ACE Working Group
Internet-Draft
Intended status: Standards Track
Expires: 2 September 2026

G. Selander
J. Preu Mattsson
Ericsson
M. Tiloca
R. Hglund
RISE

1 March 2026

Ephemeral Diffie-Hellman Over COSE (EDHOC) and Object Security for Constrained Environments (OSCORE) Profile for Authentication and Authorization for Constrained Environments (ACE) draft-ietf-ace-edhoc-oscore-profile-10

Abstract

This document specifies a profile for the Authentication and Authorization for Constrained Environments (ACE) framework. It utilizes Ephemeral Diffie-Hellman Over COSE (EDHOC) for achieving mutual authentication between an ACE-OAuth client and resource server, and it binds an authentication credential of the client to an ACE-OAuth access token. EDHOC also establishes an Object Security for Constrained RESTful Environments (OSCORE) Security Context, which is used to secure communications between the client and resource server when accessing protected resources according to the authorization information indicated in the access token. This profile can be used to delegate management of authorization information from a resource-constrained server to a trusted host with less severe limitations regarding processing power and memory.

About This Document

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

Status information for this document may be found at https://datatracker.ietf.org/doc/draft-ietf-ace-edhoc-oscore-profile/.

Discussion of this document takes place on the Authentication and Authorization for Constrained Environments (ace) Working Group mailing list (mailto:ace@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/ace/. Subscribe at https://www.ietf.org/mailman/listinfo/ace/.

Source for this draft and an issue tracker can be found at https://github.com/ace-wg/ace-edhoc-oscore-profile.

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 2 September 2026.

Table of Contents

1. Introduction

This document defines the "coap_edhoc_oscore" profile of the ACE-OAuth framework [RFC9200]. This profile addresses a "zero-touch" constrained setting where authenticated and authorized operations can be performed with low overhead without endpoint specific configurations.

Like in the "coap_oscore" profile [RFC9203], also in this profile a client (C) and a resource server (RS) use the Constrained Application Protocol (CoAP) [RFC7252] to communicate, and Object Security for Constrained RESTful Environments (OSCORE) [RFC8613] to protect their communications. However, this profile uses the Ephemeral Diffie-Hellman Over COSE (EDHOC) protocol [RFC9528] to establish the OSCORE Security Context. The processing of requests for specific protected resources is identical to what is defined in the "coap_oscore" profile.

When using this profile, C accesses protected resources hosted at RS according to the authorization information specified in an access token, which is issued by a trusted authorization server (AS) and is bound to an authentication credential of C. This differs from the "coap_oscore" profile, where the access token is bound to a secret that is generated by AS and is used to derive the OSCORE Security Context.

Whereas [RFC9200] recommends the use of CBOR Web Tokens (CWTs) [RFC8392] as access tokens, this profile requires it (see Section 3.3.1). Furthermore, this profile requires that, for messages exchanged between C and AS to request and provide an access token, the payload is encoded in CBOR [RFC8949] (see Section 3.1 and Section 3.3), which ACE requires if CoAP is used (see Section 5 of [RFC9200]) and recommends otherwise (see Section 3 of [RFC9200]).

An authentication credential can be a raw public key, e.g., encoded as a CWT Claims Set (CCS) [RFC8392]; or a public key certificate, e.g., an X.509 certificate [RFC5280] or a CBOR-encoded C509 certificate [I-D.ietf-cose-cbor-encoded-cert]); or a different type of data structure containing the public key of the peer in question.

The ACE framework establishes what those authentication credentials are, and may transport the actual authentication credentials by value or uniquely refer to them. If an authentication credential is pre-provisioned or can be obtained over less constrained links, then it suffices that ACE provides a unique reference such as a certificate hash (e.g., by using the COSE header parameter "x5t" [RFC9360]). This is in the same spirit as EDHOC, where the authentication credentials specified in the ID_CRED_x message fields can be transported by value or referred to (see Section 3.5.3 of [RFC9528]).

In general, AS and RS are likely to have trusted access to each other's authentication credentials, since AS acts on behalf of RS as per the trust model of ACE. Also, AS needs to have some information about C, including the relevant authentication credential, in order to identify C when it requests an access token and to determine what access rights it can be granted. However, the authentication credential of C may potentially be conveyed (or uniquely referred to) within the request for an access token that C makes to AS.

The establishment of an association between RS and AS in an ACE ecosystem is out of scope, but one solution is to build on the same primitives as used in this document, i.e., EDHOC for authentication and OSCORE for communication security, using for example [I-D.ietf-lake-authz] for onboarding RS with AS and [I-D.ietf-ace-coap-est-oscore] for establishing a trust anchor in RS. A similar procedure can also be applied between C and AS for registering a client and for the establishment of a trust anchor.

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.

Certain security-related terms such as "authentication", "authorization", "confidentiality", "(data) integrity", "Message Authentication Code (MAC)", "Hash-based Message Authentication Code (HMAC)", and "verify" are taken from [RFC4949].

RESTful terminology follows HTTP [RFC9110].

Readers are expected to be familiar with the terms and concepts defined in CoAP [RFC7252], OSCORE [RFC8613], and EDHOC [RFC9528].

Readers are also expected to be familiar with the terms and concepts of the ACE framework described in [RFC9200] and in [RFC9201].

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

Note that the term "endpoint" is used here, as in [RFC9200], following its OAuth definition, which is to denote resources such as /token and /introspect at AS, and /authz-info at RS. The CoAP [RFC7252] definition, which is "An entity participating in the CoAP protocol", is not used in this document.

The authorization information (authz-info) resource refers to the authorization information endpoint as specified in [RFC9200]. The term "claim" is used in this document with the same semantics as in [RFC9200], i.e., it denotes information carried in the access token or returned from introspection.

Concise Binary Object Representation (CBOR) [RFC8949][RFC8742] and Concise Data Definition Language (CDDL) [RFC8610] are used in this document. CDDL predefined type names, especially bstr for CBOR byte strings and tstr for CBOR text strings, are used extensively in this document.

Examples throughout this document are expressed in CBOR diagnostic notation as defined in Section 8 of [RFC8949] and Appendix G of [RFC8610]. Diagnostic notation comments are often used to provide a textual representation of the numeric parameter names and values.

In the CBOR diagnostic notation used in this document, constructs of the form e'SOME_NAME' are replaced by the value assigned to SOME_NAME in the CDDL model shown in Figure 11 of Appendix C. For example, {e'session_id' : h'01', e'cipher_suites': 3} stands for {0 : h'01', 2 : 3}.

Note to RFC Editor: Please delete the paragraph immediately preceding this note. Also, in the CBOR diagnostic notation used in this document, please replace the constructs of the form e'SOME_NAME' with the value assigned to SOME_NAME in the CDDL model shown in Figure 11 of Appendix C. Finally, please delete this note.

2. Protocol Overview

This section gives an overview of how to use the ACE framework [RFC9200] together with the lightweight authenticated key exchange protocol EDHOC [RFC9528]. By doing so, the client (C) and the resource server (RS) generate an OSCORE Security Context [RFC8613] associated with authorization information, and use that Security Context to protect their communications. The parameters needed by C to negotiate the use of this profile with the authorization server (AS), as well as the OSCORE setup process, are described in detail in the following sections.

RS maintains a collection of authentication credentials. These are associated with OSCORE Security Contexts and with authorization information for all clients that RS is communicating with. The authorization information is used to enforce policies for processing requests from those clients.

The ACE framework describes how integrity protected authorization information propagates from AS to RS. This profile describes how C requests from AS an access token specifying authorization information for the resources that C wants to access at RS, by sending an access token request to the /token endpoint at AS (see Section 5.8 of [RFC9200]).

If the request is granted, then AS can provide C with an access token when sending a response to C, or instead upload the access token directly to RS as per the alternative workflow defined in [I-D.ietf-ace-workflow-and-params]. The latter option is not detailed further in this document.

After that, C and RS run the EDHOC protocol, with C using the authentication credential of RS obtained from AS. If C has obtained an access token from AS, then C specifies the access token within an External Authorization Data (EAD) field of an EDHOC message sent during the EDHOC session (see Section 3.8 of [RFC9528]). RS uses the authentication credential of C bound to and specified in the access token. How C and RS run EDHOC is detailed in Section 4.2.

If C and RS successfully complete the EDHOC execution and the validation of each other’s authentication credential, they are mutually authenticated and derive the OSCORE Security Context as per Appendix A.1 of [RFC9528].

Figure 1 outlines an example of the message flow. A more detailed description of the message flow is shown in Appendix A.1.

From then on and as long as there is a valid access token, C effectively gains authorized and secure access to protected resources at RS using the established OSCORE Security Context. When RS receives a request from C protected with an OSCORE Security Context derived from an EDHOC session implementing this profile, then that OSCORE Security Context is used to retrieve the uniquely associated access token determining the access rights of C.

The OSCORE Security Context is discarded when an access token (whether the same or a different one) is used to successfully derive a new OSCORE Security Context for C.

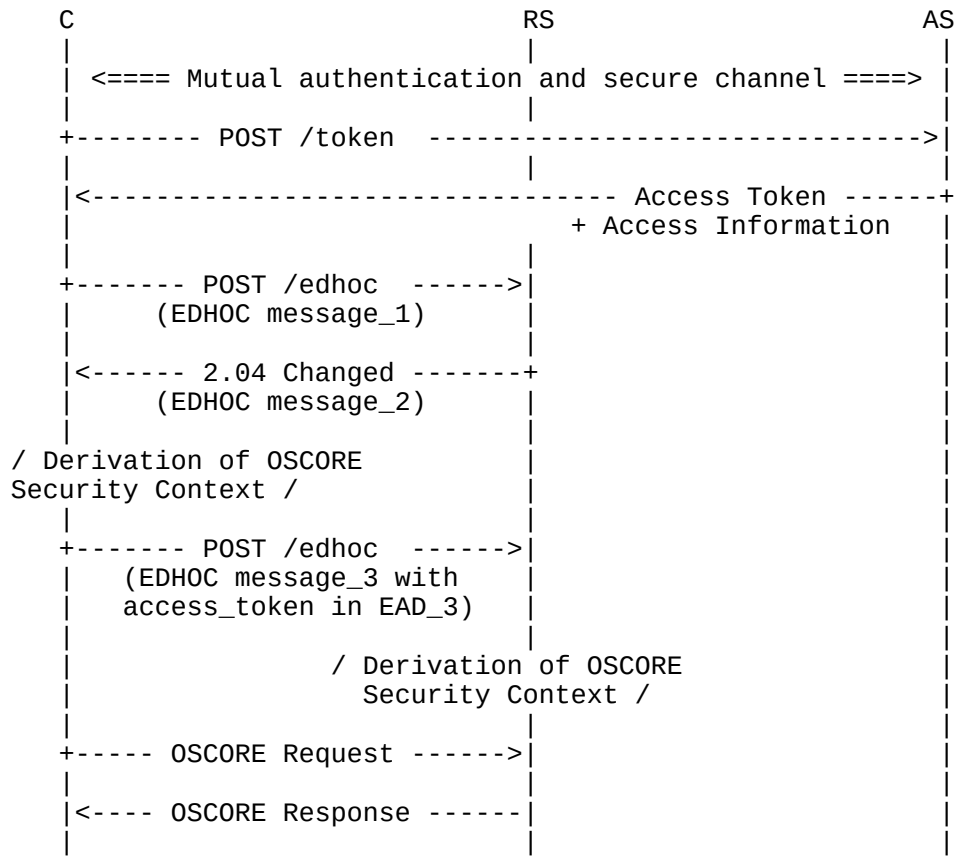


Figure 1: Protocol Outline using the EDHOC Forward Message Flow.

As long as the OSCORE Security Context and the access token are valid, C can contact AS to request an update of its access rights, by sending a similar request as described above to the /token endpoint. This request also includes a "session identifier" (see Section 3.4) provided by AS in the response to the initial access token request, which allows AS to retrieve the data that it previously shared with C. The session identifier is assigned by AS and used to identify a series of access tokens, called a "token series" (see Section 3.2).

If C has obtained an access token from AS for updating its access rights belonging to the same token series, then C transfers the access token to RS using the /authz-info endpoint as specified in Section 5.10 of [RFC9200], where the exchanged CoAP messages are protected by the previously established OSCORE Security Context (see Section 4.6). If the access token is valid, RS replies to the request with a 2.01 (Created) response.

Upon successful update of access rights, the new issued access token effectively becomes the latest in its token series also for RS, but the session identifier remains the same. When the latest access token of a token series becomes invalid (e.g., when it expires or gets revoked), that token series ends.

Figure 2 outlines the message flow for updating access rights.

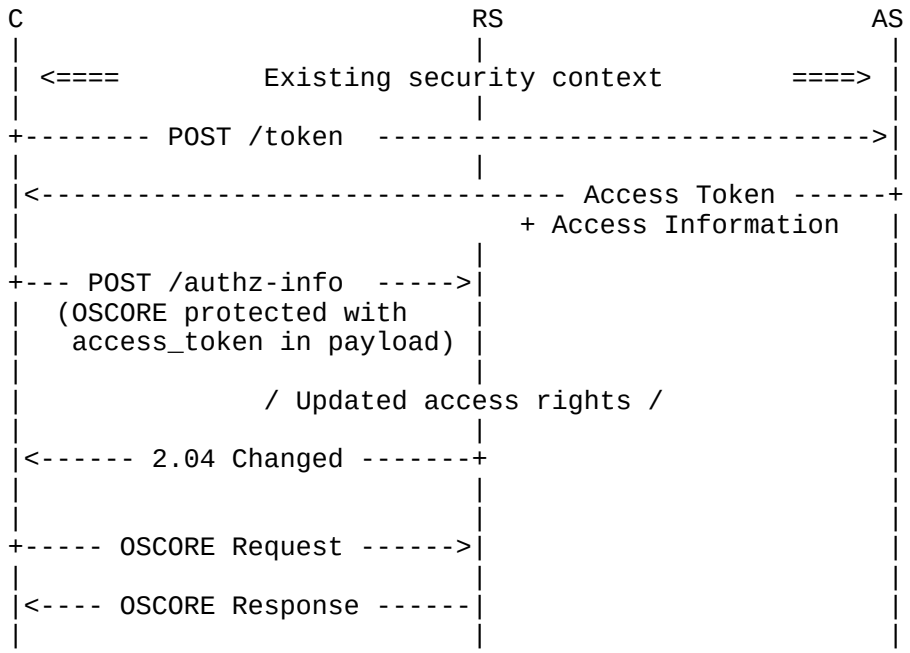


Figure 2: Protocol Outline for Updating Access Rights.

3. Client-AS Communication

The following subsections describe the details of the POST request and response to the /token endpoint between C and AS.

In this exchange, C provides AS with its own authentication credential AUTH_CRED_C. Then, AS issues the access token as securely bound to AUTH_CRED_C, by including it or uniquely referring to it in the access token. Together with the access token, AS provides C with a set of parameters that enable C to run EDHOC with RS. In particular, these parameters include information about the authentication credential AUTH_CRED_RS of RS, which is transported by value or uniquely referred to.

The request to the /token endpoint and the corresponding response can include EDHOC_Information, which is a CBOR map object containing information related to an EDHOC implementation (see Section 3.4). This object is transported in the "edhoc_info" parameter registered in Section 13.2 and Section 13.3.

3.1. C-to-AS: POST to /token endpoint

The client-to-AS request is specified in Section 5.8.1 of [RFC9200].

The client MUST send this POST request to the /token endpoint over a secure channel that guarantees authentication, message integrity, and confidentiality (see Section 5).

When using this profile, the payload of the POST request MUST be encoded in CBOR [RFC8949], i.e., the request has media-type "application/ace+cbor". In order to reduce the number of libraries that C has to support, it is RECOMMENDED that C and AS use CoAP as message transfer protocol, OSCORE as security protocol, and EDHOC to establish an OSCORE Security Context.

AUTH_CRED_C is specified in the "req_cnf" parameter [RFC9201] of the POST request, either transported by value or uniquely referred to. AS could explicitly ask C to provide its authentication credential AUTH_CRED_C by value in the Access Token Request, e.g., by relying on the method defined in [I-D.ietf-ace-workflow-and-params].

For AUTH_CRED_C, its authentication credential type MUST be one of those supported by EDHOC, e.g., CBOR Web Tokens (CWTs) and CWT Claims Sets (CCSs) [RFC8392], X.509 certificates [RFC5280], and C509 certificates [I-D.ietf-cose-cbor-encoded-cert]. Consequently, the "req_cnf" parameter MUST specify a confirmation method suitable for the type of AUTH_CRED_C, e.g., "x5chain" or "x5t" when AUTH_CRED_C is an X.509 certificate transported by value or referred to, respectively.

Note that EDHOC does not admit the use of "naked" COSE_Keys as authentication credentials. The closest admitted authentication credential type is a CCS containing a COSE_Key in a "cnf" claim and possibly other claims, which can be transported by value using the confirmation method "kccs". Therefore, the "req_cnf" parameter MUST NOT specify the confirmation method "COSE_Key" (CBOR abbreviation: 1).

When receiving an Access Token request including the "req_cnf" parameter, AS checks whether it is already storing the authentication credential of C, namely AUTH_CRED_C, specified in "req_cnf" by value or reference.

If this is not the case, AS retrieves AUTH_CRED_C, either using the "req_cnf" parameter or some other trusted source. After that, AS validates the actual AUTH_CRED_C.

In either case, the AS also needs to verify that C is in possession of the private key corresponding to the public key associated with AUTH_CRED_C. This may already have been accomplished, for example, by C authenticating to AS using AUTH_CRED_C as authentication credential, or by some other trusted party vouching for C to the AS. Alternatively, it is possible to use an approach for achieving proof of possession similar to that in Section 3.2 of [I-D.ietf-ace-group-oscore-profile].

In case of successful validations, AS stores AUTH_CRED_C as a valid authentication credential. Otherwise, the Client-to-AS request MUST be declined.

An example of client-to-AS request is shown in Figure 3. In this example, C specifies its own authentication credential by reference, as the hash of an X.509 certificate carried in the "x5t" field of the "req_cnf" parameter.

```
Header: POST (Code=0.02)
Uri-Host: "as.example.com"
Uri-Path: "token"
Content-Format: application/ace+cbor
Payload:
{
  / audience / 5 : "tempSensor4711",
  / scope /    9 : "read",
  / req_cnf /  4 : {
    e'x5t' : [-15, h'79F2A41B510C1F9B']
  }
}
```

Figure 3: Example of C-to-AS POST /token request for an access token.

If C wants to update its access rights without changing an existing OSCORE Security Context, it MUST specify an EDHOC_Information object in the "edhoc_info" parameter of its POST request to the /token endpoint. The EDHOC_Information object MUST include the "session_id" field. This POST request MUST NOT include the "req_cnf" parameter. An example of such a request is shown in Figure 4.

The identifier "session_id" is assigned by AS as discussed in Section 3.2, and identifies an ongoing token series associated with the pair (AUTH_CRED_C, AUTH_CRED_RS). That is, previous access tokens in that series were issued by AS to C, as bound to AUTH_CRED_C and intended for RS as identified by AUTH_CRED_RS.

Note that the same "session_id" value might identify multiple ongoing token series, e.g., if those are associated with the same client but different resource servers. In this case, AS can use the "session_id" value together with other information such as the targeted audience (see Section 5.8.1 of [RFC9200]) and the authenticated identity of C, in order to determine the exact token series to which the new requested access token has to be added.

