOC message_3.

The message field ID_CRED_I contains data enabling the authenticator to retrieve the public key of the device, PK_U. Since the authenticator before sending message_2 received a certificate of PK_U from the authorization server (see Section 4.5), ID_CRED_I SHALL be a COSE header_map of type 'kid' with the empty byte string as value:

```
ID_CRED_I =
{
  4 : h''
}
```

The Sig_or_MAC_3 field calculated using the private key corresponding to PK_U is either signature or MAC depending on EDHOC method.

EAD_3 MAY contain an enrolment request, see e.g. CSR specified in [I-D.mattsson-cose-cbor-cert-compress], or other request which the device is now authorized to make.

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

4.4.3.2. Authenticator processing

The authenticator performs the normal EDHOC verifications of message_3, with the exception that the Sig_or_MAC_3 field MUST be verified using the public key included in Cert_PK_U (see Section 4.5.2) received from the authorization server. The authenticator MUST ignore any key related information obtained in ID_CRED_I.

This enables the authenticator to verify that message_3 was generated by the device authorized by the authorization server as part of the associated Voucher Request/Response procedure (see Section 4.5).

4.5. Authenticator <-> Authorization Server

The authenticator and authorization server are assumed to have, or to be able to, set up a secure connection, for example TLS 1.3 authenticated with certificates. The authenticator 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 ephemeral public key G_X sent in EDHOC message_1 from device to authenticator serves as challenge/response nonce for the Voucher Request/Response Protocol, and binds together instances of the two protocols.

4.5.1. Voucher Request

4.5.1.1. Authenticator processing

Unless already in place, the authenticator and the authorization server establish a secure connection. The autenticator uses G_X received from the device as a nonce associated to this connection with the authorization server. If the same value of the nonce G_X is already used for a connection with this or other authorization server, the protocol SHALL be discontinued.

The authenticator sends the voucher request to the authorization server. The Voucher Request SHALL be a CBOR array as defined below:

```
Voucher_Request = [
    SS:              int,
    G_X:             bstr,
    ENC_ID:          bstr,
  ? PoP_V:           bstr,
]
```

where all 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

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. Authorization Server processing

The authorization server receives the voucher request, verifies and decrypts the identity ID_U of the device, and associates the nonce G_X to ID_U. If G_X is not unique among nonces associated to this identity, the protocol SHALL be discontinued. If ENC_ID also included the identity of V, ID_V, then the authorization server performs an additional check to verify that the identity of the authenticator who sent the voucher request over a secure session between V-W matches the identity of the authenticator as observed by U. If the identities of V as observed by U, and as observed by W, do not match, the protocol SHALL be discontinued.

4.5.2. Voucher Response

4.5.2.1. Authorization Server processing

The authorization server uses the identity of the device, ID_U, to look up the device certificate, Cert_PK_U.

The authorization server retrieves the public key of V used to authenticate the secure connection with the authenticator, and constructs the CWT Claims Set and the Voucher as defined in Section 4.3.2.

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

```
Voucher_Response = [
    G_X:            bstr,
    CERT_PK_U:      bstr,
    Voucher:        bstr
]
```

where

* G_X is copied from the associated voucher request.

* CERT_PK_U is the device certificate of the public key PK_U, issued by a trusted third party. The format of this certificate is out of scope.

* The Voucher is defined in Section 4.3.2.

4.5.2.2. Authenticator processing

The authenticator receives the voucher response from the authorization server over the secure connection. If the received G_X does not match the value of the nonce associated to the secure connection, the protocol SHALL be discontinued.

The authenticator verifies the certificate CERT_PK_U and that U is an admissible device, or else discontinues the protocol.

5. ACE Profile

The messages specified in this document may be carried between the endpoints in various protocols. This section defines an embedding as a profile of the ACE framework (see Appendix C of [I-D.ietf-ace-oauth-authz]).

* U plays the role of the ACE Resource Server (RS).

* V plays the role of the ACE Client (C).

* W plays the role of the ACE Authorization Server (AS).

Many readers who are used to the diagram having the Client on the left may be surprised at the cast of characters. The "resource" which C (V) is trying to access is the "ownership" of U. The AS (W) is the manufacturer (or previous owner) of RS (U), and is therefore in a position to grant C (V) ownership of RS (U).

C and RS use EDHOC's EAD to communicate. C and RS use the EDHOC protocol to protect their communication. EDHOC also provides mutual authentication of C and RS, assisted by the AS.

