   OSCOAP profile of the Authentication and Authorization for Constrained
                         Environments Framework
                    draft-seitz-ace-oscoap-profile-04

Abstract

   This memo specifies a profile for the Authentication and
   Authorization for Constrained Environments (ACE) framework. It
   utilizes Object Security of CoAP (OSCOAP) and Ephemeral Diffie-
   Hellman over COSE (EDHOC) to provide communication security, server
   authentication, and proof-of-possession for a key owned by the client
   and bound to an OAuth 2.0 access token.

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 http://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 January 24, 2018.

Copyright Notice

Table of Contents

1. Introduction

This memo specifies a profile of the ACE framework [I-D.ietf-ace-oauth-authz]. In this profile, a client and a resource server use CoAP [RFC7252] to communicate. The client uses an access token, bound to a key (the proof-of-possession key) to authorize its access to the resource server. In order to provide communication security, proof of possession, and server authentication they use Object Security of CoAP (OSCOAP) [I-D.ietf-core-object-security] and Ephemeral Diffie-Hellman Over COSE (EDHOC) [I-D.selander-ace-cose-ecdhe]. Optionally the client and the resource server may also use CoAP and OSCOAP to communicate with the authorization server. The use of EDHOC in this profile in addition to OSCOAP, provides perfect forward secrecy (PFS) and the initial proof-of-possession, which ties the proof-of-possession key to an OSCOAP security context.

OSCOAP specifies how to use CBOR Object Signing and Encryption (COSE) [RFC8152] to secure CoAP messages. In order to provide replay and reordering protection OSCOAP also introduces sequence numbers that are used together with COSE. EDHOC specifies an authenticated Diffie-Hellman protocol that allows two parties to use CBOR [RFC7049]

and COSE in order to establish a shared secret key with perfect
forward secrecy.

1.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT",
"SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this
document are to be interpreted as described in [RFC2119]. These
words may also appear in this document in lowercase, absent their
normative meanings.

Certain security-related terms such as "authentication",
"authorization", "confidentiality", "(data) integrity", "message
authentication code", and "verify" are taken from [RFC4949].

Since we describe exchanges as RESTful protocol interactions HTTP
[RFC7231] offers useful terminology.

Terminology for entities in the architecture is defined in OAuth 2.0
[RFC6749] and [I-D.ietf-ace-actors], such as client (C), resource
server (RS), and authorization server (AS). It is assumed in this
document that a given resource on a specific RS is associated to a
unique AS.

Note that the term "endpoint" is used here following its OAuth
definition, which is to denote resources such as /token and
/introspect at the AS and /authz-info at the RS. The CoAP [RFC7252]
definition, which is "An entity participating in the CoAP protocol"
is not used in this memo.

2. Client to Resource Server

The use of OSCOAP for arbitrary CoAP messages is specified in
[I-D.ietf-core-object-security]. This section defines the specific
uses and their purpose for securing the communication between a
client and a resource server, and the parameters needed to negotiate
the use of this profile with the token endpoint at the authorization
server as specified in section 5.5 of the ACE framework
[I-D.ietf-ace-oauth-authz].

2.1. Signaling the use of OSCOAP

A client requests a token at an AS via the /token endpoint. This
follows the message formats specified in section 5.5.1 of the ACE
framework [I-D.ietf-ace-oauth-authz].

The AS responding to a successful access token request as defined in
section 5.5.2 of the ACE framework can signal that the use of OSCOAP

is REQUIRED for a specific access token by including the "profile"
parameter with the value "coap_oscoap" or "coap_oscoap_edhoc" in the
access token response. This means that the client MUST use OSCOAP
towards all resource servers for which this access token is valid,
and follow Section 2.2.1 to derive the security context to run
OSCOAP, if the profile has value "coap_oscoap", or follow
Section 2.2.2 to derive the security context to run OSCOAP, if the
profile has value "coap_oscoap_edhoc".

The error response procedures defined in section 5.5.3 of the ACE
framework are unchanged by this profile.

