ACE Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 4, 2021

F. Palombini
Ericsson

July 03, 2020

Pub-Sub Profile for Authentication and Authorization for Constrained Environments (ACE) draft-ietf-ace-pubsub-profile-01

Abstract

This specification defines an application profile for authentication and authorization for publishers and subscribers in a pub-sub setting scenario in a constrained environment, using the ACE framework. This profile relies on transport layer or application layer security to authorize the publisher to the broker. Moreover, it relies on application layer security for publisher-broker and subscriber-broker communication.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on January 4, 2021.

Table of Contents

1. Introduction

The publisher-subscriber setting allows for devices with limited reachability to communicate via a broker that enables store-and-forward messaging between the devices. The pub-sub scenario using the Constrained Application Protocol (CoAP) is specified in [I-D.ietf-core-coap-pubsub], while the one using MQTT is specified in REF MQTT. This document defines a way to authorize nodes in a CoAP pub-sub type of setting, using the ACE framework [I-D.ietf-ace-oauth-authz], and to provide the keys for protecting the communication between these nodes. This document gives detailed specifications for MQTT and CoAP pub-sub, but can easily be adapted for other transport protocol as well.

1.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

Readers are expected to be familiar with the terms and concepts described in [I-D.ietf-ace-oauth-authz], [I-D.ietf-ace-key-groupcomm]. In particular, analogously to [I-D.ietf-ace-oauth-authz], terminology for entities in the architecture such as Client (C), Resource Server (RS), and Authorization Server (AS) is defined in OAuth 2.0 [RFC6749] and [I-D.ietf-ace-actors], and terminology for entities such as the Key Distribution Center (KDC) and Dispatcher in [I-D.ietf-ace-key-groupcomm].

Readers are expected to be familiar with terms and concepts of pub-sub group communication, as described in [I-D.ietf-core-coap-pubsub], or MQTT (REF MQTT pubsub).

2. Application Profile Overview

The objective of this document is to specify how to authorize nodes, provide keys, and protect a pub-sub communication, using [I-D.ietf-ace-key-groupcomm], which itself expands the Ace framework ([I-D.ietf-ace-oauth-authz]), and transport profiles ([I-D.ietf-ace-dtls-authorize], [I-D.ietf-ace-oscore-profile], REF MQTT profile). The pub-sub communication protocol can be based on CoAP, as described in [I-D.ietf-core-coap-pubsub], MQTT (see REF MQTT comm), or other transport.

The architecture of the scenario is shown in Figure 1.

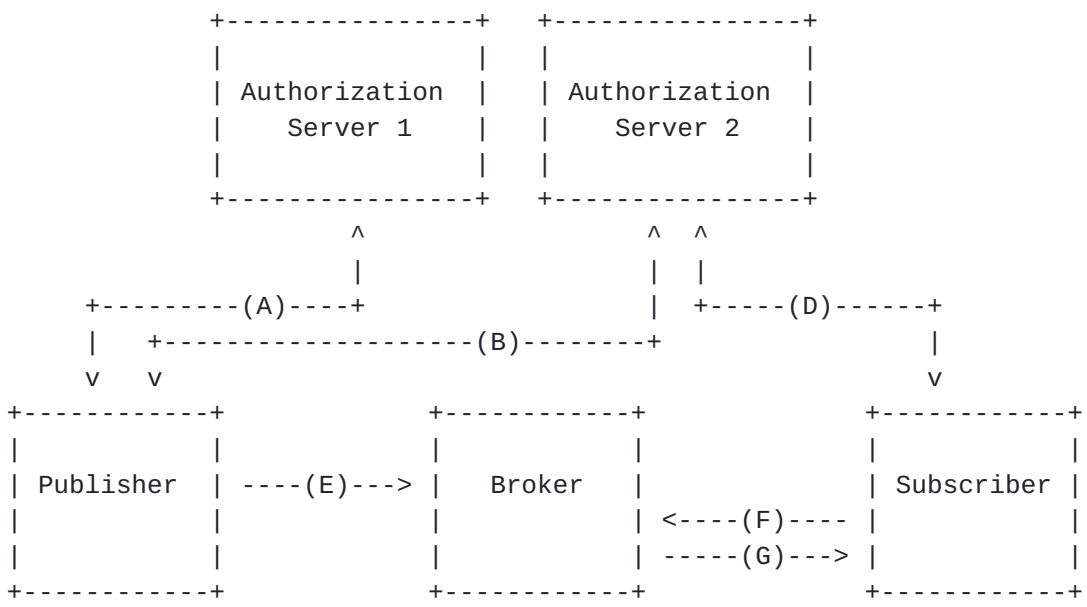


Figure 1: Architecture CoAP pubsub with Authorization Servers

The RS is the broker, which contains the topic. This node corresponds to the Dispatcher, in [I-D.ietf-ace-key-groupcomm]. The AS1 hosts the policies about the Broker: what endpoints are allowed to Publish on the Broker. The Clients access this node to get write access to the Broker. The AS2 hosts the policies about the topic: what endpoints are allowed to access what topic. This node represents both the AS and Key Distribution Center roles from [I-D.ietf-ace-key-groupcomm].