If the identifier specified in the "session_id" parameter of the POST request identifies multiple, ongoing token series of which C has an access token, then C MUST specify the "audience" parameter in the POST request. In particular, the value of the "audience" parameter MUST be the same value of the "audience" parameter in the POST request that C previously sent to AS, for requesting the first access token in the token series to which the new requested access token has to be added.

AS MUST verify that the received "session_id" identifies a token series to which a still valid access token issued for C and RS belongs. If that is not the case, the Client-to-AS request MUST be declined with the error code "invalid_request" as defined in Section 5.8.3 of [RFC9200].

```
Header: POST (Code=0.02)
Uri-Host: "as.example.com"
Uri-Path: "token"
Content-Format: application/ace+cbor
Payload:
{
  / audience /      5 : "tempSensor4711",
  / scope /         9 : "write",
  e'edhoc_info_param' : {
     e'session_id' : h'01'
  }
}
```

Figure 4: Example of C-to-AS POST /token request for updating access rights to an access token.

3.2. Token Series

This document refers to "token series" as a series of access tokens that are sorted in chronological order of release and are characterized by the following properties:

* Issued by the same AS.

* Issued to the same C, and associated with the same authentication credential of C.

* Issued for the same RS as identified by the same authentication credential.

Upon a successful update of access rights (see Section 3.3.3), the new issued access token becomes the latest in its token series. When the latest access token of a token series becomes invalid (e.g., due to its expiration or revocation), the token series it belongs to ends.

In this profile, a token series comprises access tokens that are used between a given pair (C, RS), are bound to the same authentication credential AUTH_CRED_C of C, and specify the same value in the "session_id" field of the EDHOC_Information object (see Section 3.4) in their "edhoc_info" claim.

AS assigns the value of "session_id" when issuing the first access token of a new series. That "session_id" value remains fixed throughout the series lifetime.

When assigning the "session_id" value, AS MUST ensure that it was not used in a previous series whose access tokens share both the following properties with the access tokens of the new series, irrespective of the used ACE profile:

* issued to the same client C; and

* issued for the same RS.

In case the access token is issued for a group-audience (see Section 6.9 of [RFC9200]), what is defined above applies, with the difference that the token series is associated with all the RSs in the group-audience, as indicated by their respective AUTH_CRED_RS.

3.3. AS-to-C: Response

After verifying the POST request to the /token endpoint and that C is authorized to access protected resources at RS, AS responds as defined in Section 5.8.2 of [RFC9200], with potential modifications as detailed below.

When using this profile, consistent with what is specified in Section 3.1, the payload of the response from AS MUST be encoded in CBOR [RFC8949], i.e., the response has media-type "application/ace+cbor".

If the request from C was invalid or not authorized, AS returns an error response as described in Section 5.8.3 of [RFC9200].

AS can signal that the use of EDHOC and OSCORE as per this profile is REQUIRED for a specific access token, by including the "ace_profile" parameter with the value "coap_edhoc_oscore" in the access token response. This means that C MUST use EDHOC with RS and derive an OSCORE Security Context, as specified in Section 4.2. After that, C MUST use the established OSCORE Security Context to protect communications with RS, when accessing protected resources at RS according to the authorization information indicated in the access token. Usually, it is assumed that constrained devices will be pre-configured with the necessary profile, so that this kind of profile signaling can be omitted.

According to this document, the AS provides the access token to C, by specifying it in the "access_token" parameter of the access token response. An alternative workflow where the access token is uploaded by AS directly to RS is described in [I-D.ietf-ace-workflow-and-params].

When issuing the first access token of a token series, AS MUST include the following data in the response to C.

* The "edhoc_info" parameter conveying an EDHOC_Information object (see Section 3.4). The EDHOC_Information object MUST include the "session_id" field specifying the identifier of the token series which the issued access token belongs to.

  The EDHOC_Information object MAY include additional fields (see Section 3.4) to convey information about RS. This information is based on knowledge that AS has about RS, e.g., from a previous onboarding process, with particular reference to what RS supports as EDHOC peer.

In case the access token is issued for a group-audience (see Section 6.9 of [RFC9200]), the information specified in the EDHOC_Information object refers to the group-audience as a whole. Therefore, it is appropriate for AS to define group-audiences comprising RSs that are all aligned in terms of supported EDHOC features and configurations.

* A unique identification of the authentication credential of RS, AUTH_CRED_RS. This is specified in the "rs_cnf" parameter defined in [RFC9201]. AUTH_CRED_RS can be transported by value or referred to by means of an appropriate identifier. C could explicitly ask AS to provide AUTH_CRED_RS by value in the Access Token Response, e.g., by relying on the method defined in [I-D.ietf-ace-workflow-and-params].

  When issuing the first access token ever to a pair (C, RS) using a pair of corresponding authentication credentials (AUTH_CRED_C, AUTH_CRED_RS), it is expected that the response to C includes AUTH_CRED_RS by value.

  When later issuing further access tokens to the same pair (C, RS) using the same AUTH_CRED_RS, it is expected that the response to C includes AUTH_CRED_RS by reference.

  For AUTH_CRED_RS, its authentication credential type MUST be one of those supported by EDHOC. Consequently, the "rs_cnf" parameter MUST specify a confirmation method suitable for the type of AUTH_CRED_RS. That is, the same considerations about AUTH_CRED_C and the "req_cnf" parameter made in Section 3.1 hold for AUTH_CRED_RS and the "rs_cnf" parameter.

When issuing any access token of a token series, the response from AS MUST NOT include the "cnf" parameter.

When issuing an access token for dynamically updating access rights (i.e., the access token is not the first in its token series), the response from AS MUST NOT include the "edhoc_info" and "rs_cnf" parameters (see Section 3.3.3).

Figure 5 shows an example of an AS response. The "rs_cnf" parameter specifies the authentication credential of RS, as an X.509 certificate transported by value in the "x5chain" field. The access token and the authentication credential of RS have been truncated for readability.

```
Header: Created (Code=2.01)
   Content-Format: application/ace+cbor
   Payload:
   {
     / access_token / 1 : h'8343a1010aa2044c53...0f6a'
       / remainder of access token (CWT) omitted for brevity /,
     / ace_profile / 38 : e'coap_edhoc_oscore',
     / expires_in /   2 : 3600,
     / rs_cnf /      41 : {
       e'x5chain' : h'3081ee3081a1a00302...77bc'
         / remainder of the credential omitted for brevity /
     }
     e'edhoc_info_param' : {
       e'session_id'    : h'01',
       e'methods'       : [0, 1, 2, 3],
       e'cipher_suites' : 0
     }
   }
```

Figure 5: Example of AS-to-C Access Token response with EDHOC and OSCORE profile.

3.3.1. Access Token

To avoid the complexity of different encodings, an access token of this profile SHALL be a CBOR Web Token (CWT) [RFC8392].

When issuing any access token of a token series, AS MUST include the following claims in the access token:

* The "edhoc_info" claim defined in Section 13.5 and conveying an EDHOC_Information object (see Section 3.4).

  The EDHOC_Information object MUST include the "session_id" field specifying the identifier of the token series which the issued access token belongs to. The "session_id" value is the same one included in the EDHOC_Information object of the response to C from the /token endpoint (see Section 3.3), when providing C with the first access token in the series.

* The "cnf" claim, specifying the authentication credential AUTH_CRED_C that C specified in its POST request to the /token endpoint, when requesting the first access token in the series which the issued access token belongs to (see Section 3.1).

  In the access token, AUTH_CRED_C can be transported by value or uniquely referred to by means of an appropriate identifier. Yet, consistent with the considerations about AUTH_CRED_C and the

"req_cnf" parameter made in Section 3.1, the "cnf" claim of the access token MUST specify a confirmation method suitable for the type of AUTH_CRED_C.

When issuing the first access token of a token series, the confirmation method used in the "cnf" claim MUST be the same one used in the "req_cnf" parameter of the corresponding POST request from C to the /token endpoint.

When issuing the first access token ever to a pair (C, RS) using a pair of corresponding authentication credentials (AUTH_CRED_C, AUTH_CRED_RS), it is expected that AUTH_CRED_C is included by value in the "cnf" claim of the access token.

When later issuing further access tokens to the same pair (C, RS) using the same AUTH_CRED_C, it is expected that AUTH_CRED_C is identified by reference in the "cnf" claim of the access token.

Either transported by value or identified by reference, the authentication credential specified in the "cnf" claim MUST be exactly the one that was specified in the "req_cnf" parameter of the POST request to the /token endpoint, when C requested the first access token in the series which the issued access token belongs to. That is, AS MUST NOT bind to the access token an authentication credential other than the one specified by C.

When issuing the first access token of a token series, AS MAY include additional fields in the EDHOC_Information object (see Section 3.4) specified in the "edhoc_info" claim of the access token.

The access token needs to be protected for various reasons. To prevent manipulation of the content, it needs to be integrity protected. Also, RS has to be able to verify that the access token is issued by a trusted AS, by achieving source authentication. Depending on the use case and deployment, the access token may need to be confidentiality protected, for example due to privacy reasons.

AS protects the access token using a COSE method [RFC9052] as specified in [RFC8392]. Depending on the audience, there can be different ways to most appropriately ensure the confidentiality of an access token. For an audience comprising a single RS, the CWT Claims Set may be wrapped in COSE_Encrypt / COSE_Encrypt0. Instead, if the access token needs to be read by multiple RSs, then the CWT Claims Set may be wrapped in COSE_Sign / COSE_Sign1 and confidentiality protection is applied during transport, by including the access token in the EAD_3 field of EDHOC message_3 sent by C to RS, when using the EDHOC forward message flow (see Section 4.2).

Figure 6 shows an example of CWT Claims Set, including the relevant EDHOC parameters in the "edhoc_info" claim. The "cnf" claim specifies the authentication credential of C, as an X.509 certificate transported by value in the "x5chain" field. The authentication credential of C has been truncated for readability.

```
   {
    / aud /   3 : "tempSensorInLivingRoom",
    / iat /   6 : 1563451500,
    / exp /   4 : 1563453000,
    / scope / 9 :  "temperature_g firmware_p",
    / cnf /   8 : {
      e'x5chain' : h'3081ee3081a1a00302...77bc'
        / remainder of the credential omitted for brevity /
    }
    e'edhoc_info_claim' : {
      e'session_id'    : h'01',
      e'methods'       : [0, 1, 2, 3],
      e'cipher_suites' : 0
    }
  }
```

Figure 6: Example of CWT Claims Set with EDHOC parameters.

3.3.2. Processing at C

When receiving an access token response including the "rs_cnf" parameter, C checks whether it is already storing the authentication credential of RS, namely AUTH_CRED_RS, specified in "rs_cnf" by value or reference.

If this is not the case, C retrieves AUTH_CRED_RS, either using the "rs_cnf" parameter or some other trusted source. After that, C validates the actual AUTH_CRED_RS. In case of successful validation, C stores AUTH_CRED_RS as a valid authentication credential. Otherwise, C MUST delete the access token.

3.3.3. Update of Access Rights

If C has a valid OSCORE Security Context associated with a valid access token, then C can send a request to AS for updating its access rights while preserving the same OSCORE Security Context.

If the request is granted, then AS generates a new access token, where the EDHOC_Information object specified in the "edhoc_info" claim MUST include only the "session_id" field. The access token is provisioned to RS either via C as specified in this document, or directly as described in [I-D.ietf-ace-workflow-and-params]. In either case, the access token response from AS to C MUST NOT include the "edhoc_info" and "rs_cnf" parameters.

As defined in Section 3.3.1, the "session_id" field is included in the EDHOC_Information object specified in the "edhoc_info" claim of the new access token. This allows RS to identify the old access token to supersede, as well as the OSCORE Security Context already shared between C and RS and to be associated with the new access token.

3.4. EDHOC_Information

EDHOC_Information is an object including information that guides two peers towards executing the EDHOC protocol. In particular, the EDHOC_Information is defined to be serialized and transported between nodes, as specified by this document, but it can also be used by other specifications.

In the "coap_edhoc_oscore" profile of the ACE-OAuth framework, which is specified in this document, the EDHOC_Information object MUST be encoded as CBOR. However, for easy applicability to other contexts, we define also the JSON encoding.

The EDHOC_Information can be encoded either as a JSON object or as a CBOR map. The set of common fields that can appear in an EDHOC_Information can be found in the IANA "EDHOC Information" registry defined in Section 13.10 for extensibility.

All EDHOC_Information parameters are optional. The initial set of parameters defined in this document is summarized in Table 1 and is specified below.

EDHOC_Information parameters are also categorized, i.e., each has one of two possible types, namely prescriptive or non-prescriptive:

* A prescriptive parameter is used to provide an authoritative statement about how an execution of EDHOC has to be performed. An example is the "message_4" parameter indicating whether the use of EDHOC message_4 in an EDHOC session is mandatory or not.

* A non-prescriptive parameter is used to provide convenient information to consider when executing EDHOC, e.g., in terms of features supported by other peers. Such information is not necessarily exhaustive. An example is the "methods" parameter indicating a set of supported EDHOC methods.

This categorization helps coordinate the use of EDHOC application profiles Section 3.9 of [RFC9528] in a robust way, e.g., by using the means defined in [I-D.ietf-lake-app-profiles].

Name	CBOR label	CBOR type	Registry	Description	Type
session_id	0	bstr		Identifier of a session	P
methods	1	int or array	EDHOC Method Type registry	Set of supported EDHOC methods	NP
cipher_suites	2	int or array	EDHOC Cipher Suites registry	Set of supported EDHOC cipher suites	NP
message_4	3	True or False		Mandatory use of EDHOC message_4	P
comb_req	4	True or False		Support for the EDHOC + OSCORE combined request	NP
uri_path	5	tstr		URI-path of the EDHOC resource	P
cred_types	6	int or array	EDHOC Authentication Credential Types registry	Set of supported types of authentication credentials for EDHOC	NP
id_cred_types	7	int	COSE Header	Set of	NP

		or tstr or array	Parameters registry	supported types of authentication credential identifiers for EDHOC		
eads	8	uint or array	EDHOC External Authorization Data registry	Set of supported EDHOC External Authorization Data (EAD) items	NP	
initiator	9	True or False		Support for the EDHOC Initiator role	NP	
responder	10	True or False		Support for the EDHOC Responder role	NP	
trust_anchors	11	map	EDHOC Trust Anchor Purposes registry and EDHOC Trust Anchor Types registry	Set of supported trust anchors	NP	

Table 1: EDHOC_Information Parameters. Types: P (Prescriptive), NP (Non- Prescriptive)

* session_id: This parameter identifies a ’session’ which the EDHOC information is associated with, but does not necessarily identify a specific EDHOC session. In this document, "session_id" identifies a token series. In JSON, the "session_id" value is a Base64 encoded byte string. In CBOR, the "session_id" type is a byte string, and has label 0.

* methods: This parameter specifies a set of supported EDHOC methods (see Section 3.2 of [RFC9528]). If the set is composed of a single EDHOC method, this is encoded as an integer. Otherwise, the set is encoded as an array of integers, where each array element encodes one EDHOC method. In JSON, the "methods" value is an integer or an array of integers. In CBOR, the "methods" is an integer or an array of integers, and has label 1.

* cipher_suites: This parameter specifies a set of supported EDHOC cipher suites (see Section 3.6 of [RFC9528]). If the set is composed of a single EDHOC cipher suite, this is encoded as an integer. Otherwise, the set is encoded as an array of integers, where each array element encodes one EDHOC cipher suite. In JSON, the "cipher_suites" value is an integer or an array of integers. In CBOR, the "cipher_suites" is an integer or an array of integers, and has label 2.

* message_4: This parameter indicates whether the EDHOC message_4 (see Section 5.5 of [RFC9528]) is supported. In JSON, the "message_4" value is a boolean. In CBOR, "message_4" is the simple value true (0xf5) or false (0xf4), and has label 3.

* comb_req: This parameter indicates whether the combined EDHOC + OSCORE request defined in [RFC9668]) is supported. In JSON, the "comb_req" value is a boolean. In CBOR, "comb_req" is the simple value true (0xf5) or false (0xf4), and has label 4.

* uri_path: This parameter specifies the path component of the URI of the EDHOC resource where EDHOC messages have to be sent as requests. In JSON, the "uri_path" value is a string. In CBOR, "uri_path" is a text string, and has label 5.

* cred_types: This parameter specifies a set of supported types of authentication credentials for EDHOC (see Section 3.5.2 of [RFC9528]). If the set is composed of a single type of authentication credential, this is encoded as an integer. Otherwise, the set is encoded as an array of integers, where each array element encodes one type of authentication credential. In JSON, the "cred_types" value is an integer or an array of integers. In CBOR, "cred_types" is an integer or an array of integers, and has label 6. The integer values are taken from the "EDHOC Authentication Credential Types" registry defined in [RFC9668].

* id_cred_types: This parameter specifies a set of supported types of authentication credential identifiers for EDHOC (see Section 3.5.3 of [RFC9528]). If the set is composed of a single type of authentication credential identifier, this is encoded as an integer or a text string. Otherwise, the set is encoded as an array, where each array element encodes one type of authentication credential identifier, as an integer or a text string. In JSON, the "id_cred_types" value is an integer, or a text string, or an array of integers and text strings. In CBOR, "id_cred_types" is an integer or a text string, or an array of integers and text strings, and has label 7. The integer or text string values are taken from the 'Label' column of the "COSE Header Parameters" registry [COSE.Header.Parameters].

* eads: This parameter specifies a set of supported EDHOC External Authorization Data (EAD) items, identified by their ead_label (see Section 3.8 of [RFC9528]). If the set is composed of a single ead_label, this is encoded as an unsigned integer. Otherwise, the set is encoded as an array of unsigned integers, where each array element encodes one ead_label. In JSON, the "eads" value is an unsigned integer or an array of unsigned integers. In CBOR, "eads" is an unsigned integer or an array of unsigned integers, and has label 8. The unsigned integer values are taken from the 'Label' column of the "EDHOC External Authorization Data" registry defined in [RFC9528].

* initiator: This parameter specifies whether the EDHOC Initiator role is supported. In JSON, the "initiator" value is a boolean. In CBOR, "initiator" is the simple value true (0xf5) or false (0xf4), and has label 9.

* responder: This parameter specifies whether the EDHOC Responder role is supported. In JSON, the "responder" value is a boolean. In CBOR, "responder" is the simple value true (0xf5) or false (0xf4), and has label 10.

* trust_anchors: This parameter specifies a collection of supported trust anchors for performing authentication. According to what is specified within the collection, these trust anchors are used for different purposes, e.g., for verifying authentication credentials of other EDHOC peers in EDHOC sessions.

  More in detail, the collection of trust anchors is composed of one or more sets. Each set includes one or more trust anchors to use for one specific purpose associated with that set.

In particular, each set is composed of pairs, each of which specifies a trust anchor type and an identifier of a trust anchor of that type. If the set is composed of a single pair, this pair is specified as a single item. If the set is composed of multiple pairs, these pairs are specified as elements of an array.

Trust anchor purposes are selected from the "EDHOC Trust Anchor Purposes" registry defined in Section 13.11 of this document. Trust anchor types are selected from the "EDHOC Trust Anchor Types" registry defined in Section 13.12 of this document.

