MQTT-TLS Profile of ACE

draft-ietf-ace-mqtt-tls-profile-04

Cigdem Sengul

Cigdem.Sengul@nominet.uk

Interim, ACE WG

April 12, 2020

Updates since the last interim

- Submitted draft-ietf-ace-mqtt-tls-profile-04
  - A number of changes including
    - Clarified format of AS Creation Hints in CONNACK: https://github.com/ace-wg/mqtt-tls-profile/issues/46
    - Format of the AUTH data: https://github.com/ace-wg/mqtt-tls-profile/issues/40
    - Text improvements/clarifications: Github issues 37-39, 41, 43-45; Formative vs Informative references.

- Clarified with Hannes that
  - “you can do pretty much everything defined in draft-ietf-oauth-pop-key-distribution with the ACE-OAuth framework.”

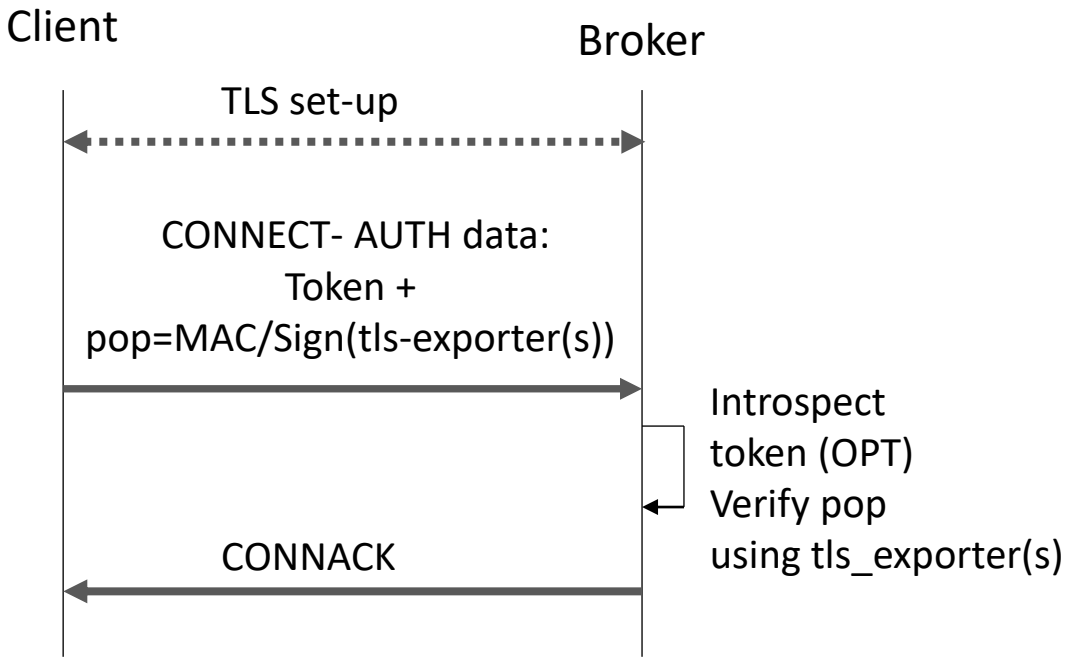
Client Authentication-Authorisation

MQTT / TLS	None	ACE
Anon	Public topics Authz-info	**Recommended option: Token in CONNECT** **AS-Discovery [DISCUSS]**
Known (RPK/PSK)	RPK – token via authz-info PSK– token “psk_identity” [I-D.ietf-ace-dtls-authorize]	SHOULD NOT be chosen Token in CONNECT overwrites any permission during TLS handshake*

* [Discuss: Daniel’s comment, is this the only way to do this?]