There are four phases, the first three can be done in parallel.

1. The Publisher requests publishing access to the Broker at the AS1, and communicates with the Broker to set up security.

2. The Publisher requests access to a specific topic at the AS2

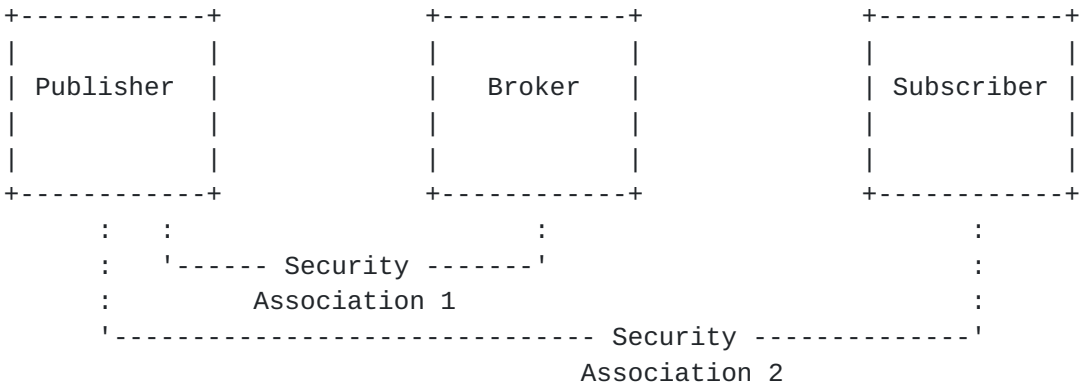
3. The Subscriber requests access to a specific topic at the AS2.

4. The Publisher and the Subscriber securely post to and get publications from the Broker.

This exchange aims at setting up 2 different security associations: on the one hand, the Publisher has a security association with the Broker, to protect the communication and securely authorize the Publisher to publish on a topic (Security Association 1). On the other hand, the Publisher has a security association with the Subscriber, to protect the publication content itself (Security Association 2). The Security Association 1 is set up using AS1 and a

transport profile of [I-D.ietf-ace-oauth-authz], the Security Association 2 is set up using AS2 and [I-D.ietf-ace-key-groupcomm].

Note that, analogously to the Publisher, the Subscriber can also set up an additional security association with the Broker, using an AS, in the same way the Publisher does with AS1. In this case, only authorized Subscribers would be able to get notifications from the Broker. The overhead would be that each Subscriber should access the AS and get all the information to start a secure exchange with the Broker.



Note that AS1 and AS2 might either be co-resident or be 2 separate physical entities, in which case access control policies must be exchanged between AS1 and AS2, so that they agree on rights for joining nodes about specific topics. How the policies are exchanged is out of scope for this specification.

3. PubSub Application Profiles

Each profile defined in this document uses [I-D.ietf-ace-key-groupcomm], which expands the ACE framework. This section defines which exact parameters from [I-D.ietf-ace-key-groupcomm] have to be used, and the values for each parameter. Since [I-D.ietf-ace-oauth-authz] recommends the use of CoAP anc CBOR, this document describes the exchanges assuming CoAP and CBOR are used. However, using HTTP instead of CoAP is possible, using the corresponding parameters and methods. Analogously, JSON [RFC8259] can be used instead of CBOR, using the conversion method specified in Sections 4.1 and 4.2 of [RFC7049]. In case JSON is used, the Content Format or Media Type of the message has to be changed accordingly.

The Publisher and the Subscriber map to the Client in [I-D.ietf-ace-key-groupcomm], the AS2 maps to the AS and to the KDC, the Broker maps to the Dispatcher.

Note that both publishers and subscribers use the same profile.

3.1. Retrieval of COSE Key for protection of content

This phase is common to both Publisher and Subscriber. To maintain the generality, the Publisher or Subscriber is referred as Client in this section.