In JSON, the "trust_anchors" value is an object with one or more outer entries, each of which is associated with a trust anchor purpose. The following applies for each outer entry:

- The outer entry's key specifies the associated trust anchor purpose taken from the 'Name' column' of the "EDHOC Trust Anchor Purposes" registry.

- The outer entry's value is an object or an array of at least two objects. Each object includes one inner entry, specifying the pair for a trust anchor TA of type TYPE. The inner entry is formatted as follows:

  - The inner entry's key specifies the TA's type TYPE taken from the 'Name' column of the "EDHOC Trust Anchor Types" registry.

  - The inner entry's value is the identifier of TA, whose encoding depends on TYPE. Such an encoding is what results from applying the conversion in Section 6.1 of [RFC8949] to the CBOR encoding of the identifier of TA when "trust_anchors" is encoded in CBOR (see below).

In CBOR, the "trust_anchors" value is a map and has label 11. The map includes one or more outer entries, each of which is associated with a trust anchor purpose. The following applies for each outer entry:

- The outer entry's key specifies the associated trust anchor purpose encoded as a CBOR integer, with integer value taken from the 'CBOR label' column of the "EDHOC Trust Anchor Purposes" registry.

- The outer entry's value is a map or an array of at least two maps. Each map includes one inner entry, specifying the pair for a trust anchor TA of type TYPE. The inner entry is formatted as follows:

- The inner entry’s key specifies the TA’s type TYPE encoded as a CBOR integer, with integer value taken from the ’CBOR label’ column of the "EDHOC Trust Anchor Types" registry.

- The inner entry’s value specifies the identifier of TA, whose encoding depends on TYPE and is specified by the ’Value type’ column of the "EDHOC Trust Anchor Types" registry, for the registry entry that has TYPE as value of the ’Name’ column.

An example of JSON EDHOC_Information is given in Figure 7.

```
"edhoc_info" : {
    "session_id"    : b64’AQ==’,
    "methods"       : 1,
    "cipher_suites" : 0,
    "trust_anchors" : {
      "edhoc_cred" : [
        { "c5u" : "coap://certs.c509.example" },
        { "x5u" : "coap://certs.x509.example" }
      ]
    }
}
```

Figure 7: Example of JSON EDHOC_Information

An example of CBOR EDHOC_Information is given in Figure 8.

```
e’edhoc_info_param’  : {
  e’session_id’    : h’01’,
  e’methods’       : 1,
  e’cipher_suites’ : 0,
  e’trust_anchors’ : {
    e’edhoc_cred’ : [
      { e’c5t_ta_type’ : [-15, h’81DC2F32CB87E163’] },
      { e’c5u_ta_type’ : "coap://certs.c509.example" },
      { e’x5t_ta_type’ : [-15, h’79F2A41B510C1F9B’] },
      { e’x5u_ta_type’ : "coap://certs.x509.example" }
    ]
  }
}
```

Figure 8: Example of CBOR EDHOC_Information

The CDDL grammar describing the CBOR EDHOC_Information is:

```
EDHOC_Information = {
  ?  0 => bstr,                                ; id
  ?  1 => int / [2* int],                      ; methods
  ?  2 => int / [2* int],                      ; cipher_suites
  ?  3 => true / false,                        ; message_4
  ?  4 => true / false,                        ; comb_req
  ?  5 => tstr,                                ; uri_path
  ?  6 => int / [2* int],                      ; cred_types
  ?  7 => int / tstr / [2* (int / tstr)],      ; id_cred_types
  ?  8 => uint / [2* uint],                    ; eads
  ?  9 => true / false,                        ; initiator
  ? 10 => true / false,                        ; responder
  ? 11 => trust_anchors_value,                 ; trust_anchors
  * (int / tstr) => any
}

trust_anchors_value = {
  1* int => trust_anchors_outer_entry_value
}

trust_anchors_outer_entry_value =
  trust_anchors_container / [2* trust_anchors_container]

trust_anchors_container = {
  int => trust_anchors_inner_entry_value
}

trust_anchors_inner_entry_value = any
```

4. Client-RS Communication

This section describes the exchange between C and RS, including the execution of the EDHOC protocol and the uploading of the access token from C to RS. The alternative workflow, where AS uploads the access token directly to RS, is described in [I-D.ietf-ace-workflow-and-params].

C and RS run the EDHOC protocol (see Section 4.2), and C uploads the access token in an EAD field (see Section 4.1) of an EDHOC message. Once successfully completed the EDHOC session, C and RS derive an OSCORE Security Context (see Section 4.5). After that, OSCORE is used for protecting communications when C accesses resources at RS, as per the access rights specified in the access token (see Section 4.9).

Detailed examples are given in Appendix A.

4.1. EAD items for Access Token and Session Identifier

This document defines EAD items (see Section 3.8 of [RFC9528]) for transporting an access token or a session identifier in EDHOC.

* EAD_ACCESS_TOKEN = (ead_label, ead_value), where:

  - ead_label is the integer value TBD registered in Section 13.8.

  - ead_value is a CBOR byte string equal to the value of the "access_token" field of the access token response from AS (see Section 3.3).

  This EAD item is critical, i.e., it is used only with the negative value of its ead_label, indicating that the receiving RS must progress the protocol using the received access token, or else abort the EDHOC session (see Section 3.8 of [RFC9528]). A client or resource server supporting the profile of ACE defined in this document MUST support this EAD item.

  EAD_ACCESS_TOKEN is used only when uploading the first access token of a token series, but not for the update of access rights (see Section 4.6).

  Example: Assuming IANA label 26 and critical, so ead_label = -26 (0x3819), and 180 bytes access_token = h'8343a1010aa2044c53...0f6a' (partly omitted for brevity):

  - EAD_ACCESS_TOKEN = 0x381958B48343a1010aa2044c53...0f6a

  Editor's note: Replace IANA label with TBD value registered for ACE-OAuth Access Token in Section 13.8.

* EAD_SESSION_ID = (ead_label, ead_value), where:

  - ead_label is the integer value TBD registered in Section 13.8.

  - ead_value is a CBOR byte string equal to the value of the "session_id" field within the EDHOC_Information object specified by AS in the "edhoc_info" parameter of the response from the /token endpoint, when issuing the first access token of a token series (see Section 3.3).

  This EAD item is critical, i.e., it is used only with the negative value of its ead_label, indicating that the receiving RS must progress the protocol using the access token associated with the identifier specified in "ead_value" and with the AUTH_CRED_C used in the EDHOC session, or else abort the EDHOC session (see

Section 3.8 of [RFC9528]). A client or resource server supporting the profile of ACE defined in this document MUST support this EAD item.

EAD_SESSION_ID is used only if the access token has been provisioned to RS and is valid, but there is a need to establish a (new) OSCORE Security Context between C and RS through EDHOC.

Example: Assuming IANA label 5 and critical, so ead_label = -5 (0x24), and session_id = h'1645'

- EAD_SESSION_ID = 0x24421645

Editor's note: Replace IANA label with TBD value for Session ID registered in Section 13.8.

4.2. EDHOC Session

In order to mutually authenticate and establish secure communication for authorized access according to the profile described in this document, C and RS run the EDHOC protocol augmented with an access token. During the EDHOC session, C specifies the access token to RS by value as conveyed in EAD item EAD_ACCESS_TOKEN, or by reference through a session identifier SESSION_ID conveyed in the EAD item EAD_SESSION_ID (see Section 4.1).

As per Appendix A.2 of [RFC9528], EDHOC can be transferred over CoAP using either the forward or the reverse message flow, thus manifesting the two possible mappings between the ACE roles (client and resource server) and the EDHOC roles (Initiator and Responder), whereas the CoAP roles (client and server) remain the same. The choice of message flow and corresponding mapping depends on the deployment setting and in particular on which identity to protect the most, since EDHOC protects the identity of the Initiator against active attackers.

In case the EDHOC forward message flow is used (see Section 4.3), C acts as EDHOC Initiator, and the access token MUST be specified by value or by reference in the EAD_3 field of EDHOC message_3. In case the EDHOC reverse message flow is used (see Section 4.4), C acts as EDHOC Responder, and the access token MUST be specified by value or by reference either in the EAD_2 field of EDHOC message_2 or in the EAD_4 field of EDHOC message_4. By doing so, the access token or the associated session identifier gets at least the same confidentiality protection by EDHOC as provided to the authentication credential used by C in the EDHOC session (see Section 9.1 of [RFC9528]).

When RS processes the EAD item EAD_ACCESS_TOKEN or EAD_SESSION_ID, RS MUST verify that the authentication credential AUTH_CRED_C that C specifies in the ID_CRED_X field during the EDHOC session is the same authentication credential correlated with the EAD item. If such a verification fails, RS MUST abort the EDHOC session. Note that:

* The ID_CRED_X field in question is the ID_CRED_I or ID_CRED_R field, when using the EDHOC forward or reverse message flow, respectively.

* If the processed EAD item is EAD_ACCESS_TOKEN, then the authentication credential correlated with the EAD item is specified in the 'cnf' claim of the access token conveyed in the EAD item.

* If the processed EAD item is EAD_SESSION_ID, then the authentication credential correlated with the EAD item is specified in the 'cnf' claim of an access token stored at the RS, which is associated with the authentication credential specified by ID_CRED_X and with the SESSION_ID conveyed in the EAD item.

RS MUST have successfully validated the access token before completing the EDHOC session. If completed successfully, then the EDHOC session is associated with both the access token and the pair (SESSION_ID, AUTH_CRED_C). If the EAD item used in the EDHOC session is EAD_ACCESS_TOKEN, then SESSION_ID is specified by the "session_id" field, within the EDHOC_Information object specified by the "cnf" claim of the access token.

Any previous EDHOC session associated with the same access token and with the same pair (SESSION_ID, AUTH_CRED_C) MUST be deleted. The OSCORE Security Context derived from that EDHOC session MUST also be deleted.

Depending on the message flow used, the EDHOC messages will be carried either in CoAP POST requests or in CoAP 2.04 (Changed) responses, as detailed in Appendix A.2 of [RFC9528].

C MUST target the EDHOC resource at RS with the URI path specified in the "uri_path" field (if present) of the EDHOC_Information object within the access token response received from AS, through which C obtained the first access token of the token series (see Section 3.1). If the "uri_path" field is not present in that EDHOC_Information object, C assumes the target resource at RS to be the well-known EDHOC resource at the path /.well-known/edhoc.

In this profile of ACE, RS MUST implement the CoAP Uri-Path-Abbrev Option specified in [I-D.ietf-core-uri-path-abbrev] and MUST understand the corresponding Uri-Path-Abbrev value 2 that abbreviates the path /.well-known/edhoc. While there is no equivalent requirement for C, the above ensures that the CoAP Uri-Path-Abbrev Option and its value are going to be understood by RS, if C includes the Option in a CoAP request that carries an EDHOC message and targets the well-known EDHOC resource at the path /.well-known/edhoc.

RS has to ensure that no requests can be performed on an EDHOC resource other than for running the EDHOC protocol. Specifically, it SHOULD NOT be possible to perform any other operation than POST on an EDHOC resource.

4.3. Forward Message Flow

This section details the case where the EDHOC forward message flow is used (see Appendix A.2.1 of [RFC9528]), i.e., where C acts as the Initiator I and RS acts as the Responder R.

Consistently with the EDHOC forward message flow, C sends EDHOC message_1 and EDHOC message_3 to an EDHOC resource at RS, as CoAP POST requests. RS sends EDHOC message_2 and (optionally) EDHOC message_4 as CoAP 2.04 (Changed) responses.

4.3.1. EDHOC message_1

The processing of EDHOC message_1 is specified in Section 5.2 of [RFC9528], with the following additions:

* The EDHOC method MUST be one of the EDHOC methods specified in the "methods" field (if present) of the EDHOC_Information object within the access token response received from AS, through which C obtained the first access token of the token series (see Section 3.1)

* The selected cipher suite MUST be an EDHOC cipher suite specified in the "cipher_suites" field (if present) of the EDHOC_Information object within the access token response received from AS, through which C obtained the first access token of the token series (see Section 3.1)

4.3.2. EDHOC message_2

The processing of EDHOC message_2 is specified in Section 5.3 of [RFC9528], with the following additions:

* The authentication credential CRED_R specified by the message field ID_CRED_R is AUTH_CRED_RS.

4.3.3. EDHOC message_3

The processing of EDHOC message_3 is specified in Section 5.4 of [RFC9528], with the following additions:

* The authentication credential CRED_I specified by the message field ID_CRED_I is AUTH_CRED_C.

* Exactly one of the EAD items EAD_ACCESS_TOKEN or EAD_SESSION_ID MUST be included in the EAD_3 field. If this is not the case, RS MUST abort the EDHOC session.

* If the EAD_3 field includes the EAD item EAD_ACCESS_TOKEN, then RS MUST ensure that the access token specified in the EAD item is valid. If the EAD_3 field includes the EAD item EAD_SESSION_ID, then RS MUST ensure that the access token associated with the session identifier SESSION_ID specified in the EAD item and with the AUTH_CRED_C used in the EDHOC session is valid.

  The validation follows the procedure specified in Section 4.6.2. If such validation fails, RS MUST reply to C with an EDHOC error message with ERR_CODE = 1 (see Section 6 of [RFC9528]) and it MUST abort the EDHOC session.

  Editor’s note: Instead of ERR_CODE = 1, consider using ERR_CODE = 3 "Access Denied" defined in draft-ietf-lake-authz

4.4. Reverse Message Flow

This section details the case where the EDHOC reverse message flow is used (see Appendix A.2.2 of [RFC9528]), i.e., where C acts as the Responder R and RS acts as the Initiator I.

Consistently with the EDHOC reverse message flow, C sends a trigger message, EDHOC message_2, and (optionally) EDHOC message_4 to RS as CoAP POST requests. RS sends EDHOC message_1 and EDHOC message_3 as CoAP 2.04 (Changed) responses.

In this profile of ACE, if RS implements the EDHOC reverse message flow, then RS MUST implement EDHOC message_4.

Exactly one of the EAD items EAD_ACCESS_TOKEN or EAD_SESSION_ID MUST be included in either the EAD_2 field of EDHOC message_2 or the EAD_4 field of EDHOC message_4. If this is not the case, RS MUST abort the EDHOC session.

Specific instructions for the different messages are provided in the following subsections.

4.4.1. Trigger Message

As specified in Appendix A.2.2 of [RFC9528], the trigger message is an empty POST request that C sends to the EDHOC resource at RS, as intended to trigger a response conveying EDHOC message_1.

In case the access token is issued for a group-audience (see Section 6.9 of [RFC9200]), then C can perform an EDHOC "roll call", by sending the trigger message as a group request over IP multicast [I-D.ietf-core-groupcomm-bis]. For the sake of efficiency, it is expected that the group-audience is appropriately associated with a CoAP group and/or application group (see Section 2 of [I-D.ietf-core-groupcomm-bis]), so that only the RSs belonging to the group-audience receive the trigger message. After that, C can receive a different EDHOC message_1 from each of the targeted RSs and separately progresses the corresponding EDHOC sessions, by sending a different EDHOC message_2 to each RS that has replied with an EDHOC message_1.

4.4.2. EDHOC message_1

The processing of EDHOC message_1 is specified in Section 5.2 of [RFC9528].

4.4.3. EDHOC message_2

The processing of EDHOC message_2 is specified in Section 5.3 of [RFC9528], with the following additions:

* The authentication credential CRED_R specified by the message field ID_CRED_R is AUTH_CRED_C.

* If the EAD_2 field includes the EAD item EAD_ACCESS_TOKEN, then RS MUST ensure that the access token specified in the EAD item is valid. If the EAD_2 field includes the EAD item EAD_SESSION_ID, then RS MUST ensure that the access token associated with the session identifier SESSION_ID specified in the EAD item and with the AUTH_CRED_C used in the EDHOC session is valid.

Note that in this case C uploads the Access Token or session ID before RS is authenticated, since C will not learn about the identity of RS until having verified message_3. In particular, in the case of a group-audience, when there may be multiple legitimate RS, C does not yet know which member of the group-audience it communicates with (if any).

The validation follows the procedure specified in Section 4.6.2. If such validation fails, RS MUST reply to C with an EDHOC error message with ERR_CODE = 1 (see Section 6 of [RFC9528]) and it MUST abort the EDHOC session.

Editor’s note: Instead of ERR_CODE = 1, consider using ERR_CODE = 3 "Access Denied" defined in draft-ietf-lake-authz

4.4.4. EDHOC message_3

The processing of EDHOC message_3 is specified in Section 5.4 of [RFC9528], with the following additions:

* The authentication credential CRED_I specified by the message field ID_CRED_I is AUTH_CRED_RS.

4.4.5. EDHOC message_4

The processing of EDHOC message_4 is specified in Section 5.5 of [RFC9528], with the following additions:

* If the EAD_4 field includes the EAD item EAD_ACCESS_TOKEN, then RS MUST ensure that the access token specified in the EAD item is valid. If the EAD_4 field includes the EAD item EAD_SESSION_ID, then RS MUST ensure that the access token associated with the session identifier SESSION_ID specified in the EAD item and with the AUTH_CRED_C used in the EDHOC session is valid.

  The validation follows the procedure specified in Section 4.6.2. If such validation fails, RS MUST reply to C with an EDHOC error message with ERR_CODE = 1 (see Section 6 of [RFC9528]) and it MUST abort the EDHOC session.

  Editor’s note: Instead of ERR_CODE = 1, consider using ERR_CODE = 3 "Access Denied" defined in draft-ietf-lake-authz

4.5. OSCORE Security Context

Once successfully completed the EDHOC session, C and RS derive an OSCORE Security Context as defined in Appendix A.1 of [RFC9528].

In addition, RS associates the latest EDHOC session and the derived OSCORE Security Context with the stored access token, which is bound to the authentication credential AUTH_CRED_C used in the EDHOC session. The access token is also associated with the pair (SESSION_ID, AUTH_CRED_C), where SESSION_ID is the identifier of the token series to which the access token belongs.

If supported by C, C MAY use the EDHOC + OSCORE combined request defined in [RFC9668], unless the EDHOC_Information object specified by the "edhoc_info" parameter of the access token response included the "comb_req" field encoding the CBOR simple value false (0xf4).

In the combined request, both EDHOC message_3 and the first OSCORE-protected application request are combined together in a single OSCORE-protected CoAP request, thus saving one round trip. This requires C to derive the OSCORE Security Context with RS already after having successfully processed the received EDHOC message_2 and before sending EDHOC message_3. An example is provided in Appendix A.2.

4.6. Update of Access Rights

If C has a valid OSCORE Security Context associated with a valid access token at RS, then C can request from AS an update of the access rights as described in Section 3.1.

If the request is granted, then AS generates a new access token containing updated access rights for C (see Section 3.3.3), in the same token series of the current access token (see Section 3.2).

According to this document, AS provides the new access token to C (see Section 3.3) for further uploading to RS. Alternatively, the new access token can be uploaded by AS directly to RS, as described in [I-D.ietf-ace-workflow-and-params].

If all validations are successful, C can access protected resources at RS according to the updated access rights, using the previously established OSCORE Security Context.

The rest of this section describes the message exchange for the uploading of the new access token from C to RS.

