ACE Working Group   M. Tiloca
Internet-Draft   RISE AB
Intended status: Standards Track   L. Seitz
Expires: 28 April 2022   Combitech
                                                        F. Palombini
                                                         Ericsson AB
                                                       S. Echeverria
                                                            G. Lewis
                                                             CMU SEI
                                                     25 October 2021

**Notification of Revoked Access Tokens in the Authentication and
Authorization for Constrained Environments (ACE) Framework
draft-tiloca-ace-revoked-token-notification-06**

Abstract

   This document specifies a method of the Authentication and
   Authorization for Constrained Environments (ACE) framework, which
   allows an Authorization Server to notify Clients and Resource Servers
   (i.e., registered devices) about revoked Access Tokens. The method
   relies on resource observation for the Constrained Application
   Protocol (CoAP), with Clients and Resource Servers observing a Token
   Revocation List on the Authorization Server. Resulting unsolicited
   notifications of revoked Access Tokens complement alternative
   approaches such as token introspection, while not requiring
   additional endpoints on Clients and Resource Servers.

Discussion Venues

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

   Discussion of this document takes place on the Constrained RESTful
   Environments Working Group mailing list (ace@ietf.org), which is
   archived at https://mailarchive.ietf.org/arch/browse/ace/.

   Source for this draft and an issue tracker can be found at
   https://gitlab.com/crimson84/draft-tiloca-ace-revoked-token-
   notification.

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 28 April 2022.

Table of Contents

1. Introduction

   Authentication and Authorization for Constrained Environments (ACE)
   [I-D.ietf-ace-oauth-authz] is a framework that enforces access
   control on IoT devices acting as Resource Servers. In order to use
   ACE, both Clients and Resource Servers have to register with an
   Authorization Server and become a registered device. Once
   registered, a Client can send a request to the Authorization Server,
   to obtain an Access Token for a Resource Server. For a Client to
   access the Resource Server, the Client must present the issued Access
   Token at the Resource Server, which then validates it before storing
   it (see Section 5.10.1.1 of [I-D.ietf-ace-oauth-authz]).

   Even though Access Tokens have expiration times, there are
   circumstances by which an Access Token may need to be revoked before
   its expiration time, such as: (1) a registered device has been
   compromised, or is suspected of being compromised; (2) a registered
   device is decommissioned; (3) there has been a change in the ACE
   profile for a registered device; (4) there has been a change in
   access policies for a registered device; and (5) there has been a
   change in the outcome of policy evaluation for a registered device
   (e.g., if policy assessment depends on dynamic conditions in the
   execution environment, the user context, or the resource
   utilization).

   As discussed in Section 6.1 of [I-D.ietf-ace-oauth-authz], only
   client-initiated revocation is currently specified [RFC7009] for
   OAuth 2.0 [RFC6749], based on the assumption that Access Tokens in
   OAuth are issued with a relatively short lifetime. However, this is
   not expected to be the case for constrained, intermittently connected
   devices, that need Access Tokens with relatively long lifetimes.

This document specifies a method for allowing registered devices to access and possibly subscribe to a Token Revocation List (TRL) resource on the Authorization Server, in order to get an updated list of revoked, but yet not expired, pertaining Access Tokens. In particular, registered devices can subscribe to the TRL at the Authorization Server by using resource observation [RFC7641] for the Constrained Application Protocol (CoAP) [RFC7252].

Unlike in the case of token introspection (see Section 5.9 of [I-D.ietf-ace-oauth-authz]), a registered device does not provide an owned Access Token to the Authorization Server for inquiring about its current state. Instead, registered devices simply obtain an updated list of revoked, but yet not expired, pertaining Access Tokens, as efficiently identified by corresponding hash values.

In fact, the benefits of this method are that it complements token introspection, and it does not require any additional endpoints on the registered devices. The only additional requirements for registered devices are a request/response interaction with the Authorization Server to access and possibly subscribe to the TRL (see Section 2), and the lightweight computation of hash values as Token identifiers (see Section 3).

1.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

Readers are expected to be familiar with the terms and concepts described in the ACE framework for Authentication and Authorization [I-D.ietf-ace-oauth-authz], as well as with terms and concepts related to CBOR Web Tokens (CWTs) [RFC8392], and JSON Web Tokens (JWTs) [RFC7519]. The terminology for entities in the considered architecture is defined in OAuth 2.0 [RFC6749]. In particular, this includes Client, Resource Server, and Authorization Server.

Readers are also expected to be familiar with the terms and concepts related to CBOR [RFC8949], JSON [RFC8259], the CoAP protocol [RFC7252], CoAP Observe [RFC7641], and the use of hash functions to name objects as defined in [RFC6920].

Note that, unless otherwise indicated, the term "endpoint" is used
here following its OAuth definition, aimed at denoting resources such
as /token and /introspect at the Authorization Server, and /authz-
info at the Resource Server. This document does not use the CoAP
definition of "endpoint", which is "An entity participating in the
CoAP protocol."

This specification also refers to the following terminology.

* Token hash: identifier of an Access Token, in binary format
   encoding. The token hash has no relation to other possibly used
   token identifiers, such as the "cti" (CWT ID) claim of CBOR Web
   Tokens (CWTs) [RFC8392].

* Token Revocation List (TRL): a collection of token hashes, in
   which the corresponding Access Tokens have been revoked but are
   not expired yet.

* TRL resource: a resource on the Authorization Server, with a TRL
   as its representation.

* TRL endpoint: an endpoint at the Authorization Server associated
   to the TRL resource. The default name of the TRL endpoint in a
   url-path is '/revoke/trl'. Implementations are not required to
   use this name, and can define their own instead.

* Registered device: a device registered at the Authorization
   Server, i.e., as a Client, or a Resource Server, or both. A
   registered device acts as caller of the TRL endpoint.

* Administrator: entity authorized to get full access to the TRL at
   the Authorization Server, and acting as caller of the TRL
   endpoint. An administrator is not necessarily a registered device
   as defined above, i.e., a Client requesting Access Tokens or a
   Resource Server consuming Access Tokens. How the administrator
   authorization is established and verified is out of the scope of
   this specification.

* Pertaining Access Token:

   - With reference to an administrator, an Access Token issued by
      the Authorization Server.

   - With reference to a registered device, an Access Token intended
      to be owned by that device. An Access Token pertains to a
      Client if the Authorization Server has issued the Access Token
      and provided it to that Client. An Access Token pertains to a
      Resource Server if the Authorization Server has issued the
      Access Token to be consumed by that Resource Server.

2. Protocol Overview

   This protocol defines how a CoAP-based Authorization Server informs
   Clients and Resource Servers, i.e., registered devices, about
   pertaining revoked Access Tokens. How the relationship between the
   registered device and the Authorization Server is established is out
   of the scope of this specification.

   At a high level, the steps of this protocol are as follows.

   * Upon startup, the Authorization Server creates a single TRL
      resource. At any point in time, the TRL resource represents the
      list of all revoked Access Tokens issued by the Authorization
      Server that are not expired yet.

   * When a device registers at the Authorization Server, it receives
      the url-path to the TRL resource.

      After the registration procedure is finished, the registered
      device sends an Observation Request to that TRL resource as
      described in [RFC7641], i.e., a GET request with an Observe option
      set to 0 (register). By doing so, the registered device
      effectively subscribes to the TRL resource, as interested to
      receive notifications about its update. Upon receiving the
      request, the Authorization Server adds the registered device to
      the list of observers of the TRL resource.

      At any time, the registered device can send a GET request to the
      TRL endpoint. When doing so, it can request for: the current list
      of pertaining revoked Access Tokens (see Section 5.1); or the most
      recent TRL updates occurred over the list of pertaining revoked
      Access Tokens (see Section 5.2). In either case, the registered
      device may especially rely on an Observation Request for
      subscribing to the TRL resource as discussed above.

