ACE Working Group M. Tiloca
Internet-Draft RISE AB
Intended status: Standards Track J. Park
Expires: 13 January 2022 Universitaet Duisburg-Essen
F. Palombini
Ericsson AB
12 July 2021

Key Management for OSCORE Groups in ACE
draft-ietf-ace-key-groupcomm-oscore-11

Abstract

This document defines an application profile of the ACE framework for Authentication and Authorization, to request and provision keying material in group communication scenarios that are based on CoAP and secured with Group Object Security for Constrained RESTful Environments (OSCORE). This application profile delegates the authentication and authorization of Clients that join an OSCORE group through a Resource Server acting as Group Manager for that group. This application profile leverages protocol-specific transport profiles of ACE to achieve communication security, server authentication and proof-of-possession for a key owned by the Client and bound to an OAuth 2.0 Access Token.

Status of This Memo

This Internet-Draft will expire on 13 January 2022.

Copyright Notice

Table of Contents

1. Introduction

Object Security for Constrained RESTful Environments (OSCORE) [RFC8613] is a method for application-layer protection of the Constrained Application Protocol (CoAP) [RFC7252], using CBOR Object Signing and Encryption (COSE) [I-D.ietf-cose-rfc8152bis-struct][I-D.ietf-cose-rfc8152bis-algs] and enabling end-to-end security of CoAP payload and options.

As described in [I-D.ietf-core-oscore-groupcomm], Group OSCORE is used to protect CoAP group communication over IP multicast [I-D.ietf-core-groupcomm-bis]. This relies on a Group Manager, which is responsible for managing an OSCORE group and enables the group members to exchange CoAP messages secured with Group OSCORE. The Group Manager can be responsible for multiple groups, coordinates the joining process of new group members, and is entrusted with the distribution and renewal of group keying material.

This document is an application profile of [I-D.ietf-ace-key-groupcomm], which itself builds on the ACE framework for Authentication and Authorization [I-D.ietf-ace-oauth-authz]. Message exchanges among the participants as well as message formats and processing follow what specified in [I-D.ietf-ace-key-groupcomm] for provisioning and renewing keying material in group communication scenarios, where Group OSCORE is used to protect CoAP group communication over IP multicast.

1.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119][RFC8174] when, and only when, they appear in all capitals, as shown here.

Readers are expected to be familiar with:

- The terms and concepts described in the ACE framework for authentication and authorization [I-D.ietf-ace-oauth-authz] and in the Authorization Information Format (AIF) [I-D.ietf-ace-aif] to express authorization information. The terminology for entities in the considered architecture is defined in OAuth 2.0 [RFC6749]. In particular, this includes Client (C), Resource Server (RS), and Authorization Server (AS).

- The terms and concept related to the message formats and processing specified in [I-D.ietf-ace-key-groupcomm], for provisioning and renewing keying material in group communication scenarios.

- The terms and concepts described in CBOR [RFC8949] and COSE [I-D.ietf-cose-rfc8152bis-struct][I-D.ietf-cose-rfc8152bis-algs].

- The terms and concepts described in CoAP [RFC7252] and group communication for CoAP [I-D.ietf-core-groupcomm-bis]. Unless otherwise indicated, the term "endpoint" is used here following its OAuth definition, aimed at denoting resources such as /token and /introspect at the AS, and /authz-info at the RS. This document does not use the CoAP definition of "endpoint", which is "An entity participating in the CoAP protocol".

- The terms and concepts for protection and processing of CoAP messages through OSCORE [RFC8613] and through Group OSCORE [I-D.ietf-core-oscore-groupcomm] in group communication scenarios. These include the concept of Group Manager, as the entity responsible for a set of groups where communications are secured with Group OSCORE. In this document, the Group Manager acts as Resource Server.

Additionally, this document makes use of the following terminology.

- Requester: member of an OSCORE group that sends request messages to other members of the group.

- Responder: member of an OSCORE group that receives request messages from other members of the group. A responder may reply back, by sending a response message to the requester which has sent the request message.

- Monitor: member of an OSCORE group that is configured as responder and never replies back to requesters after receiving request messages. This corresponds to the term "silent server" used in [I-D.ietf-core-oscore-groupcomm].

- Signature verifier: entity external to the OSCORE group and intended to verify the signature of messages exchanged in the group (see Sections 3.1 and 8.5 of [I-D.ietf-core-oscore-groupcomm]). An authorized signature verifier does not join the OSCORE group as an actual member, yet it can retrieve the public keys of the current group members from the Group Manager.

- Signature-only group: an OSCORE group that uses only the group mode (see Section 8 of [I-D.ietf-core-oscore-groupcomm]).

- Pairwise-only group: an OSCORE group that uses only the pairwise mode (see Section 9 of [I-D.ietf-core-oscore-groupcomm]).

2. Protocol Overview

Group communication for CoAP over IP multicast has been enabled in [I-D.ietf-core-groupcomm-bis] and can be secured with Group Object Security for Constrained RESTful Environments (OSCORE) [RFC8613] as described in [I-D.ietf-core-oscore-groupcomm]. A network node joins an OSCORE group by interacting with the responsible Group Manager. Once registered in the group, the new node can securely exchange messages with other group members.

This document describes how to use [I-D.ietf-ace-key-groupcomm] and [I-D.ietf-ace-oauth-authz] to perform a number of authentication, authorization and key distribution actions, as defined in Section 2 of [I-D.ietf-ace-key-groupcomm], for an OSCORE group.

With reference to [I-D.ietf-ace-key-groupcomm]:

- The node wishing to join the OSCORE group, i.e., the joining node, is the Client.

- The Group Manager is the Key Distribution Center (KDC), acting as a Resource Server.

- The Authorization Server associated to the Group Manager is the AS.

All communications between the involved entities MUST be secured.

In particular, communications between the Client and the Group Manager leverage protocol-specific transport profiles of ACE to achieve communication security, proof-of-possession and server authentication. It is expected that, in the commonly referred base-case of this document, the transport profile to use is pre-configured and well-known to nodes participating in constrained applications.

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

2.1. Overview of the Joining Process

A node performs the steps described in Section 4.3 of [I-D.ietf-ace-key-groupcomm] in order to join an OSCORE group. The format and processing of messages exchanged among the participants are further specified in Section 4 and Section 6 of this document.

2.2. Overview of the Group Rekeying Process

In a number of cases, the Group Manager has to generate new keying material and distribute it to the group (rekeying), as also discussed in Section 3.2 of [I-D.ietf-core-oscore-groupcomm].

To this end the Group Manager MUST support the Group Rekeying Process described in Section 20 of this document. Future application profiles may define alternative rekeying message formats and group rekeying schemes, which MUST comply with the functional steps defined in Section 3.2 of [I-D.ietf-core-oscore-groupcomm].

Upon generating the new group keying material and before starting its distribution, the Group Manager MUST increment the version number of the group keying material. When rekeying a group, the Group Manager MUST preserve the current value of the OSCORE Sender ID of each member in that group.

The data distributed to a group through a rekeying MUST include:

* The new version number of the group keying material for the group.

* A new Group Identifier (Gid) for the group as introduced in [I-D.ietf-ace-key-groupcomm], used as ID Context parameter of the Group OSCORE Common Security Context of that group (see Section 2 of [I-D.ietf-core-oscore-groupcomm]).

  Note that the Gid differs from the group name also introduced in [I-D.ietf-ace-key-groupcomm], which is a plain, stable and invariant identifier, with no cryptographic relevance and meaning.

* A new value for the Master Secret parameter of the Group OSCORE Common Security Context of the group (see Section 2 of [I-D.ietf-core-oscore-groupcomm]).

- A set of stale Sender IDs, which allows each rekeyed node to purge public keys and Recipient Contexts used in the group and associated to those Sender IDs. This in turn allows every group member to rely on owned public keys to confidently assert the group membership of other sender nodes, when receiving protected messages in the group (see Section 3.2 of [I-D.ietf-core-oscore-groupcomm]). More details on the maintenance of stale Sender IDs are provided in Section 2.2.1.

Also, the data distributed through a group rekeying MAY include a new value for the Master Salt parameter of the Group OSCORE Common Security Context of that group.

The Group Manager MUST rekeying the group in the following cases.

- The application requires backward security - In this case, the group is rekeyed when a node joins the group as a new member. Therefore, a joining node cannot access communications in the group prior its joining.

- One or more nodes leave the group - That is, the group is rekeyed when one or more current members spontaneously request to leave the group (see Section 18), or when the Group Manager forcibly evicts them from the group, e.g., due to expired or revoked authorization (see Section 19). Therefore, a leaving node cannot access communications in the group after its leaving, thus ensuring forward security in the group.

  Due to the set of stale Sender IDs distributed through the rekeying, this ensures that a node owning the latest group keying material does not store the public keys of former group members (see Sections 3.2 and 10.1 of [I-D.ietf-core-oscore-groupcomm]).

- Extension of group lifetime - That is, the group is rekeyed when the expiration time for the group keying material approaches or has passed, if it is appropriate to extend the group operation beyond that.

The Group Manager MAY rekey the group for other reasons, e.g., according to an application-dependent rekeying period or scheduling.

2.2.1. Stale OSCORE Sender IDs

Throughout the lifetime of every group, the Group Manager MUST maintain a collection of stale Sender IDs for that group.

The collection associated to a group MUST include up to N > 1 ordered sets of stale OSCORE Sender IDs. It is up to the application to specify the value of N, possibly on a per-group basis.

The N-th set includes the Sender IDs that have become "stale" under the current version V of the group keying material. The (N-1)-th set refers to the immediately previous version (V - 1) of the group keying material, and so on.

In the following cases, the Group Manager MUST add a new element to the most recent set X, i.e., the set associated to the current version V of the group keying material.

- When a current group member obtains a new Sender ID, its old Sender ID is added to X. This happens when the Group Manager assigns a new Sender ID upon request from the group member (see Section 9), or in case the group member re-joins the group (see Section 6.2 and Section 6.4), thus also obtaining a new Sender ID.

- When a current group member leaves the group, its current Sender ID is added to X. This happens when a group member requests to leave the group (see Section 18) or is forcibly evicted from the group (see Section 19).

The value of N can change throughout the lifetime of the group. If the new value N' is smaller than N, the Group Manager MUST preserve the (up to) N' most recent sets in the collection and MUST delete any possible older set from the collection.

Finally, the Group Manager MUST perform the following actions, when the group is rekeyed and the group shifts to the next version V' = (V + 1) of the group keying material.

1. The Group Manager rekeys the group. This includes also distributing the set of stale Sender IDs X associated to the old group keying material with version V (see Section 2.2).

