Additional Authentication Credentials for the Datagram Transport Layer Security (DTLS) Profile for Authentication and Authorization for Constrained Environments (ACE)

draft-tiloca-ace-authcred-dtls-profile-00

Marco Tiloca, RISE
John P. Mattsson, Ericsson

IETF 117 Meeting – San Francisco – July 24th, 2023
Motivation

› The DTLS profile of ACE is defined in RFC 9202
  – It has an “RPK mode” based on asymmetric authentication credentials
  – Authentication credentials are raw public keys (RPKs), only as COSE Keys

› Other types of asymmetric authentication credentials exist
  – Other representations of RPKs → DTLS handshake: just like in the main case above
  – Public key certificates → DTLS handshake: ready to use those

› Good to support other credential formats for the Client (C) and Resource Server (RS)

› Early idea shared during the ACE session at IETF 116
Contribution

› Proposed update to RFC 9202
   – Enable the use of alternative formats for public authentication credentials

› Update breakdown
   – Extend the “RPK mode”, to support also CWT Claims Sets (CCSs) [1]

› Seamlessly applicable if TLS is used between C and RS, as defined in RFC 9430

Mechanics

› Use of new CWT Confirmation Methods
  – "kccs", "x5bag", "x5chain", "c5b", and "c5c“ – Defined in draft-ietf-ace-edhoc-oscore-profile

› As usual, specify authentication credentials in
  – “req_cnf” parameter of C-to-AS token request (C’s authentication credential)
  – “cnf” claim of the access token (C’s authentication credential)
  – “rs_cnf” parameter of AS-to-C token response (RS’ authentication credential)

› Possible to combine different credential formats
  – Public keys in the “RPK mode”: both as CCS / both as COSE key / one as CCS and one as COSE Key
  – Certificates in the “Certificate mode”: both X.509 / both C509 / one X.509 and one C509
  – One public key as RPK, the other one in a certificate – Ok for (D)TLS, see Section 5.3 of RFC 7250
Example in “RPK mode”

Client → Authorization Server

Access Token Request in “RPK mode”

POST coaps://as.example.com/token
Content-Format: application/ace+cbor
Payload:
{
  "grant_type" : 2,
  "audience" : "tempSensor4711",
  "req_cnf" : {
    "kccs" : {
      "cnf" : {
        "COSKey" : {
          "kty" : 2,
          "crv" : 1,
          "x" : h'd7cc072de2205bdc1537a543d53c60a6
               acb62eccd890c7fa27c9e354089b9e13',
          "y" : h'f95e1d4b851a2cc80fff87d8e23f22af
               b725d535e515d020731e79a3b4e47120'
        }
      }
    }
  }
}

Client’s RPK as CCS

Authorization Server → Client

Access Token Response in “RPK mode”

2.01 Created
Content-Format: application/ace+cbor
Max-Age: 3560
Payload:
{
  "access_token" : b64'SlvAV32hk'/...
  (remainder of CWT omitted for brevity),
  "CWT contains the client's RPK in the cnf claim),
  "expires_in" : 3600,
  "rs_cnf" : {
    "kccs" : {
      "sub" : "AA-BB-CC-00-01-02-03-04",
      "cnf" : {
        "COSKey" : {
          "kty" : 2,
          "crv" : 1,
          "x" : h'bb34960526ea4d32e940cad2a234148
               ddc21791a12abfdabc93622046dd4f0',
          "y" : h'4519e257236b2a0ce2023f931f1f386
               ca7afda64fcde0108c224c51eabf6072'
        }
      }
    }
}

Resource Server’s RPK as CCS
Example in “Certificate mode”

Client → Authorization Server

Access Token Request in “Certificate Mode”

POST coaps://as.example.com/token
Content-Format: application/ace+cbor
Payload:
{
  "grant_type": 2,
  "audience": "tempSensor4711",
  "reg_cnf": {
    "x5chain": "h'3081ee3081a1a003020102020462319ee4c30
056032b6570301d311b3019063504030c
124544484f4320526f6f7420456432353531
39031e170d3232333331363038323433365a
170d32393132333132333030305a0a3223a
2030e106035504030c174544484f43205265
73706f6e6465722045643235353139302a30
056032b6570032100a1db47b9184854ad1
2a0ca1a354e418ace3aa0f2c66200b0a3a5c
5de92f9359300506032b6570034100b723bc
01ea0b928e8b26c98de19cc3823de46ed69
87b032478fecea14537a1af14cc8ce829c6
b73044101837be4abc94956586dce51cf8e
52ab82c152cb02"}
}

Client’s X.509 Certificate

Authorization Server → Client

Access Token Response in “Certificate Mode”

2.01 Created
Content-Format: application/ace+cbor
Max-Age: 3560
Payload:
{
  "access_token": "b64'SlAV32hk'/...
  (remainder of CWT omitted for brevity;
  CWT contains the client's X.509 certificate in the cnf claim)/,
  "expires_in": 3600,
  "rs_cnf": {
    "x5chain": "h'3081ee3081a1a003020102020462319ea030
056032b6570301d311b3019063504030c
124544484f4320526f6f7420456432353531
39031e170d3232333331363038323433365a
170d32393132333132333030305a0a3223a
2030e106035504030c174544484f43205265
73706f6e6465722045643235353139302a30
056032b6570032100a1db47b9184854ad1
2a0ca1a354e418ace3aa0f2c66200b0a3a5c
5de92f9359300506032b6570034100b723bc
01ea0b928e8b26c98de19cc3823de46ed69
87b032478fecea14537a1af14cc8ce829c6
b73044101837be4abc94956586dce51cf8e
52ab82c152cb02"
}

Resource Server’s X.509 Certificate
Next steps

› Consider the transfer of certificates by reference
  – Using CWT Confirmation Methods “x5t” and “c5t”
  – Already defined in *draft-ietf-ace-edhoc-oscore-profile*

› More security considerations
  – E.g., on validating a CCS, see also Section 9.8 of *draft-ietf-lake-edhoc-20*

› Comments and feedback are welcome!
Thank you!

Comments/questions?

https://gitlab.com/crimson84/draft-tiloca-ace-authcred-dtls-profile