   * When an Access Token is revoked, the Authorization Server adds the
      corresponding token hash to the TRL. Also, when a revoked Access
      Token eventually expires, the Authorization Server removes the
      corresponding token hash from the TRL.

In either case, after updating the TRL, the Authorization Server sends Observe Notifications as per [RFC7641]. That is, an Observe Notification is sent to each registered device subscribed to the TRL resource and to which the Access Token pertains.

Depending on the specific subscription established through the observation request, the notification provides the current updated list of revoked Access Tokens in the portion of the TRL pertaining to that device (see Section 5.1), or rather the most recent TRL updates occurred over that list of pertaining revoked Access Tokens (see Section 5.2).

Further Observe Notifications may be sent, consistently with ongoing additional observations of the TRL resource.

* An administrator can access and subscribe to the TRL like a registered device, while getting the full updated representation of the TRL.

Figure 1 shows a high-level overview of the service provided by this protocol. In particular, it shows the Observe Notifications sent by the Authorization Server to one administrator and four registered devices, upon revocation of the issued Access Tokens t1, t2 and t3, with token hash th1, th2 and th3, respectively. Each dotted line associated to a pair of registered devices indicates the Access Token that they both own.
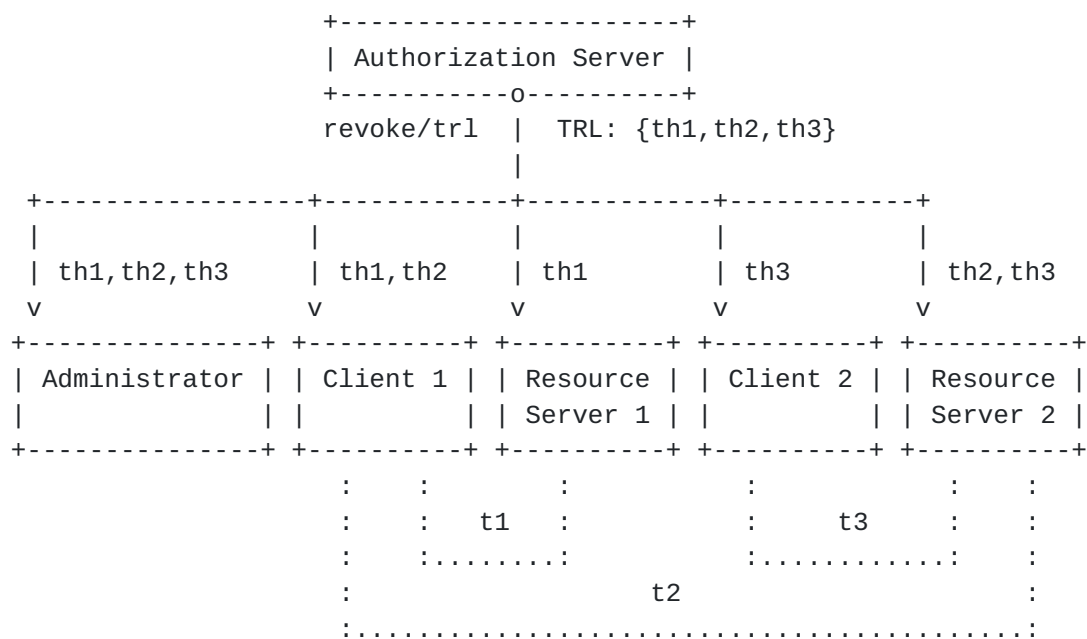
```
                     +----------------------+
                     | Authorization Server |
                     +-----------O----------+
                     revoke/trl  |  TRL: {th1,th2,th3}
                                 |
  +-----------------+-----------+-----------+------------+
  |                 |           |           |            |
  | th1,th2,th3     | th1,th2   | th1       | th3        | th2,th3
  v                 v           v           v            v
 +---------------+ +----------+ +----------+ +----------+ +----------+
 | Administrator | | Client 1 | | Resource | | Client 2 | | Resource |
 |               | |          | | Server 1 | |          | | Server 2 |
 +---------------+ +----------+ +----------+ +----------+ +----------+
                  :    :           :            :              :    :
                  :    :   t1      :            :      t3      :    :
                  :    :........:               :............:     :
                  :                   t2                           :
                  :.............................................:

                       Figure 1: Protocol Overview
```

Section 8 provides examples of the protocol flow and message exchange between the Authorization Server and a registered device.

3. Token Hash

The token hash of an Access Token is computed as follows.

1. The Authorization Server defines ENCODED_TOKEN, as the content of the 'access_token' parameter in the Authorization Server response (see Section 5.8.2 of [I-D.ietf-ace-oauth-authz]), where the Access Token was included and provided to the requesting Client.

    Note that the content of the 'access_token' parameter is either:

    * A CBOR byte string, if the Access Token was transported using CBOR. With reference to the example in Figure 2, and assuming the string's length in bytes to be 119 (i.e., 0x77 in hexadecimal), then ENCODED_TOKEN takes the bytes {0x58 0x77 0xd0 0x83 0x44 0xa1 ...}, i.e., the raw content of the parameter 'access_token'.

    * A text string, if the Access Token was transported using JSON. With reference to the example in Figure 3, ENCODED_TOKEN takes "2YotnFZFEjr1zCsicMWpAA", i.e., the raw content of the parameter 'access_token'.

2. The Authorization Server defines HASH_INPUT as follows.

    * If CBOR was used to transport the Access Token (as a CWT or JWT), HASH_INPUT takes the same value of ENCODED_TOKEN.

    * If JSON was used to transport the Access Token (as a CWT or JWT), HASH_INPUT takes the serialization of ENCODED_TOKEN.

       In either case, HASH_INPUT results in the binary representation of the content of the 'access_token' parameter from the Authorization Server response.

3. The Authorization Server generates a hash value of HASH_INPUT as per Section 6 of [RFC6920]. The resulting output in binary format is used as the token hash. Note that the used binary format embeds the identifier of the used hash function, in the first byte of the computed token hash.

    The specifically used hash function MUST be collision-resistant on byte-strings, and MUST be selected from the "Named Information Hash Algorithm" Registry [Named.Information.Hash.Algorithm].

        The Authorization Server specifies the used hash function to
        registered devices during their registration procedure (see
        [Section 6]).

    2.01 Created
    Content-Format: application/ace+cbor
    Max-Age: 85800
    Payload:
    {
       access_token : h'd08344a1...'
       (remainder of the Access Token omitted for brevity)
       token_type : pop,
       expires_in : 86400,
       profile   : coap_dtls,
       (remainder of the response omitted for brevity)
    }

        Figure 2: Example of Authorization Server response using CBOR

    HTTP/1.1 200 OK
    Content-Type: application/json
    Cache-Control: no-store
    Pragma: no-cache
    Payload:
    {
       "access_token" : "2YotnFZFEjr1zCsicMWpAA"
       (remainder of the Access Token omitted for brevity)
       "token_type" : "pop",
       "expires_in" : 86400,
       "profile"   : "coap_dtls",
       (remainder of the response omitted for brevity)
    }

        Figure 3: Example of Authorization Server response using JSON

[4]. The TRL Resource

   Upon startup, the Authorization Server creates a single TRL resource,
   encoded as a CBOR array.

   Each element of the array is a CBOR byte string, with value the token
   hash of an Access Token. The order of the token hashes in the CBOR
   array is irrelevant, and the CBOR array MUST be treated as a set in
   which the order has no significant meaning.

   The TRL is initialized as empty, i.e., the initial content of the TRL
   resource representation MUST be an empty CBOR array.

[4.1](). Update of the TRL Resource

The Authorization Server updates the TRL in the following two cases.