2. After completing the group rekeying, the Group Manager creates a new empty set X' associated to the new version V' of the newly established group keying material, i.e., V' = (V + 1).

3. If the current collection of stale Sender IDs has size N, the Group Manager deletes the oldest set in the collection.

4. The Group Manager adds the new set X' to the collection of stale Sender IDs, as the most recent set.

3. Format of Scope

Building on Section 3.1 of [I-D.ietf-ace-key-groupcomm], this section defines the exact format and encoding of scope to use.

To this end, this profile uses the Authorization Information Format (AIF) [I-D.ietf-ace-aif], and defines the following AIF specific data model AIF-OSCORE-GROUPCOMM.

With reference to the generic AIF model

```
AIF-Generic<Toid, Tperm> = [* [Toid, Tperm]]
```

the value of the CBOR byte string used as scope encodes the CBOR array [* [Toid, Tperm]], where each [Toid, Tperm] element corresponds to one scope entry.

Then, for each scope entry:

* the object identifier ("Toid") is specialized as a CBOR text string, specifying the group name for the scope entry;

* the permission set ("Tperm") is specialized as a CBOR unsigned integer with value R, specifying the role(s) that the client wishes to take in the group (REQ2). The value R is computed as follows:

  - each role in the permission set is converted into the corresponding numeric identifier X from the "Value" column of the table in Figure 1.

  - the set of N numbers is converted into the single value R, by taking each numeric identifier X_1, X_2, ..., X_N to the power of two, and then computing the inclusive OR of the binary representations of all the power values.

Name	Value	Description
Reserved	0	This value is reserved
Requester	1	Send requests; receive responses
Responder	2	Send responses; receive requests
Monitor	3	Receive requests; never send requests/responses
Verifier	4	Verify signature of intercepted messages

Figure 1: Numeric identifier of roles in the OSCORE group

The CDDL [RFC8610] definition of the AIF-OSCORE-GROUPCOMM data model is as follows:

```
AIF-OSCORE-GROUPCOMM = AIF_Generic<path, permissions>

path = tstr  ; Group name
permissions = uint . bits roles
roles = &(
   Requester: 1,
   Responder: 2,
   Monitor: 3,
   Verifier: 4
)
```

Future specifications that define new roles MUST register a corresponding numeric identifier in the "Group OSCORE Roles" Registry defined in Section 23.11 of this document.

4. Joining Node to Authorization Server

This section describes how the joining node interacts with the AS in order to be authorized to join an OSCORE group under a given Group Manager. In particular, it considers a joining node that intends to contact that Group Manager for the first time.

The message exchange between the joining node and the AS consists of the messages Authorization Request and Authorization Response defined in Section 3 of [I-D.ietf-ace-key-groupcomm]. Note that what is defined in [I-D.ietf-ace-key-groupcomm] applies, and only additions or modifications to that specification are defined here.

4.1. Authorization Request

The Authorization Request message is as defined in Section 3.1 of [I-D.ietf-ace-key-groupcomm], with the following additions.

* If the 'scope' parameter is present:

  - The value of the CBOR byte string encodes a CBOR array, whose format MUST follow the data model AIF-OSCORE-GROUPCOMM defined in Section 3. In particular, for each OSCORE group to join:

    o The group name is encoded as a CBOR text string.

    o The set of requested roles is expressed as a single CBOR unsigned integer, computed as defined in Section 3 (REQ2) from the numerical abbreviations defined in Figure 1 for each requested role (OPT7).

4.2. Authorization Response

The Authorization Response message is as defined in Section 3.2 of [I-D.ietf-ace-key-groupcomm], with the following additions:

* The AS MUST include the 'expires_in' parameter. Other means for the AS to specify the lifetime of Access Tokens are out of the scope of this document.

* The AS MUST include the 'scope' parameter, when the value included in the Access Token differs from the one specified by the joining node in the request. In such a case, the second element of each scope entry MUST be present, and specifies the set of roles that the joining node is actually authorized to take in the OSCORE group for that scope entry, encoded as specified in Section 4.1.

Furthermore, if the AS uses the extended format of scope defined in Section 6 of [I-D.ietf-ace-key-groupcomm] for the 'scope' claim of the Access Token, the first element of the CBOR sequence [RFC8742] MUST be the CBOR integer with value SEM_ID_TBD, defined in Section 23.13 of this document (REQ24). This indicates that the second element of the CBOR sequence, as conveying the actual access control information, follows the scope semantics defined for this application profile in Section 3 of this document.

5. Interface at the Group Manager

The Group Manager provides the interface defined in Section 4.1 of [I-D.ietf-ace-key-groupcomm], with the additional sub-resources defined from Section 5.1 to Section 5.4 of this document.

Furthermore, Section 5.5 provides a summary of the CoAP methods admitted to access different resources at the Group Manager, for nodes with different roles in the group or as non members (REQ8).

The GROUPNAME segment of the URI path MUST match with the group name specified in the scope entry of the Access Token scope (i.e., 'gname' in Section 3.1 of [I-D.ietf-ace-key-groupcomm]) (REQ1).

The Resource Type (rt=) Link Target Attribute value "core.osc.gm" is registered in Section 23.12 (REQ7), and can be used to describe group-membership resources and its sub-resources at a Group Manager, e.g., by using a link-format document [RFC6690].

Applications can use this common resource type to discover links to group-membership resources for joining OSCORE groups, e.g., by using the approach described in [I-D.tiloca-core-oscore-discovery].

5.1. ace-group/GROUPNAME/active

This resource implements a GET handler.

5.1.1. GET Handler

The handler expects a GET request.

The handler verifies that the group name in the /ace-group/GROUPNAME/active path is a subset of the 'scope' stored in the Access Token associated to the requesting client.

The handler also verifies that the roles granted to the requesting client in the group allow it to perform this operation on this resource (REQ8). If either verification fails, the Group Manager MUST respond with a 4.01 (Unauthorized) error message.

Additionally, the handler verifies that the requesting client is a current member of the group. If verification fails, the Group Manager MUST respond with a 4.01 (Unauthorized) error message. The response MUST have Content-Format set to application/ace-groupcomm+cbor and is formatted as defined in Section 4 of [I-D.ietf-ace-key-groupcomm]. The value of the 'error' field MUST be set to 0 ("Operation permitted only to group members").

If the verifications above succeed, the handler returns a 2.05 (Content) message, specifying the current status of the group, i.e., active or inactive. The payload of the response is formatted as defined in Section 16.

The method to set the current group status is out of the scope of this document, and is defined for the administrator interface of the Group Manager specified in [I-D.ietf-ace-oscore-gm-admin].

5.2. ace-group/GROUPNAME/gm-pub-key

This resource implements a GET handler.

5.2.1. GET Handler

The handler expects a GET request.

The handler verifies that the group name in the /ace-group/GROUPNAME/gm-pub-key path is a subset of the 'scope' stored in the Access Token associated to the requesting client.

The handler also verifies that the roles granted to the requesting client allow it to perform this operation on this resource (REQ8). If either verification fails, the Group Manager MUST respond with a 4.01 (Unauthorized) error message.

If the requesting client is not a current group member and GROUPNAME denotes a pairwise-only group, the Group Manager MUST respond with a 4.00 (Bad Request) error message. The response MUST have Content-Format set to application/ace-groupcomm+cbor and is formatted as defined in Section 4 of [I-D.ietf-ace-key-groupcomm]. The value of the 'error' field MUST be set to 7 ("Signatures not used in the group").

If the verifications above succeed, the handler returns a 2.05 (Content) message, specifying the Group Manager's public key together with a proof-of-possession evidence. The response MUST have Content-Format set to application/ace-groupcomm+cbor. The payload of the response is a CBOR map, which is formatted as defined in Section 12.

5.3. ace-group/GROUPNAME/verif-data

This resource implements a GET handler.

5.3.1. GET Handler

The handler expects a GET request.

The handler verifies that the group name in the /ace-group/GROUPNAME/verif-data path is a subset of the 'scope' stored in the Access Token associated to the requesting client.

The handler also verifies that the roles granted to the requesting client allow it to perform this operation on this resource (REQ8). If either verification fails, the Group Manager MUST respond with a 4.01 (Unauthorized) error message.

If the requesting client is a current group member, the Group Manager MUST respond with a 4.01 (Unauthorized) error message. The response MUST have Content-Format set to application/ace-groupcomm+cbor and is formatted as defined in Section 4 of [I-D.ietf-ace-key-groupcomm]. The value of the 'error' field MUST be set to 8 ("Operation permitted only to signature verifiers").

If GROUPNAME denotes a pairwise-only group, the Group Manager MUST respond with a 4.00 (Bad Request) error message. The response MUST have Content-Format set to application/ace-groupcomm+cbor and is formatted as defined in Section 4 of [I-D.ietf-ace-key-groupcomm]. The value of the 'error' field MUST be set to 7 ("Signatures not used in the group").

If the verifications above succeed, the handler returns a 2.05 (Content) message, specifying data that allows also a signature verifier to verify signatures of messages protected with the group mode and sent to the group (see Sections 3.1 and 8.5 of [I-D.ietf-core-oscore-groupcomm]). The response MUST have Content-Format set to application/ace-groupcomm+cbor. The payload of the response is a CBOR map, which is formatted as defined in Section 13.

5.4. ace-group/GROUPNAME/stale-sids

This resource implements a FETCH handler.

5.4.1. FETCH Handler

The handler expects a FETCH request, whose payload specifies a version number of the group keying material, encoded as an unsigned CBOR integer.

The handler verifies that the group name in the /ace-group/GROUPNAME/stale-sids path is a subset of the 'scope' stored in the Access Token associated to the requesting client.

The handler also verifies that the roles granted to the requesting client allow it to perform this operation on this resource (REQ8). If either verification fails, the Group Manager MUST respond with a 4.01 (Unauthorized) error message.

Additionally, the handler verifies that the requesting client is a current member of the group. If verification fails, the Group Manager MUST respond with a 4.01 (Unauthorized) error message. The response MUST have Content-Format set to application/ace-groupcomm+cbor and is formatted as defined in Section 4 of [I-D.ietf-ace-key-groupcomm]. The value of the 'error' field MUST be set to 0 ("Operation permitted only to group members").

If the verifications above succeed, the handler returns a 2.05 (Content) message, specifying data that allows the requesting client to delete the Recipient Contexts and public keys associated to former members of the group (see Section 3.2 of [I-D.ietf-core-oscore-groupcomm]. The payload of the response is formatted as defined in Section 20.3.1.

5.5. Admitted Methods

The table in Figure 2 summarizes the CoAP methods admitted to access different resources at the Group Manager, for (non-)members of a group with group name GROUPNAME, and considering different roles. The last two rows of the table apply to a node with node name NODENAME.