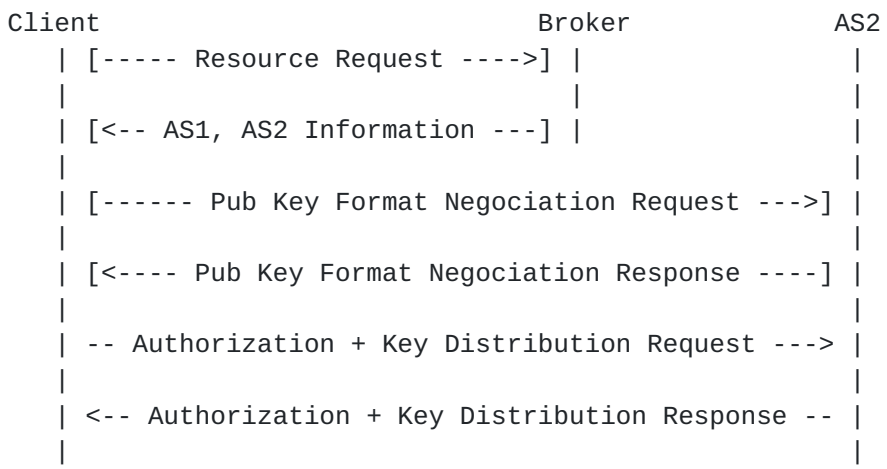


Figure 2: B: Access request - response

Complementary to what is defined in [I-D.ietf-ace-oauth-authz] (Section 5.1.1), to determine the AS2 in charge of a topic hosted at the Broker, the Broker MAY send the address of both the AS in charge of the topic back to the Client in the 'AS' parameter in the AS Information, as a response to an Unauthorized Resource Request (Section 5.1.2). The uri of AS2 is concatenated to the uri of AS1, and separated by a comma. An example using CBOR diagnostic notation and CoAP is given below:

```
4.01 Unauthorized
Content-Format: application/ace+cbor
{"AS": "coaps://as1.example.com/token,
coaps://as2.example.com/pubsubkey"}
```

Figure 3: AS1, AS2 Information example

After retrieving the AS2 address, the Client MAY send a request to the AS, in order to retrieve necessary information concerning the public keys in the group, as well as concerning the algorithm and related parameters for computing signatures in the group. This request is a subset of the Token POST request defined in Section 3.3 of [I-D.ietf-ace-key-groupcomm], specifically a CoAP POST request to a specific resource at the AS, including only the parameters 'sign_info' and 'pub_key_enc' in the CBOR map in the payload. The

default url-path for this resource is /ace-group/gid/cs-info, where "gid" is the topic identifier, but implementations are not required to use this name, and can use their own instead. The AS MUST respond with the response defined in Section 3.3 of [I-D.ietf-ace-key-groupcomm], specifically including the parameters 'sign_info', 'pub_key_enc', and 'rsnonce' (8 bytes pseudo-random nonce generated by the AS).

After that, the Client sends an Authorization + Joining Request, which is an Authorization Request merged with a Joining Request, as described in [I-D.ietf-ace-key-groupcomm], Sections 3.1 and 4.2. The reason for merging these two messages is that the AS2 is both the AS and the KDC, in this setting, so the Authorization Response and the Post Token message are not necessary.

More specifically, the Client sends a POST request to the /ace-group/gid endpoint on AS2, with Content-Format = "application/ace+cbor" that MUST contain in the payload (formatted as a CBOR map):

- the following fields from the Joining Request (Section 4.2 of [I-D.ietf-ace-key-groupcomm]):
  - 'scope' parameter set to a CBOR array containing:
    - the broker's topic as first element, and
    - the text string "publisher" if the client request to be a publisher, "subscriber" if the client request to be a subscriber, or a CBOR array containing both, if the client request to be both.
  - 'get_pub_keys' parameter set to the empty array if the Client needs to retrieve the public keys of the other pubsub members,
  - 'client_cred' parameter containing the Client's public key formatted as a COSE_Key, if the Client needs to directly send that to the AS2,
  - 'cnonce', set to a 8 bytes long pseudo-random nonce, if 'client_cred' is present,
  - 'client_cred_verify', set to a singature computed over the rsnonce concatenated with cnonce, if 'client_cred' is present,
  - OPTIONALLY, if needed, the 'pub_keys_repos' parameter
- the following fields from the Authorization Request (Section 3.1 of [I-D.ietf-ace-key-groupcomm]):

* OPTIONALLY, if needed, additional parameters such as 'client_id'

TODO: 'cnonce' might change name. TODO: register media type ace+json for HTTP?