*   When a non-expired Access Token is revoked, the token hash of the
    Access Token is added to the TRL resource representation. That
    is, a CBOR byte string with the token hash as its value is added
    to the CBOR array used as TRL resource representation.

*   When a revoked Access Token expires, the token hash of the Access
    Token is removed from the TRL resource representation. That is,
    the CBOR byte string with the token hash as its value is removed
    from the CBOR array used as TRL resource representation.

[5](). The TRL Endpoint

Consistent with Section 6.5 of [[I-D.ietf-ace-oauth-authz]()], all
communications between a caller of the TRL endpoint and the
Authorization Server MUST be encrypted, as well as integrity and
replay protected. Furthermore, responses from the Authorization
Server to the caller MUST be bound to the caller's request.

The Authorization Server MUST implement measures to prevent access to
the TRL endpoint by entities other than registered devices and
authorized administrators.

The TRL endpoint supports only the GET method, and allows two types
of query of the TRL.

*   Full query: the Authorization Server returns the token hashes of
    the revoked Access Tokens currently in the TRL and pertaining to
    the requester. The Authorization Server MUST support this type of
    query. The processing of a full query and the related response
    format are defined in [Section 5.1]().

*   Diff query: the Authorization Server returns a list of diff
    entries. Each diff entry is related to one of the most recent
    updates, in the portion of the TRL pertaining to the requester.
    The Authorization Server MAY support this type of query.

    The entry associated to one of such updates contains a list of
    token hashes, such that: i) the corresponding revoked Access
    Tokens pertain to the requester; and ii) they were added to or
    removed from the TRL at that update. The processing of a diff
    query and the related response format are defined in [Section 5.2]().

The TRL endpoint allows the following query parameter in a GET
request.

   * 'pmax': if included, it follows the semantics defined in
      Section 3.2.2 of [I-D.ietf-core-conditional-attributes]. This
      query parameter is relevant only in case the GET request is
      specifically an Observation Request, i.e., if it includes the CoAP
      Observe option set to 0 (register). In such a case, this
      parameter indicates the maximum time, in seconds, between two
      consecutive notifications for the observation in question,
      regardless whether the TRL resource has changed or not.

      If the Observation Request does not include the 'pmax' parameter,
      the maximum time to consider is up to the Authorization Server.
      If the Observation Request includes the 'pmax' parameter, its
      value MUST be greater than zero, otherwise the Authorization
      Server MUST return a 4.00 (Bad Request) response.

      If the GET request is not an Observation Request, the
      Authorization Server MUST ignore the 'pmax' parameter, in case
      this is included.

   * 'diff': if included, it indicates to perform a diff query of the
      TRL. Its value MUST be either:

      - the integer 0, indicating that a (notification) response should
         include as many diff entries as the Authorization Server can
         provide in the response; or

      - a positive integer greater than 0, indicating the maximum
         number of diff entries that a (notification) response should
         include.

5.1. Full Query of the TRL

   In order to produce a (notification) response to a GET request asking
   for a full query of the TRL, the Authorization Server performs the
   following actions.

   1. From the current TRL resource representation, the Authorization
       Server builds a set HASHES, such that:

       * If the requester is a registered device, HASHES specifies the
          token hashes of the Access Tokens pertaining to that
          registered device. The Authorization Server can use the
          authenticated identity of the registered device to perform the
          necessary filtering on the TRL resource representation.

       * If the requester is an administrator, HASHES specifies all the
          token hashes in the current TRL resource representation.

2. The Authorization Server sends a 2.05 (Content) Response to the
    requester, with a CBOR array 'full' as payload. Each element of
    the array specifies one of the token hashes from the set HASHES,
    encoded as a CBOR byte string.

    The order of the token hashes in the CBOR array is irrelevant,
    i.e., the CBOR array MUST be treated as a set in which the order
    has no significant meaning.

The CDDL definition [RFC8610] of the CBOR array 'full' specified as
payload in the response from the Authorization Server is provided
below.

```
token-hash = bytes
full = [* token-hash]
```

        Figure 4: CDDL definition of the response payload following a
                   Full Query request to the TRL endpoint

5.2. Diff Query of the TRL

In order to produce a (notification) response to a GET request asking
for a diff query of the TRL, the Authorization Server performs the
following actions.

1. The Authorization Server defines the positive integer NUM. If
    the value N specified in the query parameter 'diff' of the GET
    request is equal to 0 or greater than a pre-defined positive
    integer N_MAX, then NUM takes the value of N_MAX. Otherwise, NUM
    takes N.

2. The Authorization Server prepares U = min(NUM, SIZE) diff
    entries, where SIZE <= N_MAX is the number of TRL updates
    pertaining to the requester and currently stored at the
    Authorization Server. That is, the diff entries are related to
    the U most recent TRL updates pertaining to the requester. In
    particular, the first entry refers to the most recent of such
    updates, the second entry refers to the second from last of such
    updates, and so on.

    Each diff entry is a CBOR array 'diff-entry', which includes the
    following two elements.

    * The first element is a CBOR array 'removed'. Each element of
       the array is a CBOR byte string, with value the token hash of
       an Access Token such that: it pertained to the requester; and
       it was removed from the TRL during the update associated to
       the diff entry.

   * The second element is a CBOR array 'added'. Each element of
      the array is a CBOR byte string, with value the token hash of
      an Access Token such that: it pertains to the requester; and
      it was added to the TRL during the update associated to the
      diff entry.

   The order of the token hashes in the CBOR arrays 'removed' and
   'added' is irrelevant. That is, the CBOR arrays 'removed' and
   'added' MUST be treated as a set in which the order of elements
   has no significant meaning.

3. The Authorization Server prepares a 2.05 (Content) Response for
    the requester, with a CBOR array 'diff' of U elements as payload.
    Each element of the CBOR array 'diff' specifies one of the CBOR
    arrays 'diff-entry' prepared at step 2 as diff entries.

    Within the CBOR array 'diff', the CBOR arrays 'diff-entry' MUST
    be sorted to reflect the corresponding updates to the TRL in
    reverse chronological order. That is, the first 'diff-entry'
    element of 'diff' relates to the most recent update to the
    portion of the TRL pertaining to the requester. The second
    'diff-entry' element of 'diff' relates to the second from last
    most recent update to that portion, and so on.

The CDDL definition [RFC8610] of the CBOR array 'diff' specified as
payload in the response from the Authorization Server is provided
below.

```
token-hash = bytes
trl-patch = [* token-hash]
diff-entry = [removed: trl-patch, added: trl-patch]
diff = [* diff-entry]
```

       Figure 5: CDDL definition of the response payload following a
                   Diff Query request to the TRL endpoint

If the Authorization Server supports diff queries:

* The Authorization Server MUST return a 4.00 (Bad Request) response
   in case the 'diff' parameter specifies a value other than 0 or
   than a positive integer.

* The Authorization Server MUST keep track of N_MAX most recent
   updates to the portion of the TRL that pertains to each caller of
   the TRL endpoint. The particular method to achieve this is
   implementation-specific.

   * When SIZE is equal to N_MAX, and a new TRL update occurs as
      pertaining to a registered device, the Authorization Server MUST
      first delete the oldest stored update for that device, before
      storing this latest update as the most recent one for that device.

   * The Authorization Server SHOULD provide registered devices and
      administrators with the value of N_MAX, upon their registration
      (see Section 6).

   If the Authorization Server does not support diff queries, it
   proceeds as when processing a full query (see Section 5.1).

   Appendix A discusses how the diff query of the TRL is in fact a usage
   example of the Series Transfer Pattern defined in
   [I-D.bormann-t2trg-stp].

   Appendix B discusses how the diff query of the TRL can be further
   improved by using the "Cursor" pattern defined in Section 3.3 of
   [I-D.bormann-t2trg-stp].

6. Upon Registration

   During the registration process at the Authorization Server, an
   administrator or a registered device receives the following
   information as part of the registration response.