TLS:none – MQTT:ACE

MQTT v5: Authentication Using AUTH Property

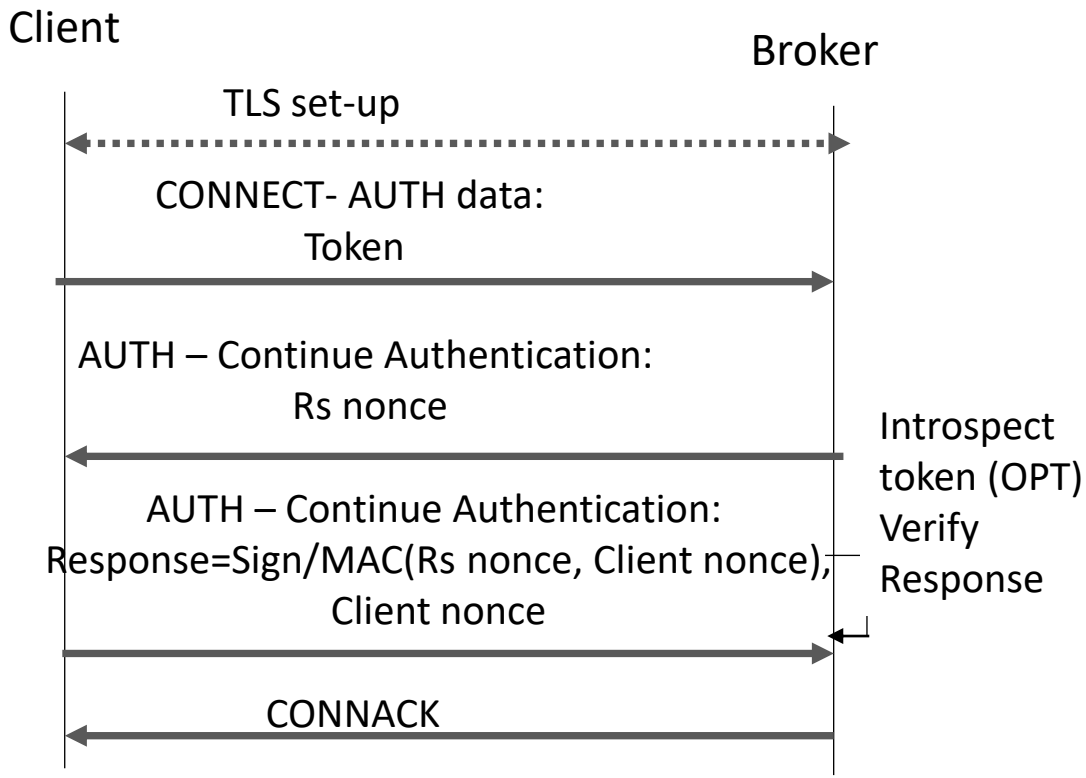


Proof-of-Possession using a secret from the TLS session

Only option for MQTT v3.1.1: Username=Token; Password= pop

MQTT Binary Data encoding for token + pop

tls-exporter label? EXPORTER-ACE-MQTT-Sign-Challenge; Length 32B.



Proof-of-Possession using a challenge/response

Challenge: 8B RS nonce + 8B Client Nonce.

[DISCUSS]

Does not use channel binding. Should it?

Jim in e-mail: Sign the triple of <server challenge, client challenge, tls-exporter value>?

Other discussion points

Review comments from Jim

1. MQTT v3.1 client and v5 server operation; v5 server support for v3.1 clients (Also commented on by Hannes)
2. The clean session requirement - put to avoid server keeping state unnecessarily. Is MUST too strong? Alternative: SHOULD and other SHOULDs/MUSTs on broker behavior i.e.,
   a. Session state includes information of the token used in the session.
   b. There is a MAX session expiry set by the broker admin policy allowing capping what the client puts for session expiry.
   c. The client still submits a token in every CONNECT - the broker checks if the token matches the current token (may or may not introspect in case of a token match); if new token, validate token, replace the session token.
   d. Session state is updated: Subscription Identifiers, that are part of the session state, validated if new permissions. (Although it will do this anyway when publishing a packet to the subscriber.)
   e. Pending messages MUST be re-evaluated based on permissions.

Other review comment from Hannes:

1. Terminology: MQTT broker or server?