Resource	Type1	Type2	Type3	Type4
ace-group/	F	F	F	-
ace-group/GROUPNAME/	G Po	G Po	Po *	Po
ace-group/GROUPNAME/active	G	G	-	-
ace-group/GROUPNAME/gm-pub-key	G	G	G	-
ace-group/GROUPNAME/verif-data	-	-	G	-
ace-group/GROUPNAME/pub-key	G F	G F	G F	-
ace-group/GROUPNAME/stale-sids	F	F	-	-
ace-group/GROUPNAME/policies	G	G	-	-
ace-group/GROUPNAME/num	G	G	-	-
ace-group/GROUPNAME/nodes/NODENAME	G Pu D	G D	-	-
ace-group/GROUPNAME/nodes/NODENAME/pub-key	Po	-	-	-

Type1 = Member as Requester and/or Responder
Type2 = Member as Monitor
Type3 = Non-member (authorized to be Verifier)
(*) = cannot join the group as Verifier
Type4 = Non-member (not authorized to be Verifier)

G = GET
F = FETCH
Po = POST
Pu = PUT
D = DELETE

Figure 2: Admitted CoAP Methods on the Group Manager Resources

6. Token POST and Group Joining

The rest of this section describes the interactions between the joining node and the Group Manager, i.e., the sending of the Access Token and the Request-Response exchange to join the OSCORE group. The message exchange between the joining node and the Group Manager consists of the messages defined in Sections 3.3 and 4.3 of [I-D.ietf-ace-key-groupcomm]. Note that what is defined in [I-D.ietf-ace-key-groupcomm] applies, and only additions or modifications to that specification are defined here.

A signature verifier provides the Group Manager with an Access Token, as described in Section 6.1, just as any another joining node does. However, unlike candidate group members, it does not join any OSCORE group, i.e., it does not perform the joining process defined in Section 6.2. After successfully posting an Access Token, a signature verifier is authorized to perform only the operations specified in Section 10, to retrieve the public keys of group members, and only for the OSCORE groups specified in the validated Access Token. The Group Manager MUST respond with a 4.01 (Unauthorized) error message, in case a signature verifier attempts to access any other endpoint than /ace-group/GROUPNAME/pub-key at the Group Manager.

6.1. Token Post

The Token post exchange is defined in Section 3.3 of [I-D.ietf-ace-key-groupcomm].

Additionally to what defined in [I-D.ietf-ace-key-groupcomm], the following applies.

- The CoAP POST request MAY additionally contain the following parameters, which, if included, MUST have the corresponding values:
    - 'ecdh_info' defined in Section 6.1.1, encoding the CBOR simple value Null to require information on the ECDH algorithm, the ECDH algorithm parameters, the ECDH key parameters and on the exact encoding of public keys used in the group, in case the joining node supports the pairwise mode of Group OSCORE [I-D.ietf-core-oscore-groupcomm].
    - 'gm_dh_pub_keys' defined in Section 6.1.2, encoding the CBOR simple value Null to require the Diffie-Hellman public key of the Group Manager in the group, in case the joining node supports the pairwise mode of Group OSCORE [I-D.ietf-core-oscore-groupcomm].

  Alternatively, the joining node may retrieve this information by other means.

- The 'kdcchallenge' parameter contains a dedicated nonce N_S generated by the Group Manager. For the N_S value, it is RECOMMENDED to use a 8-byte long random nonce. The joining node can use this nonce in order to prove the possession of its own private key, upon joining the group (see Section 6.2).

The 'kdcchallenge' parameter MAY be omitted from the 2.01 (Created) response, if the 'scope' of the Access Token specifies only the role "monitor" or only the role "verifier" or both of them, for each and every of the specified groups.

* If the 'sign_info' parameter is present in the response, the following applies for each element 'sign_info_entry'.

  - 'id' MUST NOT refer to OSCORE groups that are pairwise-only groups.

  - 'sign_alg' takes value from the "Value" column of the "COSE Algorithms" Registry [COSE.Algorithms].

  - 'sign_parameters' is a CBOR array. Its format and value are the same of the COSE capabilities array for the algorithm indicated in 'sign_alg', as specified for that algorithm in the "Capabilities" column of the "COSE Algorithms" Registry [COSE.Algorithms] (REQ4).

  - 'sign_key_parameters' is a CBOR array. Its format and value are the same of the COSE capabilities array for the COSE key type of the keys used with the algorithm indicated in 'sign_alg', as specified for that key type in the "Capabilities" column of the "COSE Key Types" Registry [COSE.Key.Types] (REQ5).

  - 'pub_key_enc' takes value from the "Label" column of the "COSE Header Parameters" Registry [COSE.Header.Parameters] (REQ6). Consistently with Section 2.3 of [I-D.ietf-core-oscore-groupcomm], acceptable values denote an encoding that MUST explicitly provide the full set of information related to the public key algorithm, including, e.g., the used elliptic curve (when applicable).

    At the time of writing this specification, acceptable public key encodings are CWTs [RFC8392], unprotected CWT claim sets [I-D.ietf-rats-uccs], X.509 certificates [RFC7925] and C509 certificates [I-D.ietf-cose-cbor-encoded-cert]. Further encodings may be available in the future, and would be acceptable to use as long as they comply with the criteria defined above.

    [As to CWTs and unprotected CWT claim sets, there is a pending registration requested by draft-ietf-lake-edhoc.]

    [As to C509 certificates, there is a pending registration requested by draft-ietf-cose-cbor-encoded-cert.]

This format is consistent with every signature algorithm currently considered in [I-D.ietf-cose-rfc8152bis-algs], i.e., with algorithms that have only the COSE key type as their COSE capability. Appendix B of [I-D.ietf-ace-key-groupcomm] describes how the format of each 'sign_info_entry' can be generalized for possible future registered algorithms having a different set of COSE capabilities.

* If 'ecdh_info' is included in the request, the Group Manager MAY include in the response the 'ecdh_info' parameter defined in Section 6.1.1, with the same encoding. Note that the field 'id' takes as value the group name, or array of group names, for which the corresponding 'ecdh_info_entry' applies to.

* If 'gm_dh_pub_keys' is included in the request and any of the groups that the client has been authorized to join is a pairwise-only group, then the Group Manager MUST include in the response the 'gm_dh_pub_keys' parameter defined in Section 6.1.2, with the same encoding. Otherwise, the Group Manager MAY include the 'gm_dh_pub_keys' parameter. Note that the field 'id' takes as value the group name, or array of group names, for which the corresponding 'gm_dh_pub_keys' applies to.

Note that, other than through the above parameters as defined in Section 3.3 of [I-D.ietf-ace-key-groupcomm], the joining node MAY have previously retrieved this information by other means. For example, information conveyed in the 'sign_info' and 'ecdh_info' parameters can be obtained by using the approach described in [I-D.tiloca-core-oscore-discovery], to discover the OSCORE group and the link to the associated group-membership resource at the Group Manager (OPT1).

Additionally, if allowed by the used transport profile of ACE, the joining node may instead provide the Access Token to the Group Manager by other means, e.g., during a secure session establishment (see Section 3.3.2 of [I-D.ietf-ace-dtls-authorize]).

6.1.1. 'ecdh_info' Parameter

The 'ecdh_info' parameter is an OPTIONAL parameter of the Token Post response message defined in Section 5.10.1 of [I-D.ietf-ace-oauth-authz].

This parameter is used to require and retrieve from the Group Manager information and parameters about the ECDH algorithm and about the public keys to be used in the OSCORE group to compute a static-static Diffie-Hellman shared secret [NIST-800-56A], in case the group uses the pairwise mode of Group OSCORE [I-D.ietf-core-oscore-groupcomm].

When used in the request, the 'ecdh_info' parameter encodes the CBOR simple value Null, to require information and parameters on the ECDH algorithm and on the public keys to be used to compute Diffie-Hellman shared secrets in the OSCORE group.

The CDDL notation [RFC8610] of the 'ecdh_info' parameter formatted as in the request is given below.

```
ecdh_info_req = nil
```

The 'ecdh_info' parameter of the 2.01 (Created) response is a CBOR array of one or more elements. The number of elements is at most the number of OSCORE groups the client has been authorized to join.

Each element contains information about ECDH parameters and about public keys, for one or more OSCORE groups that use the pairwise mode of Group OSCORE and that the client has been authorized to join. Each element is formatted as follows.

* The first element 'id' is the group name of the OSCORE group or an array of group names for the OSCORE groups for which the specified information applies. In particular 'id' MUST NOT refer to OSCORE groups that are signature-only groups.

* The second element 'ecdh_alg' is a CBOR integer or a CBOR text string indicating the ECDH algorithm used in the OSCORE group identified by 'gname'. Values are taken from the "Value" column of the "COSE Algorithms" Registry [COSE.Algorithms].

* The third element 'ecdh_parameters' is a CBOR array indicating the parameters of the ECDH algorithm used in the OSCORE group identified by 'gname'. Its format and value are the same of the COSE capabilities array for the algorithm indicated in 'ecdh_alg', as specified for that algorithm in the "Capabilities" column of the "COSE Algorithms" Registry [COSE.Algorithms].

* The fourth element 'ecdh_key_parameters' is a CBOR array indicating the parameters of the keys used with the ECDH algorithm in the OSCORE group identified by 'gname'. Its content depends on the value of 'ecdh_alg'. In particular, its format and value are the same of the COSE capabilities array for the COSE key type of the keys used with the algorithm indicated in 'ecdh_alg', as specified for that key type in the "Capabilities" column of the "COSE Key Types" Registry [COSE.Key.Types].

* The fifth element 'pub_key_enc' is a CBOR integer indicating the encoding of public keys used in the OSCORE group identified by 'gname'. It takes value from the "Label" column of the "COSE

Header Parameters" Registry [COSE.Header.Parameters] (REQ6). Acceptable values denote an encoding that MUST explicitly provide the full set of information related to the public key algorithm, including, e.g., the used elliptic curve (when applicable). The same considerations and guidelines for the 'pub_key_enc' element of 'sign_info' (see Section 6.1) apply.

The CDDL notation [RFC8610] of the 'ecdh_info' parameter formatted as in the response is given below.

```
ecdh_info_res = [ + ecdh_info_entry ]

ecdh_info_entry =
[
  id : gname / [ + gname ],
  ecdh_alg : int / tstr,
  ecdh_parameters : [ any ],
  ecdh_key_parameters : [ any ],
  pub_key_enc = int
]

gname = tstr
```

This format is consistent with every ECDH algorithm currently considered in [I-D.ietf-cose-rfc8152bis-algs], i.e., with algorithms that have only the COSE key type as their COSE capability. Appendix B of this document describes how the format of each 'ecdh_info_entry' can be generalized for possible future registered algorithms having a different set of COSE capabilities.