   * The url-path to the TRL endpoint at the Authorization Server.

   * The hash function used to compute token hashes. This is specified
      as an integer or a text string, taking value from the "ID" or
      "Hash Name String" column of the "Named Information Hash
      Algorithm" Registry [Named.Information.Hash.Algorithm],
      respectively.

   * Optionally, a positive integer N_MAX, if the Authorization Server
      supports diff queries of the TRL resource (see Section 5.2).

   After the registration procedure is finished, the administrator or
   registered device performs a GET request to the TRL resource,
   including the CoAP Observe option set to 0 (register), in order to
   start an observation of the TRL resource at the Authorization Server,
   as per Section 3.1 of [RFC7641]. The GET request can express the
   wish for a full query (see Section 5.1) or a diff query (see
   Section 5.2) of the TRL.

   In case the request is successfully processed, The Authorization
   Server replies using the CoAP response code 2.05 (Content) and
   including the CoAP Observe option in the response. The payload of

the response is formatted as defined in Section 5.1 or in
Section 5.2, in case the GET request was for a full query or a diff
query of the TRL, respectively.

Further details about the registration process at the Authorization
Server are out of scope for this specification. Note that the
registration process is also out of the scope of the ACE framework
for Authentication and Authorization (see Section 5.5 of
[I-D.ietf-ace-oauth-authz]).

7. Notification of Revoked Tokens

When the TRL is updated (see Section 4.1), the Authorization Server
sends Observe Notifications to the observers of the TRL resource.
Observe Notifications are sent as per Section 4.2 of [RFC7641].

If the 'pmax' query parameter was specified in the Observation
Request starting an observation (see Section 5), the Authorization
Server might accordingly send additional Observe Notifications to the
associated observer. That is, the Authorization Server ensures that
no more than pmax seconds elapse between two consecutive
notifications sent to that observer, regardless whether the TRL
resource has changed or not. If the 'pmax' query parameter was not
specified in the Observation Request, a possible maximum time to
consider is up to the Authorization Server.

The payload of each Observe Notification is formatted as defined in
Section 5.1 or in Section 5.2, in case the original Observation
Request was for a full query or a diff query of the TRL,
respectively.

Furthermore, an administrator or a registered device can send
additional GET requests to the TRL endpoint at any time, in order to
retrieve the token hashes of the pertaining revoked Access Tokens.
When doing so, the caller of the TRL endpoint can perform a full
query (see Section 5.1) or a diff query (see Section 5.2).

8. Interaction Examples

This section provides examples of interactions between a Resource
Server RS as registered device and an Authorization Server AS. The
Authorization Server supports both full query and diff query of the
TRL, as defined in Section 5.1 and Section 5.2, respectively.

The details of the registration process are omitted, but it is
assumed that the Resource Server sends an unspecified payload to the
Authorization Server, which replies with a 2.01 (Created) response.

The payload of the registration response is a CBOR map, which
includes the following entries:

* a "trl" parameter, specifying the path of the TRL resource;

* a "trl_hash" parameter, specifying the hash function used to
   computed token hashes as defined in Section 3;

* an "n_max" parameter, specifying the value of N_MAX, i.e., the
   maximum number of TRL updates pertaining to each registered device
   that the Authorization Server retains for that device (see
   Section 5.2);

* possible further parameters related to the registration process.

Furthermore, 'h(x)' refers to the hash function used to compute the
token hashes, as defined in Section 3 of this specification and
according to [RFC6920]. Assuming the usage of CWTs transported in
CBOR, 'bstr.h(t1)' and 'bstr.h(t2)' denote the byte-string
representations of the token hashes for the Access Tokens t1 and t2,
respectively.

8.1. Full Query with Observation

Figure 6 shows an interaction example considering a CoAP observation
and a full query of the TRL.

```
                RS                                    AS
                |                                     |
                | Registration: POST                  |
                +------------------------------------->|
                |                                     |
                |<-------------------------------------+
                |           2.01 CREATED              |
                |             Payload: {              |
                |                ...                  |
                |                "trl" = "revoke/trl", |
                |                "trl_hash" = "sha-256", |
                |                "n_max" = 10         |
                |             }                       |
                |                                     |
                | GET Observe: 0                      |
                |   coap://example.as.com/revoke/trl/ |
                +------------------------------------->|
                |                                     |
                |<-------------------------------------+
                |           2.05 CONTENT Observe: 42 |
                |             Payload: []             |
```

```
           |                 .                   |
           |                 .                   |
           |                 .                   |
           |                                     |
           |    (Access Tokens t1 and t2 issued  |
           |   and successfully submitted to RS) |
           |                 .                   |
           |                 .                   |
           |                 .                   |
           |                                     |
           |                                     |
           |     (Access Token t1 is revoked)    |
           |                                     |
           |<------------------------------------+
           |             2.05 CONTENT Observe: 53 |
           |               Payload: [bstr.h(t1)]  |
           |                 .                   |
           |                 .                   |
           |                 .                   |
           |                                     |
           |     (Access Token t2 is revoked)    |
           |                                     |
           |<------------------------------------+
           |             2.05 CONTENT Observe: 64 |
           |               Payload: [bstr.h(t1),  |
           |                         bstr.h(t2)]  |
           |                 .                   |
           |                 .                   |
           |                 .                   |
           |                                     |
           |       (Access Token t1 expires)     |
           |                                     |
           |<------------------------------------+
           |             2.05 CONTENT Observe: 75 |
           |               Payload: [bstr.h(t2)]  |
           |                 .                   |
           |                 .                   |
           |                 .                   |
           |                                     |
           |       (Access Token t2 expires)     |
           |                                     |
           |<------------------------------------+
           |             2.05 CONTENT Observe: 86 |
           |               Payload: []            |
           |                                     |
```
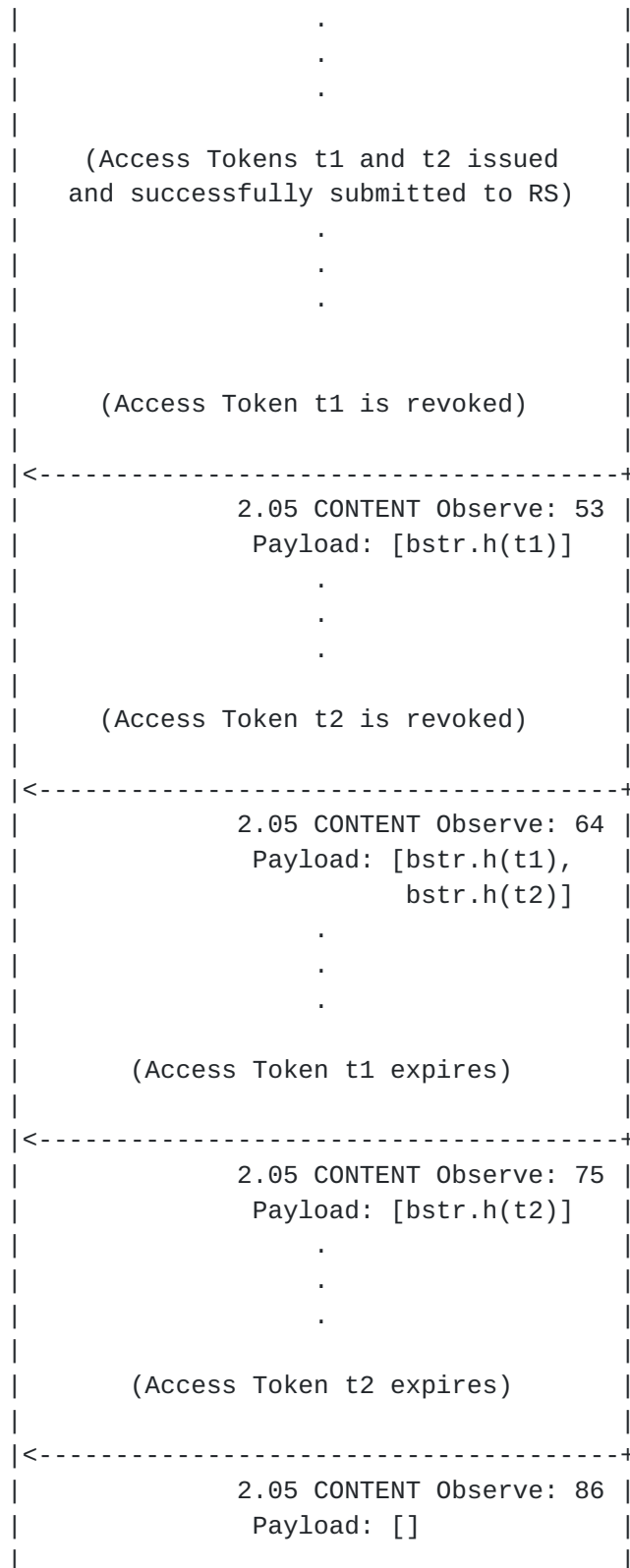
Figure 6: Interaction for Full Query with Observation