Note the the client and the authorization server MAY OPTIONALLY use
OSCOAP to protect the interaction via the /token endpoint. See
section 3 for details.

2.2. Key establishment for OSCOAP

Section 3.2 of OSCOAP [I-D.ietf-core-object-security] defines how to
derive a security context based on a shared master secret and a few
other parameters, established between client and server. The proof-
of-possession key (pop-key) provisioned from the AS MAY, in case of
pre-shared keys, be used directly as master secret in OSCOAP.
Alternatively the pop-key (symmetric or asymmetric) MAY be used to
authenticate the messages in the key exchange protocol EDHOC
[I-D.selander-ace-cose-ecdhe], from which a master secret is derived.

2.2.1. Using the pop-key with OSCOAP directly (OSCOAP)

If OSCOAP is used directly with the symmetric pop-key as master
secret, then the AS MUST provision the following data, in response to
the access token request:

o a master secret

o the sender identifier

o the recipient identifier

Additionally, the AS MAY provision the following data, in the same
response. In case these parameters are omitted, the default values
are used as described in section 3.2. of
[I-D.ietf-core-object-security].

o an AEAD algorithm

o a KDF algorithm

o a salt

The master secret MUST be communicated as COSE_Key in the 'cnf'
parameter of the access token response as defined in section 5.5.4.5
of [I-D.ietf-ace-oauth-authz]. The AEAD algorithm MAY be included as
the 'alg' parameter in the COSE_Key; the KDF algorithm MAY be
included as the 'kdf' parameter of the COSE_Key and the salt MAY be
included as the 'slt' parameter of the COSE_Key as defined in table
1. The same parameters MUST be included as metadata of the access
token, if the token is a CWT [I-D.ietf-ace-cbor-web-token], the same
COSE_Key structure MUST be placed in the 'cnf' claim of this token.
The AS MUST also assign identifiers to both client and RS, which are
then used as Sender ID and Recipient ID in the OSCOAP context as
described in section 3.1. of [I-D.ietf-core-object-security]. These
identifiers MUST be unique in the set of all clients and RS
identifiers for a certain AS. Moreover, these MUST be included in
the COSE_Key as header parameters, as defined in table 1.

We assume in this document that a resource is associated to one
single AS, which makes it possible to assume unique identifiers for
each client requesting a particular resource to a RS. If this is not
the case, collisions of identifiers may appear in the RS, in which
case the RS needs to have a mechanism in place to disambiguate
identifiers or mitigate their effect.

Note that C should set the Sender ID of its security context to the
clientId value received and the Recipient ID to the serverId value,
and RS should do the opposite.

```
+----------+-------+---------------+-----------+------------------+
| name     | label | CBOR type     | registry  | description      |
+----------+-------+---------------+-----------+------------------+
| clientId | TBD   | bstr          |           | Identifies the   |
|          |       |               |           | client in an     |
|          |       |               |           | OSCOAP context   |
|          |       |               |           | using this key   |
|          |       |               |           |                  |
| serverId | TBD   | bstr          |           | Identifies the   |
|          |       |               |           | server in an     |
|          |       |               |           | OSCOAP context   |
|          |       |               |           | using this key   |
|          |       |               |           |                  |
| kdf      | TBD   | bstr          |           | Identifies the   |
|          |       |               |           | KDF algorithm in |
|          |       |               |           | an OSCOAP context|
|          |       |               |           | using this key   |
|          |       |               |           |                  |
| slt      | TBD   | bstr          |           | Identifies the   |
|          |       |               |           | master salt in   |
|          |       |               |           | an OSCOAP context|
|          |       |               |           | using this key   |
+----------+-------+---------------+-----------+------------------+
```
Table 1: Additional common header parameters for COSE_Key

Figure 1 shows an example of such an AS response, in CBOR diagnostic
notation without the tag and value abbreviations.

```
        Header: Created (Code=2.01)
        Content-Type: "application/cose+cbor"
        Payload:
        {
          "access_token" : b64'SlAV32hkKG ...
           (remainder of access token omitted for brevity)',
          "profile" : "coap_oscoap",
          "expires_in" : "3600",
          "cnf" : {
            "COSE_Key" : {
              "kty" : "Symmetric",
              "alg" : "AES-CCM-16-64-128",
              "clientId" : b64'qA',
              "serverId" : b64'Qg',
              "k" : b64'+a+Dg2jjU+eIiOFCa9lObw'
            }
          }
        }
```

Figure 1: Example AS response with OSCOAP parameters.