6.1.2. 'gm_dh_pub_keys' Parameter

The 'gm_dh_pub_keys' parameter is an OPTIONAL parameter of the Token Post response message defined in Section 5.10.1 of [I-D.ietf-ace-oauth-authz].

This parameter is used to require and retrieve from the Group Manager its Diffie-Hellman public key used in the OSCORE group. The Group Manager has specifically a Diffie-Hellman public key if and only if the OSCORE group is a pairwise-only group. In this case, the early retrieval of the Group Manager's public key is necessary in order for the joining node to prove the possession of its own private key, upon joining the group (see Section 6.2).

When used in the request, the 'gm_dh_pub_keys' parameter encodes the CBOR simple value Null, to require the Diffie-Hellman public key that the Group Manager uses in the OSCORE group.

The CDDL notation [RFC8610] of the 'gm_dh_pub_keys' parameter formatted as in the request is given below.

```
   gm_dh_pub_keys_req = nil
```

The 'gm_dh_pub_keys' parameter of the 2.01 (Created) response is a CBOR array of one or more elements. The number of elements is at most the number of OSCORE groups the client has been authorized to join.

Each element 'gm_dh_pub_keys_entry' contains information about the Group Manager's Diffie-Hellman public keys, for one or more OSCORE groups that are pairwise-only groups and that the client has been authorized to join. Each element is formatted as follows.

* The first element 'id' is the group name of the OSCORE group or an array of group names for the OSCORE groups for which the specified information applies. In particular 'id' MUST refer exclusively to OSCORE groups that are pairwise-only groups.

* The second element 'pub_key_enc' is a CBOR integer indicating the encoding of public keys used in the OSCORE group identified by 'gname'. It takes value from the "Label" column of the "COSE Header Parameters" Registry [COSE.Header.Parameters] (REQ6). Acceptable values denote an encoding that MUST explicitly provide the full set of information related to the public key algorithm, including, e.g., the used elliptic curve (when applicable). The same considerations and guidelines for the 'pub_key_enc' element of 'sign_info' (see Section 6.1) apply.

* The third element 'pub_key' is a CBOR byte string, which encodes the Group Manager's Diffie-Hellman public key in its original binary representation made available to other endpoints in the group. In particular, the original binary representation complies with the encoding specified by the 'pub_key_enc' parameter. Note that the public key provides the full set of information related to its public key algorithm, i.e., the ECDH algorithm used in the OSCORE group as pairwise key agreement algorithm.

The CDDL notation [RFC8610] of the 'gm_dh_pub_keys' parameter formatted as in the response is given below.

```
   gm_dh_pub_keys_res = [ + gm_dh_pub_keys_entry ]

   gm_dh_pub_keys_entry =
   [
     id : gname / [ + gname ],
     pub_key_enc = int,
     pub_key = bstr
   ]

   gname = tstr
```

6.2. Sending the Joining Request

The joining node requests to join the OSCORE group by sending a Joining Request message to the related group-membership resource at the Group Manager, as per Section 4.3 of [I-D.ietf-ace-key-groupcomm].

Additionally to what defined in Section 4.1.2.1 of [I-D.ietf-ace-key-groupcomm], the following applies.

* The 'scope' parameter MUST be included. Its value encodes one scope entry with the format defined in Section 3, indicating the group name and the role(s) that the joining node wants to take in the group.

* The 'get_pub_keys' parameter is present only if the joining node wants to retrieve the public keys of the group members from the Group Manager during the joining process (see Section 7). Otherwise, this parameter MUST NOT be present.

  If this parameter is present and its value is non-null, each element of the inner CBOR array 'role_filter' is encoded as a CBOR unsigned integer, with the same value of a permission set ("Tperm") indicating that role or combination of roles in a scope entry, as defined in Section 3.

* 'cnonce' contains a dedicated nonce N_C generated by the joining node. For the N_C value, it is RECOMMENDED to use a 8-byte long random nonce.

* The proof-of-possession (PoP) evidence included in 'client_cred_verify' is computed as defined below (REQ20). In either case, the N_S used to build the PoP input is as defined in Section 6.2.1.

- If the group is not a pairwise-only group, the PoP evidence MUST be a signature. The joining node computes the signature by using the same private key and signature algorithm it intends to use for signing messages in the OSCORE group.

- If the group is a pairwise-only group, the PoP evidence MUST be a MAC computed as follows, by using the HKDF Algorithm HKDF SHA-256, which consists of composing the HKDF-Extract and HKDF-Expand steps [RFC5869].

  MAC = HKDF(salt, IKM, info, L)

  The input parameters of HKDF are as follows.

  - salt takes as value the empty byte string.

  - IKM is computed as a cofactor Diffie-Hellman shared secret, see Section 5.7.1.2 of [NIST-800-56A], using the ECDH algorithm used in the OSCORE group. The joining node uses its own Diffie-Hellman private key and the Diffie-Hellman public key of the Group Manager. For X25519 and X448, the procedure is described in Section 5 of [RFC7748].

  - info takes as value the PoP input.

  - L is equal to 8, i.e., the size of the MAC, in bytes.

6.2.1. Value of the N_S Challenge

The value of the N_S challenge is determined as follows.

1. If the joining node has posted the Access Token to the /authz-info endpoint of the Group Manager as in Section 6.1, N_S takes the same value of the most recent 'kdcchallenge' parameter received by the joining node from the Group Manager. This can be either the one specified in the 2.01 (Created) response to the Token POST, or the one possibly specified in a 4.00 (Bad Request) response to a following Joining Request (see Section 6.3).

2. If the Token posting has relied on the DTLS profile of ACE [I-D.ietf-ace-dtls-authorize] with the Access Token as content of the "psk_identity" field of the ClientKeyExchange message [RFC6347], N_S is an exporter value computed as defined in Section 7.5 of [RFC8446]. Specifically, N_S is exported from the DTLS session between the joining node and the Group Manager, using an empty 'context_value', 32 bytes as 'key_length', and the exporter label "EXPORTER-ACE-Sign-Challenge-coap-group-oscore-app" defined in Section 23.7 of this document.

It is up to applications to define how N_S is computed in further alternative settings.

Section 22.3 provides security considerations on the reusage of the N_S challenge.

6.3. Receiving the Joining Request

The Group Manager processes the Joining Request as defined in Section 4.1.2.1 of [I-D.ietf-ace-key-groupcomm]. Additionally, the following applies.

* The Group Manager MUST return a 5.03 (Service Unavailable) response in case the OSCORE group that the joining node has been trying to join is currently inactive (see Section 5.1). The response MUST have Content-Format set to application/ace-groupcomm+cbor and is formatted as defined in Section 4 of [I-D.ietf-ace-key-groupcomm]. The value of the 'error' field MUST be set to 9 ("Group currently not active").

* In case the joining node is not going to join the group exclusively as monitor, the Group Manager MUST return a 5.03 (Service Unavailable) response if there are currently no OSCORE Sender IDs available to assign in the OSCORE group. The response MUST have Content-Format set to application/ace-groupcomm+cbor and is formatted as defined in Section 4 of [I-D.ietf-ace-key-groupcomm]. The value of the 'error' field MUST be set to 4 ("No available node identifiers").

* In case the joining node is not going to join the group exclusively as monitor and the Joining Request does not include the 'client_cred' parameter, the joining process fails if the Group Manager either: i) does not store a public key with an accepted format for the joining node; or ii) stores multiple public keys with an accepted format for the joining node.

* In order to verify the PoP evidence contained in 'client_cred_verify', the Group Manager proceeds as follows.

    - As PoP input, the Group Manager uses the value of the 'scope' parameter from the Joining Request as a CBOR byte string, concatenated with N_S encoded as a CBOR byte string, concatenated with N_C encoded as a CBOR byte string. In particular, N_S is determined as described in Section 6.2.1, while N_C is the nonce provided in the 'cnonce' parameter of the Joining Request;

  - As public key of the joining node, the Group Manager uses either the one retrieved from the 'client_cred' parameter of the Joining Request, or the one already stored as acquired from previous interactions with the joining node.

  - The Group Manager verifies the PoP evidence as defined below.

    - If the group is not a pairwise-only group, the PoP evidence is a signature. The Group Manager verifies it by using the public key of the joining node, as well as the signature algorithm used in the OSCORE group and possible corresponding parameters.

    - If the group is a pairwise-only group, the PoP evidence is a MAC. The Group Manager recomputes the MAC through the same process taken by the joining node when preparing the value of the 'client_cred_verify' parameter for the Joining Request (see Section 6.2), with the difference that the Group Manager uses its own Diffie-Hellman private key and the Diffie-Hellman public key of the joining node. The verification succeeds if and only if the recomputed MAC is equal to the MAC conveyed as PoP evidence in the Joining Request.

* A 4.00 (Bad Request) response from the Group Manager to the joining node MUST have content format application/ace-groupcomm+cbor. The response payload is a CBOR map formatted as follows.

  - If the group uses (also) the group mode of Group OSCORE, the CBOR map MUST contain the 'sign_info' parameter, whose CBOR label is defined in Section 7 of [I-D.ietf-ace-key-groupcomm]. This parameter has the same format of 'sign_info_res' defined in Section 3.3.1 of [I-D.ietf-ace-key-groupcomm]. In particular, it includes a single element 'sign_info_entry' pertaining to the OSCORE group that the joining node has tried to join with the Joining Request.

  - If the group uses (also) the pairwise mode of Group OSCORE, the CBOR map MUST contain the 'ecdh_info' parameter, whose CBOR label is defined in Section 23.3. This parameter has the same format of 'ecdh_info_res' defined in Section 6.1.1. In particular, it includes a single element 'ecdh_info_entry' pertaining to the OSCORE group that the joining node has tried to join with the Joining Request.

  - If the group is a pairwise-only group, the CBOR map MUST contain the 'gm_dh_pub_keys' parameter, whose CBOR label is defined in Section 23.3. This parameter has the same format of 'gm_dh_pub_keys_res' defined in Section 6.1.2. In particular, it includes a single element 'gm_dh_pub_keys_entry' pertaining to the OSCORE group that the joining node has tried to join with the Joining Request.

  - The CBOR map MAY include the 'kdcchallenge' parameter, whose CBOR label is defined in Section 7 of [I-D.ietf-ace-key-groupcomm]. If present, this parameter is a CBOR byte string, which encodes a newly generated 'kdcchallenge' value that the Client can use when preparing a Joining Request (see Section 6.2). In such a case the Group Manager MUST store the newly generated value as the 'kdcchallenge' value associated to the joining node, possibly replacing the currently stored value.