8.2. Diff Query with Observation

   Figure 7 shows an interaction example considering a CoAP observation
   and a diff query of the TRL.

   The Resource Server indicates N=3 as value of the query parameter
   "diff", i.e., as the maximum number of diff entries to be specified
   in a response from the Authorization Server.

```
                RS                                           AS
                |                                             |
                | Registration: POST                          |
                +-------------------------------------------->|
                |                                             |
                |<--------------------------------------------+
                |                  2.01 CREATED               |
                |                    Payload: {               |
                |                       ...                   |
                |                       "trl" = "revoke/trl", |
                |                       "trl_hash" = "sha-256", |
                |                       "n_max" = 10          |
                |                    }                        |
                |                                             |
                | GET Observe: 0                              |
                |   coap://example.as.com/revoke/trl?diff=3   |
                +-------------------------------------------->|
                |                                             |
                |<--------------------------------------------+
                |                       2.05 CONTENT Observe: 42 |
                |                         Payload: []         |
                |                            .                |
                |                            .                |
                |                            .                |
                |                                             |
                |         (Access Tokens t1 and t2 issued     |
                |         and successfully submitted to RS)   |
                |                            .                |
                |                            .                |
                |                            .                |
                |                                             |
                |         (Access Token t1 is revoked)        |
                |                                             |
                |<--------------------------------------------+
                |             2.05 CONTENT Observe: 53        |
                |               Payload: [                    |
                |                       [ [], [bstr.h(t1)] ]  |
                |                       ]                     |
                |                                             |
```

```
                      |                                      |
                      |                    .                 |
                      |                    .                 |
                      |                    .                 |
                      |                                      |
                      |         (Access Token t2 is revoked) |
                      |                                      |
                      |<-------------------------------------+
                      |         2.05 CONTENT Observe: 64     |
                      |           Payload: [                 |
                      |                     [ [], [bstr.h(t2)] ], |
                      |                     [ [], [bstr.h(t1)] ]  |
                      |                   ]                  |
                      |                    .                 |
                      |                    .                 |
                      |                    .                 |
                      |                                      |
                      |         (Access Token t1 expires)    |
                      |                                      |
                      |<-------------------------------------+
                      |         2.05 CONTENT Observe: 75     |
                      |           Payload: [                 |
                      |                     [ [bstr.h(t1)], [] ], |
                      |                     [ [], [bstr.h(t2)] ], |
                      |                     [ [], [bstr.h(t1)] ]  |
                      |                   ]                  |
                      |                    .                 |
                      |                    .                 |
                      |                    .                 |
                      |                                      |
                      |         (Access Token t2 expires)    |
                      |                                      |
                      |<-------------------------------------+
                      |         2.05 CONTENT Observe: 86     |
                      |           Payload: [                 |
                      |                     [ [bstr.h(t2)], [] ], |
                      |                     [ [bstr.h(t1)], [] ], |
                      |                     [ [], [bstr.h(t2)] ]  |
                      |                   ]                  |
                      |                                      |
```
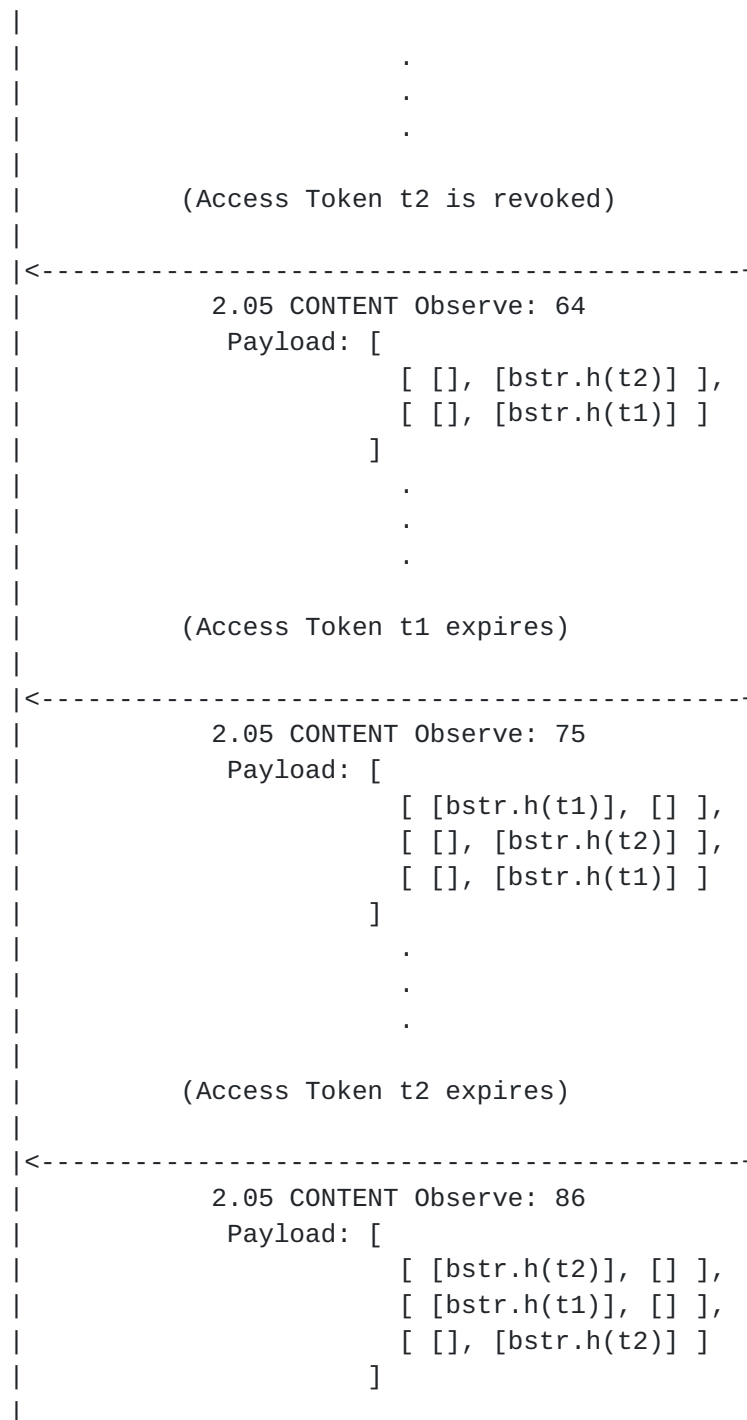
        Figure 7: Interaction for Diff Query with Observation

8.3. Full Query with Observation and Additional Diff Query

   Figure 8 shows an interaction example considering a CoAP observation
   and a full query of the TRL.

