Pub-Sub Profile for Authentication and Authorization for Constrained Environments (ACE)

draft-ietf-ace-pubsub-profile-06

Francesca Palombini
Cigdem Sengul
Marco Tiloca

IETF 116
March 2023
The document defines a way to authorize pub-sub clients and protect group communication using the ACE framework

- Mainly focuses on securing group communications for CoAP pub-sub clients (may support MQTT)
  - Relies on transport or application layer security profiles of ACE (e.g. DTLS [RFC9202] or OSCORE [RFC9203]) to achieve communication security, server authentication and proof-of-possession for Access Tokens

- The architecture has evolved from the one proposed in IETF 110

- Message exchanges/message formats and processing follow the specifications for provisioning and renewing keying material in key-groupcomm
  - Work in progress to complete the Client Group Communication Interface at the KDC
  - Work in progress to satisfy key-groupcomm requirements (30) / options (15)
  - Evolved to include removal from the group, and group rekeying (In Progress)
Recap: Architecture Overview

Workflow

(O) Optional AS Discovery (ACE framework) + KDC discovery (Core-CoAP-PubSub resource discovery – in Progress)

(A) Authorisation Request/Response (Clients, AS) – for Broker and KDC

(B) Establish secure connection - Join Request (Clients, KDC) – to join a security group and obtain security keys

Publishers: authentication credential/Proof-of-Possession (token passed via authz-info/DTLS handshake)

(C) Establish secure connection (Clients, Broker) – using token obtained in (A)

(D) Pub/sub operations but with protected payloads – using group symmetric key protect message payload; publisher public key crypto for source authentication
Discussion 1 – Application group vs Security Group

**Draft text:** Pub/sub clients communicate within their application groups mapped to a collection of **pub/sub topics**. The pub/sub topics [as defined in core-coap-pubsub] may consist of one or more sub-topics, which may have their own sub-topics, forming a hierarchy. The applications decide how to map this hierarchy into different **application groups**, and a **security group** SHOULD be associated with a single application group.

**Draft leaves it to the applications to decide on the correct mapping**

More alignment with core-coap-pubsub (topic collection vs topic)
Discussion 2 – Authorisation Request & Scope

- **Draft text:** The Client sends **two** Authorisation Requests to the AS for **two audiences:** the Broker and the KDC

- Audience (required) and Scope (optional) and MUST be of data model AIF-PUBSUB-GROUPCOMM
  - The object identifier ("Toid") is a CBOR text string, specifying the topic name for the scope entry.
  - The permission set ("Tperm") is a CBOR unsigned integer with value, specifying the Client role, based on the operations the Client can execute on Topic Data in the group.
    - Publish (1), Subscribe (or Read) (2), Delete (3) and Admin (0)

- **Proposal 1:** Have scope encode the audience of the token inside pubsub-roles with ‘Key-Beneficiary’ (i.e., KDC or a Broker) rather than relying on audience

- **Proposal 2:** Embed broker identifier into pubsub-perm - BROKERNAME:TOPICNAME (differentiate the same topic hosted at different brokers)

```plaintext
AIF-PUBSUB-GROUPCOMM = AIF-Generic<pubsub-topic, pubsub-perm>
pubsub-topic = tstr ; Pub/sub topic name
                ; (the associated security group)
pubsub-perm = uint . bits pubsub-roles
pubsub-roles = &(
    Admin: 0,
    Pub: 1,
    Sub: 2,
    Delete: 3
)
scope_entry = [pubsub-topic, pubsub-perm]
```
Protecting Group Communication

• Authorisation Response
  • Contains group key and Base IV defined by the KDC
  • For a Publisher Client, **the KDC assigns an available Sender ID** that has not been used in the group (similar to groupcomm-oscore)

• Protect Messages
  • COSE_Encrypt0 – countersigned by Publisher
  • Unprotected headers
    • Partial IV (Sender Sequence Number) – incremented at each message
    • the IV, generated following the construction in Section 5.2 of [RFC8613] using the sender ID, and Base IV from the symmetric COSE Key received
  • 'external_aad' is an empty string
Next steps

• Needs a few iterations to
  • Wrap up work in progress
  • Close the questions raised in discussions