Workgroup: Network Working Group
Internet-Draft:
draft-selander-ace-ake-authz-02
Published: 2 November 2020
Intended Status: Informational
Expires: 6 May 2021
Authors: G. Selander, Ericsson AB; J. Mattsson, Ericsson AB; M. Vucinic, INRIA
M. Richardson, Sandelman Software Works
A. Schellenbaum, Institute of Embedded Systems, ZHAW

Lightweight Authorization for Authenticated Key Exchange.

Abstract

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

Note to Readers

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

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction
   1.1. Terminology
2. Problem Description
3. Assumptions
   3.1. Device
   3.2. Domain Authenticator
   3.3. Authorization Server
4. The Protocol
   4.1. Device <-> Authorization Server
      4.1.1. Voucher
   4.2. Device <-> Authenticator
      4.2.1. Message 1
      4.2.2. Message 2
      4.2.3. Message 3
   4.3. Authenticator <-> Authorization Server
      4.3.1. Voucher Request
      4.3.2. Voucher Response
5. ACE Profile
   5.1. Protocol Overview
   5.2. AS Request Creation Hints
   5.3. Client-to-AS Request
   5.4. AS-to-Client Response
6. Security Considerations
7. IANA Considerations
8. Informative References
Authors' Addresses

1. Introduction

For constrained IoT deployments [RFC7228] the overhead 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 Auxiliary Data defined in EDHOC.

In this document we consider the target interaction to be "enrollment", for example certificate enrollment (such as [I-D.selander-ace-coap-est-oscore]) or joining a network for the first time (e.g. [I-D.ietf-6tisch-minimal-security]), 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.

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.

```
                   Voucher
             EDHOC  Info
+----------+  |    |   +---------------+  Voucher  +---------------+
|          |  |    |   |               |  Request  |               |
|  Device  |--|----o-->|    Domain     |---------->| Authorization |
|          |<-|---o----| Authenticator |<----------|     Server    |
|    (U)   |--|---|--->|      (V)      |  Voucher  |      (W)      |
|          |      |    |               |  Response |               |
+----------+      |    +---------------+           +---------------+
                  Voucher
```

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 Auxiliary 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 4.1).

* 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 4.1).

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 the device's certificate Cert_PK_U.

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

4. The Protocol

Three security sessions are going on in parallel (as detailed in the subsections):

- Between device (U) and (domain) authenticator (V),

- between authenticator and authorization server (W), and *between device and authorization server mediated by the authenticator.

The most relevant message fields of EDHOC [I-D.ietf-lake-edhoc] in this specification are shown within brackets { ... } (see Figure 2):

*G_X: the x-coordinate of the ephemeral public Diffie-Hellman key of party U

*AD_1: Auxiliary Data of message_1

*AD_2: Auxiliary Data of message_2

*ID_CRED_R: data enabling the party U to obtain the credentials containing the public authentication key of the responder V

*ID_CRED_I: data enabling the party V to obtain the credentials containing the public authentication key of the initiator U

*Sig_or_MAC_2: a signature or MAC made by party V with use of the private key of V

*Sig_or_MAC_3: a signature or MAC made by party U with use of the private key of U

```
U                                    V                              W
|                                    |                              |
|            {G_X, AD_1}             |                              |
+----------------------------------->|                              |
|          EDHOC message_1           |    G_X, CC, AEAD(K_1; ID_U)  |
|                                    +----------------------------->|
|                                    |    Voucher Request (VREQ)    |
|                                    |                              |
|                                    |    G_X, CERT_PK_U, Voucher   |
|                                    |<-----------------------------+
|                                    |    Voucher Response (VRES)   |
|  {ID_CRED_R, Sig_or_MAC_2, AD_2}   |                              |
|<-----------------------------------+                              |
|          EDHOC message_2           |                              |
|                                    |                              |
|      {ID_CRED_I, Sig_or_MAC_3}     |                              |
+----------------------------------->|                              |
|          EDHOC message_3           |                              |

where
AD_1 = (T0, LOC_W, CC, AEAD(K1; ID_U))
AD_2 = (T1, Voucher)
Voucher = AEAD(K_2; V_TYPE, PK_V, G_X, ID_U)
```

Figure 2: W-assisted authorization of AKE between U and V: EDHOC between U and V, and Voucher Request/Response between V and W.

4.1. Device <-> Authorization Server

The communication between device and authorization server is carried out via the authenticator protected between the endpoints (protocol between U and W in Figure 2) using an ECIES hybrid encryption scheme (see [I-D.irtf-cfrg-hpke]): The device uses the private key corresponding to its ephemeral DH key G_X generated for EDHOC message_1 (see Section 4.2) together with the static public DH key of the authorization server G_W to generate a shared secret G_XW. The shared secret is used to derive AEAD encryption keys to protect data between device and authorization server. The data is carried in AD_1 and AD_2 (between device and authenticator) and in Voucher Request/Response (between authenticator and authorization server).

TODO: Reference relevant ECIES scheme in [I-D.irtf-cfrg-hpke].