The example also considers one of the notifications from the
Authorization Server to get lost in transmission, and thus not
reaching the Resource Server.

When this happens, and after a waiting time defined by the
application has elapsed, the Resource Server sends a GET request with
no observation to the Authorization Server, to perform a diff query
of the TRL. The Resource Server indicates N=8 as value of the query
parameter "diff", i.e., as the maximum number of diff entries to be
specified in a response from the Authorization Server.

```
             RS                                             AS
             |                                              |
             | Registration: POST                          |
             +--------------------------------------------->|
             |                                              |
             |<---------------------------------------------+
             |                   2.01 CREATED               |
             |                     Payload: {               |
             |                           ...                |
             |                           "trl" = "revoke/trl",   |
             |                           "trl_hash" = "sha-256", |
             |                           "n_max" = 10       |
             |                         }                    |
             |                                              |
             | GET Observe: 0                               |
             |   coap://example.as.com/revoke/trl/          |
             +--------------------------------------------->|
             |                                              |
             |<---------------------------------------------+
             |                         2.05 CONTENT Observe: 42 |
             |                           Payload: []        |
             |                             .                |
             |                             .                |
             |                             .                |
             |                                              |
             |        (Access Tokens t1 and t2 issued       |
             |         and successfully submitted to RS)    |
             |                             .                |
             |                             .                |
             |                             .                |
             |                                              |
             |         (Access Token t1 is revoked)         |
             |                                              |
             |<---------------------------------------------+
             |                         2.05 CONTENT Observe: 53 |
             |                           Payload: [bstr.h(t1)]  |
             |                                              |
```

```
              |                   .                   |
              |                   .                   |
              |                   .                   |
              |                                       |
              |       (Access Token t2 is revoked)    |
              |                                       |
              |<--------------------------------------+
              |                   2.05 CONTENT Observe: 64 |
              |                   Payload: [bstr.h(t1),    |
              |                                 bstr.h(t2)] |
              |                   .                   |
              |                   .                   |
              |                   .                   |
              |                                       |
              |       (Access Token t1 expires)       |
              |                                       |
              |<--------------------------------------+
              |                   2.05 CONTENT Observe: 75 |
              |                   Payload: [bstr.h(t2)]     |
              |                   .                   |
              |                   .                   |
              |                   .                   |
              |                                       |
              |       (Access Token t2 expires)       |
              |                                       |
              |      X<--------------------------------+
              |                   2.05 CONTENT Observe: 86 |
              |                   Payload: []         |
              |                   .                   |
              |                   .                   |
              |                   .                   |
              |                                       |
              |       (Enough time has passed since   |
              |        the latest received notification) |
              |                                       |
              | GET                                   |
              |  coap://example.as.com/revoke/trl?diff=8   |
              +-------------------------------------->|
              |                                       |
              |<--------------------------------------+
              |            2.05 CONTENT               |
              |              Payload: [               |
              |                    [ [bstr.h(t2)], [] ], |
              |                    [ [bstr.h(t1)], [] ], |
              |                    [ [], [bstr.h(t2)] ], |
              |                    [ [], [bstr.h(t1)] ]  |
              |                    ]                  |
              |                                       |
```

   Figure 8: Interaction for Full Query with Observation and Diff Query

9. Security Considerations

   Security considerations are inherited from the ACE framework for
   Authentication and Authorization [I-D.ietf-ace-oauth-authz], from
   [RFC8392] as to the usage of CWTs, from [RFC7519] as to the usage of
   JWTs, from [RFC7641] as to the usage of CoAP Observe, and from
   [RFC6920] with regard to resource naming through hashes. The
   following considerations also apply.

   The Authorization Server MUST ensure that each registered device can
   access and retrieve only its pertaining portion of the TRL. To this
   end, the Authorization Server can perform the required filtering
   based on the authenticated identity of the registered device, i.e., a
   (non-public) identifier that the Authorization Server can securely
   relate to the registered device and the secure association they use
   to communicate.

   Disclosing any information about revoked Access Tokens to entities
   other than the intended registered devices may result in privacy
   concerns. Therefore, the Authorization Server MUST ensure that,
   other than registered devices accessing their own pertaining portion
   of the TRL, only authorized and authenticated administrators can
   retrieve the full TRL. To this end, the Authorization Server may
   rely on an access control list or similar.

   If a registered device has many non-expired Access Tokens associated
   to itself that are revoked, the pertaining portion of the TRL could
   grow to a size bigger than what the registered device is prepared to
   handle upon reception, especially if relying on a full query of the
   TRL resource (see Section 5.1). This could be exploited by attackers
   to negatively affect the behavior of a registered device. Issuing
   Access Tokens with not too long expiration time could help reduce the
   size of a TRL, but an Authorization Server SHOULD take measures to
   limit this size.

   Most of the communication about revoked Access Tokens presented in
   this specification relies on CoAP Observe Notifications sent from the
   Authorization Server to a registered device. The suppression of
   those notifications by an external attacker that has access to the
   network would prevent registered devices from ever knowing that their
   pertaining Access Tokens have been revoked. To avoid this, a
   registered device SHOULD NOT rely solely on the CoAP Observe
   notifications. In particular, a registered device SHOULD also
   regularly poll the Authorization Server for the most current
   information about revoked Access Tokens, by sending GET requests to
   the TRL endpoint according to a related application policy.

10. IANA Considerations

   This document has the following actions for IANA.

10.1. Media Type Registrations

   IANA is asked to register the 'application/ace-trl+cbor' media type
   for messages of the protocols defined in this document encoded in
   CBOR. This registration follows the procedures specified in
   [RFC6838].

   Type name: application

   Subtype name: ace-trl+cbor

   Required parameters: N/A

   Optional parameters: N/A

   Encoding considerations: Must be encoded as CBOR map containing the
   protocol parameters defined in [this document].

   Security considerations: See Section 9 of this document.

   Interoperability considerations: N/A

   Published specification: [this document]

   Applications that use this media type: The type is used by
   Authorization Servers, Clients and Resource Servers that support the
   notification of revoked Access Tokens, according to a Token
   Revocation List maintained by the Authorization Server as specified
   in [this document].

   Fragment identifier considerations: N/A

   Additional information: N/A

   Person & email address to contact for further information:
   <iesg@ietf.org>

   Intended usage: COMMON

   Restrictions on usage: None

   Author: Marco Tiloca <marco.tiloca@ri.se>

   Change controller: IESG

10.2. CoAP Content-Formats Registry

IANA is asked to add the following entry to the "CoAP Content-Formats" registry within the "CoRE Parameters" registry group.

Media Type: application/ace-trl+cbor

Encoding: -

ID: TBD

Reference: [this document]

10.3. Token Revocation List Registry

This specification establishes the "Token Revocation List" IANA registry. The registry has been created to use the "Expert Review" registration procedure [RFC8126]. Expert review guidelines are provided in Section 10.4. It should be noted that, in addition to the expert review, some portions of the registry require a specification, potentially a Standards Track RFC, to be supplied as well.

The columns of this registry are:

* Name: This is a descriptive name that enables easier reference to the item. The name MUST be unique. It is not used in the encoding.

* CBOR Value: This is the value used as CBOR abbreviation of the item. These values MUST be unique. The value can be a positive integer or a negative integer. Different ranges of values use different registration policies [RFC8126]. Integer values from -256 to 255 are designated as Standards Action. Integer values from -65536 to -257 and from 256 to 65535 are designated as Specification Required. Integer values greater than 65535 are designated as Expert Review. Integer values less than -65536 are marked as Private Use.

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

* Reference: This contains a pointer to the public specification for the item.

This registry has been initially populated with the values from Figure 9 in Appendix B.4.4.