Note that the alg parameter in the 'client_cred' COSE_Key MUST be a signing algorithm, as defined in section 8 of [RFC8152], and that it is the same algorithm used to compute the signature sent in 'client_cred_verify'.

Examples of the payload of a Authorization + Joining Request are specified in Figure 5 and Figure 8.

The AS2 verifies that the Client is authorized to access the topic and, if the 'client_cred' parameter is present, stores the public key of the Client.

The AS2 response is an Authorization + Joining Response, with Content-Format = "application/ace+cbor". The payload (formatted as a CBOR map) MUST contain:

o the following fields from the Joining Response (Section 4.2 of [I-D.ietf-ace-key-groupcomm]):

  * 'kty' identifies a key type "COSE_Key", as defined in Section 8.2.

  * 'key', which contains a "COSE_Key" object (defined in [RFC8152], containing:

    + 'kty' with value 4 (symmetric)

    + 'alg' with value defined by the AS2 (Content Encryption Algorithm)

    + 'Base IV' with value defined by the AS2

    + 'k' with value the symmetric key value

    + OPTIONALLY, 'kid' with an identifier for the key value

  * OPTIONALLY, 'exp' with the expiration time of the key

  * 'pub_keys', containing the public keys of all authorized signing members formatted as COSE_Keys, if the 'get_pub_keys' parameter was present and set to the empty array in the Authorization + Key Distribution Request

- the following fields from the Authorization Response (Section 3.2 of [I-D.ietf-ace-key-groupcomm]):
  - 'profile' set to the corresponding value, see Section 3.2 or Section 3.3
  - OPTIONALLY 'scope', set to a CBOR array containing:
    - the broker's topic as first element, and
    - the string "publisher" if the client is an authorized publisher, "subscriber" if the client is an authorized subscriber, or a CBOR array containing both, if the client is authorized to be both.

Examples for the response payload are detailed in Figure 6 and Figure 9.

3.2. coap_pubsub_app Application Profile

In case CoAP PubSub is used as communication protocol:

- 'profile' set to "coap_pubsub_app", as specified in Section 8.1.1.

3.3. mqtt_pubsub_app Application Profile

In case mQTT PubSub is used as communication protocol:

- 'profile' set to "mqtt_pubsub_app", as specified in Section 8.1.2.

4. Publisher

In this section, it is specified how the Publisher requests, obtains and communicates to the Broker the access token, as well as the retrieval of the keying material to protect the publication.

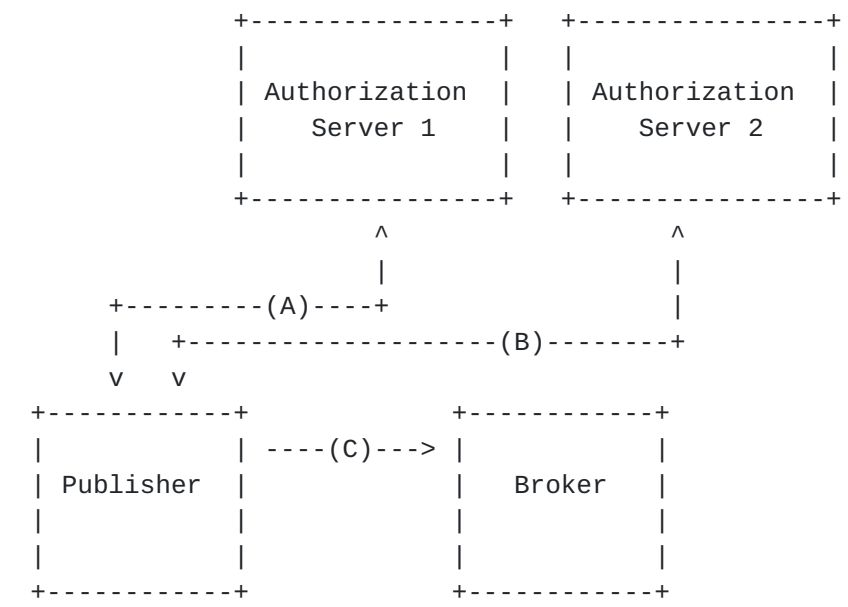


Figure 4: Phase 1: Publisher side

This is a combination of two independent phases:

- one is the establishment of a secure connection between Publisher and Broker, using an ACE transport profile such as DTLS [I-D.ietf-ace-dtls-authorize], OSCORE [I-D.ietf-ace-oscore-profile] or REF MQTT Profile. (A)(C)

- the other is the Publisher's retrieval of keying material to protect the publication. (B)

In detail:

(A) corresponds to the Access Token Request and Response between Publisher and Authorization Server to retrieve the Access Token and RS (Broker) Information. As specified, the Publisher has the role of a CoAP client, the Broker has the role of the CoAP server.