Figure 2 shows an example CWT, containing the necessary OSCOAP parameters in the 'cnf' claim, in CBOR diagnostic notation without tag and value abbreviations.

```
    {
      "aud" : "tempSensorInLivingRoom",
      "iat" : "1360189224",
      "exp" : "1360289224",
      "scope" :  "temperature_g firmware_p",
      "cnf" : {
        "COSE_Key" : {
          "kty" : "Symmetric",
          "alg" : "AES-CCM-16-64-128",
          "clientId" : b64'Qg',
          "serverId" : b64'qA',
          "k" : b64'+a+Dg2jjU+eIiOFCa9lObw'
        }
      }
    }
```

Figure 2: Example CWT with OSCOAP parameters.

2.2.2. Using the pop-key with EDHOC (EDHOC+OSCOAP)

If EDHOC is used together with OSCOAP, and the pop-key (symmetric or asymmetric) is used to authenticate the messages in EDHOC, then the AS MUST provision the following data, in response to the access token request:

   o a symmetric or asymmetric key (associated to the RS)

   o a key identifier;

   How these parameters are communicated depends on the type of key
   (asymmetric or symmetric).

   Note that in the case described in this section, the 'expires_in'
   parameter, defined in section 4.2.2. of [RFC6749] defines the
   lifetime in seconds of both the access token and the shared secret.
   After expiration, C MUST acquire a new access token from the AS, and
   run EDHOC again, as specified in this section

2.2.2.1. Using Asymmetric Keys

   In case of an asymmetric key, C MUST communicate its own asymmetric
   key to the AS in the 'cnf' parameter of the access token request, as
   specified in section 5.5.1 of [I-D.ietf-ace-oauth-authz]; the AS MUST
   communicate the RS's public key to C in the response, in the 'rs_cnf'
   parameter, as specified in section 5.5.1 of
   [I-D.ietf-ace-oauth-authz]. Note that the RS's public key MUST
   include a 'kid' parameter, and that the value of the 'kid' MUST be
   included in the access token, to let the RS know which of its public
   keys C used. If the access token is a CWT
   [I-D.ietf-ace-cbor-web-token], the key identifier MUST be placed
   directly in the 'cnf' structure (if the key is only referenced).

   Figure 3 shows an example of such a request in CBOR diagnostic
   notation without tag and value abbreviations.

```
Header: POST (Code=0.02)
Uri-Host: "server.example.com"
Uri-Path: "token"
Content-Type: "application/cose+cbor"
Payload:
{
  "grant_type" : "client_credentials",
  "cnf" : {
    "COSE_Key" : {
      "kid" : "client_key"
      "kty" : "EC",
      "crv" : "P-256",
      "x" : b64'usWxHK2PmfnHKwXPS54m0kTcGJ90UiglWiGahtagnv8',
      "y" : b64'IBOL+C3BttVivg+lSreASjpkttcsz+1rb7btKLv8EX4'
    }
  }
}
```

Figure 3: Example access token request (OSCOAP+EDHOC, asymmetric).