* The Group Manager MUST return a 4.00 (Bad Request) response in case the 'scope' parameter is not present in the Joining Request, or if it is present and specifies any set of roles not included in the following list: "requester", "responder", "monitor", ("requester", "responder"). Future specifications that define a new role MUST define possible sets of roles including the new one and existing ones, that are acceptable to specify in the 'scope' parameter of a Joining Request.

* The Group Manager MUST return a 4.00 (Bad Request) response in case the Joining Request includes the 'client_cred' parameter but does not include both the 'cnonce' and 'client_cred_verify' parameters.

* The Group Manager MUST return a 4.00 (Bad Request) response in case an eligible public key for the joining node is neither present in the 'client_cred' parameter nor already stored.

* The Group Manager MAY return a 4.00 (Bad Request) response in case all the following conditions hold.

  - The OSCORE group uses the pairwise mode of Group OSCORE.

  - The OSCORE group uses EdDSA public keys [RFC8032].

  - The public key of the joining node from the 'client_cred' parameter:

    - Is for the elliptic curve Ed25519 and has its Y coordinate equal to -1 or 1 (mod p), with p = (2^255 - 19), see Section 4.1 of [RFC7748]; or

    - Is for the elliptic curve Ed448 and has its Y coordinate equal to -1 or 1 (mod p), with p = (2^448 - 2^224 - 1), see Section 4.2 of [RFC7748].

  This prevents the acceptance of Ed25519 and Ed448 public keys that cannot be successfully converted to Montgomery coordinates, and thus cannot be used for the derivation of pairwise keys (see Section 2.3.1 of [I-D.ietf-core-oscore-groupcomm]).

* When receiving a 4.00 (Bad Request) response, the joining node SHOULD send a new Joining Request to the Group Manager, where:

  - The 'cnonce' parameter MUST include a new dedicated nonce N_C generated by the joining node.

  - The 'client_cred' parameter MUST include a public key compatible with the encoding, signature or ECDH algorithm, and possible associated parameters indicated by the Group Manager.

  - The 'client_cred_verify' parameter MUST include a PoP evidence computed as described in Section 6.2, by using the public key indicated in the current 'client_cred' parameter, with the signature or ECDH algorithm, and possible associated parameters indicated by the Group Manager. If the error response from the Group Manager includes the 'kdcchallenge' parameter, the joining node MUST use its content as new N_S challenge to compute the PoP evidence.

6.4. Sending the Joining Response

If the processing of the Joining Request described in Section 6.3 is successful, the Group Manager updates the group membership by registering the joining node NODENAME as a new member of the OSCORE group GROUPNAME, as described in Section 4.1.2.1 of [I-D.ietf-ace-key-groupcomm].

If the joining node has not taken exclusively the role of monitor, the Group Manager performs also the following actions.

* The Group Manager selects an available OSCORE Sender ID in the OSCORE group, and exclusively assigns it to the joining node. The Group Manager MUST NOT assign an OSCORE Sender ID to the joining node if this joins the group exclusively with the role of monitor, according to what specified in the Access Token (see Section 4.2).

Consistently with Section 3.1 of [I-D.ietf-core-oscore-groupcomm], the Group Manager MUST assign an OSCORE Sender ID that has not been used in the OSCORE group since the latest time when the current Gid value was assigned to the group.

If the joining node is recognized as a current group member, e.g., through the ongoing secure communication association, the following also applies.

- The Group Manager MUST assign a new OSCORE Sender ID different than the one currently used by the joining node in the OSCORE group.

- The Group Manager MUST add the old, relinquished OSCORE Sender ID of the joining node to the most recent set of stale Sender IDs, in the collection associated to the group (see Section 2.2.1).

* The Group Manager stores the association between i) the public key of the joining node; and ii) the Group Identifier (Gid), i.e., the OSCORE ID Context, associated to the OSCORE group together with the OSCORE Sender ID assigned to the joining node in the group. The Group Manager MUST keep this association updated over time.

Then, the Group Manager replies to the joining node, providing the updated security parameters and keying meterial necessary to participate in the group communication. This success Joining Response is formatted as defined in Section 4.1.2.1 of [I-D.ietf-ace-key-groupcomm], with the following additions:

* The 'gkty' parameter identifies a key of type "Group_OSCORE_Input_Material object", defined in Section 23.4 of this document.

* The 'key' parameter includes what the joining node needs in order to set up the Group OSCORE Security Context as per Section 2 of [I-D.ietf-core-oscore-groupcomm].

  This parameter has as value a Group_OSCORE_Input_Material object, which is defined in this document and extends the OSCORE_Input_Material object encoded in CBOR as defined in Section 3.2.1 of [I-D.ietf-ace-oscore-profile]. In particular, it contains the additional parameters 'group_senderId', 'pub_key_enc', 'sign_enc_alg', 'sign_alg', 'sign_params', 'ecdh_alg' and 'ecdh_params' defined in Section 23.6 of this document.

  More specifically, the 'key' parameter is composed as follows.

- The 'hkdf' parameter, if present, has as value the HKDF Algorithm used in the OSCORE group. This parameter MAY be omitted, if the HKDF Algorithm used in the group is HKDF SHA-256. Otherwise, this parameter MUST be present.

- The 'salt' parameter, if present, has as value the OSCORE Master Salt used in the OSCORE group. This parameter MAY be omitted, if the Master Salt used in the group is the empty byte string. Otherwise, this parameter MUST be present.

- The 'ms' parameter includes the OSCORE Master Secret value used in the OSCORE group. This parameter MUST be present.

- The 'contextId' parameter MUST be present and has as value the Group Identifier (Gid), i.e., the OSCORE ID Context of the OSCORE group. This parameter MUST be present.

- The 'group_senderId' parameter, if present, has as value the OSCORE Sender ID assigned to the joining node by the Group Manager, as described above. This parameter MUST NOT be present if the node joins the OSCORE group exclusively with the role of monitor, according to what specified in the Access Token (see Section 4.2). In any other case, this parameter MUST be present.

- The 'pub_key_enc' parameter MUST be present and specifies the encoding of public keys used in the OSCORE group. It takes value from the "Label" column of the "COSE Header Parameters" Registry [COSE.Header.Parameters] (REQ6). Consistently with Section 2.3 of [I-D.ietf-core-oscore-groupcomm], acceptable values denote an encoding that MUST explicitly provide the full set of information related to the public key algorithm, including, e.g., the used elliptic curve (when applicable).

  At the time of writing this specification, acceptable public key encodings are CWTs [RFC8392], unprotected CWT claim sets [I-D.ietf-rats-uccs], X.509 certificates [RFC7925] and C509 certificates [I-D.ietf-cose-cbor-encoded-cert]. Further encodings may be available in the future, and would be acceptable to use as long as they comply with the criteria defined above.

  [As to CWTs and unprotected CWT claim sets, there is a pending registration requested by draft-ietf-lake-edhoc.]

  [As to C509 certificates, there is a pending registration requested by draft-ietf-cose-cbor-encoded-cert.]

- The 'sign_enc_alg' parameter MUST NOT be present if the OSCORE group is a pairwise-only group. Otherwise, it MUST be present and specifies the Signature Encryption Algorithm used in the OSCORE group to encrypt messages protected with the group mode. This parameter takes values from the "Value" column of the "COSE Algorithms" Registry [COSE.Algorithms].

- The 'sign_alg' parameter MUST NOT be present if the OSCORE group is a pairwise-only group. Otherwise, it MUST be present and specifies the Signature Algorithm used to sign messages in the OSCORE group. This parameter takes values from the "Value" column of the "COSE Algorithms" Registry [COSE.Algorithms].

- The 'sign_params' parameter MUST NOT be present if the OSCORE group is a pairwise-only group. Otherwise, it MUST be present and specifies the parameters of the Signature Algorithm. This parameter is a CBOR array, which includes the following two elements:

    - o 'sign_alg_capab': a CBOR array, with the same format and value of the COSE capabilities array for the Signature Algorithm indicated in 'sign_alg', as specified for that algorithm in the "Capabilities" column of the "COSE Algorithms" Registry [COSE.Algorithms].

    - o 'sign_key_type_capab': a CBOR array, with the same format and value of the COSE capabilities array for the COSE key type of the keys used with the Signature Algorithm indicated in 'sign_alg', as specified for that key type in the "Capabilities" column of the "COSE Key Types" Registry [COSE.Key.Types].

- The 'alg' parameter MUST NOT be present if the OSCORE group is a signature-only group. Otherwise, it MUST be present and specifies the AEAD Algorithm used in the OSCORE group to encrypt messages protected with the pairwise mode.

- The 'ecdh_alg' parameter MUST NOT be present if the OSCORE group is a signature-only group. Otherwise, it MUST be present and specifies the Pairwise Key Agreement Algorithm used in the OSCORE group. This parameter takes values from the "Value" column of the "COSE Algorithms" Registry [COSE.Algorithms].

- The 'ecdh_params' parameter MUST NOT be present if the OSCORE group is a signature-only group. Otherwise, it MUST be present and specifies the parameters of the Pairwise Key Agreement Algorithm. This parameter is a CBOR array, which includes the following two elements:

  - o 'ecdh_alg_capab': a CBOR array, with the same format and value of the COSE capabilities array for the algorithm indicated in 'ecdh_alg', as specified for that algorithm in the "Capabilities" column of the "COSE Algorithms" Registry [COSE.Algorithms].

  - o 'ecdh_key_type_capab': a CBOR array, with the same format and value of the COSE capabilities array for the COSE key type of the keys used with the algorithm indicated in 'ecdh_alg', as specified for that key type in the "Capabilities" column of the "COSE Key Types" Registry [COSE.Key.Types].

  The format of 'key' defined above is consistent with every signature algorithm and ECDH algorithm currently considered in [I-D.ietf-cose-rfc8152bis-algs], i.e., with algorithms that have only the COSE key type as their COSE capability. Appendix B of this document describes how the format of the 'key' parameter can be generalized for possible future registered algorithms having a different set of COSE capabilities.

* The 'exp' parameter MUST be present.

* The 'ace-groupcomm-profile' parameter MUST be present and has value coap_group_oscore_app (PROFILE_TBD), which is defined in Section 23.5 of this document.

* The 'pub_keys' parameter, if present, includes the public keys requested by the joining node by means of the 'get_pub_keys' parameter in the Joining Request.

  If the joining node has asked for the public keys of all the group members, i.e., 'get_pub_keys' had value Null in the Joining Request, then the Group Manager provides only the public keys of the group members that are relevant to the joining node. That is, in such a case, 'pub_keys' includes only: i) the public keys of the responders currently in the OSCORE group, in case the joining node is configured (also) as requester; and ii) the public keys of the requesters currently in the OSCORE group, in case the joining node is configured (also) as responder or monitor.

