Authentication and Authorization for Constrained Environments (ACE)

draft-ietf-ace-oauth-authz-02

Ludwig Seitz (ludwig@sics.se)

IETF ACE WG interim meeting
June 16, 2016

Major changes from -01 to -02

- Separation of Framework and Profiles
- OAuth Endpoints
- Proof-of-possession Key Distribution
- Key Confirmation
- Client Tokens
- IANA
- Deployment Scenarios

Separation of Framework and Profiles

- This draft is the ACE framework
  - Defines OAuth endpoints
    - Note: “endpoint” defined differently in OAuth and CoAP
- ACE Profiles specify
  - Communication protocol
  - Communication security
  - Mutual authentication
  - Proof-of-Possession method for access tokens (could coincide with client authentication)
  - *Optionally: New methods of token transfer*

  We will try to provide an example profile for IETF 96

OAuth Endpoints

- /token
  - Hosted by AS
  - Used by client to request access tokens
  - Informs client about the profile to use
- /introspect
  - Hosted by AS
  - Used by RS to get information about access tokens
  - Can provide information for the client → *client-token*
- /authz-info
  - Hosted by RS
  - Used by client to submit access tokens

Proof-of-possession Key Distribution

- /token endpoint (like in plain OAuth 2.0)
- Additional response parameters:
  - profile : Specifies ACE profile between client and RS
  - token_type : here always “pop”
  - alg : Proof-of-possession method, specified by profiles
  - cnf : Proof-of-possession key (See next slide)
- Client can also use these to indicate preferences in the request
- Duplicates some work from draft-ietf-oauth-pop-key-distribution
  - Status of this draft unclear

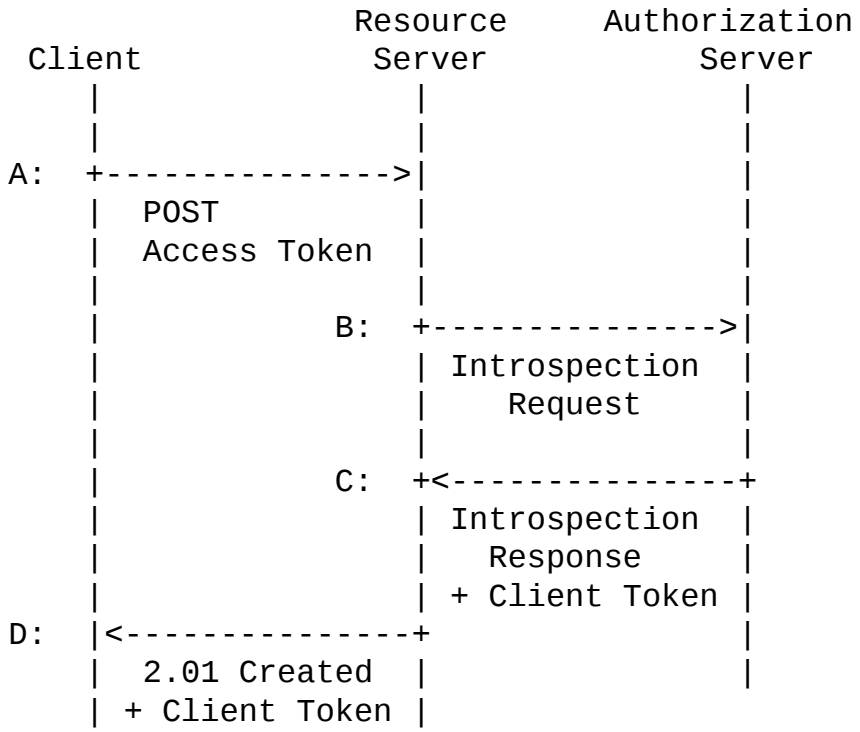
Key Confirmation

- Uses 'cnf' claim/parameter
  - Analogous to RFC 7800, but for CBOR/COSE
  - Either holds a COSE_Key or a key-identifier
- Defined for use in:
  - Access Token
  - Client Token
  - Access Token request
  - Access Token response
  - Introspection response

Client Tokens

Scenario:

- Client with limited connectivity and long-lived token
- Client Token informs client about RS's key (and possibly about other access token metadata)
- New concept, please review for usefulness!