5.1. Protocol Overview

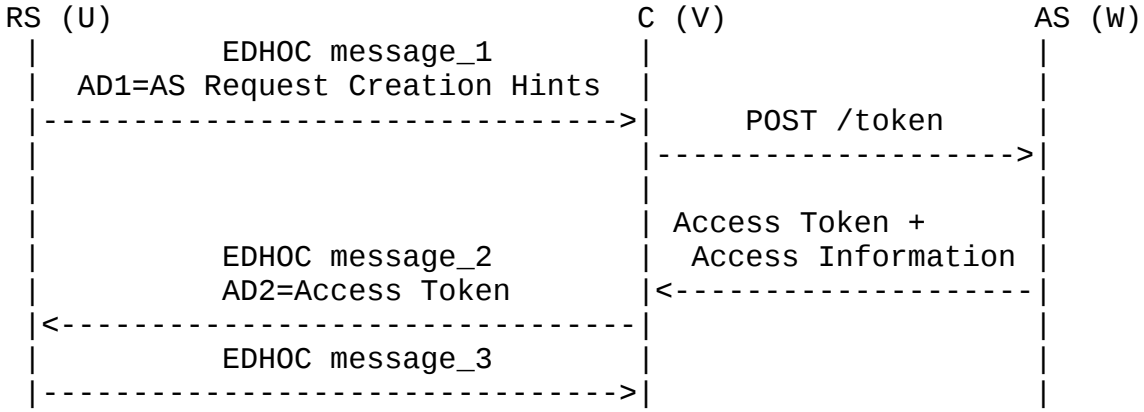


Figure 3: Overview of the protocol mapping to ACE

1. RS proactively sends the AS Request Creation Hints message to C to signal the information on where C can reach the AS.

2. RS piggybacks the AS Request Creation Hints message using Auxiliary Data of EDHOC message_1.

3. Before continuing the EDHOC exchange, based on the AS Request Creation Hints information, C sends a POST request to the token endpoint at the AS requesting the access token.

4. The AS issues an assertion to C that is cryptographically protected based on the secret shared between the AS and RS. In this profile, the assertion is encoded as a Bearer Token.

5. C presents this token to RS in EAD_2.

6. RS verifies the token based on the possession of the shared secret with the AS and authenticates C.

5.2. AS Request Creation Hints

Parameters that can appear in the AS Request Creation Hints message are specified in Section 5.3 of [I-D.ietf-ace-oauth-authz]. RS MUST use the "AS" parameter to transport LOC_W, i.e. an absolute URI where C can reach the AS. RS MUST use the "audience" parameter to transport the CBOR sequence consisting of two elements: SS, the selected cipher suite; ENC_ID, the AEAD encrypted blob containing identities. The "cnonce" parameter MUST be implied to G_X, i.e. the ephemeral public key of the RS in the underlying EDHOC exchange. The "cnonce" parameter is not carried in the AS Request Creation Hints message for byte saving reasons. AS Request Creation Hints MUST be carried within EAD_1.

An example EAD_1 value in CBOR diagnostic notation is shown below:

```
EAD_1:
{
    "AS" : "coaps://as.example.com/token",
    "audience": << h'73',h'737570657273...' >>
}
```

5.3. Client-to-AS Request

The protocol that provides the secure connection between C and the AS is out-of-scope. This can, for example, be TLS 1.3. What is important is that the two peers are mutually authenticated, and that the secure connection provides message integrity, confidentiality and freshness. It is also necessary for the AS to be able to extract the public key of C used in the underlying security handshake.

C sends the POST request to the token endpoint at the AS following Section 5.8.1. of [I-D.ietf-ace-oauth-authz]. C MUST set the "audience" parameter to the value received in AS Request Creation Hints. C MUST set the "cnonce" parameter to G_X, the ephemeral public key of RS in the EDHOC exchange.

An example exchange using CoAP and CBOR diagnostic notation is shown below:

```
Header: POST (Code=0.02)
Uri-Host: "as.example.com"
Uri-Path: "token"
Content-Format: "application/ace+cbor"
Payload:
{
    "audience" : << h'73',h'737570657273...' >>
    "cnonce" : h'756E73686172...'
}
```

5.4. AS-to-Client Response

Given successful authorization of C at the AS, the AS responds by issuing a Bearer token and retrieves the certificate of RS on behalf of C. The access token and the certificate are passed back to C, who uses it to complete the EDHOC exchange. This document extends the ACE framework by registering a new Access Information parameter:

rsp_ad: OPTIONAL. Carries additional information from the AS to C associated with the access token.

The AS-to-Client responsE MUST contain:

* ace_profileparameter set to "edhoc-authz"

* token_type parameter set to "Bearer"

* access_token as specified in Section 4.3.2

* rsp_ad = bstr .cbor cert_gx

```
cert_gx = (
    CERT_PK_U:        bstr,
    G_X:              bstr
)
```

where:

* CERT_PK_U is the RS's certificate, as discussed in Section 4.5.2. To be able to retrieve this certificate, the AS first needs to decrypt the audience value and obtain the RS's identity.

* G_X is the ephemeral key generated by RS in EDHOC message_1.