* The 'peer_identifiers' parameter includes the OSCORE Sender ID of each group member whose public key is specified in the 'pub_keys' parameter. That is, a group member's Sender ID is used as identifier for that group member (REQ12).

* The 'group_policies' parameter SHOULD be present, and SHOULD include the following elements:

  - "Key Update Check Interval" defined in Section 4.1.2.1 of [I-D.ietf-ace-key-groupcomm], with default value 3600;

  - "Expiration Delta" defined in Section 4.1.2.1 of [I-D.ietf-ace-key-groupcomm], with default value 0.

Furthermore, the CBOR map in the payload of the Joining Response MUST also include the following new parameters, defined in Section 23.3 of this document.

* The 'kdc_nonce' parameter, which is a CBOR byte string encoding a dedicated nonce N_KDC generated by the Group Manager. For N_KDC, it is RECOMMENDED to use a 8-byte long random nonce.

* The 'kdc_cred' parameter, which is a CBOR byte string encoding the Group Manager's public key in its original binary representation. The Group Manager's public key MUST be compatible with the encoding, signature or ECDH algorithm, and possible associated parameters used in the OSCORE group.

* The 'kdc_cred_verify' parameter, which is a CBOR byte string encoding a proof-of-possession (PoP) evidence computed by the Group Manager. The PoP evidence is computed over the nonce N_KDC, which is specified in the 'kdc_nonce' parameter and taken as PoP input. The PoP evidence is computed as defined below.

  - If the group is not a pairwise-only group, the PoP evidence MUST be a signature. The Group Manager computes the signature by using the signature algorithm used in the OSCORE group, as well as its own private key associated to the public key specified in the 'kdc_cred' parameter.

  - If the group is a pairwise-only group, the PoP evidence MUST be a MAC computed as follows, by using the HKDF Algorithm HKDF SHA-256, which consists of composing the HKDF-Extract and HKDF-Expand steps [RFC5869].

    MAC = HKDF(salt, IKM, info, L)

    The input parameters of HKDF are as follows.

    o salt takes as value the empty byte string.

- o IKM is computed as a cofactor Diffie-Hellman shared secret, see Section 5.7.1.2 of [NIST-800-56A], using the ECDH algorithm used in the OSCORE group. The Group Manager uses its own Diffie-Hellman private key and the Diffie-Hellman public key of the joining node. For X25519 and X448, the procedure is described in Section 5 of [RFC7748].
- o info takes as value the PoP input.
- o L is equal to 8, i.e., the size of the MAC, in bytes.

As a last action, the Group Manager MUST store the Gid specified in the 'contextId' parameter of the 'key' parameter, as the Birth Gid of the joining node in the joined group (see Section 3.1 of [I-D.ietf-core-oscore-groupcomm]). This applies also in case the node is in fact re-joining the group; in such a case, the newly determined Birth Gid overwrites the one currently stored.

6.5. Receiving the Joining Response

Upon receiving the Joining Response, the joining node retrieves the Group Manager's public key from the 'kdc_cred' parameter. The joining node MUST verify the proof-of-possession (PoP) evidence specified in the 'kdc_cred_verify' parameter of the Joining Response as defined below.

* If the group is not a pairwise-only group, the PoP evidence is a signature. The joining node verifies it by using the public key of the Group Manager, as well as the signature algorithm used in the OSCORE group and possible corresponding parameters.

* If the group is a pairwise-only group, the PoP evidence is a MAC. The joining node recomputes the MAC through the same process taken by the Group Manager when computing the value of the 'kdc_cred_verify' parameter (see Section 6.4), with the difference that the joining node uses its own Diffie-Hellman private key and the Diffie-Hellman public key of the Group Manager. The verification succeeds if and only if the recomputed MAC is equal to the MAC conveyed as PoP evidence in the Joining Response.

In case of failed verification of the PoP evidence, the joining node MUST stop processing the Joining Response and MAY send a new Joining Request to the Group Manager (see Section 6.2).

In case of successful verification of the PoP evidence, the joining node uses the information received in the Joining Response to set up the Group OSCORE Security Context, as described in Section 2 of [I-D.ietf-core-oscore-groupcomm]. If the following parameters were

not included in the 'key' parameter of the Joining Response, the joining node considers the following default values, consistently with Section 3.2 of [RFC8613].

* Absent the 'hkdf' parameter, the joining node considers HKDF SHA-256 as HKDF Algorithm to use in the OSCORE group.

* Absent the 'salt' parameter, the joining node considers the empty byte string as Master Salt to use in the OSCORE group.

In addition, the joining node maintains an association between each public key retrieved from the 'pub_keys' parameter and the role(s) that the corresponding group member has in the OSCORE group.

From then on, the joining node can exchange group messages secured with Group OSCORE as described in [I-D.ietf-core-oscore-groupcomm]. When doing so:

* The joining node MUST NOT process an incoming request message, if protected by a group member whose public key is not associated to the role "Requester".

* The joining node MUST NOT process an incoming response message, if protected by a group member whose public key is not associated to the role "Responder".

* The joining node MUST NOT use the pairwise mode of Group OSCORE to process messages in the group, if the Joining Response did not include the 'ecdh_alg' parameter.

If the application requires backward security, the Group Manager MUST generate updated security parameters and group keying material, and provide it to the current group members upon the new node's joining (see Section 20). As a consequence, the joining node is not able to access secure communication in the OSCORE group occurred prior its joining.

7. Public Keys of Joining Nodes

Source authentication of a message sent within the group and protected with Group OSCORE is ensured by means of a digital signature embedded in the message (in group mode), or by integrity-protecting the message with pairwise keying material derived from the asymmetric keys of sender and recipient (in pairwise mode).

Therefore, group members must be able to retrieve each other's public key from a trusted key repository, in order to verify source authenticity of incoming group messages.

As also discussed in [I-D.ietf-core-oscore-groupcomm], the Group Manager acts as trusted repository of the public keys of the group members, and provides those public keys to group members if requested to. Upon joining an OSCORE group, a joining node is thus expected to provide its own public key to the Group Manager.

In particular, one of the following four cases can occur when a new node joins an OSCORE group.

- The joining node is going to join the group exclusively as monitor, i.e., it is not going to send messages to the group. In this case, the joining node is not required to provide its own public key to the Group Manager, which thus does not have to perform any check related to the public key encoding, to a signature or ECDH algorithm, and to possible associated parameters. In case that joining node still provides a public key in the 'client_cred' parameter of the Joining Request (see Section 6.2), the Group Manager silently ignores that parameter, as well as the related parameters 'cnonce' and 'client_cred_verify'.

- The Group Manager already acquired the public key of the joining node during a past joining process. In this case, the joining node MAY choose not to provide again its own public key to the Group Manager, in order to limit the size of the Joining Request. The joining node MUST provide its own public key again if it has provided the Group Manager with multiple public keys during past joining processes, intended for different OSCORE groups. If the joining node provides its own public key, the Group Manager performs consistency checks as per Section 6.3 and, in case of success, considers it as the public key associated to the joining node in the OSCORE group.

- The joining node and the Group Manager use an asymmetric proof-of-possession key to establish a secure communication association. Then, two cases can occur.

  1. The proof-of-possession key is compatible with the encoding, as well as with the signature or ECDH algorithm, and with possible associated parameters used in the OSCORE group. Then, the Group Manager considers the proof-of-possession key as the public key associated to the joining node in the OSCORE group. If the joining node is aware that the proof-of-possession key is also valid for the OSCORE group, it MAY not provide it again as its own public key to the Group Manager. The joining node MUST provide its own public key again if it has provided the Group Manager with multiple public keys during past joining processes, intended for different OSCORE

groups. If the joining node provides its own public key in the 'client_cred' parameter of the Joining Request (see Section 6.2), the Group Manager performs consistency checks as per Section 6.3 and, in case of success, considers it as the public key associated to the joining node in the OSCORE group.

2. The proof-of-possession key is not compatible with the encoding, or with the signature or algorithm, and with possible associated parameters used in the OSCORE group. In this case, the joining node MUST provide a different compatible public key to the Group Manager in the 'client_cred' parameter of the Joining Request (see Section 6.2). Then, the Group Manager performs consistency checks on this latest provided public key as per Section 6.3 and, in case of success, considers it as the public key associated to the joining node in the OSCORE group.

* The joining node and the Group Manager use a symmetric proof-of-possession key to establish a secure communication association. In this case, upon performing a joining process with that Group Manager for the first time, the joining node specifies its own public key in the 'client_cred' parameter of the Joining Request targeting the group-membership endpoint (see Section 6.2).

8. Retrieval of Updated Keying Material

At some point, a group member considers the Group OSCORE Security Context invalid and to be renewed. This happens, for instance, after a number of unsuccessful security processing of incoming messages from other group members, or when the Security Context expires as specified by the 'exp' parameter of the Joining Response.

When this happens, the group member retrieves updated security parameters and group keying material. This can occur in the two different ways described below.

8.1. Retrieval of Group Keying Material

If the group member wants to retrieve only the latest group keying material, it sends a Key Distribution Request to the Group Manager.

In particular, it sends a CoAP GET request to the endpoint /ace-group/GROUPNAME at the Group Manager.

The Group Manager processes the Key Distribution Request according to Section 4.1.2.2 of [I-D.ietf-ace-key-groupcomm]. The Key Distribution Response is formatted as defined in Section 4.1.2.2 of [I-D.ietf-ace-key-groupcomm]. In addition:

- The 'key' parameter is formatted as defined in Section 6.4 of this document, with the difference that it does not include the 'group_SenderId' parameter.

- The 'exp' parameter MUST be present.

- The 'ace-groupcomm-profile' parameter MUST be present and has value coap_group_oscore_app.

Upon receiving the Key Distribution Response, the group member retrieves the updated security parameters and group keying material, and, if they differ from the current ones, uses them to set up the new Group OSCORE Security Context as described in Section 2 of [I-D.ietf-core-oscore-groupcomm].

8.2. Retrieval of Group Keying Material and OSCORE Sender ID

If the group member wants to retrieve the latest group keying material as well as the OSCORE Sender ID that it has in the OSCORE group, it sends a Key Distribution Request to the Group Manager.

In particular, it sends a CoAP GET request to the endpoint /ace-group/GROUPNAME/nodes/NODENAME at the Group Manager.

The Group Manager processes the Key Distribution Request according to Section 4.1.6.2 of [I-D.ietf-ace-key-groupcomm]. The Key Distribution Response is formatted as defined in Section 4.1.6.2 of [I-D.ietf-ace-key-groupcomm]. In addition:

- The 'key' parameter is formatted as defined in Section 6.4 of this document. In particular, if the requesting group member has exclusively the role of monitor, no 'group_SenderId' is specified within the 'key' parameter.