4.6.1. C-to-RS: POST to /authz-info endpoint

C can update its access rights by uploading the updated access token to RS using CoAP [RFC7252] and the Authorization Information endpoint as described in Section 5.10.1 of [RFC9200].

That is, C sends a POST request to the /authz-info endpoint at RS, with the request payload containing the access token without any CBOR wrapping. As per Section 5.10.1 of [RFC9200], the Content-Format of the POST request MUST be "application/cwt" to reflect the format of the transported access token.

C MUST protect the POST request using the current OSCORE Security Context shared with RS.

Upon receiving an access token from C, RS MUST follow the procedures defined in Section 5.10.1 of [RFC9200]. That is, RS MUST verify the validity of the access token. RS MAY make an introspection request (see Section 5.9.1 of [RFC9200]) to validate the access token at AS.

RS MUST check the following conditions:

* RS checks whether it stores an access token T_OLD, such that the "session_id" field of the EDHOC_Information object specified by the "cnf" claim matches the "session_id" field of the EDHOC_Information object specified by the "cnf" claim of the new access token T_NEW.

* RS checks whether the OSCORE Security Context CTX used to protect the request matches the OSCORE Security Context associated with the stored access token T_OLD.

If both the conditions above hold, RS MUST supersede the old access token T_OLD by replacing the corresponding authorization information with the one specified in the new access token T_NEW, and MUST associate T_NEW with the OSCORE Security Context CTX.

Note that C and RS do not execute the EDHOC protocol, they do not establish a new OSCORE Security Context, and AUTH_CRED_C remains the same.

4.6.2. RS-to-C: 2.01 (Created)

If all validations are successful, RS stores the new access token in such a way that it is possible to retrieve it based on the pair (SESSION_ID, AUTH_CRED_C), where SESSION_ID is the identifier of the token series to which the access token belongs. Note that SESSION_ID is specified in the "session_id" field of the EDHOC_Information object, within the "cnf" claim of the access token.

Then, RS MUST reply to the POST request by sending a 2.01 (Created) response with no payload. The response is protected with the same OSCORE Security Context used to protect the corresponding request. After that, C can access protected resources at RS according to the updated access rights, using the previously established OSCORE Security Context.

Instead, if any validation fails, RS MUST respond with a 4.01 (Unauthorized) error response. RS MAY provide additional information in the payload of the error response, in order to clarify what went wrong.

As specified in Section 5.10.1 of [RFC9200], when receiving a valid access token with updated authorization information from C (see Section 4.6.1), it is recommended that RS overwrites the previous access token. That is, only the latest authorization information in the access token received by RS is valid. This simplifies the process needed by RS to keep track of authorization information for a given client.

4.7. Discarding the OSCORE Security Context

There are a number of cases where C or RS have to discard the OSCORE Security Context that they share, and may establish a new one (see Section 4.8).

C MUST discard the current OSCORE Security Context shared with RS when any of the following occurs.

* The OSCORE Sender Sequence Number space of C is exhausted.

* The access token associated with the OSCORE Security Context becomes invalid, for example, due to expiration or revocation.

* C receives a number of unprotected 4.01 (Unauthorized) responses to OSCORE-protected requests, which are sent to RS and protected using the same OSCORE Security Context. The exact number of such received responses needs to be specified by the application. This can happen, for example, due to lack of storage at RS, which then sends the "AS Request Creation Hints" message (see Section 5.3 of [RFC9200]).

* The authentication credential of C (of RS) becomes invalid, e.g., due to expiration or revocation, and it was used as AUTH_CRED_C (AUTH_CRED_RS) in the EDHOC session to establish the OSCORE Security Context.

RS MUST discard the current OSCORE Security Context shared with C when any of the following occurs:

* The OSCORE Sender Sequence Number space of RS is exhausted.

* The access token associated with the OSCORE Security Context becomes invalid, for example, due to expiration or revocation.

* The authentication credential of C (of RS) becomes invalid (e.g., due to expiration or revocation), and it was used as AUTH_CRED_C (AUTH_CRED_RS) in the EDHOC session to establish the OSCORE Security Context.

After a new access token is successfully uploaded to RS and a new OSCORE Security Context is established between C and RS, messages still in transit that were protected with the previous OSCORE Security Context might not be successfully verified by the recipient, since the old OSCORE Security Context might have been discarded. This means that messages sent shortly before C has uploaded the new access token to RS might not be successfully accepted by the recipient.

Furthermore, implementations may want to cancel CoAP observations at RS, if registered before the new OSCORE Security Context has been established. Alternatively, applications need to implement a mechanism to ensure that, from then on, messages exchanged within those observations are going to be protected with the newly derived OSCORE Security Context.

4.8. Establishing a New OSCORE Security Context

The procedure for provisioning a new access token to RS specified in this section applies to various cases when an OSCORE Security Context shared between C and RS has been deleted, for example as described in Section 4.7.

Another exceptional case is when there is still a valid OSCORE Security Context but it needs to be updated, e.g., due to a policy limiting its use in terms of time or amount of processed data, or to the imminent exhaustion of the OSCORE Sender Sequence Number space. In this case, C and RS MAY alternatively attempt to run a key update protocol that they both support. One lightweight example is KUDOS [I-D.ietf-core-oscore-key-update], which is independent of ACE and EDHOC and does not require the uploading of an access token. If C and RS does not support a common key update protocol to use for updating their still valid OSCORE Security Context, then C and RS fall back to EDHOC as outlined above.

In either case, C and RS establish a new OSCORE Security Context that replaces the old one and will be used for protecting their communications from then on. In particular, RS MUST associate the new OSCORE Security Context with the current (potentially re-uploaded) access token. Furthermore, the SESSION_ID identifying the token series to which the access token belongs remains unchanged, even if C and RS have established a new EDHOC session. Unless C and RS re-run the EDHOC protocol, they preserve their OSCORE identifiers, i.e., their OSCORE Sender/Recipient IDs.

4.9. Access Rights Verification

RS MUST follow the procedures defined in Section 5.10.2 of [RFC9200]. That is, if RS receives an OSCORE-protected request targeting a protected resource from C, then RS processes the request according to [RFC8613].

If OSCORE verification succeeds and the target resource requires authorization, RS retrieves the authorization information using the access token associated with the OSCORE Security Context. Then, RS MUST verify that the authorization information covers the target resource and the action intended by C on it.

4.10. Access Token Invalidity

When an access token becomes invalid (e.g., due to its expiration or revocation), RS MUST delete the access token and the associated OSCORE Security Context, and MUST notify C with an error response with code 4.01 (Unauthorized) for any long running request, as specified in Section 5.8.3 of [RFC9200].

4.11. Authentication Credential Invalidity

If an authentication credential AUTH_CRED_C of C is invalidated (e.g., it expires), then the following applies:

* RS MUST delete all the stored access tokens that specify AUTH_CRED_C in the "cnf" claim.

* C MUST delete every stored access token such that C obtained the first access token of the same series through the response to an access token request specifying AUTH_CRED_C, e.g., in the "req_cnf" parameter (see Section 3.1).

* RS and C MUST abort and purge all the EDHOC sessions that used AUTH_CRED_C and successfully completed, as well as the OSCORE Security Context derived from those sessions (see Section 4.7).

If an authentication credential AUTH_CRED_RS of RS is invalidated (e.g., it expires), then the following applies:

* C MUST delete every stored access token such that C obtained the first access token of the same series through an access token response specifying AUTH_CRED_RS, e.g., in the 'rs_cnf' parameter (see Section 3.3).

* C MUST delete every stored access token that C specified (by value or by reference) during an EDHOC session that used AUTH_CRED_RS and successfully completed.

* RS and C MUST abort and purge all the EDHOC sessions that used AUTH_CRED_RS and successfully completed, as well as the OSCORE Security Context derived from those sessions (see Section 4.7).

4.12. EDHOC Session Invalidity

If an EDHOC session is aborted and purged for other reasons than those in Section 4.11, then RS and C that established the session MUST delete the OSCORE Security Context derived from that session (see Section 4.7).

4.13. Using AS Request Creation Hints

When replying to an unauthorized resource request message from a client, RS can send an unprotected AS Request Creation Hints message as a 4.01 (Unauthorized) error response (see Section 5.3 of [RFC9200]).

The message payload can specify a number of parameters that help the sender client acquire a valid access token from AS. These parameters include "audience" and "scope".

When using this profile and running EDHOC per its reverse message flow (see Section 4.4), RS acts as EDHOC Initiator. A compelling reason to do so is the wish to protect the identity of RS against active attackers, consistently with the EDHOC security properties.

However, the identity protection achieved through EDHOC can be defeated if RS exposes information such as audience and scope, when specifying the corresponding parameters in an unprotected AS Request Creation Hints message.

Therefore, if RS supports the EDHOC reverse message flow and sends an AS Request Creation Hints, the following applies:

* The message payload MUST NOT include the "audience" parameter.

* The message payload SHOULD NOT include the "scope" parameter, unless its value cannot contribute to expose the identity of RS.

AS Request Creation Hints may also be requested and retrieved through a new EAD item defined here, see Figure 9, Section 13.8, and an example of its usage in Appendix A.3.

```
ead_request_creation_hints = (
  ead_label : e’ead_request_creation_hints_label’,
  ? ead_value : bstr .cbor AS_request_creation_hints
)
AS_request_creation_hints = map
```

Figure 9: EAD item EAD_REQUEST_CREATION_HINTS.

The AS_request_creation_hints is a CBOR map with keys defined in the IANA registry "ACE Authorization Server Request Creation Hints".

Example: Assuming IANA label 2 and non-critical, so ead_label = 2 (0x02), and the AS_request_creation_hints map containing one CBOR text string "coap://www.example.com/token" with key 1 (the absolute URI of the /token endpoint at the AS):

* EAD_REQUEST_CREATION_HINTS = 0x025820A101781C636F61703A2F2F7777772E6578616D706C652E636F6D2F746F6B656E

Editor’s note: Replace IANA label with TBD value registered for EAD_REQUEST_CREATION_HINTS in Section 13.8.

This EAD item is intended to be used in EAD fields of EDHOC messages exchanged between C and RS: in the forward message flow in EAD_1 and EAD_2, and in the reverse message flow in EAD_2 and EAD_3. In the first EDHOC message from C to RS, an EAD item with ead_label = TBD with no ead_value asks the RS to include in the next EDHOC message the same EAD item with ead_value encoding the AS_request_creation_hints map. This EAD item is non-critical, i.e., it can be ignored by the receiving peer. It is OPTIONAL to implement.

Since C has not made an actual request targeting a specific application resource, the RS may not know what resource C is interested in accessing. Moreover, such information needs to be matched against the privacy policy of the application. Since EDHOC message_2 is only protected against passive attackers, the AS_request_creation_hints map SHOULD NOT include "audience" and "scope" when present in the EAD item conveyed in the EAD_2 field.

4.14. Requesting Authentication Credential By Value

EDHOC peers need access to each other's authentication credentials to complete the protocol. However, to reduce unnecessary overhead, the EDHOC protocol enables credential references to be sent instead of authentication credentials. The ACE protocol includes the initial transport of authentication credentials of C and RS and thus matches the use of credential references in EDHOC.

However, if one of the parties has deleted the other party's authentication credential from its local storage, then there should be a way to restore it without requesting a new access token.

Consider first the EDHOC forward message flow. If the ACE Client / EDHOC Initiator sends a credential by reference in message_3, then Responder may return error code 3, Unknown credential referenced. This enables the Initiator to restart the protocol using some other ID_CRED, typically the authentication credential by value thereby resolving the issue. However, in case the ACE Resource Server / EDHOC Responder sends a credential by reference in message_2, then returning a code 3 EDHOC error message does not automatically solve the problem. Having aborted the protocol, the Responder has no reliable way to act differently in a following EDHOC session since it never authenticated the Initiator.

In order to remediate this situation, this section specifies a new EAD item for requesting the peer's authentication credential by value, see Figure 10, Section 13.8, and an example of its usage in Appendix A.3.

```
ead_credential_by_value = (
  ead_label : e'ead_credential_by_value_label'
)
```

Figure 10: EAD item EAD_CRED_BY_VALUE.

This EAD item has no ead_value. When present in EAD_1, it requests the Responder's authentication credential by value in ID_CRED_R of message_2. When present in EAD_2, it requests the Initiator's authentication credential by value in ID_CRED_I of message_3. The EAD item is non-critical, i.e., it can be ignored by the receiving peer. It is OPTIONAL to implement.

Example: Assuming IANA label 15 and non-critical, so ead_label = 15 (0x0F), and considering that this EAD item has no ead_value:

* EAD_CRED_BY_VALUE = 0x0F

Editor's note: Replace IANA label with TBD value registered for EAD_CRED_BY_VALUE in Section 13.8.

In the EDHOC reverse message flow this EAD item can be applied for better control of the use of credential by value. Note that in the reverse flow both C and RS may recover from error code 3, but at the cost of more round trips which can be avoided by using the EAD item.

* In case the ACE Client / EDHOC Responder sends a credential by reference in message_2 and receives a code 3 error message, then it can trigger a new EDHOC session and send credential by value this time.

* In case the ACE Resource Server / EDHOC Initiator sends a credential by reference in message_3 and receives a code 3 error message, then since the RS has authenticated C, it can store the authentication credential of C, and in the next session with C, the RS can send its credential by value.

5. Secure Communication with AS

As specified in the ACE framework (see Sections 5.8 and 5.9 of [RFC9200]), the requesting entity (RS and/or C) and AS communicate via the /token or /introspect endpoint. When using this profile, the use of CoAP [RFC7252] and OSCORE [RFC8613] for this communication is RECOMMENDED. Other protocols fulfilling the security requirements defined in Section 5 of [RFC9200] (such as HTTP and DTLS [RFC9147] or TLS [RFC8446]) MAY be used instead.

If OSCORE is used, the requesting entity and AS need to have an OSCORE Security Context in place. While this can be pre-installed, the requesting entity and AS can establish such an OSCORE Security Context, for example, by running the EDHOC protocol, as shown between C and AS by the examples in Appendix A.1 and Appendix A.2. This also applies for communication between RS and AS, for example to protect the upload of access tokens from AS directly to RS as described in [I-D.ietf-ace-workflow-and-params].

6. Use with Application Profiles of ACE for Group Key Provisioning

The EDHOC and OSCORE profile specified in the present document can be used in combination with application profiles of ACE for key provisioning for group communication [RFC9594], such as [I-D.ietf-ace-key-groupcomm-oscore] and [I-D.ietf-ace-coap-pubsub-profile].

When doing so, the EDHOC and OSCORE profile is used by a client and an RS that acts as Key Distribution Center (KDC) responsible for a security group. According to what is defined in the present document, the client does not upload the first access token of a token series to the /authz-info endpoint at RS over an unprotected channel. Consequently, the client does not obtain the N_S challenge from RS (see Section 3.3 of [RFC9594]).

Therefore, the following holds for computing the N_S challenge used in the two application profiles of [RFC9594] mentioned above.

* In the ACE application profile specified in [I-D.ietf-ace-key-groupcomm-oscore], when the provisioning of the access token to the Group Manager (i.e., the KDC) has relied on the EDHOC and OSCORE profile of ACE specified in the present document and the access token was specified in the EAD item ACE-OAuth Access Token (see Section 4.1), then N_S is derived using the EDHOC_Exporter interface defined in Section 4.2.1 of [RFC9528].

  Specifically, N_S is exported from the EDHOC session associated with the access token and established between the client and the Group Manager. The following is provided as input to the EDHOC_Exporter interface:

  - The 'exporter_label' parameter is the unsigned integer EXPORTER_LABEL_TBD_26, which is registered in Section 13.9 of the present document.

  - The 'context' parameter is h'' (0x40), i.e., the empty CBOR byte string.

  - The 'length' parameter is 32, i.e., the intended length of N_S in bytes.

* In the ACE application profile specified in [I-D.ietf-ace-coap-pubsub-profile], when the provisioning of the access token to the KDC has relied on the EDHOC and OSCORE profile of ACE specified in the present document and the access token was specified in the EAD item ACE-OAuth Access Token (see Section 4.1), then N_S is derived using the EDHOC_Exporter interface defined in Section 4.2.1 of [RFC9528].

  Specifically, N_S is exported from the EDHOC session associated with the access token and established between the client and the KDC. The following is provided as input to the EDHOC_Exporter interface:

- The 'exporter_label' parameter is the unsigned integer EXPORTER_LABEL_TBD_27, which is registered in Section 13.9 of the present document.

- The 'context' parameter is h'' (0x40), i.e., the empty CBOR byte string.

- The 'length' parameter is 32, i.e., the intended length of N_S in bytes.

7. CWT Confirmation Methods

This document defines a number of new CWT confirmation methods, which are registered in Section 13.7. The semantics of each confirmation method is defined below.

7.1. Ordered Chain of X.509 Certificates

The confirmation method "x5chain" specifies an ordered array of X.509 certificates [RFC5280]. The semantics of "x5chain" is like that of the "x5chain" COSE Header Parameter specified in [RFC9360].

7.2. Unordered Bag of X.509 Certificates

The confirmation method "x5bag" specifies a bag of X.509 certificates [RFC5280]. The semantics of "x5bag" is like that of the "x5bag" COSE Header Parameter specified in [RFC9360].

7.3. Hash of an X.509 Certificate

The confirmation method "x5t" specifies the hash value of the end-entity X.509 certificate [RFC5280]. The semantics of "x5t" is like that of the "x5t" COSE Header Parameter specified in [RFC9360].

7.4. URI Pointing to an Ordered Chain of X.509 Certificates

The confirmation method "x5u" specifies the URI [RFC3986] of an ordered chain of X.509 certificates [RFC5280]. The semantics of "x5u" is like that of the "x5u" COSE Header Parameter specified in [RFC9360].

7.5. Ordered Chain of C509 Certificates

The confirmation method "c5c" specifies an ordered array of C509 certificates [I-D.ietf-cose-cbor-encoded-cert]. The semantics of "c5c" is like that of the "c5c" COSE Header Parameter specified in [I-D.ietf-cose-cbor-encoded-cert].

7.6. Unordered Bag of C509 Certificates

The confirmation method "c5b" specifies a bag of C509 certificates [I-D.ietf-cose-cbor-encoded-cert]. The semantics of "c5b" is like that of the "c5b" COSE Header Parameter specified in [I-D.ietf-cose-cbor-encoded-cert].

7.7. Hash of a C509 Certificate

The confirmation method "c5t" specifies the hash value of the end-entity C509 certificate [I-D.ietf-cose-cbor-encoded-cert]. The semantics of "c5t" is like that of the "c5t" COSE Header Parameter specified in [I-D.ietf-cose-cbor-encoded-cert].

7.8. URI Pointing to an Ordered Chain of C509 Certificates

The confirmation method "c5u" specifies the URI [RFC3986] of a COSE_C509 containing an ordered chain of C509 certificates [I-D.ietf-cose-cbor-encoded-cert]. COSE_C509 is defined in [I-D.ietf-cose-cbor-encoded-cert]. The semantics of "c5u" is like that of the "c5u" COSE Header Parameter specified in [I-D.ietf-cose-cbor-encoded-cert].

7.9. CWT Containing a COSE_Key

The confirmation method "kcwt" specifies a CBOR Web Token (CWT) [RFC8392] containing a COSE_Key [RFC9053] in a ’cnf’ claim and possibly other claims. The semantics of "kcwt" is like that of the "kcwt" COSE Header Parameter specified in [RFC9528].

