Notification of Revoked Access Tokens in the ACE Framework

draft-tiloca-ace-revoked-tokens-notification-01

Marco Tiloca, RISE
Ludwig Seitz, Combitech
Francesca Palombini, Ericsson
Sebastian Echeverria, CMU SEI
Grace Lewis, CMU SEI

IETF ACE WG, Virtual Interim, June 22nd, 2020
Motivation

› An Access Token may be revoked, before expiration
  – Client or RS has been compromised, or decommissioned
  – Changed access policies
  – Changed ACE profile to use

› In OAuth
  – Token revocation by Client exists (RFC 7009)
  – No revocation by Resource Owner or RS
  – Not a problem, Tokens expire fast

› Different assumptions in ACE
  – E.g. RS has intermittent connectivity, Tokens don’t expire fast
  – How can the AS tell C and RS about revoked tokens?
Contribution

› New interface at the AS
  – The AS maintains one Token Revocation List (TRL) resource
  – The TRL contains the hashes of revoked, not-yet-expired tokens
  – C/RS can GET or GET-Observe from the TRL
  – C/RS retrieve only their own pertaining portion of the TRL

› Benefits
  – Complement token introspection at the AS
  – No need for new endpoints at C or RS

› Updates in -01 from Travis’ review [1] and Jim input – Thanks!

[1] https://mailarchive.ietf.org/arch/msg/ace/1UK5QuLh4kmzIh211JBtotdchfQ/
Rationale

› Token hash, as Token name/ID
  – Not ‘cti’, the Token is opaque to the Client
  – Computed as per RFC 6920, Section 6
  – Support for both CBOR and JSON transport

› Token Revocation List (TRL) at the AS
  – CBOR array of Token hashes
  – Add token hashes when Tokens are revoked
  – Remove token hashes when revoked Tokens expire

› Interaction
  – C and RS get the URL to the TRL endpoint upon registration
  – C and RS obtain only hashes of their own pertaining Tokens
  – A registered Administrator gets all Token hashes in the TRL
Protocol overview

Authorization Server

- revoke/trl
- TRL: {th1, th2, th3}

Administrator

Client 1

Resource Server 1

Client 2

Resource Server 2

- t1
- t2
- t3
Two types of TRL queries

› Common features
   – Limited to the portion of the TRL pertaining the requester
   – TRL filtering based on authenticated identity of the requester (secure session)

› Full query – GET [Observe: 0] example_as/revoke/trl
   – Request for all pertaining token hashes in the TRL
   – Return a CBOR array, with the Token hashes as elements

› Diff query – GET [Observe: 0] example_revoke/trl?diff=true[&N=3]
   – Request for the latest N updates to the pertaining portion of the TRL list
   – Build N entries as CBOR maps. Each entry refers to an update and has:
     › A field “deleted”, with a CBOR array of Token hashes as element.
     › A field “added”, with a CBOR array of Token hashes as element.
   – Return a CBOR array with the N entries as element, in reverse chronological order
   – Work in progress to make it simpler and more efficient – Thanks Carsten!
Example

Registration: POST

2.01 CREATED
Payload: {
    ...
    "tr1" = "revoke/tr1"
}

GET Observe: 0
coap://example.as.com/revoke/tr1/

2.05 CONTENT Observe: 1
Payload: []
    .
    .
    .

(Access Tokens t1 and t2 issued and successfully submitted to RS)
    .
    .
Example (ctd.)

RS

\[(Access\, \text{Token}\, t_1\, \text{is revoked})\]

\[\leftarrow \]

2.05 CONTENT Observe: 2
Payload: \([h(\text{bstr.t1})]\)

\[
\vdots
\]

\[(Access\, \text{Token}\, t_2\, \text{is revoked})\]

\[\leftarrow \]

2.05 CONTENT Observe: 3
Payload: \([h(\text{bstr.t1}),\]
\[\quad h(\text{bstr.t2})]\]

\[
\vdots
\]

\[(Access\, \text{Token}\, t_1\, \text{expires})\]

\[\leftarrow \]

2.05 CONTENT Observe: 4
Payload: \([h(\text{bstr.t2})]\)
Summary

› Notification of revoked Access Token
  – GET or GET-Observe; full query and diff query
  – Complement token introspection at the AS
  – No need for new endpoints on Clients and Resource Servers

› Version -01 incorporates:
  – Review from Travis Spencer
  – Input and comments from Jim

› Next steps
  – Submit version -02 before the cut-off
  – Address review of version -01 from Carsten [1] – Thank you!

Thank you!

Comments/questions?