Figure 4 shows an example of a corresponding response in CBOR diagnostic notation without tag and value abbreviations.

```
Header: Created (Code=2.01)
Content-Type: "application/cose+cbor"
Payload:
{
  "access_token" : b64'SlAV32hkKG ...
   (contains "kid" : "server_key")',
  "profile" : "coap_oscoap_edhoc",
  "expires_in" : "3600",
  "rs_cnf" : {
    "COSE_Key" : {
      "kid" : "server_key"
      "kty" : "EC",
      "crv" : "P-256",
      "x" : b64'cGJ90UiglWiGahtagnv8usWxHK2PmfnHKwXPS54m0kT',
      "y" : b64'reASjpkttcsz+1rb7btKLv8EX4IBOL+C3BttVivg+lS'
    }
  }
}
```

Figure 4: Example AS response (EDHOC+OSCOAP, asymmetric).

2.2.2.2. **Using Symmetric Keys**

   In the case of a symmetric key, the AS MUST communicate the key to
   the client in the 'cnf' parameter of the access token response, as
   specified in section 5.5.2. of [I-D.ietf-ace-oauth-authz]. AS MUST
   also select a key identifier, that MUST be included as the 'kid'
   parameter either directly in the 'cnf' structure, as in figure 4 of
   [I-D.ietf-ace-oauth-authz], or as the 'kid' parameter of the
   COSE_key, as in figure 6 of [I-D.ietf-ace-oauth-authz].

   Figure 5 shows an example of the necessary parameters in the AS
   response to the access token request when EDHOC is used. The example
   uses CBOR diagnostic notation without tag and value abbreviations.

```
     Header: Created (Code=2.01)
     Content-Type: "application/cose+cbor"
     Payload:
     {
       "access_token" : b64'SlAV32hkKG ...
        (remainder of access token omitted for brevity)',
       "profile" : "coap_oscoap_edhoc",
       "expires_in" : "3600",
       "cnf" : {
         "COSE_Key" : {
           "kty" : "Symmetric",
           "kid" : b64'5tOS+h42dkw',
           "k" : b64'+a+Dg2jjU+eIiOFCa9lObw'
         }
       }
     }
```

       Figure 5: Example AS response (EDHOC+OSCOAP, symmetric).

   In both cases, the AS MUST also include the same key identifier as
   'kid' parameter in the access token metadata. If the access token is
   a CWT [I-D.ietf-ace-cbor-web-token], the key identifier MUST be
   placed inside the 'cnf' claim as 'kid' parameter of the COSE_Key or
   directly in the 'cnf' structure (if the key is only referenced).

   Figure 6 shows an example CWT containing the necessary EDHOC+OSCOAP
   parameters in the 'cnf' claim, in CBOR diagnostic notation without
   tag and value abbreviations.

```
      {
        "aud" : "tempSensorInLivingRoom",
        "iat" : "1360189224",
        "exp" : "1360289224",
        "scope" :   "temperature_g firmware_p",
        "cnf" : {
          "COSE_Key" : {
            "kty" : "Symmetric",
            "kid" : b64'5tOS+h42dkw',
            "k" : b64'+a+Dg2jjU+eIiOFCa9lObw'
        }
      }
```

        Figure 6: Example CWT with EDHOC+OSCOAP, symmetric case.

   All other parameters defining OSCOAP security context are derived
   from EDHOC message exchange, including the master secret (see
   Appendix C.2 of [I-D.selander-ace-cose-ecdhe]).

2.2.2.3. Processing

   To provide forward secrecy and mutual authentication in the case of
   pre-shared keys, pre-established raw public keys or with X.509
   certificates it is RECOMMENDED to use EDHOC
   [I-D.selander-ace-cose-ecdhe] to generate the keying material. EDHOC
   MUST be used as defined in Appendix C, with the following additions
   and modifications.

   The first EDHOC message is sent after the access token is posted to
   the /authz-info endpoint of the RS as specified in section 5.7.1 of,
   as defined in [I-D.ietf-ace-oauth-authz]. Then the EDHOC message_1
   is sent and the EDHOC protocol is initiated
   [I-D.selander-ace-cose-ecdhe]).