7.10. CCS Containing a COSE_Key

The confirmation method "kccs" specifies a CWT Claims Set (CCS) [RFC8392] containing a COSE_Key [RFC9053] in a ’cnf’ claim and possibly other claims. The semantics of "kccs" is like that of the "kccs" COSE Header Parameter specified in [RFC9528].

8. JWT Confirmation Methods

This document defines a number of new JWT confirmation methods, which are registered in Section 13.6. The semantics of each confirmation method is defined below.

8.1. Ordered Chain of X.509 Certificates

The confirmation method "x5c" specifies an ordered array of X.509 certificates [RFC5280]. The semantics of "x5c" is like that of the "x5c" JSON Web Signature and Encryption Header Parameter specified in [RFC7515], with the following difference. The public key contained in the first certificate is the proof-of-possession key and does not have to correspond to a key used to digitally sign the JWS.

8.2. Unordered Bag of X.509 Certificates

The confirmation method "x5b" specifies a bag of X.509 certificates [RFC5280]. The semantics of the "x5b" is like that of the "x5c" JWT confirmation method defined in Section 8.1, with the following differences. First, the set of certificates is unordered and may contain self-signed certificates. Second, the composition and processing of "x5b" are like for the "x5bag" COSE Header Parameter defined in [RFC9360].

8.3. Hash of an X.509 Certificate

The confirmation method "x5t" specifies the hash value of the end-entity X.509 certificate [RFC5280]. The semantics of "x5t" is like that of the "x5t" JSON Web Signature and Encryption Header Parameter specified in [RFC7515].

8.4. URI Pointing to an Ordered Chain of X.509 Certificates

The confirmation method "x5u" specifies the URI [RFC3986] of an ordered chain of X.509 certificates [RFC5280]. The semantics of "x5u" is like that of the "x5u" COSE Header Parameter specified in [RFC9360], with the following difference. The public key contained in the first certificate is the proof-of-possession key and does not have to correspond to a key used to digitally sign the JWS.

8.5. Ordered Chain of C509 Certificates

The confirmation method "c5c" specifies an ordered array of C509 certificates [I-D.ietf-cose-cbor-encoded-cert]. The semantics of "c5c" is like that of the "x5c" JWT confirmation method defined in Section 8.1, with the following difference. Each string in the JSON array is a base64-encoded (Section 4 of [RFC4648] - not base64url-encoded) C509 certificate.

8.6. Unordered Bag of C509 Certificates

The confirmation method "c5b" specifies a bag of C509 certificates [I-D.ietf-cose-cbor-encoded-cert]. The semantics of "c5b" is like that of the "c5c" JWT confirmation method defined in Section 8.5, with the following differences. First, the set of certificates is unordered and may contain self-signed certificates. Second, the composition and processing of "c5b" is like for the "c5b" COSE Header Parameter defined in [I-D.ietf-cose-cbor-encoded-cert].

8.7. Hash of a C09 Certificate

The confirmation method "c5t" specifies the hash value of the end-entity C509 certificate [I-D.ietf-cose-cbor-encoded-cert]. The semantics of "c5t" is like that of the "x5t" JWT confirmation method defined in Section 8.3, with the following differences. First, the base64url-encoded SHA-1 thumbprint is computed over the C509 certificate. Second, the public key contained in the C509 certificate does not have to correspond to a key used to digitally sign the JWS.

8.8. URI Pointing to an Ordered Chain of C509 Certificates

The confirmation method "c5u" specifies the URI [RFC3986] of COSE_C509 containing an ordered chain of C509 certificates [I-D.ietf-cose-cbor-encoded-cert]. COSE_C509 is defined in [I-D.ietf-cose-cbor-encoded-cert]. The semantics of "c5u" is like that of the "x5u" JWT confirmation method defined in Section 8.4, with the following differences. First, the URI refers to a resource for the C509 certificate chain. Second, the public key contained in one of the C509 certificates and acting as proof-of-possession key does not have to correspond to a key used to digitally sign the JWS.

8.9. CWT Containing a COSE_Key

The confirmation method "kcwt" specifies a CBOR Web Token (CWT) [RFC8392] containing a COSE_Key [RFC9053] in a ’cnf’ claim and possibly other claims. The format of "kcwt" is the base64url-encoded serialization of the CWT.

8.10. CCS Containing a COSE_Key

The confirmation method "kccs" specifies a CWT Claims Set (CCS) [RFC8392] containing a COSE_Key [RFC9053] in a ’cnf’ claim and possibly other claims. The format of "kcwt" is the base64url-encoded serialization of the CWT.

9. EDHOC Trust Anchor Purposes

This document defines the following EDHOC trust anchor purpose.

Note to RFC Editor: Please replace all occurrences of "[RFC-XXXX]" with the RFC number of this specification and delete this paragraph.

Name	CBOR label	Description	Reference
edhoc_cred	0	Verifying authentication credentials of other EDHOC peers in EDHOC sessions	[RFC-XXXX][RFC9528]

Table 2: EDHOC Trust Anchor Purposes

Trust anchors with purpose "edhoc_cred" are used for verifying authentication credentials of other EDHOC peers in an EDHOC session, and they typically are authentication credentials of Certificate Authorities (CAs).

10. EDHOC Trust Anchor Types

This document defines the following EDHOC trust anchor types.

Note to RFC Editor: Please replace all occurrences of "[RFC-XXXX]" with the RFC number of this specification and delete this paragraph.

Name	CBOR label	Value type	Description	Reference
uuid	0	#6.37(bstr)	Binary CBOR-encoded UUID	[RFC-XXXX][RFC9562]
kid	4	bstr	Key identifier	[RFC-XXXX][RFC9052]
c5t	22	COSE_CertHash	Hash of a C509 certificate	[RFC-XXXX][draft-ietf-cose-cbor-encoded-cert]
c5u	23	uri	URI pointing to a COSE_C509 containing a ordered chain of certificates	[RFC-XXXX][draft-ietf-cose-cbor-encoded-cert]
x5t	34	COSE_CertHash	Hash of an X.509 certificate	[RFC-XXXX][RFC9360]
x5u	35	uri	URI pointing to an X.509 certificate	[RFC-XXXX][RFC9360]

Table 3: EDHOC Trust Anchor Types

11. Security Considerations

This document specifies a profile for the Authentication and Authorization for Constrained Environments (ACE) framework [RFC9200]. Thus, the general security considerations from the ACE framework also apply to this profile.

Furthermore, the security considerations from OSCORE [RFC8613] and from EDHOC [RFC9528] also apply to this specific use of the OSCORE and EDHOC protocols.

As previously stated, once completed the EDHOC session, C and RS are mutually authenticated through their respective authentication credentials, whose retrieval has been facilitated by AS. Also, once completed the EDHOC session, C and RS have established a long-term secret key PRK_out enjoying forward secrecy. This is in turn used by C and RS to establish an OSCORE Security Context.

Furthermore, RS achieves confirmation that C has PRK_out (proof of possession) when completing the EDHOC session. Instead, C achieves confirmation that RS has PRK_out (proof of possession) either when receiving the optional EDHOC message_4 from RS, or when successfully verifying a response from RS protected with the established OSCORE Security Context.

OSCORE is designed to secure point-to-point communication, providing a secure binding between a request and the corresponding response(s). Thus, the basic OSCORE protocol is not intended for use in point-to-multipoint communication (e.g., enforced via multicast or a publish-subscribe model). Implementers of this profile should make sure that their use case of OSCORE corresponds to the expected one, in order to prevent weakening the security assurances provided by OSCORE.

When using this profile, it is RECOMMENDED that RS stores only one access token per client. The use of multiple access tokens for a single client increases the strain on RS, since it must consider every access token associated with the client and calculate the actual permissions that client has. Also, access tokens indicating different or disjoint permissions from each other may lead RS to enforce wrong permissions. If one of the access tokens expires earlier than others, the resulting permissions may offer insufficient protection. Developers SHOULD avoid using multiple access tokens for the same client. Furthermore, RS MUST NOT store more than one access token per client per PoP-key (i.e., per client’s authentication credential).

This profile defines the confirmation methods "kcwt" and "kccs" corresponding to the use of CBOR Web Tokens (CWTs) and CWT Claims Set (CCSs), respectively. Security considerations of CWTs and CCSs, and of COSE header parameters "kcwt" and "kccs" are given in Section 9.8 of [RFC9528], and apply also to confirmation methods. In particular, the contents of the CWT or CCS must be processed as untrusted input. The application needs to define a trust-establishment mechanism and identify the relevant trust anchors.

12. Privacy Considerations

This document specifies a profile for the Authentication and Authorization for Constrained Environments (ACE) framework [RFC9200]. Thus, the general privacy considerations from the ACE framework also apply to this profile.

Furthermore, the privacy considerations from OSCORE [RFC8613] and from EDHOC [RFC9528] also apply to this specific use of the OSCORE and EDHOC protocols.

An unprotected response to an unauthorized request may disclose information about RS and/or its existing relationship with C. It is advisable to include as little information as possible in an unencrypted response (see also Section 4.13). When an OSCORE Security Context already exists between C and RS, more detailed information may be included.

The (encrypted) access token is never sent in an unprotected POST request to the /authz-info endpoint at RS. Thus, even if C uses the same single access token from multiple locations, the access token's value does not contribute to the risk of C being tracked.

The identifiers used in OSCORE, i.e., the OSCORE Sender/Recipient IDs, are negotiated by C and RS during the EDHOC session. When using the EDHOC forward (reverse) message flow:

* The EDHOC Connection Identifier C_I (C_R) of C is going to be the OSCORE Recipient ID of C, i.e., the OSCORE Sender ID of RS.

* The EDHOC Connection Identifier C_R (C_I) of RS is going to be the OSCORE Recipient ID of RS, i.e., the OSCORE Sender ID of C.

These OSCORE identifiers are privacy sensitive (see Section 12.8 of [RFC8613]). In particular, they could reveal information about C, or may be used for correlating different requests from C, e.g., across different networks that C has joined and left over time. This can be mitigated if C and RS dynamically update their OSCORE identifiers, e.g., by using the method defined in [I-D.ietf-core-oscore-id-update].

13. IANA Considerations

This document has the following actions for IANA.

Note to RFC Editor: Please replace all occurrences of "[RFC-XXXX]" with the RFC number of this specification and delete this paragraph.

13.1. ACE Profiles Registry

IANA is asked to add the following entry to the "ACE Profiles" registry, following the procedure specified in [RFC9200].

* Name: coap_edhoc_oscore
* Description: Profile for delegating client authentication and authorization in a constrained environment by establishing an OSCORE Security Context [RFC8613] between resource-constrained nodes, through the execution of the lightweight authenticated key exchange protocol EDHOC [RFC9528].
* CBOR Value: TBD (value between 1 and 23)
* Reference: [RFC-XXXX]

13.2. OAuth Parameters Registry

IANA is asked to add the following entry to the "OAuth Parameters" registry.

* Name: edhoc_info
* Parameter Usage Location: token request and token response
* Change Controller: IETF
* Reference: [RFC-XXXX]

13.3. OAuth Parameters CBOR Mappings Registry

IANA is asked to add the following entry to the "OAuth Parameters CBOR Mappings" registry, following the procedure specified in [RFC9200].

* Name: edhoc_info
* CBOR Key: TBD (value between 1 and 255)
* Value Type: map
* Reference: [RFC-XXXX]
* Original Specification: [RFC-XXXX]

13.4. JSON Web Token Claims Registry

IANA is asked to add the following entries to the "JSON Web Token Claims" registry, following the procedure specified in [RFC7519].

* Claim Name: edhoc_info
* Claim Description: Information for EDHOC session
* Change Controller: IETF
* Reference: [RFC-XXXX]

13.5. CBOR Web Token (CWT) Claims Registry

IANA is asked to add the following entries to the "CBOR Web Token (CWT) Claims" registry, following the procedure specified in [RFC8392].

* Claim Name: edhoc_info
* Claim Description: Information for EDHOC session
* JWT Claim Name: edhoc_info
* Claim Key: TBD (value between 1 and 255)
* Claim Value Type: map
* Change Controller: IETF
* Reference: [RFC-XXXX]

13.6. JWT Confirmation Methods Registry

IANA is asked to add the following entries to the "JWT Confirmation Methods" registry, following the procedure specified in [RFC7800].

* Confirmation Method Value: x5c
* Confirmation Method Description: An ordered chain of X.509 certificates
* Change Controller: IETF
* Reference: Section 8.1 of [RFC-XXXX]

* Confirmation Method Value: x5b
* Confirmation Method Description: An unordered bag of X.509 certificates
* Change Controller: IETF
* Reference: Section 8.2 of [RFC-XXXX]

* Confirmation Method Value: x5t
* Confirmation Method Description: Hash of an X.509 certificate
* Change Controller: IETF
* Reference: Section 8.3 of [RFC-XXXX]

* Confirmation Method Value: x5u
* Confirmation Method Description: URI pointing to an ordered chain of X.509 certificates
* Change Controller: IETF
* Reference: Section 8.4 of [RFC-XXXX]

* Confirmation Method Value: c5c
* Confirmation Method Description: An ordered chain of C509 certificates
* Change Controller: IETF
* Reference: Section 8.5 of [RFC-XXXX]

* Confirmation Method Value: c5b
* Confirmation Method Description: An unordered bag of C509 certificates

* Change Controller: IETF

* Reference: Section 8.6 of [RFC-XXXX]

* Confirmation Method Value: c5t

* Confirmation Method Description: Hash of a C509 certificate

* Change Controller: IETF

* Reference: Section 8.7 of [RFC-XXXX]

* Confirmation Method Value: c5u

* Confirmation Method Description: URI pointing to a COSE_C509 containing an ordered chain of C509 certificates

* Change Controller: IETF

* Reference: Section 8.8 of [RFC-XXXX]

* Confirmation Method Value: kcwt

* Confirmation Method Description: A CBOR Web Token (CWT) containing a COSE_Key in a 'cnf' claim and possibly other claims

* Change Controller: IETF

* Reference: Section 8.9 of [RFC-XXXX]

* Confirmation Method Value: kccs

* Confirmation Method Description: A CWT Claims Set (CCS) containing a COSE_Key in a 'cnf' claim and possibly other claims

* Change Controller: IETF

* Reference: Section 8.10 of [RFC-XXXX]

13.7. CWT Confirmation Methods Registry

IANA is asked to add the following entries to the "CWT Confirmation Methods" registry, following the procedure specified in [RFC8747].

* Confirmation Method Name: x5chain
* Confirmation Method Description: An ordered chain of X.509 certificates
* JWT Confirmation Method Name: x5c
* Confirmation Key: TBD (value between 24 and 255)
* Confirmation Value Type: COSE_X509
* Change Controller: IETF
* Reference: Section 7.1 of [RFC-XXXX]

* Confirmation Method Name: x5bag
* Confirmation Method Description: An unordered bag of X.509 certificates
* JWT Confirmation Method Name: x5b
* Confirmation Key: TBD (value between 24 and 255)
* Confirmation Value Type: COSE_X509
* Change Controller: IETF
* Reference: Section 7.2 of [RFC-XXXX]

* Confirmation Method Name: x5t
* Confirmation Method Description: Hash of an X.509 certificate
* JWT Confirmation Method Name: x5t
* Confirmation Key: TBD (value between 1 and 23)
* Confirmation Value Type: COSE_CertHash

* Change Controller: IETF

* Reference: Section 7.3 of [RFC-XXXX]

* Confirmation Method Name: x5u

* Confirmation Method Description: URI pointing to an ordered chain of X.509 certificates

* JWT Confirmation Method Name: x5u

* Confirmation Key: TBD (value between 24 and 255)

* Confirmation Value Type: uri

* Change Controller: IETF

* Reference: Section 7.4 of [RFC-XXXX]

* Confirmation Method Name: c5c

* Confirmation Method Description: An ordered chain of C509 certificates

* JWT Confirmation Method Name: c5c

* Confirmation Key: TBD (value between 24 and 255)

* Confirmation Value Type: COSE_C509

* Change Controller: IETF

* Reference: Section 7.5 of [RFC-XXXX]

* Confirmation Method Name: c5b

* Confirmation Method Description: An unordered bag of C509 certificates

* JWT Confirmation Method Name: c5b

* Confirmation Key: TBD (value between 24 and 255)

* Confirmation Value Type: COSE_C509
* Change Controller: IETF
* Reference: Section 7.6 of [RFC-XXXX]

* Confirmation Method Name: c5t
* Confirmation Method Description: Hash of a C509 certificate
* JWT Confirmation Method Name: c5t
* Confirmation Key: TBD (value between 1 and 23)
* Confirmation Value Type: COSE_CertHash
* Change Controller: IETF
* Reference: Section 7.7 of [RFC-XXXX]

* Confirmation Method Name: c5u
* Confirmation Method Description: URI pointing to a COSE_C509 containing an ordered chain of C509 certificates
* JWT Confirmation Method Name: c5u
* Confirmation Key: TBD (value between 24 and 255)
* Confirmation Value Type: uri
* Change Controller: IETF
* Reference: Section 7.8 of [RFC-XXXX]

* Confirmation Method Name: kcwt
* Confirmation Method Description: A CBOR Web Token (CWT) containing a COSE_Key in a ’cnf’ claim and possibly other claims
* JWT Confirmation Method Name: kcwt

* Confirmation Key: TBD (value between 1 and 23)

* Confirmation Value Type: COSE_Messages

* Change Controller: IETF

* Reference: Section 7.9 of [RFC-XXXX]

* Confirmation Method Name: kccs

* Confirmation Method Description: A CWT Claims Set (CCS) containing a COSE_Key in a 'cnf' claim and possibly other claims

* JWT Confirmation Method Name: kccs

* Confirmation Key: TBD (value between 1 and 23)

* Confirmation Value Type: map / #6(map)

* Change Controller: IETF

* Reference: Section 7.10 of [RFC-XXXX]

13.8. EDHOC External Authorization Data Registry

IANA is asked to add the following entries to the "EDHOC External Authorization Data" registry defined in Section 10.5 of [RFC9528].

* Name: ACE-OAuth Access Token

* Label: TBD (value between 24 and 255)

* Description: An Access Token as used in the ACE-OAuth framework [RFC9200]

* Reference: [RFC-XXXX], Section 4.1

* Name: Session ID

* Label: TBD (value between 1 and 23)

* Description: The identifier of an EDHOC session

* Reference: [RFC-XXXX], Section 4.1

* Name: Credential By Value

* Label: TBD (value between 1 and 23)

* Description: The sender peer wishes to receive the other peer’s credential transported by value (and not identified by reference)

* Reference: [RFC-XXXX], Section 4.14

* Name: Request Creation Hints

* Label: TBD (value between 1 and 23)

* Description: Request or convey AS Request Creation Hints

* Reference: [RFC-XXXX], Section 4.13

13.9. EDHOC Exporter Labels

IANA is asked to register the following entries in the "EDHOC Exporter Labels" registry, within the "Ephemeral Diffie-Hellman Over COSE (EDHOC)" registry group.

* Label: 26 (suggested value)

* Description: Derived N_S challenge for key provisioning for Group OSCORE using the ACE framework [I-D.ietf-ace-key-groupcomm-oscore].