TODO: Define derivation of encryption keys (K_1, K_2) and nonces (N_1, N_2) for the both directions

AD_1 SHALL be the following CBOR sequence:

```
AD_1 = (
    T0:              int,
    LOC_W:           tstr,
    CC:              bstr,
    CIPHERTEXT_RQ:   bstr
)
```

where

* T0 is the Auxiliary Data Type (TBD in relevant IANA registry)

and the rest is Voucher Info:

* LOC_W is location information about the authorization server

* CC is a crypto context identifier for the security context between the device and the authorization server

* 'CIPHERTEXT_RQ' is the authenticated encrypted identity of the device with CC as Additional Data, more specifically:

'CIPHERTEXT_RQ' is 'ciphertext' of COSE_Encrypt0 (Section 5.2-5.3 of [RFC8152]) computed from the following:

* the secret key K_1

* the nonce N_1

* 'protected' is a byte string of size 0

* 'plaintext and 'external_aad' as below:

```
plaintext = (
    ID_U:             bstr
 )

external_aad = (
    CC:               bstr
 )
```

where

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

* CC is defined above.

AD_2 SHALL be the following CBOR sequence:

```
AD_2 = (
    T1:              int,
    Voucher:         bstr
)
```

where

* T1 is the Auxiliary Data Type (TBD in relevant IANA registry)

and 'Voucher' is defined in Section 4.1.1.

4.1.1. 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 consists essentially of a message authentication code which binds the identity of the authenticator to message_1 of EDHOC.

More specifically 'Voucher' is the 'ciphertext' of COSE_Encrypt0 (Section 5.2 of [RFC8152]) computed from the following:

* the secret key K_2

* the nonce N_2

* 'protected' is a byte string of size 0

* 'plaintext' is empty (plaintext = nil)

* 'external_aad' as below:

```
external_aad = bstr .cbor external_aad_array

external_aad_array = [
    V_TYPE:        int,
    PK_V:          bstr,
    G_X:           bstr,
    CC:            bstr,
    ID_U:          bstr
]
```

where

* 'V_TYPE' indicates the type of voucher used

* PK_V is a COSE_Key containing the public authentication key of the authenticator. The public key MUST be an Elliptic Curve Diffie-Hellman key, COSE key type 'kty' = 'EC2' or 'OKP'.

  - COSE_Keys of type OKP SHALL only include the parameters 1 (kty), -1 (crv), and -2 (x-coordinate). COSE_Keys of type EC2 SHALL only include the parameters 1 (kty), -1 (crv), -2 (x-coordinate), and -3 (y-coordinate). The parameters SHALL be encoded in decreasing order.

* G_X is the ephemeral key of the device sent in EDHOC message_1

* CC and ID_U are defined in Section 4.1

All parameters, except 'V_TYPE', are as received in the voucher request (see Section 4.3).

TODO: Consider making the voucher a CBOR Map to indicate type of voucher, to indicate the feature (cf. Section 4.3). Alternatively, include V_TYPE in 'unprotected'.

4.2. Device <-> Authenticator

The device and authenticator run the EDHOC protocol authenticated with public keys (PK_U and PK_V) of the device and the authenticator, see protocol between U and V in Figure 2. The normal EDHOC processing is omitted here.

4.2.1. Message 1

4.2.1.1. Device processing

The device composes EDHOC message_1 with specific parameters pre-configured, such as EDHOC method. The correlation properties (see Section 3.1 of [I-D.ietf-lake-edhoc]) are defined by the transport of the messages. The static public DH key G_W of the authorization server defines the ECDH curve of the selected cipher suite in SUITES_I. As part of the normal EDHOC processing, the device generates the ephemeral public key G_X. A new G_X MUST be generated for each execution of the protocol. The ephemeral key G_X is reused in the ECIES scheme, see Section 4.1.

The device sends EDHOC message_1 with AD_1 as specified in Section 4.1.

4.2.1.2. Authenticator processing

The authenticator receives EDHOC message_1 from the device, which triggers the voucher request to the authorization server as described in Section 4.3.

4.2.2. Message 2

4.2.2.1. Authenticator processing

The authenticator receives the voucher response from the authorization server as described in Section 4.3.

The authenticator sends EDHOC message_2 to the device with the voucher (see Section 4.1) in AD_2. The public key PK_V is carried in ID_CRED_R of message_2 encoded as a COSE header_map, see Section 4.1 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.

4.2.2.2. Device processing

In addition to normal EDHOC verifications, the device MUST verify the voucher by calculating the same message authentication code as when it was generated (see Section 4.1.1) and compare with what was received in message_2.

The input in this calculation includes:

* The ephemeral key G_X, sent in message_1.

* The identity ID_U, sent in message_1.

*The public key of the authenticator PK_V, received in message_2.

If the voucher does not verify, the device MUST discontinue the protocol.

4.2.3. Message 3

4.2.3.1. Device processing

If all verifications are passed, 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 4.3), 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.

AD_3 MAY contain an enrolment request, see [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].

4.2.3.2. 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 4.3.2) 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 4.3).

4.3. 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.