10.4. Expert Review Instructions

   The IANA registry established in this document is defined as expert
   review. 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:

   * 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 track range of point
      assignment. Specifications should exist for specification
      required ranges, but early assignment before a specification is
      available is considered to be permissible. Specifications are
      needed for the first-come, first-serve range if they are expected
      to be used outside of closed environments in an interoperable way.
      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. The fact that there is a range for
      standards track documents does not mean that a standards track
      document cannot have points assigned outside of that 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.

11. References

11.1. Normative References

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

[Named.Information.Hash.Algorithm]
          IANA, "Named Information Hash Algorithm",
          <https://www.iana.org/assignments/named-information/named-
          information.xhtml>.

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

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

[RFC6838] Freed, N., Klensin, J., and T. Hansen, "Media Type
          Specifications and Registration Procedures", BCP 13,
          RFC 6838, DOI 10.17487/RFC6838, January 2013,
          <https://www.rfc-editor.org/info/rfc6838>.

[RFC6920] Farrell, S., Kutscher, D., Dannewitz, C., Ohlman, B.,
          Keranen, A., and P. Hallam-Baker, "Naming Things with
          Hashes", RFC 6920, DOI 10.17487/RFC6920, April 2013,
          <https://www.rfc-editor.org/info/rfc6920>.

[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/info/rfc7252>.

[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/info/rfc7519>.

[RFC7641] Hartke, K., "Observing Resources in the Constrained
          Application Protocol (CoAP)", RFC 7641,
          DOI 10.17487/RFC7641, September 2015,
          <https://www.rfc-editor.org/info/rfc7641>.

[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/info/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/info/rfc8174>.

   [RFC8259] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data
              Interchange Format", STD 90, RFC 8259,
              DOI 10.17487/RFC8259, December 2017,
              <https://www.rfc-editor.org/info/rfc8259>.

   [RFC8392] Jones, M., Wahlstroem, E., Erdtman, S., and H. Tschofenig,
              "CBOR Web Token (CWT)", RFC 8392, DOI 10.17487/RFC8392,
              May 2018, <https://www.rfc-editor.org/info/rfc8392>.

   [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/info/rfc8610>.

   [RFC8949] Bormann, C. and P. Hoffman, "Concise Binary Object
              Representation (CBOR)", STD 94, RFC 8949,
              DOI 10.17487/RFC8949, December 2020,
              <https://www.rfc-editor.org/info/rfc8949>.

11.2. Informative References

   [I-D.bormann-t2trg-stp]
              Bormann, C. and K. Hartke, "The Series Transfer Pattern
              (STP)", Work in Progress, Internet-Draft, draft-bormann-
              t2trg-stp-03, 7 April 2020,
              <https://www.ietf.org/archive/id/draft-bormann-t2trg-stp-
              03.txt>.

   [I-D.ietf-core-conditional-attributes]
              Koster, M. and B. Silverajan, "Conditional Attributes for
              Constrained RESTful Environments", Work in Progress,
              Internet-Draft, draft-ietf-core-conditional-attributes-00,
              12 July 2021, <https://www.ietf.org/archive/id/draft-ietf-
              core-conditional-attributes-00.txt>.

   [RFC7009] Lodderstedt, T., Ed., Dronia, S., and M. Scurtescu, "OAuth
              2.0 Token Revocation", RFC 7009, DOI 10.17487/RFC7009,
              August 2013, <https://www.rfc-editor.org/info/rfc7009>.

Appendix A. Usage of the Series Transfer Pattern

   This section discusses how the diff query of the TRL defined in
   Section 5.2 is a usage example of the Series Transfer Pattern defined
   in [I-D.bormann-t2trg-stp].

A diff query enables the transfer of a series of TRL updates, with the Authorization Server specifying U <= N_MAX diff entries as the U most recent updates to the portion of the TRL pertaining to a registered device.

For each registered device, the Authorization Server maintains an update collection of maximum N_MAX items. Each time the TRL changes, the Authorization Server performs the following operations for each registered device.

1. The Authorization Server considers the portion of the TRL pertaining to that registered device. If the TRL portion is not affected by this TRL update, the Authorization Server stops the processing for that registered device.

2. Otherwise, the Authorization Server creates two sets 'trl_patch' of token hashes, i.e., one "removed" set and one "added" set, as related to this TRL update.

3. The Authorization Server fills the two sets with the token hashes of the removed and added Access Tokens, respectively, from/to the TRL portion from step 1.

4. The Authorization Server creates a new series item including the two sets from step 3, and adds the series item to the update collection associated to the registered device.

When responding to a diff query request from a registered device (see Section 5.2), 'diff' is a subset of the collection associated to the requester, where each 'diff_entry' record is a series item from that collection. Note that 'diff' specifies the whole current collection when the value of U is equal to SIZE, i.e., the current number of series items in the collection.

The value N of the 'diff' query parameter in the diff query request allows the requester and the Authorization Server to trade the amount of provided information with the latency of the information transfer.

Since the collection associated to each registered device includes up to N_MAX series item, the Authorization Server deletes the oldest series item when a new one is generated and added to the end of the collection, due to a new TRL update pertaining to that registered device. This addresses the question "When can the server decide to no longer retain older items?" in Section 3.2 of [I-D.bormann-t2trg-stp].

Appendix B. Usage of the "Cursor" Pattern

   Building on Appendix A, this section describes how the diff query of
   the TRL defined in Section 5.2 can be further improved by using the
   "Cursor" pattern of the Series Transfer Pattern (see Section 3.3 of
   [I-D.bormann-t2trg-stp]).

   This has two benefits. First, the Authorization Server can avoid
   excessively big latencies when several diff entries have to be
   transferred, by delivering one adjacent subset at the time, in
   different diff query responses. Second, a requester can retrieve
   diff entries associated to TRL updates that, even if not the most
   recent ones, occurred after a TRL update indicated as reference
   point.

   To this end, each series item in an update collection is also
   associated to an unsigned integer 'index', with value the absolute
   counter of series items added to that collection minus 1. That is,
   the first series item added to a collection has 'index' with value 0.
   Then, the values of 'index' are used as cursor information.

   Furthermore, the Authorization Server defines an unsigned integer
   MAX_DIFF_BATCH <= N_MAX, specifying the maximum number of diff
   entries to be included in a single diff query response. If
   supporting diff queries, the Authorization Server SHOULD provide
   registered devices and administrators with the value of
   MAX_DIFF_BATCH, upon their registration (see Section 6).

   Finally, the full query and diff query exchanges defined in
   Section 5.1 and Section 5.2 are extended as follows.

   In particular, successul responses from the TRL endpoint MUST use the
   Content-Format "application/ace-trl+cbor" defined in Section 10.2 of
   this specification.

B.1. Full Query Request

   No changes apply to what defined in Section 5.1.

B.2. Full Query Response

   When sending a 2.05 (Content) response to a full query request (see
   Appendix B.1), the response payload includes a CBOR map with the
   following fields, whose CBOR labels are defined in Appendix B.4.4.

   * 'trl': this field MUST include a CBOR array of token hashes. The
      CBOR array is populated and formatted as defined for the CBOR
      array 'full' in Section 5.1.

   * 'cursor': this field MUST include either the CBOR simple value
      Null or a CBOR unsigned integer.

      The CBOR simple value Null MUST be used to indicate that there are
      currently no TRL updates pertinent to the requester, i.e., the
      update collection for that requester is empty. This is the case
      from when the requester registers at the Authorization Server
      until a first update pertaining that requester occurs to the TRL.

      Otherwise, the field MUST include a CBOR unsigned integer,
      encoding the 'index' value of the last series item in the
      collection, as corresponding to the most recent update pertaining
      to the requester occurred to the TRL.

B.3. Diff Query Request

   In addition to the query parameter 'diff' (see Section 5.2), the
   requester can specify a query parameter 'cursor', with value an
   unsigned integer.