* Change Controller: IETF

* Reference: [RFC-XXXX, Section 6]

* Label: 27 (suggested value)

* Description: Derived N_S challenge for key provisioning for CoAP Pub-Sub using the ACE framework [I-D.ietf-ace-coap-pubsub-profile].

* Change Controller: IETF

* Reference: [RFC-XXXX, Section 6]

13.10. EDHOC Information Registry

IANA is requested to create a new "EDHOC Information" registry within the "Ephemeral Diffie-Hellman Over COSE (EDHOC)" registry group defined in [RFC9528].

The registration policy is either "Private Use", "Standards Action with Expert Review", "Specification Required" per Section 4.6 of [RFC8126], or "Expert Review" per Section 4.5 of [RFC8126]. "Expert Review" guidelines are provided in Section 13.13.

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

The columns of the registry are:

* Name: A descriptive name that enables easier reference to this item. Because a core goal of this document is for the resulting representations to be compact, it is RECOMMENDED that the name be short.

  This name is case sensitive. Names may not match other registered names in a case-insensitive manner unless the Designated Experts determine that there is a compelling reason to allow an exception. The name is not used in the CBOR encoding.

* CBOR label: The value to be used as CBOR abbreviation of the item.

  The value MUST be unique. The value can be a positive integer, a negative integer, or a string. Integer values between -256 and 255 and strings of length 1 are designated as "Standards Action with Expert Review". Integer values from -65536 to -257 and from 256 to 65535 and strings of maximum length 2 are designated as "Specification Required". Integer values greater than 65535 and strings of length greater than 2 are designated as "Expert Review". Integer values less than -65536 are marked as "Private Use".

* CBOR type: The CBOR type of the item, or a pointer to the registry that defines its type, when that depends on another item.

* Registry: The registry that values of the item may come from, if one exists.

* Description: A brief description of the item.

* Type: The category of the item, i.e., P if prescriptive or NP if Non-Prescriptive.

* Specification: A pointer to the public specification for the item, if one exists.

This registry will be initially populated by the values in Table 1. In the "Specification" column, the value for all of these entries will be [RFC-XXXX] and [RFC9528].

13.11. EDHOC Trust Anchor Purposes Registry

IANA is requested to create a new "EDHOC Trust Anchor Purposes" registry within the "Ephemeral Diffie-Hellman Over COSE (EDHOC)" registry group defined in [RFC9528].

The registration policy is either "Private Use", "Standards Action with Expert Review", or "Specification Required" per Section 4.6 of [RFC8126]. "Expert Review" guidelines are provided in Section 13.13.

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

The columns of this registry are:

* Name: This field contains the descriptive name of the trust anchor purpose, to enable easier reference to the item. These names MUST be unique.

* CBOR label: This field contains the value used to identify the trust anchor purpose. These values MUST be unique. The value can be an unsigned integer or a negative integer. Different ranges of values use different registration policies:

  - Integer values from -24 to 23 are designated as "Standards Action with Expert Review".

- Integer values from -65536 to -25 and from 24 to 65535 are designated as "Specification Required".

- Integer values smaller than -65536 and greater than 65535 are marked as "Private Use".

* Description: This field contains a short description of the trust anchor purpose.

* Reference: This field contains a pointer to the public specification for the trust anchor purpose.

This registry has been initially populated with the values in Section 9.

13.12. EDHOC Trust Anchor Types Registry

IANA is requested to create a new "EDHOC Trust Anchor Types" registry within the "Ephemeral Diffie-Hellman Over COSE (EDHOC)" registry group defined in [RFC9528].

The registration policy is either "Private Use", "Standards Action with Expert Review", or "Specification Required" per Section 4.6 of [RFC8126]. "Expert Review" guidelines are provided in Section 13.13.

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

The columns of this registry are:

* Name: This field contains the descriptive name of the type of trust anchor, to enable easier reference to the item. These names MUST be unique.

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

  - Integer values from -24 to 23 are designated as "Standards Action with Expert Review".

- Integer values from -65536 to -25 and from 24 to 65535 are designated as "Specification Required".

- Integer values smaller than -65536 and greater than 65535 are marked as "Private Use".

* Value type: This field contains the CBOR type for the value portion of the label.

* Description: This field contains a short description of the type of trust anchor.

* Reference: This field contains a pointer to the public specification for the type of trust anchor.

This registry has been initially populated with the values in Table 3.

13.13. Expert Review Instructions

"Standards Action with Expert Review", "Specification Required", and "Expert Review" are three of the registration policies defined for the IANA registries established in this document. This section gives some general guidelines for what the experts should be looking for, but they are being designated as experts for a reason so they should be given substantial latitude.

Expert reviewers should take into consideration the following points:

* Clarity and correctness of registrations. Experts are expected to check the clarity of purpose and use of the requested entries. Experts need to make sure that the object of registration is clearly defined in the corresponding specification. Entries that do not meet these objectives of clarity and completeness must not be registered.

* Point squatting should be discouraged. Reviewers are encouraged to get sufficient information for registration requests to ensure that the usage is not going to duplicate one that is already registered and that the point is likely to be used in deployments. The zones tagged as "Private Use" are intended for testing purposes and closed environments. Code points in other ranges should not be assigned for testing.

* Specifications are required for the "Standards Action with Expert Review" range of point assignment. Specifications should exist for "Specification Required" ranges, but early assignment before a specification is available is considered to be permissible. When

specifications are not provided, the description provided needs to have sufficient information to identify what the point is being used for.

* Experts should take into account the expected usage of fields when approving point assignment. Documents published via Standards Action can also register points outside the Standards Action range. The length of the encoded value should be weighed against how many code points of that length are left, the size of device it will be used on, and the number of code points left that encode to that size.

14. References

14.1. Normative References

[COSE.Header.Parameters]
IANA, "COSE Header Parameters", <https://www.iana.org/assignments/cose/cose.xhtml#header-parameters>.

[I-D.ietf-ace-coap-pubsub-profile]
Palombini, F., Sengul, C., and M. Tiloca, "CoAP Publish-Subscribe Profile for Authentication and Authorization for Constrained Environments (ACE)", Work in Progress, Internet-Draft, draft-ietf-ace-coap-pubsub-profile-03, 9 February 2026, <https://datatracker.ietf.org/doc/html/draft-ietf-ace-coap-pubsub-profile-03>.

[I-D.ietf-ace-key-groupcomm-oscore]
Tiloca, M. and F. Palombini, "Key Management for Group Object Security for Constrained RESTful Environments (Group OSCORE) Using Authentication and Authorization for Constrained Environments (ACE)", Work in Progress, Internet-Draft, draft-ietf-ace-key-groupcomm-oscore-20, 25 February 2026, <https://datatracker.ietf.org/doc/html/draft-ietf-ace-key-groupcomm-oscore-20>.

[I-D.ietf-core-uri-path-abbrev]
Amsss, C. and M. Richardson, "URI-Path abbreviation in CoAP", Work in Progress, Internet-Draft, draft-ietf-core-uri-path-abbrev-02, 20 October 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-core-uri-path-abbrev-02>.

[I-D.ietf-cose-cbor-encoded-cert]
Mattsson, J. P., Selander, G., Raza, S., Hglund, J., and M. Furuhed, "CBOR Encoded X.509 Certificates (C509

Certificates)", Work in Progress, Internet-Draft, draft-ietf-cose-cbor-encoded-cert-16, 25 January 2026, <https://datatracker.ietf.org/doc/html/draft-ietf-cose-cbor-encoded-cert-16>.

[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/rfc/rfc2119>.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, DOI 10.17487/RFC3986, January 2005, <https://www.rfc-editor.org/rfc/rfc3986>.

[RFC4648] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 4648, DOI 10.17487/RFC4648, October 2006, <https://www.rfc-editor.org/rfc/rfc4648>.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, DOI 10.17487/RFC5280, May 2008, <https://www.rfc-editor.org/rfc/rfc5280>.

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

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

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

[RFC7515] Jones, M., Bradley, J., and N. Sakimura, "JSON Web Signature (JWS)", RFC 7515, DOI 10.17487/RFC7515, May 2015, <https://www.rfc-editor.org/rfc/rfc7515>.

[RFC7519] Jones, M., Bradley, J., and N. Sakimura, "JSON Web Token (JWT)", RFC 7519, DOI 10.17487/RFC7519, May 2015, <https://www.rfc-editor.org/rfc/rfc7519>.

[RFC7800] Jones, M., Bradley, J., and H. Tschofenig, "Proof-of-Possession Key Semantics for JSON Web Tokens (JWTs)", RFC 7800, DOI 10.17487/RFC7800, April 2016, <https://www.rfc-editor.org/rfc/rfc7800>.

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

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/rfc/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/rfc/rfc8392>.

[RFC8610] Birkholz, H., Vigano, C., and C. Bormann, "Concise Data Definition Language (CDDL): A Notational Convention to Express Concise Binary Object Representation (CBOR) and JSON Data Structures", RFC 8610, DOI 10.17487/RFC8610, June 2019, <https://www.rfc-editor.org/rfc/rfc8610>.

[RFC8613] Selander, G., Mattsson, J., Palombini, F., and L. Seitz, "Object Security for Constrained RESTful Environments (OSCORE)", RFC 8613, DOI 10.17487/RFC8613, July 2019, <https://www.rfc-editor.org/rfc/rfc8613>.

[RFC8742] Bormann, C., "Concise Binary Object Representation (CBOR) Sequences", RFC 8742, DOI 10.17487/RFC8742, February 2020, <https://www.rfc-editor.org/rfc/rfc8742>.

[RFC8747] Jones, M., Seitz, L., Selander, G., Erdtman, S., and H. Tschofenig, "Proof-of-Possession Key Semantics for CBOR Web Tokens (CWTs)", RFC 8747, DOI 10.17487/RFC8747, March 2020, <https://www.rfc-editor.org/rfc/rfc8747>.

[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/rfc/rfc8949>.

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

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

[RFC9200] Seitz, L., Selander, G., Wahlstroem, E., Erdtman, S., and H. Tschofenig, "Authentication and Authorization for Constrained Environments Using the OAuth 2.0 Framework (ACE-OAuth)", RFC 9200, DOI 10.17487/RFC9200, August 2022, <https://www.rfc-editor.org/rfc/rfc9200>.

[RFC9201] Seitz, L., "Additional OAuth Parameters for Authentication and Authorization for Constrained Environments (ACE)", RFC 9201, DOI 10.17487/RFC9201, August 2022, <https://www.rfc-editor.org/rfc/rfc9201>.

[RFC9203] Palombini, F., Seitz, L., Selander, G., and M. Gunnarsson, "The Object Security for Constrained RESTful Environments (OSCORE) Profile of the Authentication and Authorization for Constrained Environments (ACE) Framework", RFC 9203, DOI 10.17487/RFC9203, August 2022, <https://www.rfc-editor.org/rfc/rfc9203>.

[RFC9360] Schaad, J., "CBOR Object Signing and Encryption (COSE): Header Parameters for Carrying and Referencing X.509 Certificates", RFC 9360, DOI 10.17487/RFC9360, February 2023, <https://www.rfc-editor.org/rfc/rfc9360>.

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

[RFC9562] Davis, K., Peabody, B., and P. Leach, "Universally Unique IDentifiers (UUIDs)", RFC 9562, DOI 10.17487/RFC9562, May 2024, <https://www.rfc-editor.org/rfc/rfc9562>.

[RFC9594] Palombini, F. and M. Tiloca, "Key Provisioning for Group Communication Using Authentication and Authorization for Constrained Environments (ACE)", RFC 9594, DOI 10.17487/RFC9594, September 2024, <https://www.rfc-editor.org/rfc/rfc9594>.

[RFC9668] Palombini, F., Tiloca, M., Hglund, R., Hristozov, S., and G. Selander, "Using Ephemeral Diffie-Hellman Over COSE (EDHOC) with the Constrained Application Protocol (CoAP) and Object Security for Constrained RESTful Environments (OSCORE)", RFC 9668, DOI 10.17487/RFC9668, November 2024, <https://www.rfc-editor.org/rfc/rfc9668>.

14.2. Informative References

[I-D.ietf-ace-coap-est-oscore]
Selander, G., Raza, S., Furuhed, M., Vuini, M., and T. Claeys, "Protecting EST Payloads with OSCORE", Work in Progress, Internet-Draft, draft-ietf-ace-coap-est-oscore-09, 20 October 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-ace-coap-est-oscore-09>.

[I-D.ietf-ace-group-oscore-profile]
Tiloca, M., Hglund, R., and F. Palombini, "The Group Object Security for Constrained RESTful Environments (Group OSCORE) Profile of the Authentication and Authorization for Constrained Environments (ACE) Framework", Work in Progress, Internet-Draft, draft-ietf-ace-group-oscore-profile-05, 3 September 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-ace-group-oscore-profile-05>.

[I-D.ietf-ace-workflow-and-params]
Tiloca, M. and G. Selander, "Short Distribution Chain (SDC) Workflow and New OAuth Parameters for the Authentication and Authorization for Constrained Environments (ACE) Framework", Work in Progress, Internet-Draft, draft-ietf-ace-workflow-and-params-06, 20 October 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-ace-workflow-and-params-06>.

[I-D.ietf-core-groupcomm-bis]
Dijk, E. and M. Tiloca, "Group Communication for the Constrained Application Protocol (CoAP)", Work in Progress, Internet-Draft, draft-ietf-core-groupcomm-bis-18, 10 February 2026, <https://datatracker.ietf.org/doc/html/draft-ietf-core-groupcomm-bis-18>.

[I-D.ietf-core-oscore-id-update]
Hglund, R. and M. Tiloca, "Identifier Update for OSCORE", Work in Progress, Internet-Draft, draft-ietf-core-oscore-id-update-05, 20 October 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-core-oscore-id-update-05>.

[I-D.ietf-core-oscore-key-update]
Hglund, R. and M. Tiloca, "Key Update for OSCORE (KUDOS)", Work in Progress, Internet-Draft, draft-ietf-core-oscore-key-update-12, 20 October 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-core-oscore-key-update-12>.

[I-D.ietf-lake-app-profiles]
Tiloca, M. and R. Hglund, "Coordinating the Use of Application Profiles for Ephemeral Diffie-Hellman Over COSE (EDHOC)", Work in Progress, Internet-Draft, draft-ietf-lake-app-profiles-03, 20 October 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-lake-app-profiles-03>.

[I-D.ietf-lake-authz]
Selander, G., Mattsson, J. P., Vuini, M., Fedrecheski, G., and M. Richardson, "Lightweight Authorization using Ephemeral Diffie-Hellman Over COSE (ELA)", Work in Progress, Internet-Draft, draft-ietf-lake-authz-06, 20 October 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-lake-authz-06>.

[I-D.serafin-lake-ta-hint]
Serafin, M. and G. Selander, "Trust Anchor Hints in Ephemeral Diffie-Hellman Over COSE (EDHOC)", Work in Progress, Internet-Draft, draft-serafin-lake-ta-hint-00, 21 October 2024, <https://datatracker.ietf.org/doc/html/draft-serafin-lake-ta-hint-00>.

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

[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/rfc/rfc8446>.

[RFC9110] Fielding, R., Ed., Nottingham, M., Ed., and J. Reschke, Ed., "HTTP Semantics", STD 97, RFC 9110, DOI 10.17487/RFC9110, June 2022, <https://www.rfc-editor.org/rfc/rfc9110>.

[RFC9147] Rescorla, E., Tschofenig, H., and N. Modadugu, "The Datagram Transport Layer Security (DTLS) Protocol Version 1.3", RFC 9147, DOI 10.17487/RFC9147, April 2022, <https://www.rfc-editor.org/rfc/rfc9147>.

Appendix A. Examples

This appendix provides examples where this profile of ACE is used. In particular:

* Appendix A.1 does not make use of any optimization.

* Appendix A.2 makes use of the optimizations defined in [RFC9668], hence reducing the roundtrips of the interactions between C and RS.

* Appendix A.3 makes use of the EAD items EAD_REQUEST_CREATION_HINTS (see Section 4.13) and EAD_CRED_BY_VALUE (see Section 4.14), allowing C to receive AS Request Creation Hints from the RS transported in an EAD item. This is useful if C is not able to determine in advance the appropriate AS to contact.

All these examples build on the following assumptions, as relying on expected early procedures performed at AS. These include the registration of resource servers by the respective resource owners as well as the registrations of clients authorized to request access tokens for those resource servers.

* AS knows the authentication credential AUTH_CRED_C of C.

* C knows the authentication credential AUTH_CRED_AS of AS.

* AS knows the authentication credential AUTH_CRED_RS of RS.

* RS knows the authentication credential AUTH_CRED_AS of AS.

  This is relevant in case AS and RS actually require a secure association (e.g., for RS to perform token introspection at AS, or for AS to upload an access token to RS on behalf of C as described in [I-D.ietf-ace-workflow-and-params]).

As a result of the assumptions above, it is possible to limit the transport of AUTH_CRED_C and AUTH_CRED_RS by value only to the following two cases, and only when C requests an access token for RS for the first time when considering the pair (AUTH_CRED_C, AUTH_CRED_RS).

* In the access token response from AS to C, where AUTH_CRED_RS is specified by the "rs_cnf" parameter.

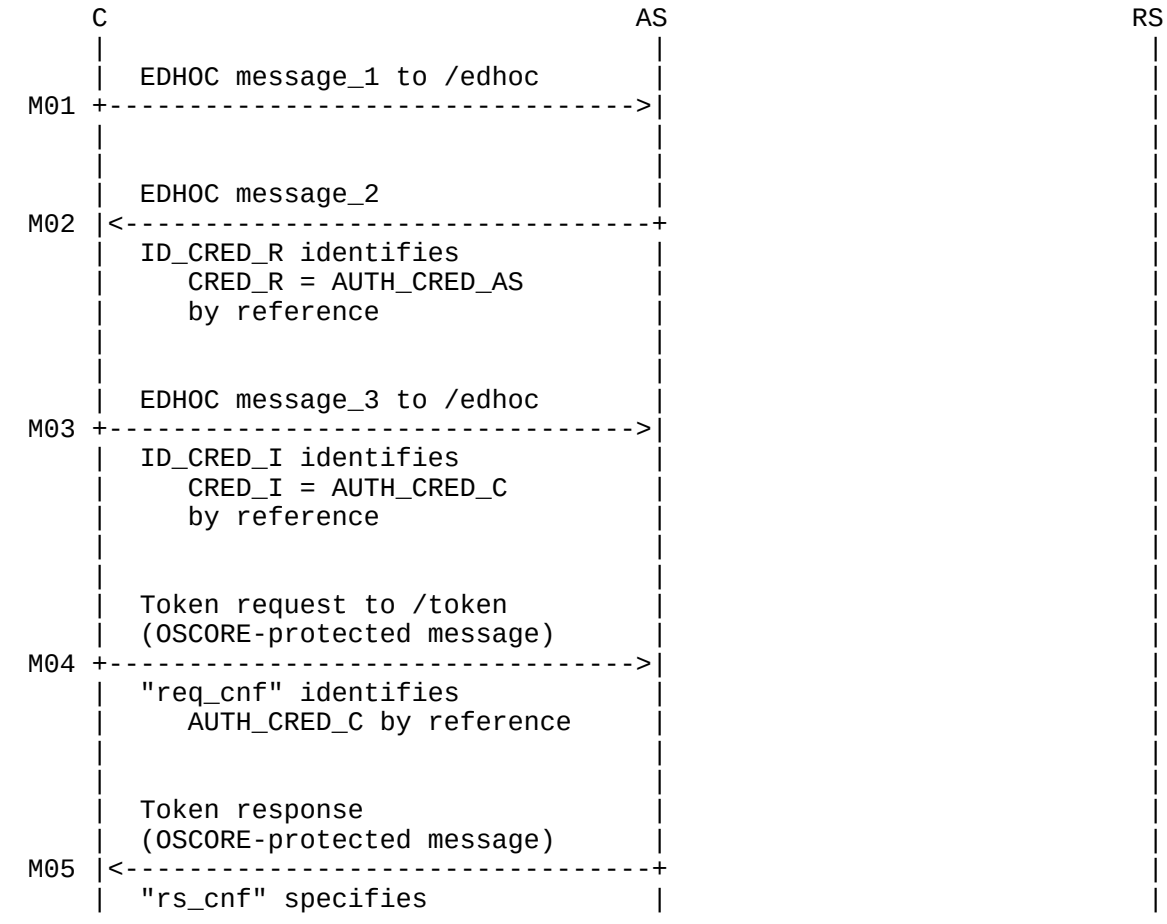
* In the access token, where AUTH_CRED_C is specified by the "cnf" claim.