4.3.1. Voucher Request

4.3.1.1. 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 = [
    G_X:             bstr,
    CC:              bstr,
    CIPHERTEXT_RQ:   bstr
]
```

where the parameters are defined in Section 4.1.

TODO: Add in VREQ the optional parameters ?PK_V:bstr, and ?PoP:bstr to support the case when V uses different keys to authenticate to U and W. One case to study is when V authenticates to U with static DH and to W with signature.

4.3.1.2. 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.

4.3.2. Voucher Response

4.3.2.1. 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 corresponding COSE_Key as defined in Section 4.1.1.

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 4.1.1.

4.3.2.2. 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.

TODO: The voucher response may contain a "Voucher-info" field as an alternative to make the Voucher a CBOR Map (see Section 4.1)

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).

C and RS use the Auxiliary Data in the EDHOC protocol 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

```
RS                                C                     AS
|          EDHOC message_1        |                      |
|  AD1=AS Request Creation Hints  |                      |
|-------------------------------->|     POST /token      |
|                                 |--------------------->|
|                                 |                      |
|                                 | Access Token +       |
|          EDHOC message_2        |  Access Information  |
|          AD2=Access Token       |<---------------------|
|<--------------------------------|                      |
|          EDHOC message_3        |                      |
|-------------------------------->|                      |
```

Figure 3: Overview of the protocol mapping to ACE

RS proactively sends the AS Request Creation Hints message to C to signal the information on where C can reach the AS. RS piggybacks the AS Request Creation Hints message using Auxiliary Data of EDHOC message_1. 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. 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. C presents this token to RS in the Auxiliary Data of the EDHOC message_2. 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 5.1.2. 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: CC, the crypto context; CIPHERTEXT_RQ, the authenticated encrypted identity of the RS. 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 Auxiliary Data of the EDHOC message_1 (AD_1).

An example AD_1 value in CBOR diagnostic notation is shown below:

```
AD_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 5.6.1. 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.

When responding to C, the AS MUST set the "ace_profile" parameter to "edhoc-authz". The AS MUST set the "token_type" parameter to "Bearer". The access token MUST be formatted as specified in Section 4.1.1. The AS MUST set the "rsp_ad" parameter to the certificate of RS. To be able to do so, AS first needs to decrypt the audience value, and based on it retrieve the corresponding RS certificate.

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'666F72657175617274657572...',
    "rsp_ad" : h'61726973746F64656D6F637261746963616C...'
}
```

TODO: Add cnonce = G_X to this message to match the current version of the voucher response.

6. Security Considerations

TODO: Identity protection of device

TODO: Use of G_X as ephemeral key between device and authenticator, and between device and authorization server

7. IANA Considerations

TODO: CC registry

TODO: Voucher type registry

TODO: register rsp_ad ACE parameter

8. Informative References

[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>.

[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>.

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

[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-35, 24 June 2020, <http://www.ietf.org/internet-drafts/draft-ietf-ace-oauth-authz-35.txt>.

[I-D.mattsson-cose-cbor-cert-compress] Raza, S., Hoglund, J., Selander, G., Mattsson, J., and M. Furuhed, "CBOR Profile of X.509 Certificates", Work in Progress, Internet-Draft, draft-mattsson-cose-cbor-cert-compress-01, 13 July 2020, <http://www.ietf.org/internet-drafts/draft-mattsson-cose-cbor-cert-compress-01.txt>.

[I-D.irtf-cfrg-hpke] Barnes, R., Bhargavan, K., Lipp, B., and C. Wood, "Hybrid Public Key Encryption", Work in Progress, Internet-Draft, draft-irtf-cfrg-hpke-06, 23 October 2020,

<http://www.ietf.org/internet-drafts/draft-irtf-cfrg-hpke-06.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-03, 9 March 2020, <http://www.ietf.org/internet-drafts/draft-selander-ace-coap-est-oscore-03.txt>.

[I-D.ietf-6tisch-minimal-security]
Vucinic, M., Simon, J., Pister, K., and M. Richardson, "Constrained Join Protocol (CoJP) for 6TiSCH", Work in Progress, Internet-Draft, draft-ietf-6tisch-minimal-security-15, 10 December 2019, <http://www.ietf.org/internet-drafts/draft-ietf-6tisch-minimal-security-15.txt>.

[I-D.ietf-lake-edhoc]
Selander, G., Mattsson, J., and F. Palombini, "Ephemeral Diffie-Hellman Over COSE (EDHOC)", Work in Progress, Internet-Draft, draft-ietf-lake-edhoc-01, 2 August 2020, <http://www.ietf.org/internet-drafts/draft-ietf-lake-edhoc-01.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-00, 13 July 2020, <http://www.ietf.org/internet-drafts/draft-palombini-core-oscore-edhoc-00.txt>.

Authors' Addresses

Goeran Selander
Ericsson AB

Email: goran.selander@ericsson.com

John Preuss Mattsson
Ericsson AB

Email: john.mattsson@ericsson.com

Malisa Vucinic
INRIA

Email: malisa.vucinic@inria.fr

Michael Richardson

Sandelman Software Works

Email: mcr+ietf@sandelman.ca

Aurelio Schellenbaum
Institute of Embedded Systems, ZHAW

Email: aureliorubendario.schellenbaum@zhaw.ch