   Before the RS continues with the EDHOC protocol and responds to this
   token submission request, additional verifications on the access
   token are done: the RS SHALL process the access token according to
   [I-D.ietf-ace-oauth-authz]. If the token is valid then the RS
   continues processing EDHOC following Appendix C of
   [I-D.selander-ace-cose-ecdhe], otherwise it discontinues EDHOC and
   responds with the error code as specified in
   [I-D.ietf-ace-oauth-authz].

   o In case the EDHOC verification fails, the RS MUST return an error
      response to the client with code 4.01 (Unauthorized).

   o If RS has an access token for C but not for the resource that C
      has requested, RS MUST reject the request with a 4.03 (Forbidden).

   o If RS has an access token for C but it does not cover the action C
      requested on the resource, RS MUST reject the request with a 4.05
      (Method Not Allowed).

   o If all verifications above succeeds, further communication between
      client and RS is protected with OSCOAP, including the RS response
      to the OSCOAP request.

   In the case of EDHOC being used with symmetric keys, the protocol in
   section 5 of [I-D.selander-ace-cose-ecdhe] MUST be used. If the key
   is asymmetric, the RS MUST also use an asymmetric key for
   authentication. This key is known to the client through the access
   token response (see section 5.5.2 of the ACE framework). In this
   case the protocol in section 4 of [I-D.selander-ace-cose-ecdhe] MUST
   be used.

   Figure 7 illustrates the message exchanges for using OSCOAP+EDHOC
   (step C in figure 1 of [I-D.ietf-ace-oauth-authz]).

```
                        Resource
                 Client    Server
                   |          |
                   |          |
                   +--------->| Header: POST (Code=0.02)
                   | POST     | Uri-Path:"authz-info"
                   |          | Content-Type: application/cbor
                   |          | Payload: access token
                   |          |
                   |          |
                   +--------->| Header: POST (Code=0.02)
                   |    POST  | Uri-Path: "/.well-known/edhoc"
                   |          | Content-Type: application/edhoc
                   |          | Payload: EDHOC message_1
                   |          |
                   |<---------+ Header: 2.04 Changed
                   |    2.04  | Content-Type: application/edhoc
                   |          | Payload: EDHOC message_2
                   |          |
                   +--------->| Header: POST (Code=0.02)
                   |    POST  | Uri-Path: "/.well-known/edhoc"
                   |          | Content-Type: application/edhoc
                   |          | Payload: EDHOC message_3
                   |          |
                   |<---------+ Header: 2.04 Changed
                   |    2.04  |
                   |          |
          start of protected communication
                   |          |
                   +--------->| CoAP request +
                   |  OSCOAP  | Object-Security option
                   | request  |
                   |          |
                   |<---------+ CoAP response +
                   |  OSCOAP  | Object-Security option
                   | response |
                   |          |
```
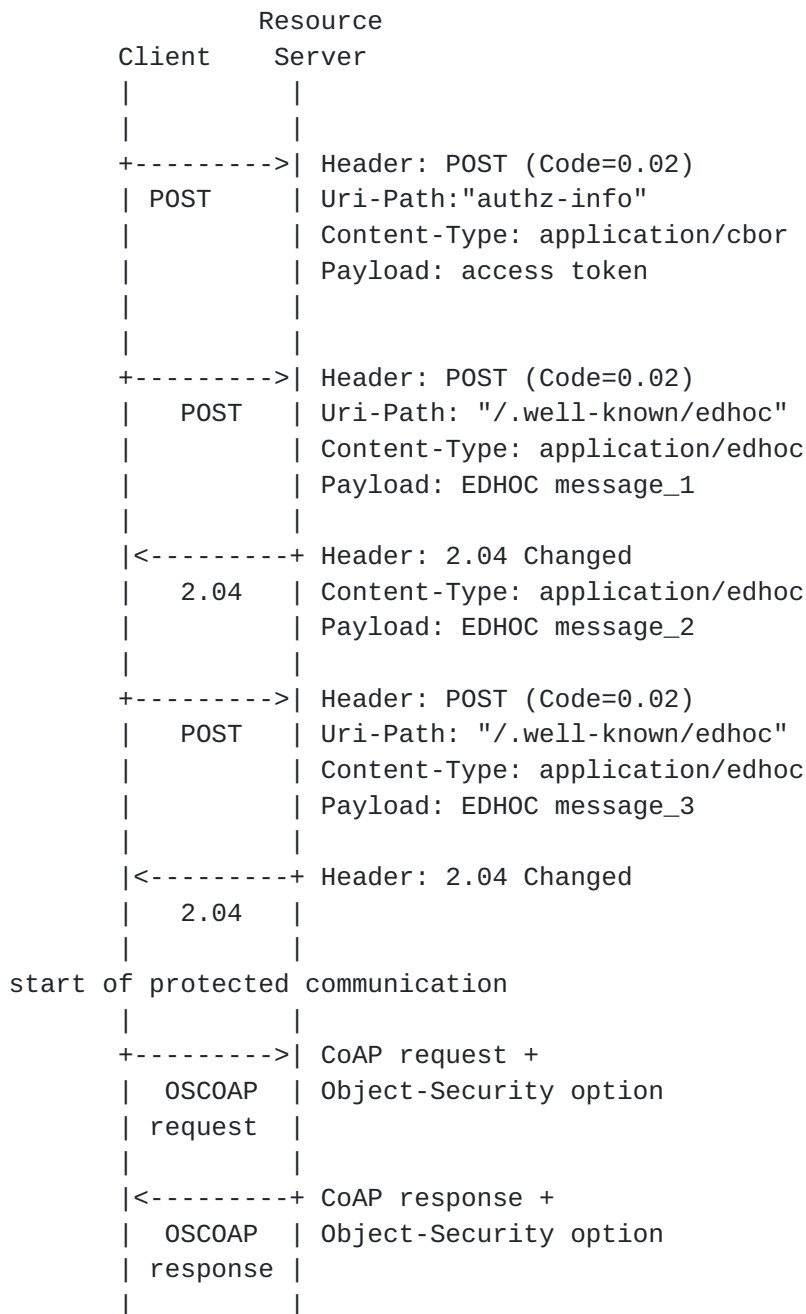
        Figure 7: Access token and key establishment with EDHOC