B.4. Diff Query Response

   The Authorization Server composes a response to a diff query request
   (see Appendix B.3) as follows, depending on the parameters specified
   in the request and on the current status of the update collection for
   the requester.

B.4.1. Empty Collection

   If the collection associated to the requester has no elements, the
   Authorization Server returns a 2.05 (Content) diff query response.

   The response payload includes a CBOR map with the following fields,
   whose CBOR labels are defined in Appendix B.4.4.

   * 'diff': this field MUST include an empty CBOR array.

   * 'cursor': this field MUST include the CBOR simple value Null.

   * 'more': this fields MUST include the CBOR simple value False.

B.4.2. Cursor Not Specified in the Diff Query Request

   If the update collection associated to the requester is not empty and
   the diff query request does not include the query parameter 'cursor',
   the Authorization Server returns a 2.05 (Content) diff query
   response.

The response payload includes a CBOR map with the following fields, whose CBOR labels are defined in Appendix B.4.4.

* 'diff': this field MUST include a CBOR array, containing L = min(U, MAX_DIFF_BATCH) diff entries. In particular, the CBOR array is populated as follows.

  - If U <= MAX_DIFF_BATCH, these diff entries are the last series items in the collection associated to the requester, corresponding to the L most recent TRL updates pertaining to the requester.

  - If U > MAX_DIFF_BATCH, these diff entries are the eldest of the last U series items in the collection associated to the requester, as corresponding to the first L of the U most recent TRL updates pertaining to the requester.

  The 'diff' CBOR array as well as the individual diff entries have the same format specified in Figure 5 and used for the response payload defined in Section 5.2.

* 'cursor': this field MUST include a CBOR unsigned integer. This takes the 'index' value of the series element of the collection included as first diff entry in the 'diff' CBOR array. That is, it takes the 'index' value of the series item in the collection corresponding to the most recent update pertaining to the requester and returned in this diff query response.

  Note that 'cursor' takes the same 'index' value of the last series item in the collection when U <= MAX_DIFF_BATCH.

* 'more': this field MUST include the CBOR simple value False if U <= MAX_DIFF_BATCH, or the CBOR simple value True otherwise.

  If 'more' has value True, the requester can send a follow-up diff query request including the query parameter 'cursor', with the same value of the 'cursor' field included in this diff query response. As defined in Appendix B.4.3, this would result in the Authorization Server transferring the following subset of series items as diff entries, i.e., resuming from where interrupted in the previous transfer.

B.4.3. Cursor Specified in the Diff Query Request

If the update collection associated to the requester is not empty and the diff query request includes the query parameter 'cursor' with value P, the Authorization Server proceeds as follows.

* The Authorization Server MUST return a 4.00 (Bad Request) response in case the 'cursor' parameter specifies a value other than 0 or than a positive integer.

* If no series item X with 'index' having value P is found in the collection associated to the requester, then that item has been previously removed from the history of updates for that requester (see Appendix A). In this case, the Authorization Server returns a 2.05 (Content) diff query response.

  The response payload includes a CBOR map with the following fields, whose CBOR labels are defined in Appendix B.4.4.

  - 'diff': this field MUST include an empty CBOR array.

  - 'cursor': this field MUST include the CBOR simple value Null.

  - 'more': this field MUST include the CBOR simple value True.

  With the combination ('cursor', 'more') = (Null, True), the Authorization Server is signaling that the update collection is in fact not empty, but that some series items have been lost due to their removal, including the item with 'index' value P that the requester wished to use as reference point.

  When receiving this diff query response, the requester should send a new full query request to the Authorization Server, in order to fully retrieve the current pertaining portion of the TRL.

* If the series item X with 'index' having value P is found in the collection associated to the requester, the Authorization Server returns a 2.05 (Content) diff query response.

  The response payload includes a CBOR map with the following fields, whose CBOR labels are defined in Appendix B.4.4.

  - 'diff': this field MUST include a CBOR array, containing L = min(SUB_U, MAX_DIFF_BATCH) diff entries, where SUB_U = min(NUM, SUB_SIZE), and SUB_SIZE is the number of series items in the collection following the series item X.

     That is, these are the L updates pertaining to the requester that immediately follow the series item X indicated as reference point. In particular, the CBOR array is populated as follows.

         o If SUB_U <= MAX_DIFF_BATCH, these diff entries are the last
            series items in the collection associated to the requester,
            corresponding to the L most recent TRL updates pertaining to
            the requester.

         o If SUB_U > MAX_DIFF_BATCH, these diff entries are the eldest
            of the last SUB_U series items in the collection associated
            to the requester, corresponding to the first L of the SUB_U
            most recent TRL updates pertaining to the requester.

         The 'diff' CBOR array as well as the individual diff entries
         have the same format specified in Figure 5 and used for the
         response payload defined in [Section 5.2](#).

      - 'cursor': this field MUST include a CBOR unsigned integer. In
         particular:

         o If L is equal to 0, i.e., the series item X is the last one
            in the collection, 'cursor' takes the same 'index' value of
            the last series item in the collection.

         o If L is different than 0, 'cursor' takes the 'index' value
            of the series element of the collection included as first
            diff entry in the 'diff' CBOR array. That is, it takes the
            'index' value of the series item in the collection
            corresponding to the most recent update pertaining to the
            requester and returned in this diff query response.

         Note that 'cursor' takes the same 'index' value of the last
         series item in the collection when SUB_U <= MAX_DIFF_BATCH.

      - 'more': this field MUST include the CBOR simple value False if
         SUB_U <= MAX_DIFF_BATCH, or the CBOR simple value True
         otherwise.

         If 'more' has value True, the requester can send a follow-up
         diff query request including the query parameter 'cursor', with
         the same value of the 'cursor' field specified in this diff
         query response. This would result in the Authorization Server
         transferring the following subset of series items as diff
         entries, i.e., resuming from where interrupted in the previous
         transfer.

[B.4.4](#). TRL Parameters

   This specification defines a number of fields used in the response to
   a diff query request to the TRL endpoint relying on the "Cursor"
   pattern, as defined in [Appendix B](#).

The table below summarizes them, and specifies the CBOR value to use as abbreviation instead of the full descriptive name. Note that the Content-Format "application/ace-trl+cbor" defined in Section 10.2 of this specification MUST be used when these fields are transported.

```
+--------+------------+---------------------+-----------+
| Name   | CBOR Value | CBOR Type           | Reference |
+--------+------------+---------------------+-----------+
| trl    | TBD        | array               | [this     |
|        |            |                     | document] |
+--------+------------+---------------------+-----------+
| cursor | TBD        | simple value null / | [this     |
|        |            | unsigned integer    | document] |
+--------+------------+---------------------+-----------+
| diff   | TBD        | array               | [this     |
|        |            |                     | document] |
+--------+------------+---------------------+-----------+
| more   | TBD        | simple value True   | [this     |
|        |            | or False            | document] |
+--------+------------+---------------------+-----------+
```

Figure 9: CBOR abbreviations for TRL parameters

Acknowledgments

Authors' Addresses

   Marco Tiloca
   RISE AB
   Isafjordsgatan 22
   SE-16440 Kista
   Sweden

   Email: marco.tiloca@ri.se

Ludwig Seitz
Combitech
Djaeknegatan 31
SE-21135 Malmoe
Sweden

Email: ludwig.seitz@combitech.com

Francesca Palombini
Ericsson AB
Torshamnsgatan 23
SE-16440 Kista
Sweden

Email: francesca.palombini@ericsson.com

Sebastian Echeverria
CMU SEI
4500 Fifth Avenue
Pittsburgh, PA, 15213-2612
United States of America

Email: secheverria@sei.cmu.edu

Grace Lewis
CMU SEI
4500 Fifth Avenue
Pittsburgh, PA, 15213-2612
United States of America

Email: glewis@sei.cmu.edu