(C) corresponds to the exchange between Publisher and Broker, where the Publisher sends its access token to the Broker and establishes a secure connection with the Broker. Depending on the Information received in (A), this can be for example DTLS handshake, or other protocols. Depending on the application, there may not be the need for this set up phase: for example, if OSCORE is used directly. Note that, in line with what defined in the ACE transport profile used, the access token includes the scope (i.e. pubsub topics on the Broker) the Publisher is allowed to publish to. For implementation semplicity, it is RECOMMENDED that the ACE transport profile used and this specification use the same format of "scope".

(A) and (C) details are specified in the profile used.

(B) corresponds to the retrieval of the keying material to protect the publication end-to-end with the subscribers (see Section 6.1), and uses [I-D.ietf-ace-key-groupcomm]. The details are defined in Section 3.1.

4.1. CoAP Publisher

An example of the payload of an Authorization + Joining Request and corresponding Response for a CoAP Publisher using CoAP and CBOR is specified in Figure 5 and Figure 6, where SIG is a signature computed using the private key associated to the public key and the algorithm in "client_cred".

```
{
  "scope" : ["Broker1/Temp", "publisher"],
  "client_id" : "publisher1",
  "client_cred" :
    { / COSE_Key /
      / type / 1 : 2, / EC2 /
      / kid / 2 : h'11',
      / alg / 3 : -7, / ECDSA with SHA-256 /
      / crv / -1 : 1 , / P-256 /
      / x / -2 : h'65eda5a12577c2bae829437fe338701a10aaa375e1bb5b5de1
      08de439c08551d',
      / y /-3 : h'1e52ed75701163f7f9e40ddf9f341b3dc9ba860af7e0ca7ca7e
      9eecd0084d19c',
  "cnonce" : h'd36b581d1eef9c7c,
  "client_cred_verify" : SIG
    }
}
```

Figure 5: Authorization + Joining Request payload for a Publisher

```
    {
      "profile" : "coap_pubsub_app",
      "kty" : "COSE_Key",
      "key" : {1: 4, 2: h'1234', 3: 12, 5: h'1f389d14d17dc7',
              -1: h'02e2cc3a9b92855220f255fff1c615bc'}
    }
```

Figure 6: Authorization + Joining Response payload for a Publisher

4.2. MQTT Publisher

TODO

5. Subscriber

In this section, it is specified how the Subscriber retrieves the keying material to protect the publication.

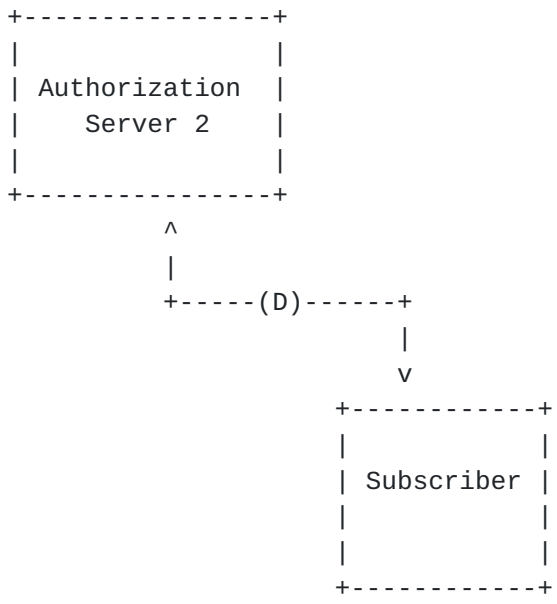


Figure 7: Phase 2: Subscriber side

Step (D) between Subscriber and AS2 corresponds to the retrieval of the keying material to verify the publication end-to-end with the publishers (see Section 6.1). The details are defined in Section 3.1

This step is the same as (B) between Publisher and AS2 (Section 3.1), with the following differences:

- The Authorization + Joining Request MUST NOT contain the 'client_cred parameter', the role element in the 'scope' parameter MUST be set to "subscriber". The Subscriber MUST have access to the public keys of all the Publishers; this MAY be achieved in the Authorization + Joining Request by using the parameter 'get_pub_keys' set to empty array.

- The Authorization + Key Distribution Response MUST contain the 'pub_keys' parameter.