  Note that, in any other case, the current Sender ID of the group member is not specified as a separate parameter, but rather specified as 'group_SenderId' within the 'key' parameter.

- The 'exp' parameter MUST be present.

Upon receiving the Key Distribution Response, the group member retrieves the updated security parameters, group keying material and Sender ID, and, if they differ from the current ones, uses them to set up the new Group OSCORE Security Context as described in Section 2 of [I-D.ietf-core-oscore-groupcomm].

9. Requesting a Change of Keying Material

As discussed in Section 2.4.2 of [I-D.ietf-core-oscore-groupcomm], a group member may at some point exhaust its Sender Sequence Numbers in the OSCORE group.

When this happens, the group member MUST send a Key Renewal Request message to the Group Manager, as per Section 4.5 of [I-D.ietf-ace-key-groupcomm]. In particular, it sends a CoAP PUT request to the endpoint /ace-group/GROUPNAME/nodes/NODENAME at the Group Manager.

Upon receiving the Key Renewal Request, the Group Manager processes it as defined in Section 4.1.6.1 of [I-D.ietf-ace-key-groupcomm], with the following additions.

The Group Manager MUST return a 5.03 (Service Unavailable) response in case the OSCORE group identified by GROUPNAME is currently inactive (see Section 5.1). The response MUST have Content-Format set to application/ace-groupcomm+cbor and is formatted as defined in Section 4 of [I-D.ietf-ace-key-groupcomm]. The value of the 'error' field MUST be set to 9 ("Group currently not active").

Otherwise, the Group Manager performs one of the following actions.

1. If the requesting group member has exclusively the role of monitor, the Group Manager replies with a 4.01 (Unauthorized) error response. The response MUST have Content-Format set to application/ace-groupcomm+cbor and is formatted as defined in Section 4 of [I-D.ietf-ace-key-groupcomm]. The value of the 'error' field MUST be set to 1 ("Request inconsistent with the current roles").

2. Otherwise, the Group Manager takes one of the following actions.

   * The Group Manager rekeys the OSCORE group. That is, the Group Manager generates new group keying material for that group (see Section 20), and replies to the group member with a group rekeying message as defined in Section 20, providing the new group keying material. Then, the Group Manager rekeys the rest of the OSCORE group, as discussed in Section 20.

     The Group Manager SHOULD perform a group rekeying only if already scheduled to occur shortly, e.g., according to an application-dependent rekeying period or scheduling, or as a reaction to a recent change in the group membership. In any other case, the Group Manager SHOULD NOT rekey the OSCORE group when receiving a Key Renewal Request (OPT8).

- The Group Manager determines and assigns a new OSCORE Sender ID for that group member, and replies with a Key Renewal Response formatted as defined in Section 4.1.6.1 of [I-D.ietf-ace-key-groupcomm]. In particular, the CBOR Map in the response payload includes a single parameter 'group_SenderId' defined in Section 23.3 of this document, specifying the new Sender ID of the group member encoded as a CBOR byte string.

  Consistently with Section 2.4.3.1 of [I-D.ietf-core-oscore-groupcomm], the Group Manager MUST assign a new Sender ID that has not been used in the OSCORE group since the latest time when the current Gid value was assigned to the group.

  Furthermore, the Group Manager MUST add the old, relinquished Sender ID of the group member to the most recent set of stale Sender IDs, in the collection associated to the group (see Section 2.2.1).

  The Group Manager MUST return a 5.03 (Service Unavailable) response in case there are currently no Sender IDs available to assign in the OSCORE group. The response MUST have Content-Format set to application/ace-groupcomm+cbor and is formatted as defined in Section 4 of [I-D.ietf-ace-key-groupcomm]. The value of the 'error' field MUST be set to 4 ("No available node identifiers").

10. Retrieval of Public Keys and Roles of Group Members

A group member or a signature verifier may need to retrieve the public keys of (other) group members. To this end, the group member or signature verifier sends a Public Key Request message to the Group Manager, as per Section 4.6 of [I-D.ietf-ace-key-groupcomm]. In particular, it sends the request to the endpoint /ace-group/GROUPNAME/pub-key at the Group Manager.

If the Public Key Request uses the method FETCH, the Public Key Request is formatted as defined in Section 4.1.3.1 of [I-D.ietf-ace-key-groupcomm]. In particular:

- Each element (if any) of the inner CBOR array 'role_filter' is formatted as in the inner CBOR array 'role_filter' of the 'get_pub_keys' parameter of the Joining Request when the parameter value is non-null (see Section 6.2).

- Each element (if any) of the inner CBOR array 'id_filter' is a CBOR byte string, which encodes the OSCORE Sender ID of the group member for which the associated public key is requested (REQ12).

Upon receiving the Public Key Request, the Group Manager processes it as per Section 4.1.3.1 or Section 4.1.3.2 of [I-D.ietf-ace-key-groupcomm], depending on the request method being FETCH or GET, respectively. Additionally, if the Public Key Request uses the method FETCH, the Group Manager silently ignores node identifiers included in the 'get_pub_keys' parameter of the request that are not associated to any current group member.

The success Public Key Response is formatted as defined in Section 4.1.3.1 or Section 4.1.3.2 of [I-D.ietf-ace-key-groupcomm], depending on the request method being FETCH or GET, respectively.

11. Update of Public Key

A group member may need to provide the Group Manager with its new public key to use in the group from then on, hence replacing the current one. This can be the case, for instance, if the signature or ECDH algorithm, and possible associated parameters used in the OSCORE group have been changed, and the current public key is not compatible with them.

To this end, the group member sends a Public Key Update Request message to the Group Manager, as per Section 4.7 of [I-D.ietf-ace-key-groupcomm], with the following addition.

- The group member computes the proof-of-possession (PoP) evidence included in 'client_cred_verify' in the same way taken when preparing a Joining Request for the OSCORE group in question, as defined in Section 6.2 (REQ20).

In particular, the group member sends a CoAP POST request to the endpoint /ace-group/GROUPNAME/nodes/NODENAME/pub-key at the Group Manager.

Upon receiving the Public Key Update Request, the Group Manager processes it as per Section 4.1.7.1 of [I-D.ietf-ace-key-groupcomm], with the following additions.

- The N_S challenge used to build the proof-of-possession input is computed as per point (1) in Section 6.2.1 (REQ21).

- The Group Manager verifies the PoP challenge included in 'client_cred_verify' in the same way taken when processing a Joining Request for the OSCORE group in question, as defined in Section 6.3 (REQ20).

- The Group Manager MUST return a 5.03 (Service Unavailable) response in case the OSCORE group identified by GROUPNAME is currently inactive (see Section 5.1). The response MUST have Content-Format set to application/ace-groupcomm+cbor and is formatted as defined in Section 4 of [I-D.ietf-ace-key-groupcomm]. The value of the 'error' field MUST be set to 9 ("Group currently not active").

- If the requesting group member has exclusively the role of monitor, the Group Manager replies with a 4.00 (Bad request) error response. The response MUST have Content-Format set to application/ace-groupcomm+cbor and is formatted as defined in Section 4 of [I-D.ietf-ace-key-groupcomm]. The value of the 'error' field MUST be set to 1 ("Request inconsistent with the current roles").

- If the request is successfully processed, the Group Manager stores the association between i) the new public key of the group member; and ii) the Group Identifier (Gid), i.e., the OSCORE ID Context, associated to the OSCORE group together with the OSCORE Sender ID assigned to the group member in the group. The Group Manager MUST keep this association updated over time.

12. Retrieval of the Group Manager's Public Key

A group member or a signature verifier may need to retrieve the public key of the Group Manager. To this end, the group member or signature verifier sends a Group Manager Public Key Request message to the Group Manager.

In particular, it sends a CoAP GET request to the endpoint /ace-group/GROUPNAME/gm-pub-key at the Group Manager defined in Section 5.2 of this document, where GROUPNAME is the name of the OSCORE group.

The payload of the 2.05 (Content) Group Manager Public Key Response is a CBOR map, which MUST contain the following parameters defined in Section 23.3.

- The 'kdc_nonce' parameter, specifying a nonce generated by the Group Manager. This parameter is encoded like the 'kdc_nonce' parameter in the Joining Response (see Section 6.4).

- The 'kdc_cred' parameter, specifying the Group Manager's public key. This parameter is encoded like the 'kdc_cred' parameter in the Joining Response (see Section 6.4).

- The 'kdc_cred_verify' parameter, specifying a proof-of-possession (PoP) evidence computed by the Group Manager. This parameter is encoded like the 'kdc_cred_verify' parameter in the Joining Response (see Section 6.4).

  The PoP evidence is computed over the nonce specified in the 'kdc_nonce' parameter and taken as PoP input, by means of the same method used when preparing the Joining Response (see Section 6.4). In particular, if the group is a pairwise-only group, the Group Manager computes IKM by using its own Diffie-Hellman private key as well as the Diffie-Hellman public key of the requesting client.

Upon receiving a 2.05 (Content) Group Manager Public Key Response, the group member or signature verifier retrieves the Group Manager's public key from the 'kdc_cred' parameter, and MUST verify the proof-of-possession (PoP) evidence specified in the 'kdc_cred_verify' parameter. That is:

- A group member verifies the PoP evidence by means of the same method used when processing the Joining Response (see Section 6.4). In particular, if the group is a pairwise-only group, the group member computes IKM by using its own Diffie-Hellman private key as well as the Diffie-Hellman public key of the Group Manager.

- A signature verifier verifies the PoP evidence as a signature, by using the public key of the Group Manager, as well as the signature algorithm used in the OSCORE group and possible corresponding parameters. Note that a signature verifier would not receive a successful response from the Group Manager, in case GROUPNAME denotes a pairwise-only group.

In case of successful verification of the PoP evidence, the group member or signature verifier MUST store the obtained Group Manager's public key, possibly replacing the currently stored one.