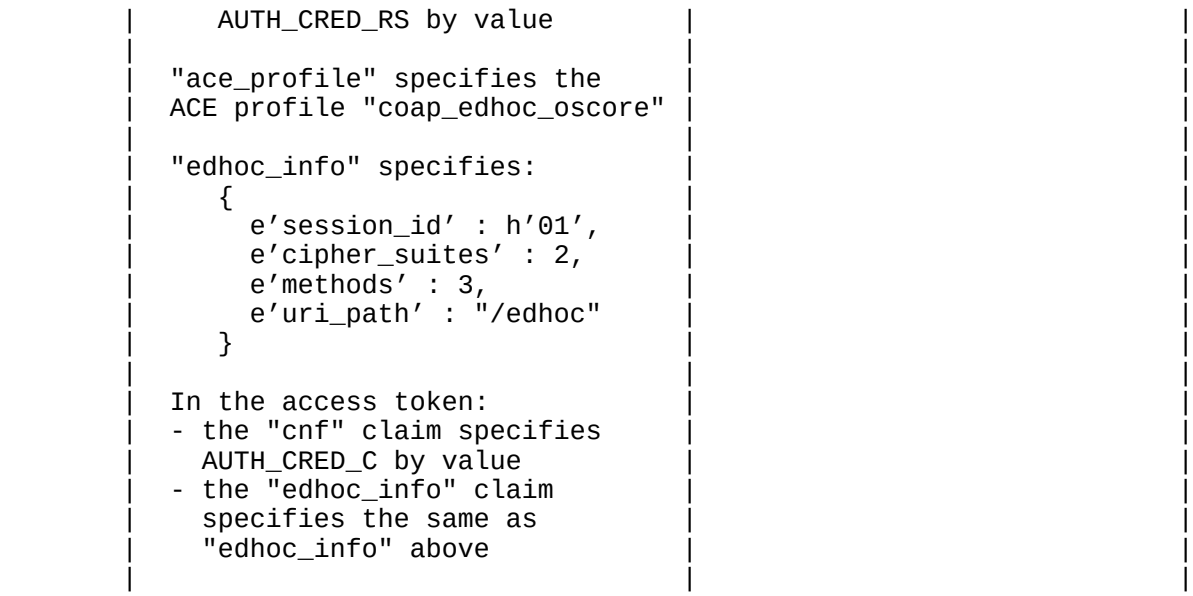
Note that, even under the circumstances mentioned above, AUTH_CRED_C might rather be identified by reference. This is possible if RS can effectively use such a reference from the access token to retrieve AUTH_CRED_C (e.g., from a trusted repository of authentication credentials reachable through a non-constrained link), and if AS is in turn aware of that.

In any other case, it is otherwise possible to identify both AUTH_CRED_C and AUTH_CRED_RS by reference, when performing the ACE access control workflow as well as later on when C and RS run EDHOC.

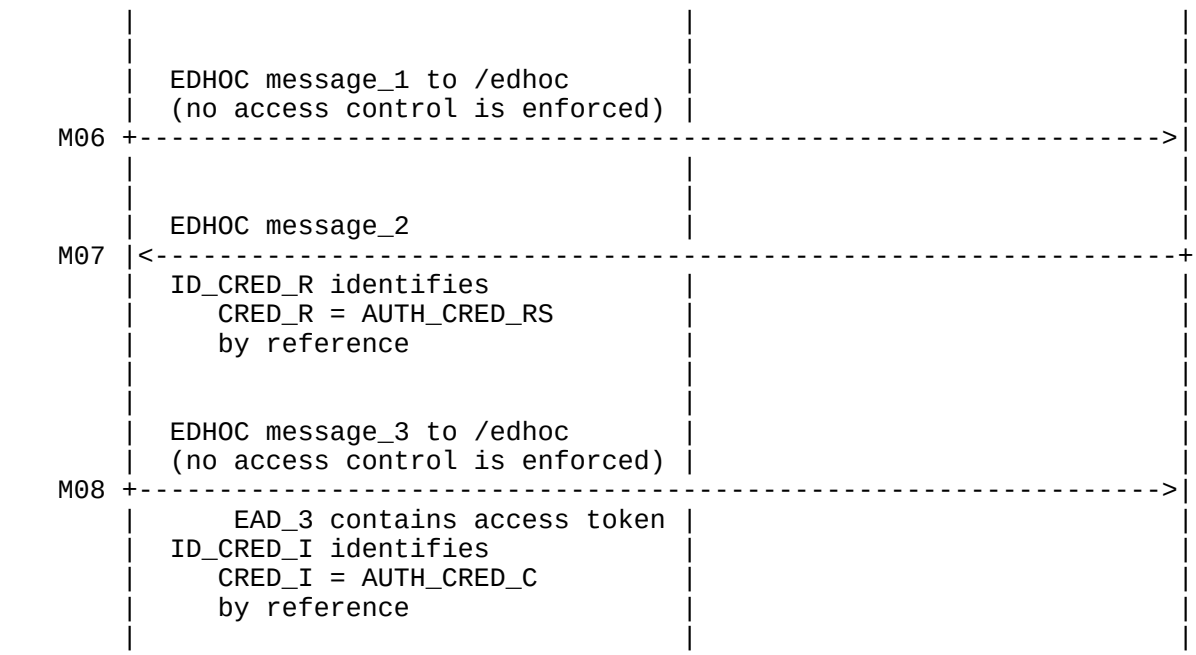
A.1. Workflow without Optimizations

The example below shows a simple interaction between C and RS: C and RS run EDHOC wherein C uploads the access token to RS, and then accesses a protected resource at RS.

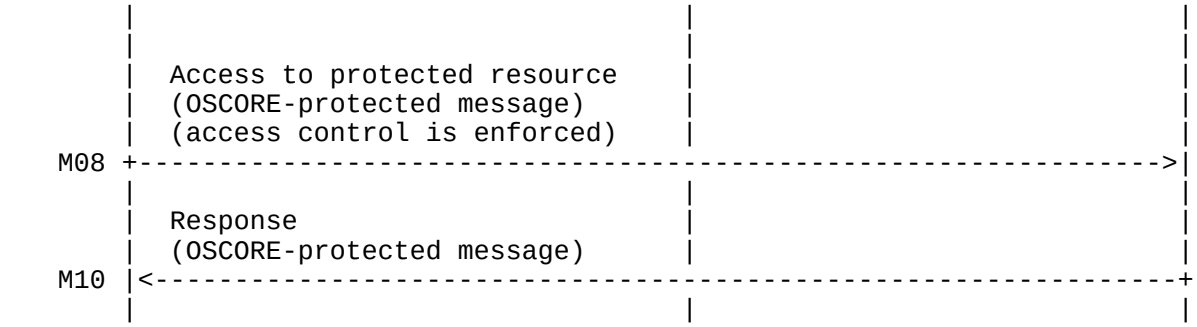




Possibly after chain verification, C adds AUTH_CRED_RS to the set of its trusted peer authentication credentials, relying on AS as trusted provider



Possibly after chain verification, RS adds AUTH_CRED_C
to the set of its trusted peer authentication credentials,
relying on AS as trusted provider



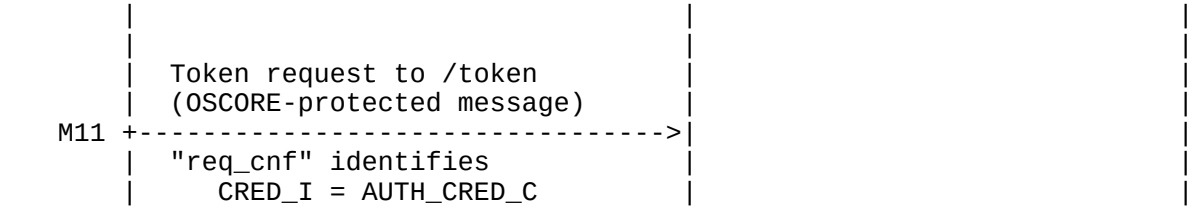
Later on, the access token expires ...

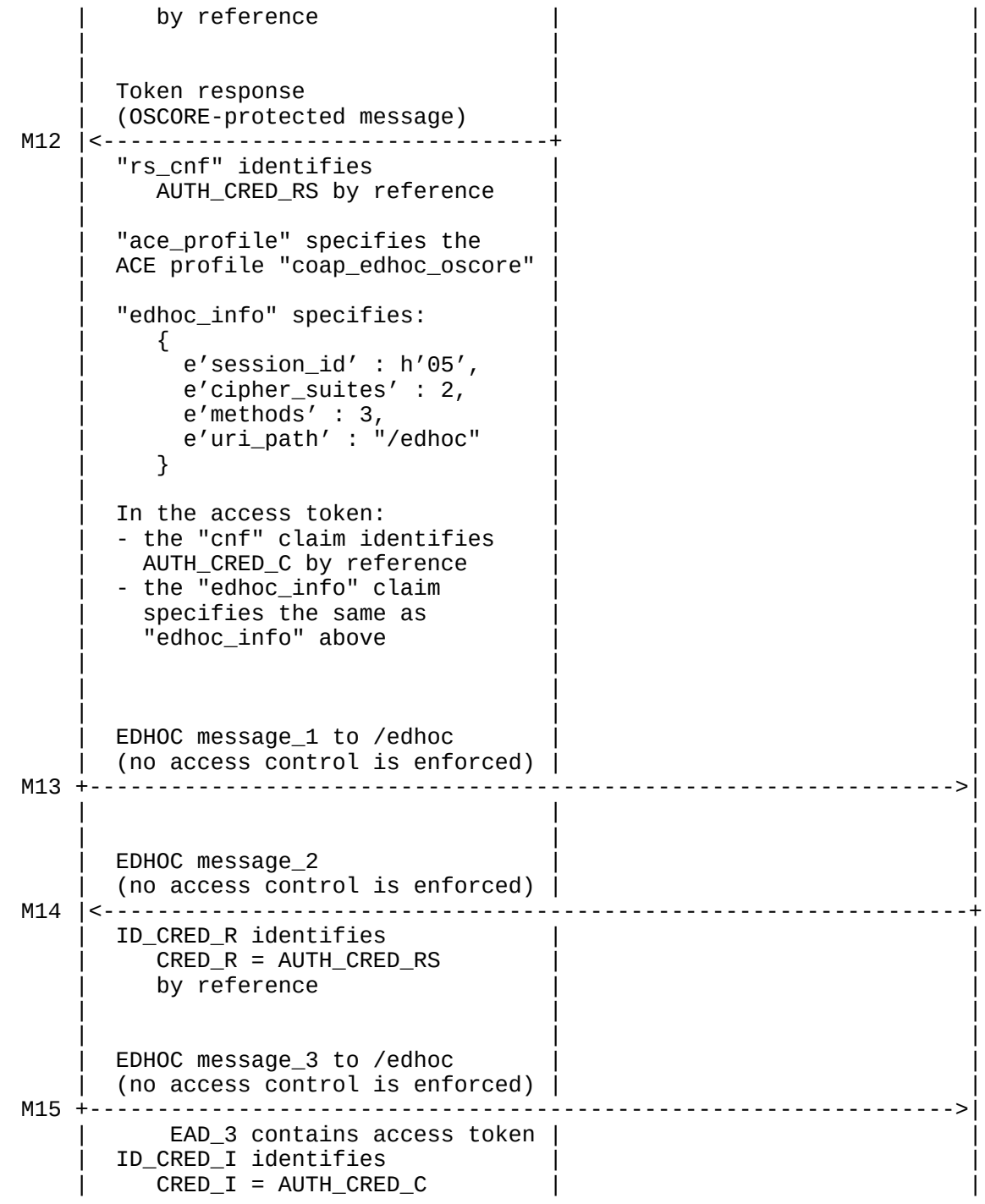
- C and RS delete their OSCORE Security Context and purge the EDHOC session used to derive it (unless the same session is also used for other reasons).
- RS retains AUTH_CRED_C as still valid, and AS knows about it.
- The Client retains AUTH_CRED_RS as still valid, and AS knows about it.

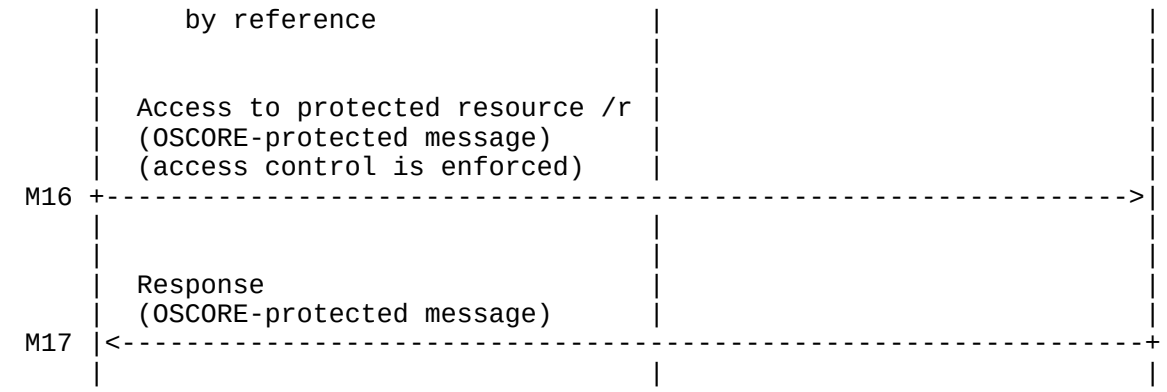
Time passes ...

C asks for a new access token; now all the
authentication credentials can be identifies by reference

The price to pay is on AS, about remembering that at least
one access token has been issued for the pair (Client, RS)
and considering the pair (AUTH_CRED_C, AUTH_CRED_RS)



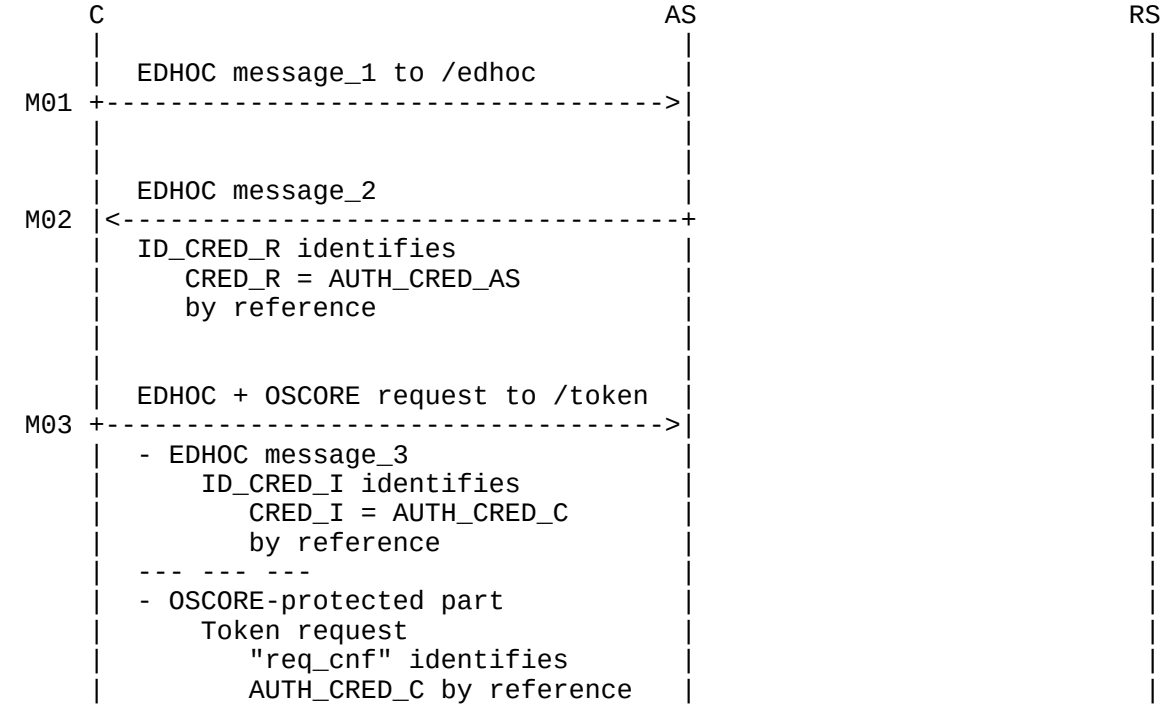

by reference
Token response
(OSCORE-protected message)
M12
"rs_cnf" identifies
AUTH_CRED_RS by reference
"ace_profile" specifies the
ACE profile "coap_edhoc_oscore"
"edhoc_info" specifies:
{
e'session_id' : h'05',
e'cipher_suites' : 2,
e'methods' : 3,
e'uri_path' : "/edhoc"
}
In the access token:
- the "cnf" claim identifies
AUTH_CRED_C by reference
- the "edhoc_info" claim
specifies the same as
"edhoc_info" above
EDHOC message_1 to /edhoc
(no access control is enforced)
M13
EDHOC message_2
(no access control is enforced)
M14
ID_CRED_R identifies
CRED_R = AUTH_CRED_RS
by reference
EDHOC message_3 to /edhoc
(no access control is enforced)
M15
EAD_3 contains access token
ID_CRED_I identifies
CRED_I = AUTH_CRED_C