5.1. CoAP Subscriber

An example of the payload of an Authorization + Joining Request and corresponding Response for a CoAP Subscriber using CoAP and CBOR is specified in Figure 8 and Figure 9.

```
{
  "scope" : ["Broker1/Temp", "subscriber"],
  "get_pub_keys" : [ ]
}
```

Figure 8: Authorization + Joining Request payload for a Subscriber

```
{
  "profile" : "coap_pubsub_app",
  "scope" : ["Broker1/Temp", "subscriber"],
  "kty" : "COSE_Key"
  "key" : {1: 4, 2: h'1234', 3: 12, 5: h'1f389d14d17dc7',
          -1: h'02e2cc3a9b92855220f255fff1c615bc'},
  "pub_keys" : [
   {
      1 : 2, / type EC2 /
      2 : h'11', / kid /
      3 : -7, / alg ECDSA with SHA-256 /
      -1 : 1 , / crv P-256 /
      -2 : h'65eda5a12577c2bae829437fe338701a10aaa375e1bb5b5de108de43
      9c08551d', / x /
      -3 : h'1e52ed75701163f7f9e40ddf9f341b3dc9ba860af7e0ca7ca7e9eecd
      0084d19c' / y /
    }
  ]
}
```

Figure 9: Authorization + Joining Response payload for a Subscriber

5.2. MQTT Subscriber

TODO

6. Pub-Sub Protected Communication

This section specifies the communication Publisher-Broker and Subscriber-Broker, after the previous phases have taken place. The operations of publishing and subscribing are defined in [I-D.ietf-core-coap-pubsub].

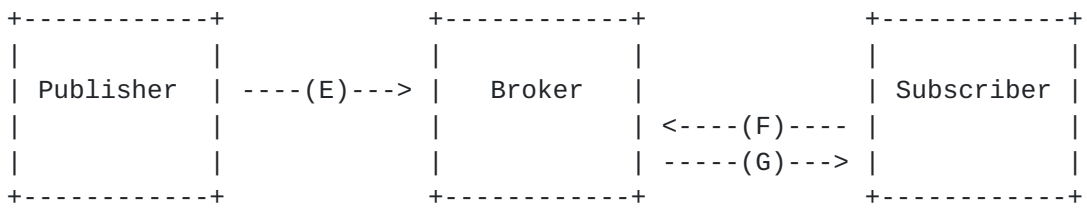


Figure 10: Phase 3: Secure communication between Publisher and Subscriber

The (E) message corresponds to the publication of a topic on the Broker. The publication (the resource representation) is protected with COSE ([RFC8152]). The (F) message is the subscription of the Subscriber, which is unprotected, unless a profile of ACE [I-D.ietf-ace-oauth-authz] is used between Subscriber and Broker. The (G) message is the response from the Broker, where the publication is protected with COSE.

The flow graph is presented below.

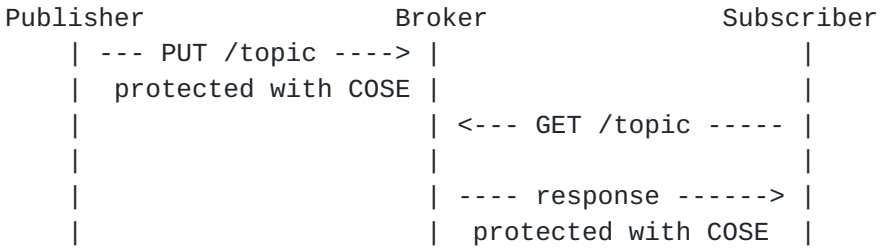


Figure 11: (E), (F), (G): Example of protected communication

6.1. Using COSE Objects To Protect The Resource Representation

The Publisher uses the symmetric COSE Key received from AS2 in exchange B (Section 3.1) to protect the payload of the PUBLISH operation (Section 4.3 of [I-D.ietf-core-coap-pubsub] and REF MQTT). Specifically, the COSE Key is used to create a COSE_Encrypt0 with algorithm specified by AS2. The Publisher uses the private key corresponding to the public key sent to the AS2 in exchange B (Section 3.1) to countersign the COSE Object as specified in Section 4.5 of [RFC8152]. The CoAP payload is replaced by the COSE object before the publication is sent to the Broker.

The Subscriber uses the kid in the countersignature field in the COSE object to retrieve the right public key to verify the countersignature. It then uses the symmetric key received from AS2 to verify and decrypt the publication received in the payload of the CoAP Notification from the Broker.

The COSE object is constructed in the following way:

- The protected Headers (as described in Section 3 of [RFC8152]) MAY contain the kid parameter, with value the kid of the symmetric COSE Key received in Section 3.1 and MUST contain the content encryption algorithm.