2.3. Client to Authorization Server

   As specified in the ACE framework section 5.5
   [I-D.ietf-ace-oauth-authz], the Client and AS can also use CoAP
   instead of HTTP to communicate via the token endpoint. This section
   specifies how to use OSCOAP between Client and AS together with CoAP.
   The use of OSCOAP for this communication is OPTIONAL in this profile,
   other security protocols (such as DTLS) MAY be used instead.

The client and the AS are expected to have pre-established
credentials (e.g. raw public keys). How these credentials are
established is out of scope for this profile. Furthermore the client
and the AS communicate using CoAP through the token endpoint as
specified in section 5.5 of [I-D.ietf-ace-oauth-authz]. At first
point of contact, prior to making the token request and response, the
client and the AS MAY perform an EDHOC exchange with the pre-
established credentials to create forward secret keying material for
use with OSCOAP. Subsequent requests and the responses MUST be
protected with OSCOAP.

3. **Resource Server to Authorization Server**

As specified in the ACE framework section 5.6
[I-D.ietf-ace-oauth-authz], the RS and AS can also use CoAP instead
of HTTP to communicate via the introspection endpoint. This section
specifies how to use OSCOAP between RS and AS together with CoAP.
The use of OSCOAP for this communication is OPTIONAL in this profile,
other security protocols (such as DTLS) MAY be used instead.

The RS and the AS are expected to have pre-established credentials
(e.g. symmetric keys). How these credentials are established is out
of scope for this profile. Furthermore the RS and the AS communicate
using CoAP through the introspection endpoint as specified in section
5.6 of [I-D.ietf-ace-oauth-authz]. At first point of contact, prior
to making the introspection request and response, the RS and the AS
MAY perform an EDHOC exchange with the pre-established credentials to
create forward secret keying material for use with OSCOAP.
Subsequent requests and the responses MUST be protected with OSCOAP

4. **Security Considerations**

TBD.