IANA

- Registering new parameters/claims for OAuth
- Registering CBOR abbreviation for existing parameters
- Please double-check!

Deployment Scenarios

- Moved to appendix
- Non-normative examples of how the framework could be used
- May be replaced by profiles

Examples – General Notes

- CBOR diagnostic notation used
- CBOR label abbreviations not used
- Binary data abbreviated
  - Keys, IVs, tokens

Profile Example: CoAP-DTLS

- *Note: This is made up on the fly*
- CoAP+DTLS with client authN for C – RS
  - Use the proof-of-possession key as either PSK or RPK in DTLS to provide proof-of-possession
- CoAP+DTLS for /token and /introspect
- Optional token transfer through resumption ticket as described in draft-seitz-ace-ticket-token-transfer

Token Endpoint Example

CoAP Request C → AS :

```
Header: POST (Code=0.02)
Uri-Host: "server.example.com"
Uri-Path: "token"
Content-Type: "application/cbor"
Payload:
{
  "grant_type" : "client_credentials",
  "aud" : "tempSensor4711",
  "client_id" : "myclient",
  "client_secret" : b64'FWRUVGZUZ...WSRlVGhA',
  "token_type" : "pop",
  "alg" : "DTLS_PSK",
  "profile" : "coap_dtls"
}
```

Token Endpoint Example ctd.

CoAP Response AS → C :

```
Header: Created (Code=2.01)
Content-Type: "application/cbor"
Payload:
{
  "access_token" : b64'SlAV32h...kKG',
  "token_type" : "pop",
  "alg" : "DTLS-PSK",
  "expires_in" : "3600",
  "profile" : "coap_dtls"
  "cnf" : { … see example later … }
}
```

Introspection Example

Request RS → AS:

```
Header: POST (Code=0.02)
Uri-Host: "server.example.com"
Uri-Path: "introspect"
Content-Type: "application/cbor"
Payload:
{
  "token" : b64'7gj0dXJQ43U',
  "token_type_hint" : "pop"
}
```

Introspection Example ctd

Response AS → RS:

```
Header: Created Code=2.01)
Content-Type: "application/cbor"
Payload:
 {
   "active" : true,
   "scope" : "read",
   "token_type" : "pop",
   "alg" : "DTLS_RPK",
   "profile" : "coap_dtls",
   "client_token" : b64'2QPhg...0OhAQo',
   "cnf" : { … see example later … }
 }
```

Key Confirmation – Examples

Raw public key:

```
"cnf" : {
   "COSE_Key" : {
      "kty" : "EC",
      "kid" : h'11',
      "crv" : "P-256",
      "x" : b64'usWxHK...iglWiGahtagnv8',
      "y" : b64'IBOL+C...3BtpkbtKLv8EX4'
   }
}
```

Key Confirmation – Examples

Encrypted symmetric key:

```
"cnf" : {
   "COSE_Encrypted" : {
      993([
         h'a1010a', # protected header
                    #  == {"alg" : "AES-CCM-16-64-128"}
         {"iv" : b64'ifUv...UmGnjA'}, # unprotected header
         b64'WXThuZo...OdGk8kNzaIhIQ' # ciphertext
      ])
   }
}
```

Plaintext:

```
{
   "kty" : "Symmetric",
   "kid" : b64'39Gqlw',
   "k" : b64'hJtXhkV8FJG+Onbc6mxCcQh'
}
```

Key Confirmation – Examples

Key identifier:

```
"cnf" : {
     "kid" : b64'39Gqlw'
}
```

Client Token – Example

Instructions AS → Client (passed through RS)

```
{
   "profile" : "coap_dtls",
   "token_type" : "pop",
   "alg" : "DTLS_RPK",
   "cnf" : { "kid" : b64'39Gqlw'},
   "rs_cnf" : {
      "COSE_Key" : {
         "kty" : "EC",
         "kid" : h'11',
         "crv" : "P-256",
         "x" : b64'usWxHK2PmfnHKwXPS54m0kTcGJ90UiglWiGahtagnv8',
         "y" : b64'IBOL+C3BttVivg+lSreASjpkttcsz+1rb7btKLv8EX4'
      }
   }
}
```

“Use this profile”

“Use this pop method”

“Use this key for pop”

“RS uses this key for authN”

Note: This shows only client token payload, not encryption wrapper

Thank you!

Questions/comments?