- The unprotected Headers MUST contain the Partial IV, with value a sequence number that is incremented for every message sent, and the counter signature that includes:

  * the algorithm (same value as in the asymmetric COSE Key received in (B)) in the protected header;

  * the kid (same value as the kid of the asymmetric COSE Key received in (B)) in the unprotected header;

  * the signature computed as specified in Section 4.5 of [RFC8152].

- The ciphertext, computed over the plaintext that MUST contain the CoAP payload.

The external_aad is an empty string.

An example is given in Figure 12

```
16(
  [
    / protected / h'a2010c04421234' / {
        \ alg \ 1:12, \ AES-CCM-64-64-128 \
        \ kid \ 4: h'1234'
      } / ,
    / unprotected / {
      / iv / 5:h'89f52f65a1c580',
      / countersign / 7:[
        / protected / h'a10126' / {
          \ alg \ 1:-7
        } / ,
        / unprotected / {
          / kid / 4:h'11'
        },
        / signature / SIG / 64 bytes signature /
      ]
    },
    / ciphertext / h'8df0a3b62fccff37aa313c8020e971f8aC8d'
  ]
)
```

Figure 12: Example of COSE Object sent in the payload of a PUBLISH operation

The encryption and decryption operations are described in sections 5.3 and 5.4 of [RFC8152].

7. Security Considerations

In the profile described above, the Publisher and Subscriber use asymmetric crypto, which would make the message exchange quite heavy for small constrained devices. Moreover, all Subscribers must be able to access the public keys of all the Publishers to a specific topic to be able to verify the publications. Such a database could be set up and managed by the same entity having control of the topic, i.e. AS2.

An application where it is not critical that only authorized Publishers can publish on a topic may decide not to make use of the asymmetric crypto and only use symmetric encryption/MAC to confidentiality and integrity protect the publication, but this is not recommended since, as a result, any authorized Subscribers with access to the Broker may forge unauthorized publications without being detected. In this symmetric case the Subscribers would only need one symmetric key per topic, and would not need to know any information about the Publishers, that can be anonymous to it and the Broker.

Subscribers can be excluded from future publications through re-keying for a certain topic. This could be set up to happen on a regular basis, for certain applications. How this could be done is out of scope for this work.

The Broker is only trusted with verifying that the Publisher is authorized to publish, but is not trusted with the publications itself, which it cannot read nor modify. In this setting, caching of publications on the Broker is still allowed.

TODO: expand on security and privacy considerations

8. IANA Considerations

8.1. ACE Groupcomm Profile Registry

The following registrations are done for the "ACE Groupcomm Profile" Registry following the procedure specified in [I-D.ietf-ace-key-groupcomm].

Note to RFC Editor: Please replace all occurrences of "[[This document]]" with the RFC number of this specification and delete this paragraph.

8.1.1. CoAP Profile Registration

Name: coap_pubsub_app

Description: Profile for delegating client authentication and authorization for publishers and subscribers in a CoAP pub-sub setting scenario in a constrained environment.

CBOR Key: TBD

Reference: [[This document]]

8.1.2. CoAP Profile Registration

Name: mqtt_pubsub_app

Description: Profile for delegating client authentication and authorization for publishers and subscribers in a MQTT pub-sub setting scenario in a constrained environment.

CBOR Key: TBD

Reference: [[This document]]

8.2. ACE Groupcomm Key Registry

The following registrations are done for the ACE Groupcomm Key Registry following the procedure specified in [I-D.ietf-ace-key-groupcomm].

Note to RFC Editor: Please replace all occurrences of "[[This document]]" with the RFC number of this specification and delete this paragraph.

Name: COSE_Key

Key Type Value: TBD

Profile: coap_pubsub_app

Description: COSE_Key object

References: [RFC8152], [[This document]]

9. References

9.1. Normative References

[I-D.ietf-ace-key-groupcomm]
Palombini, F. and M. Tiloca, "Key Provisioning for Group Communication using ACE", draft-ietf-ace-key-groupcomm-07 (work in progress), June 2020.

[I-D.ietf-ace-oauth-authz]
Seitz, L., Selander, G., Wahlstroem, E., Erdtman, S., and H. Tschofenig, "Authentication and Authorization for Constrained Environments (ACE) using the OAuth 2.0 Framework (ACE-OAuth)", draft-ietf-ace-oauth-authz-35 (work in progress), June 2020.

[I-D.ietf-core-coap-pubsub]
Koster, M., Keranen, A., and J. Jimenez, "Publish-Subscribe Broker for the Constrained Application Protocol (CoAP)", draft-ietf-core-coap-pubsub-09 (work in progress), September 2019.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC6749] Hardt, D., Ed., "The OAuth 2.0 Authorization Framework", RFC 6749, DOI 10.17487/RFC6749, October 2012, <https://www.rfc-editor.org/info/rfc6749>.