5. **Privacy Considerations**

TBD.

6. **IANA Considerations**

TBD. 'coap_oscoap' as profile id. Header parameters 'sid', 'rid',
'kdf' and 'slt' for COSE_Key.

7. **Acknowledgments**

The author wishes to thank Jim Schaad, Goeran Selander and Marco
Tiloca for the input on this memo. The error responses specified in

section 2.2. were originally specified by Gerdes et al. in
[I-D.gerdes-ace-dcaf-authorize].

8. References

8.1. Normative References

[I-D.ietf-ace-cbor-web-token]
          Jones, M., Wahlstroem, E., Erdtman, S., and H. Tschofenig,
          "CBOR Web Token (CWT)", draft-ietf-ace-cbor-web-token-07
          (work in progress), July 2017.

[I-D.ietf-ace-oauth-authz]
          Seitz, L., Selander, G., Wahlstroem, E., Erdtman, S., and
          H. Tschofenig, "Authentication and Authorization for
          Constrained Environments (ACE)", draft-ietf-ace-oauth-
          authz-06 (work in progress), March 2017.

[I-D.ietf-core-object-security]
          Selander, G., Mattsson, J., Palombini, F., and L. Seitz,
          "Object Security of CoAP (OSCOAP)", draft-ietf-core-
          object-security-04 (work in progress), July 2017.

[I-D.selander-ace-cose-ecdhe]
          Selander, G., Mattsson, J., and F. Palombini, "Ephemeral
          Diffie-Hellman Over COSE (EDHOC)", draft-selander-ace-
          cose-ecdhe-07 (work in progress), July 2017.

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

[RFC7252] Shelby, Z., Hartke, K., and C. Bormann, "The Constrained
          Application Protocol (CoAP)", RFC 7252,
          DOI 10.17487/RFC7252, June 2014,
          <http://www.rfc-editor.org/info/rfc7252>.

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

8.2. Informative References

   [I-D.gerdes-ace-dcaf-authorize]
              Gerdes, S., Bergmann, O., and C. Bormann, "Delegated CoAP
              Authentication and Authorization Framework (DCAF)", draft-
              gerdes-ace-dcaf-authorize-04 (work in progress), October
              2015.

   [I-D.ietf-ace-actors]
              Gerdes, S., Seitz, L., Selander, G., and C. Bormann, "An
              architecture for authorization in constrained
              environments", draft-ietf-ace-actors-05 (work in
              progress), March 2017.

   [RFC4949] Shirey, R., "Internet Security Glossary, Version 2",
              FYI 36, RFC 4949, DOI 10.17487/RFC4949, August 2007,
              <http://www.rfc-editor.org/info/rfc4949>.

   [RFC6749] Hardt, D., Ed., "The OAuth 2.0 Authorization Framework",
              RFC 6749, DOI 10.17487/RFC6749, October 2012,
              <http://www.rfc-editor.org/info/rfc6749>.

   [RFC7049] Bormann, C. and P. Hoffman, "Concise Binary Object
              Representation (CBOR)", RFC 7049, DOI 10.17487/RFC7049,
              October 2013, <http://www.rfc-editor.org/info/rfc7049>.

   [RFC7231] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer
              Protocol (HTTP/1.1): Semantics and Content", RFC 7231,
              DOI 10.17487/RFC7231, June 2014,
              <http://www.rfc-editor.org/info/rfc7231>.

Authors' Addresses

   Ludwig Seitz
   RISE SICS AB
   Scheelevagen 17
   Lund 22370
   SWEDEN

   Email: ludwig.seitz@ri.se

   Francesca Palombini
   Ericsson AB
   Farogatan 6
   Kista SE-16480 Stockholm
   Sweden

   Email: francesca.palombini@ericsson.com

Martin Gunnarsson
RISE SICS AB
Scheelevagen 17
Lund   22370
SWEDEN

Email: martin.gunnarsson@ri.se