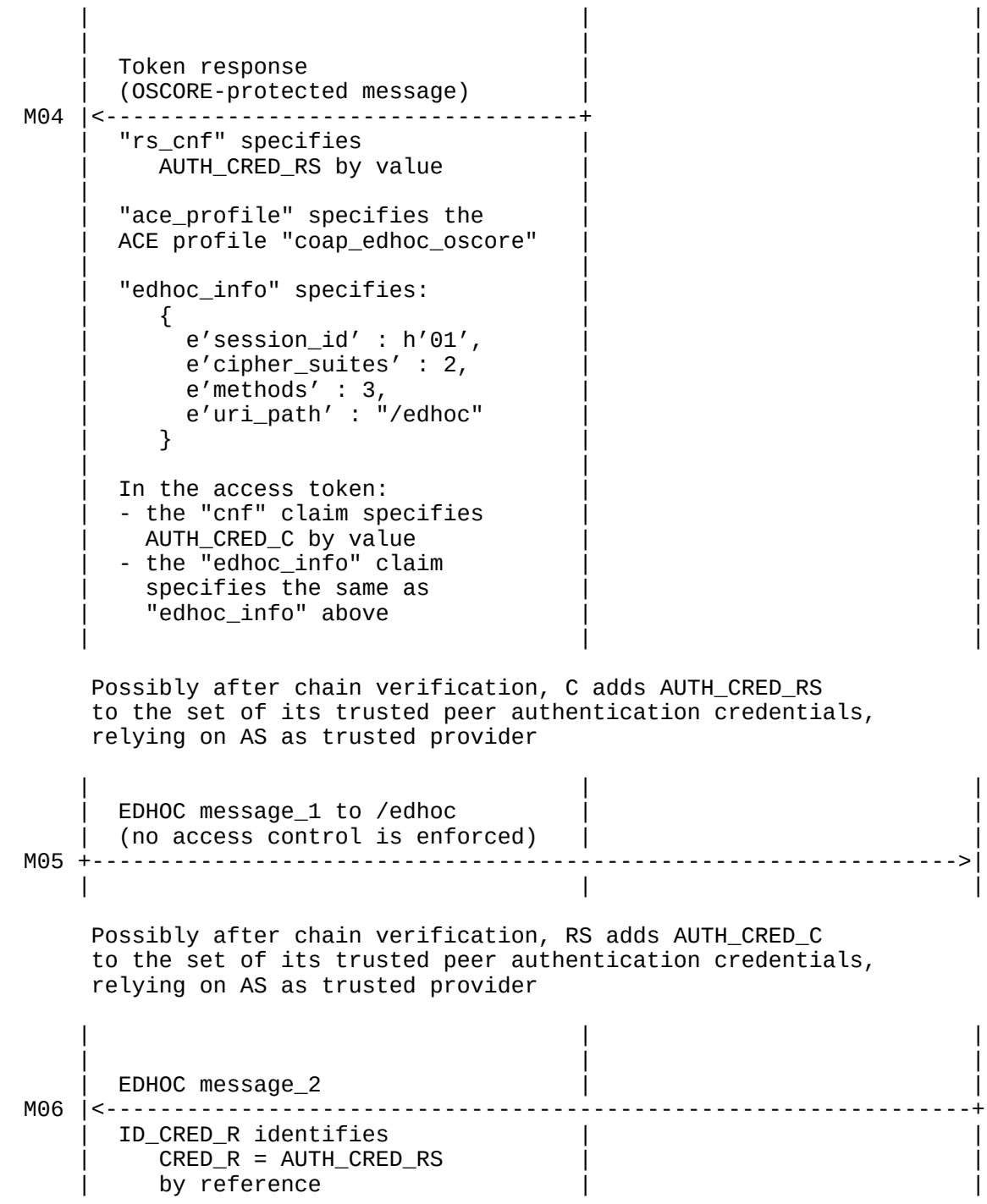


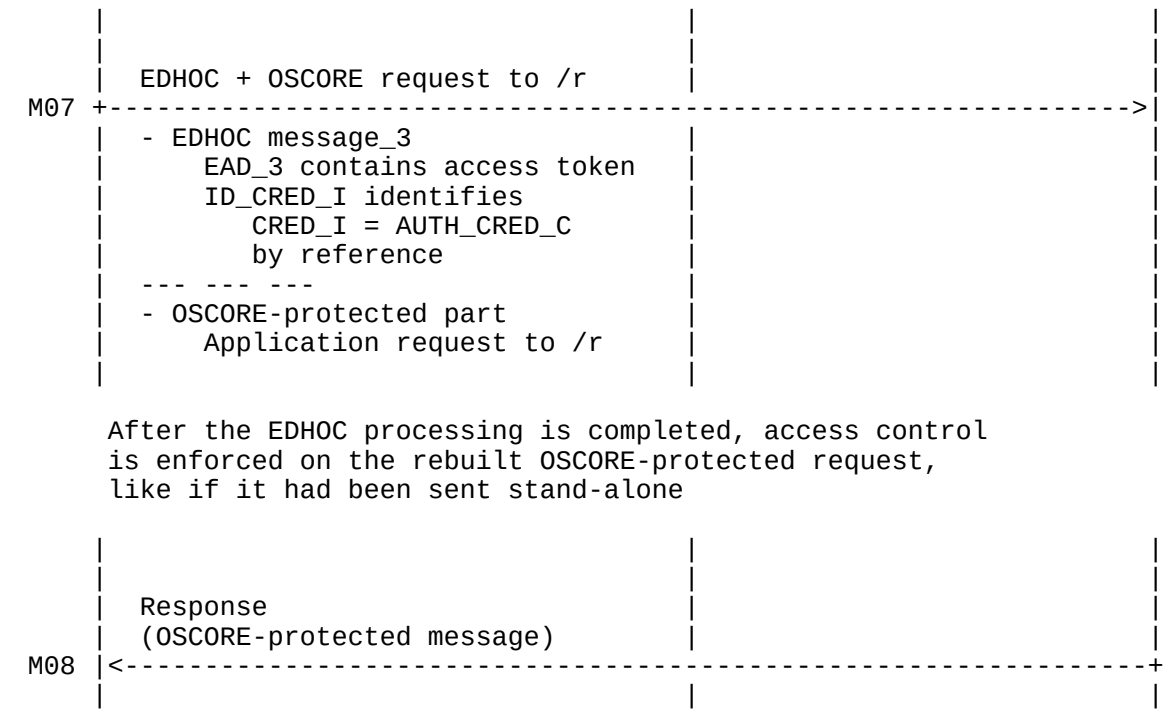
A.2. Workflow with Optimizations

The example below builds on the example in Appendix A.1, while additionally using the EDHOC + OSCORE request defined in [RFC9668] when running EDHOC both with AS and with RS.

This interaction between C and RS consists of only two roundtrips to upload the access token, run EDHOC, and access the protected resource at RS.




Token response
(OSCORE-protected message)
M04
"rs_cnf" specifies
AUTH_CRED_RS by value
"ace_profile" specifies the
ACE profile "coap_edhoc_oscore"
"edhoc_info" specifies:
{
e'session_id' : h'01',
e'cipher_suites' : 2,
e'methods' : 3,
e'uri_path' : "/edhoc"
}
In the access token:
- the "cnf" claim specifies
AUTH_CRED_C by value
- the "edhoc_info" claim
specifies the same as
"edhoc_info" above
Possibly after chain verification, C adds AUTH_CRED_RS
to the set of its trusted peer authentication credentials,
relying on AS as trusted provider
EDHOC message_1 to /edhoc
(no access control is enforced)
M05
Possibly after chain verification, RS adds AUTH_CRED_C
to the set of its trusted peer authentication credentials,
relying on AS as trusted provider
EDHOC message_2
M06
ID_CRED_R identifies
CRED_R = AUTH_CRED_RS
by reference



A.3. Non-sequential Workflow

When this profile is used, the C might not be able to determine in advance the appropriate AS to contact. In such cases, C may initiate EDHOC with RS prior to obtaining an access token and rely on RS to provide this information.

One approach, as detailed in this section, is for RS to convey this information by including an External Authorization Data (EAD) item in EDHOC message_2. The content of this EAD item is the information conveyed in a Request Creation Hints response, thereby helping C to identify the correct AS, and possibly other relevant parameters needed to request an access token.

The following describes an example scenario where this functionality is used in the EDHOC forward message flow, in particular taking advantage of the new EAD items EAD_REQUEST_CREATION_HINTS (see Section 4.13) and EAD_CRED_BY_VALUE (see Section 4.14).

1. Without having an access token, C sends EDHOC message_1 to RS, including in EAD_1:

* A new EAD item EAD_REQUEST_CREATION_HINTS, here specifying no ead_value.

  By doing so, C is asking RS to include the same EAD item in EAD_2, specifying the information that would be included in a Request Creation Hints response.

* A new EAD item EAD_CRED_BY_VALUE, asking RS to specify its authentication credential AUTH_CRED_RS by value in ID_CRED_R.

2. RS replies with EDHOC message_2, including in EAD_2:

   * The new EAD item EAD_REQUEST_CREATION_HINTS, whose CBOR byte string specified as ead_value encodes the information that would have been included in a Request Creation Hints response to an unauthorized request sent to RS. In particular, this information includes the URI to the /token endpoint at the AS.

     However, note that C has not made an actual request with a specific request method and targeting a specific application resource. That is, RS does not know yet what resources C is actually interested in accessing later on. Hence, the information specified in this EAD_REQUEST_CREATION_HINTS item is typically not expected to include "audience" and "scope".

3. C requests an access token for the right audience and scope from the right AS, based on pre-configured parameters on the client and the information from EAD_REQUEST_CREATION_HINTS within EDHOC message_2, like if C had received a Request Creation Hints response.

   C should already know the right audience and scope to specify in the Access Token Request, as that information is not expected to be provided by RS in the EAD_REQUEST_CREATION_HINTS item of EDHOC message_2. There may also be default audience and scope set at the AS to use, if none is specified by C in its Access Token Request.

4. C sends EDHOC message_3 to RS, specifying the access token in EAD_3 as value of the EAD item "EAD_ACCESS_TOKEN".

5. If a protected request from C for accessing a resource at RS is not authorized per the issued access token, then RS replies with a protected error response. Within that error response, RS can provide C with information that would have been specified in a Request Creation Hints response. This time, that information also includes "audience" and "scope". After this, C can follow-up requesting a new access token that dynamically updates access rights accordingly (see Section 3.1).

   Note that this applies also if C uses the EDHOC + OSCORE combined request (see [RFC9668]), hence combining EDHOC message_3 with the first OSCORE-protected request to access a protected resource at RS (see Appendix A.2).

If, instead, the EDHOC reverse message flow is used, then the following differences compared to what is described above apply:

* EAD_REQUEST_CREATION_HINTS and EAD_CRED_BY_VALUE are included in EDHOC message_2 by C, in order to ask RS to provide the Request Creation Hints information and to specify AUTH_CRED_RS by value in ID_CRED_I, respectively.

* EAD_REQUEST_CREATION_HINTS is included in EDHOC message_3 by RS, with value the Request Creation Hints information.

* After receiving EDHOC message_3, C requests the access token to the AS, based on the information from EAD_REQUEST_CREATION_HINTS in EDHOC message_3.

* Finally, C sends EDHOC message_4 to RS, specifying the access token in EAD_4 as value of the EAD item "EAD_ACCESS_TOKEN".

Appendix B. Profile Requirements

This section lists the specifications of this profile based on the requirements of the framework, as requested in Appendix C of [RFC9200].

* Optionally, define new methods for the client to discover the necessary permissions and AS for accessing a resource, different from the one proposed in [RFC9200]: Not specified

* Optionally, specify new grant types: Not specified

* Optionally, define the use of client certificates as client credential type: C can use authentication credentials of any type admitted by the EDHOC protocol, including public key certificates such as X.509 and C509 certificates.

* Specify the communication protocol the client and RS must use: CoAP

* Specify the security protocol the client and RS must use to protect their communication: OSCORE

* Specify how the client and RS mutually authenticate: Explicitly, by successfully executing the EDHOC protocol, after which a common OSCORE Security Context is exported from the EDHOC session. As per the EDHOC authentication method used during the EDHOC session, authentication is provided by digital signatures, or by Message Authentication Codes (MACs) computed from an ephemeral-static ECDH shared secret.

* Specify the proof-of-possession protocol(s) and how to select one, if several are available. Also specify which key types (e.g., symmetric/asymmetric) are supported by a specific proof-of-possession protocol: proof of possession is first achieved by RS when successfully processing EDHOC message_3 during the EDHOC session with C, through EDHOC algorithms and symmetric EDHOC session keys. Also, proof of possession is later achieved by C when receiving from RS: i) the optional EDHOC message_4 during the EDHOC session with RS, through EDHOC algorithms and symmetric EDHOC session keys; or ii) the first response protected with the OSCORE Security Context established after the EDHOC session with RS, through OSCORE algorithms and OSCORE symmetric keys derived from the completed EDHOC session.

* Specify a unique ace_profile identifier: coap_edhoc_oscore

* If introspection is supported, specify the communication and security protocol for introspection: HTTP/CoAP (+ TLS/DTLS/OSCORE)

* Specify the communication and security protocol for interactions between client and AS: HTTP/CoAP (+ TLS/DTLS/OSCORE)

* Specify if/how the authz-info endpoint is protected, including how error responses are protected: Not protected

* Optionally, define methods of token transport other than the authz-info endpoint: C can upload the access token when executing EDHOC with RS, by including the access token in the EAD_3 field of EDHOC message_3 (see Section 4.2).

Appendix C. CDDL Model

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

```
   ; ACE Profiles
   coap_edhoc_oscore = 4

   ; OAuth Parameters CBOR Mappings
   edhoc_info_param = 47

   ; CBOR Web Token (CWT) Claims
   edhoc_info_claim = 41

   ; CWT Confirmation Methods
   x5t = 6
   x5u = 7
   c5t = 8
   c5u = 9
   kcwt = 10
   kccs = 11
   x5chain = 24
   x5bag = 25
   c5c = 26
   c5b = 27

   ; EDHOC External Authorization Data
   ead_request_creation_hints_label = 2
   ead_credential_by_value_label = 15

   ; EDHOC Information
   session_id = 0
   methods = 1
   cipher_suites = 2
   message_4 = 3
   comb_req = 4
   uri_path = 5
   cred_types = 6
   id_cred_types = 7
   eads = 8
   initiator = 9
   responder = 10
   trust_anchors = 11

   ; EDHOC Trust Anchor Purposes
   edhoc_cred = 0

   ; EDHOC Trust Anchor Types
   c5t_ta_type = 22
   c5u_ta_type = 23
   x5t_ta_type = 34
   x5u_ta_type = 35
```

Figure 11: CDDL model

Appendix D. Document Updates

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

D.1. Version -09 to -10

* Fixed CDDL definition of the EDHOC_Information object.
* Removed excessive EDHOC_Information parameters: "max_msgsize", "coap_ct", "ep_id_types", and "transports".
* Removed definition of some IANA registries: "EDHOC Endpoint Identity Types" and "EDHOC Transports".
* Defined derivation of N_S, when combining this profile with application profiles of RFC 9594.
* Editorial fixes and improvements.

D.2. Version -08 to -09

* Parameter "cnf" explicitly forbidden in the Access Token Response.
* Clarification about content of "cnf" claim in the access token.
* RS must support the CoAP Uri-Path-Abbrev Option and its value abbreviating /.well-known/edhoc.
* Example of Request Creation Hints provided in EAD item.
* Examples showing content of new EAD items.

D.3. Version -07 to -08

* KUDOS is just an example of protocol to use for optional key update.
* message_4 is mandatory to support for an RS that supports the reverse message flow.
* Method in -workflow-and-params as example for coordinating the exchange of authentication credentials by value or reference.
* Revised initial set of EDHOC_Information parameters.
* Defined categorization of EDHOC_Information parameters.

* New EAD item for C to retrieve Request Creation Hints information from RS.

* New EAD item for requesting authentication credential by value.

* Means for the AS to achieve proof of possession of C’s private key.

* Editorial improvements.

D.4. Version -06 to -07

* Renamed id_ep_types as ep_id_types.

* Revised rules for the AS to assign session ID values.

* Added explicit validation of AUTH_CRED_C at AS.

* "edhoc_info" only in AS-to-C response for first token in a series.

* With a group-audience, "edhoc_info" refers to the whole audience.

* Defined parameters for the EDHOC_Information object:

  - Parameters moved here from draft-ietf-lake-app-profiles.

  - New parameter "trust_anchors".

* Access Token Request/Response messages must be encoded in CBOR.

* Explicit statement on admitted confirmation methods.

* First token in a series: the "cnf" claim uses the same confirmation method of the "req_cnf" request to /token.

* Revised examples in CBOR diagnostic notation.

* With a group-audience, the reverse message flow can use a roll call.

* Matching authentication credentials from ID_CRED_X and EAD item.

* Handling authentication credentials and EDHOC session that become invalid.

* Limited use of ACE Request Creation Hints when supporting the EDHOC reverse message flow.

* Changed CBOR abbreviations to not collide with existing codepoints.

* Updates and fixes in the IANA considerations.

* Updated references.

* Clarifications and editorial improvements.

D.5. Version -05 to -06

* The access token can be uploaded through EDHOC in EAD_3, EAD_2, or EAD_4.

* Ruled out the upload of the access token to the /authz-info endpoint over an unprotected channel.

* Defined an EDHOC EAD item for transporting a Session ID.

* Provided more details and added example of dynamic update of access rights.

* Defined in detail the use of the EDHOC reverse message flow.

* Provided details on access token invalidity.

* Revised examples with message exchanges.

* Clarifications and editorial improvements.

D.6. Version -04 to -05

* CBOR diagnostic notation uses placeholders from a CDDL model.

* Only CWTs are supported as access tokens in this profile.

* The alternative workflow of ACE is only mentioned as an example.

* Clarified that both the EDHOC forward and reverse message flows are possible.

* Consistent with the used EDHOC message flow, the access token can be in the EAD item of any EDHOC message.

* Explicit registration policies for the new IANA registry.

* Fixes in the IANA considerations.

* Editorial fixes and improvements.

D.7. Version -03 to -04

* Fixed column name and prefilling of the "EDHOC Information" registry.

* Added EDHOC_Information Parameters originally in draft-tiloca-lake-app-profiles-00.

* Removed the case of transporting access token in EAD_1

* Removed redundant normative text

* Clarifications of association between access token, session_id, EDHOC session and OSCORE security context

* Updated references.

* Editorial fixes and improvements.

D.8. Version -02 to -03

* Restructured presentation of content.

* Simplified description of the use of EDHOC_Information.

* Merged the concepts of EDHOC "session_id" and identifier of token series.

* Enabled the transport of the access token also in EDHOC EAD_3.

* Defined semantics of the newly defined CWT/JWT Confirmation Methods.

* Clarifications and editorial improvements.

D.9. Version -01 to -02

* Removed use of EDHOC_KeyUpdate.

* The Security Context is updated either by KUDOS or a rerun of EDHOC.

* The alternative workflow (AS token posting) is specified in separate draft.

* Fixed and updated examples.

* Editorial improvements.

D.10. Version -00 to -01

* Fixed semantics of the ead_value for transporting an Access Token in the EAD_1 field.

* Error handling aligned with EDHOC.

* Precise characterization of the EDHOC execution considered for EDHOC-KeyUpdate.

* Fixed message exchange examples.

* Added appendix with profile requirements.

* Updated references.

* Clarifications and editorial improvements.

Acknowledgments

The authors sincerely thank Christian Amsss and Carsten Bormann for their comments and feedback.

The parameter "trust_anchors" for specifying supported trust anchors builds on a proposal originally described in [I-D.serafin-lake-ta-hint].

This work was supported by the Sweden’s Innovation Agency VINNOVA within the EUREKA CELTIC-NEXT project CYPRESS; and by the H2020 project SIFIS-Home (Grant agreement 952652).

Authors’ Addresses

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

John Preu Mattsson
Ericsson
Email: john.mattsson@ericsson.com

Marco Tiloca
RISE
Email: marco.tiloca@ri.se

Rikard Hglund
RISE
Email: rikard.hoglund@ri.se