[RFC7049] Bormann, C. and P. Hoffman, "Concise Binary Object Representation (CBOR)", RFC 7049, DOI 10.17487/RFC7049, October 2013, <https://www.rfc-editor.org/info/rfc7049>.

[RFC8152] Schaad, J., "CBOR Object Signing and Encryption (COSE)", RFC 8152, DOI 10.17487/RFC8152, July 2017, <https://www.rfc-editor.org/info/rfc8152>.

9.2. Informative References

[I-D.ietf-ace-actors]
Gerdes, S., Seitz, L., Selander, G., and C. Bormann, "An architecture for authorization in constrained environments", draft-ietf-ace-actors-07 (work in progress), October 2018.

[I-D.ietf-ace-dtls-authorize]
Gerdes, S., Bergmann, O., Bormann, C., Selander, G., and L. Seitz, "Datagram Transport Layer Security (DTLS) Profile for Authentication and Authorization for Constrained Environments (ACE)", draft-ietf-ace-dtls-authorize-11 (work in progress), June 2020.

[I-D.ietf-ace-oscore-profile]
Palombini, F., Seitz, L., Selander, G., and M. Gunnarsson, "OSCORE profile of the Authentication and Authorization for Constrained Environments Framework", draft-ietf-ace-oscore-profile-11 (work in progress), June 2020.

[RFC8259] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", STD 90, RFC 8259, DOI 10.17487/RFC8259, December 2017, <https://www.rfc-editor.org/info/rfc8259>.

Appendix A. Requirements on Application Profiles

This section lists the specifications on this profile based on the requirements defined in Appendix A of [I-D.ietf-ace-key-groupcomm]

- REQ1: Specify the encoding and value of the identifier of group or topic of 'scope': see Section 3.1).

- REQ2: Specify the encoding and value of roles of 'scope': see Section 3.1).

- REQ3: Optionally, specify the acceptable values for 'sign_alg': TODO

- REQ4: Optionally, specify the acceptable values for 'sign_parameters': TODO

- REQ5: Optionally, specify the acceptable values for 'sign_key_parameters': TODO

- REQ6: Optionally, specify the acceptable values for 'pub_key_enc': TODO

- REQ7: Specify the exact format of the 'key' value: COSE_Key, see Section 3.1.

- REQ8: Specify the acceptable values of 'kty' : "COSE_Key", see Section 3.1.

- REQ9: Specity the format of the identifiers of group members: TODO

- REQ10: Optionally, specify the format and content of 'group_policies' entries: not defined

- REQ11: Specify the communication protocol the members of the group must use: CoAP pub/sub.

- REQ12: Specify the security protocol the group members must use to protect their communication. This must provide encryption, integrity and replay protection: Object Security of Content using COSE, see Section 6.1.

- REQ13: Specify and register the application profile identifier : "coap_pubsub_app", see Section 8.1.

- REQ14: Optionally, specify the encoding of public keys, of 'client_cred', and of 'pub_keys' if COSE_Keys are not used: NA.

- REQ15: Specify policies at the KDC to handle id that are not included in get_pub_keys: TODO

- REQ16: Specify the format and content of 'group_policies': TODO

- REQ17: Specify the format of newly-generated individual keying material for group members, or of the information to derive it, and corresponding CBOR label : not defined

- REQ18: Specify how the communication is secured between Client and KDC. Optionally, specify tranport profile of ACE

[I-D.ietf-ace-oauth-authz] to use between Client and KDC: pre-set, as KDC is AS.

- OPT1: Optionally, specify the encoding of public keys, of 'client_cred', and of 'pub_keys' if COSE_Keys are not used: NA

- OPT2: Optionally, specify the negotiation of parameter values for signature algorithm and signature keys, if 'sign_info' and 'pub_key_enc' are not used: NA

- OPT3: Optionally, specify the format and content of 'mgt_key_material': not defined

- OPT4: Optionally, specify policies that instruct clients to retain unsuccessfully decrypted messages and for how long, so that they can be decrypted after getting updated keying material: not defined

Acknowledgments

The author wishes to thank Ari Keraenen, John Mattsson, Ludwig Seitz, Goeran Selander, Cigdem Sengul, Jim Schaad and Marco Tiloca for the useful discussion and reviews that helped shape this document.

Author's Address

Francesca Palombini
Ericsson

Email: francesca.palombini@ericsson.com