An example AS response to C is shown below:

```
      2.01 Created
      Content-Format: application/ace+cbor
      Max-Age: 3600
      Payload:
      {
          "ace_profile" : "edhoc-authz",
          "token_type" : "Bearer",
          "access_token" : h'666F726571756172746572...',
          "rsp_ad" : h'61726973746F64656D6F637261746963616C...'
      }
```

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, disclosed to the Responder in message_3. The sending of the certificate of U in the Voucher Response provides information about the identity of the device already before message_2, which changes the identity protection properties and thus needs to be validated against a given use case. The authorization server authenticates the authenticator, receives the Voucher Request, and can perform potential other verifications before sending the Voucher Response. This allows the authorization server to restrict information about the identity of the device to parties which are authorized to have that. However, if there are multiple authorized authenticators, the authorization server may not be able to distinguish between authenticator V which the device is connecting to and a misbehaving but authorized authenticator V' constructing a Voucher Request built from an eavesdropped message_1. A mitigation for this kind of misbehaving authenticator is that the device discovers the identity of the authenticator through out-of-bands means before attempting to enroll, and include the optional ID_V in the ENC_ID encrypted blob. For example, the network's discovery mechanism can carry asserted information on the associated identity of the authenticator. The use of ID_V also changes the identity protection assumptions since it requires U to know the identity of V before the protocol starts. The identity of V is still protected against passive adversaries, unless disclosed by the out-of-band mechanism by which U acquires information about the identity of V. The privacy considerations whether the identity of the device or of the authenticator is more sensitive need to be studied depending on a specific use case.

For use cases where neither the early disclosure of the device nor of the authenticator identities are deemed acceptable, the device certificate must not be sent in the Voucher Response, and the identity of V must be omitted. Instead, the device certificate could be retrieved from the authorization server or other certificate repository after message_3 is received by the authenticator, using the device identifier provided in ID_CRED_I as lookup. This would require the device identity to be transported in both message_1 (in EAD_1) and message_3 but would make the protocol comply with the default identity protection provided by EDHOC.

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.

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 the external authorization data EAD_1.

7. IANA Considerations

TODO: register rsp_ad ACE parameter

8. Informative References

[I-D.ietf-ace-oauth-authz]
Seitz, L., Selander, G., Wahlstroem, E., Erdtman, S., and H. Tschofenig, "Authentication and Authorization for Constrained Environments (ACE) using the OAuth 2.0 Framework (ACE-OAuth)", Work in Progress, Internet-Draft, draft-ietf-ace-oauth-authz-45, 29 August 2021, <https://www.ietf.org/archive/id/draft-ietf-ace-oauth-authz-45.txt>.

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

[I-D.ietf-lake-reqs]
Vucinic, M., Selander, G., Mattsson, J. P., and D. Garcia-Carrillo, "Requirements for a Lightweight AKE for OSCORE", Work in Progress, Internet-Draft, draft-ietf-lake-reqs-04, 8 June 2020, <https://www.ietf.org/archive/id/draft-ietf-lake-reqs-04.txt>.

[I-D.irtf-cfrg-hpke]
Barnes, R. L., Bhargavan, K., Lipp, B., and C. A. Wood, "Hybrid Public Key Encryption", Work in Progress, Internet-Draft, draft-irtf-cfrg-hpke-12, 2 September 2021, <https://www.ietf.org/archive/id/draft-irtf-cfrg-hpke-12.txt>.

[I-D.mattsson-cose-cbor-cert-compress]
Raza, S., Höglund, J., Selander, G., Mattsson, J. P., and M. Furuhed, "CBOR Encoded X.509 Certificates (C509 Certificates)", Work in Progress, Internet-Draft, draft-mattsson-cose-cbor-cert-compress-08, 22 February 2021, <https://www.ietf.org/archive/id/draft-mattsson-cose-cbor-cert-compress-08.txt>.

[I-D.palombini-core-oscore-edhoc]
Palombini, F., Tiloca, M., Hoeglund, R., Hristozov, S., and G. Selander, "Combining EDHOC and OSCORE", Work in Progress, Internet-Draft, draft-palombini-core-oscore-edhoc-02, 19 February 2021, <https://www.ietf.org/archive/id/draft-palombini-core-oscore-edhoc-02.txt>.

[I-D.selander-ace-coap-est-oscore]
Selander, G., Raza, S., Furuhed, M., Vucinic, M., and T. Claeys, "Protecting EST Payloads with OSCORE", Work in Progress, Internet-Draft, draft-selander-ace-coap-est-oscore-05, 5 May 2021, <https://www.ietf.org/archive/id/draft-selander-ace-coap-est-oscore-05.txt>.

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

[RFC3748] Aboba, B., Blunk, L., Vollbrecht, J., Carlson, J., and H. Levkowetz, Ed., "Extensible Authentication Protocol (EAP)", RFC 3748, DOI 10.17487/RFC3748, June 2004, <https://www.rfc-editor.org/info/rfc3748>.

[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>.

[RFC8152] Schaad, J., "CBOR Object Signing and Encryption (COSE)", RFC 8152, DOI 10.17487/RFC8152, July 2017, <https://www.rfc-editor.org/info/rfc8152>.

[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>.

[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>.

[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>.

[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>.

[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>.

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