Figure 3 gives an overview of the exchange described above, while Figure 4 shows an example.

```
   Group                                                         Group
   Member                                                       Manager
     |                                                             |
     |               Group Manager Public Key Request              |
     |------------------------------------------------------------>|
     |              GET ace-group/GROUPNAME/gm-pub-key             |
     |                                                             |
     |<---- Group Manager Public Key Response: 2.05 (Content) -----|
     |                                                             |
```

Figure 3: Message Flow of Group Manager Public Key Request-Response

```
      Request:

      Header: GET (Code=0.01)
      Uri-Host: "kdc.example.com"
      Uri-Path: "ace-group"
      Uri-Path: "g1"
      Uri-Path: "gm-pub-key"
      Payload: -

      Response:

      Header: Content (Code=2.05)
      Content-Format: "application/ace-groupcomm+cbor"
      Payload (in CBOR diagnostic notation, with PUB_KEY_GM
               and POP_EVIDENCE being CBOR byte strings):
        {
          "kdc_nonce": h'25a8991cd700ac01',
          "kdc_cred": PUB_KEY_GM,
          "kdc_cred_verify": POP_EVIDENCE
        }
```

Figure 4: Example of Group Manager Public Key Request-Response

13. Retrieval of Signature Verification Data

A signature verifier may need to retrieve data required to verify signatures of messages protected with the group mode and sent to a group (see Sections 3.1 and 8.5 of [I-D.ietf-core-oscore-groupcomm]). To this end, the signature verifier sends a Signature Verification Data Request message to the Group Manager.

In particular, it sends a CoAP GET request to the endpoint /ace-group/GROUPNAME/verif-data at the Group Manager defined in Section 5.3 of this document, where GROUPNAME is the name of the OSCORE group.

The payload of the 2.05 (Content) Signature Verification Data Response is a CBOR map, which has the format used for the Joining Response message in Section 6.4, with the following differences.

- From the Joining Response message, only the parameters 'gkty', 'key', 'num', 'exp' and 'ace-groupcomm-profile' are present. In particular, the 'key' parameter includes only the following data.
  - The parameters 'hkdf', 'contextId', 'pub_key_enc', 'sign_enc_alg', 'sign_alg', 'sign_params'. These parameters MUST be present.
  - The parameters 'alg' and 'ecdh_alg'. These parameter MUST NOT be present if the group is a signature-only group. Otherwise, they MUST be present.
- The parameter 'group_enc_key' is also included, with CBOR label defined in Section 23.3. This parameter specifies the Group Encryption Key of the OSCORE Group, encoded as a CBOR byte string. The Group Manager derives the Group Encryption Key from the group keying material, as per Section 2.1.6 of [I-D.ietf-core-oscore-groupcomm]. This parameter MUST be present.

In order to verify signatures in the group (see Section 8.5 of [I-D.ietf-core-oscore-groupcomm]), the signature verifier relies on: the data retrieved from the 2.05 (Content) Signature Verification Data Response; the public keys of the group members signing the messages to verify, that can be retrieved as defined in Section 10; and the public key of the Group Manager, which can be retrieved as defined in Section 12.

Figure 5 gives an overview of the exchange described above, while Figure 6 shows an example.

```
Signature                                                     Group
Verifier                                                     Manager
  |                                                             |
  |             Signature Verification Data Request             |
  |------------------------------------------------------------>|
  |             GET ace-group/GROUPNAME/verif-data              |
  |                                                             |
  |<--- Signature Verification Data Response: 2.05 (Content) ---|
  |                                                             |
```

Figure 5: Message Flow of Signature Verification Data Request-Response

```
   Request:

   Header: GET (Code=0.01)
   Uri-Host: "kdc.example.com"
   Uri-Path: "ace-group"
   Uri-Path: "g1"
   Uri-Path: "verif-data"
   Payload: -

   Response:

   Header: Content (Code=2.05)
   Content-Format: "application/ace-groupcomm+cbor"
   Payload (in CBOR diagnostic notation, with GROUPCOMM_KEY_TBD
            and PROFILE_TBD being CBOR integers, while GROUP_ENC_KEY
            being a CBOR byte string):
    {
      "gkty": GROUPCOMM_KEY_TBD,
      "key": {
        'hkdf': -10,                   ; HKDF SHA-256
        'contextId': h'37fc',
        'pub_key_enc': 33,             ; x5chain
        'sign_enc_alg': 10,            ; AES-CCM-16-64-128
        'sign_alg': -8,                ; EdDSA
        'sign_params': [[1], [1, 6]]   ; [[OKP], [OKP, Ed25519]]
      },
      "num": 12,
      "exp": 1609459200,
      "ace_groupcomm_profile": PROFILE_TBD,
      "group_enc_key": GROUP_ENC_KEY
    }
```

Figure 6: Example of Signature Verification Data Request-Response

14. Retrieval of Group Policies

A group member may request the current policies used in the OSCORE group. To this end, the group member sends a Policies Request, as per Section 4.8 of [I-D.ietf-ace-key-groupcomm]. In particular, it sends a CoAP GET request to the endpoint /ace-group/GROUPNAME/policies at the Group Manager, where GROUPNAME is the name of the OSCORE group.

Upon receiving the Policies Request, the Group Manager processes it as per Section 4.1.4.1 of [I-D.ietf-ace-key-groupcomm]. The success Policies Response is formatted as defined in Section 4.1.4.1 of [I-D.ietf-ace-key-groupcomm].

15. Retrieval of Keying Material Version

A group member may request the current version of the keying material used in the OSCORE group. To this end, the group member sends a Version Request, as per Section 4.9 of [I-D.ietf-ace-key-groupcomm]. In particular, it sends a CoAP GET request to the endpoint /ace-group/GROUPNAME/num at the Group Manager, where GROUPNAME is the name of the OSCORE group.

Upon receiving the Version Request, the Group Manager processes it as per Section 4.1.5.1 of [I-D.ietf-ace-key-groupcomm]. The success Version Response is formatted as defined in Section 4.1.5.1 of [I-D.ietf-ace-key-groupcomm].

16. Retrieval of Group Status

A group member may request the current status of the the OSCORE group, i.e., active or inactive. To this end, the group member sends a Group Status Request to the Group Manager.

In particular, the group member sends a CoAP GET request to the endpoint /ace-group/GROUPNAME/active at the Group Manager defined in Section 5.1 of this document, where GROUPNAME is the name of the OSCORE group.

The payload of the 2.05 (Content) Group Status Response includes the CBOR simple value True if the group is currently active, or the CBOR simple value False otherwise. The group is considered active if it is set to allow new members to join, and if communication within the group is fine to happen.

Upon learning from a 2.05 (Content) response that the group is currently inactive, the group member SHOULD stop taking part in communications within the group, until it becomes active again.

Upon learning from a 2.05 (Content) response that the group has become active again, the group member can resume taking part in communications within the group.

Figure 7 gives an overview of the exchange described above, while Figure 8 shows an example.

```
Group                                                         Group
Member                                                       Manager
  |                                                              |
  |--- Group Status Request: GET ace-group/GROUPNAME/active --->|
  |                                                              |
  |<---------- Group Status Response: 2.05 (Content) -----------|
  |                                                              |
```

Figure 7: Message Flow of Group Status Request-Response

```
Request:

Header: GET (Code=0.01)
Uri-Host: "kdc.example.com"
Uri-Path: "ace-group"
Uri-Path: "g1"
Uri-Path: "active"
Payload: -

Response:

Header: Content (Code=2.05)
Payload (in CBOR diagnostic notation):
  true
```

Figure 8: Example of Group Status Request-Response

17. Retrieval of Group Names and URIs

A node may want to retrieve from the Group Manager the group name and the URI of the group-membership resource of a group. This is relevant in the following cases.

* Before joining a group, a joining node may know only the current Group Identifier (Gid) of that group, but not the group name and the URI to the group-membership resource.

* As current group member in several groups, the node has missed a previous group rekeying in one of them (see Section 20). Hence, it retains stale keying material and fails to decrypt received messages exchanged in that group.

Such messages do not provide a direct hint to the correct group name, that the node would need in order to retrieve the latest keying material and public keys from the Group Manager (see Section 8.1, Section 8.2 and Section 10). However, such messages may specify the current Gid of the group, as value of the 'kid_context' field of the OSCORE CoAP option (see Section 6.1 of [RFC8613] and Section 4.2 of [I-D.ietf-core-oscore-groupcomm]).

- As signature verifier, the node also refers to a group name for retrieving the required public keys from the Group Manager (see Section 10). As discussed above, intercepted messages do not provide a direct hint to the correct group name, while they may specify the current Gid of the group, as value of the 'kid_context' field of the OSCORE CoAP option. In such a case, upon intercepting a message in the group, the node requires to correctly map the Gid currently used in the group with the invariant group name.

  Furthermore, since it is not a group member, the node does not take part to a possible group rekeying. Thus, following a group rekeying and the consequent change of Gid in a group, the node would retain the old Gid value and cannot correctly associate intercepted messages to the right group, especially if acting as signature verifier in several groups. This in turn prevents the efficient verification of signatures, and especially the retrieval of required, new public keys from the Group Manager.

In either case, the node only knows the current Gid of the group, as learned from received or intercepted messages exchanged in the group. As detailed below, the node can contact the Group Manager, and request the group name and URI to the group-membership resource corresponding to that Gid. Then, it can use that information to either join the group as a candidate group member, get the latest keying material as a current group member, or retrieve public keys used in the group as a signature verifier. To this end, the node sends a Group Name and URI Retrieval Request, as per Section 4.2 of [I-D.ietf-ace-key-groupcomm].

In particular, the node sends a CoAP FETCH request to the endpoint /ace-group at the Group Manager formatted as defined in Section 4.1.1.1 of [I-D.ietf-ace-key-groupcomm]. Each element of the CBOR array 'gid' is a CBOR byte string (REQ9), which encodes the Gid of the group for which the group name and the URI to the group-membership resource are requested.

Upon receiving the Group Name and URI Retrieval Request, the Group Manager processes it as per Section 4.1.1.1 of [I-D.ietf-ace-key-groupcomm]. The success Group Name and URI

Retrieval Response is formatted as defined in Section 4.1.1.1 of [I-D.ietf-ace-key-groupcomm]. In particular, each element of the CBOR array 'gid' is a CBOR byte string (REQ9), which encodes the Gid of the group for which the group name and the URI to the group-membership resource are provided.

For each of its groups, the Group Manager maintains an association between the group name and the URI to the group-membership resource on one hand, and only the current Gid for that group on the other hand. That is, the Group Manager MUST NOT maintain an association between the former pair and any other Gid for that group than the current, most recent one.

Figure 9 gives an overview of the exchanges described above, while Figure 10 shows an example.

```
                                                                Group
Node                                                           Manager
 |                                                                 |
 |---- Group Name and URI Retrieval Request: FETCH ace-group/ --->|
 |                                                                 |
 |<--- Group Name and URI Retrieval Response: 2.05 (Content) -----|
 |                                                                 |
```

Figure 9: Message Flow of Group Name and URI Retrieval Request-Response

```
   Request:

   Header: FETCH (Code=0.05)
   Uri-Host: "kdc.example.com"
   Uri-Path: "ace-group"
   Content-Format: "application/ace-groupcomm+cbor"
   Payload (in CBOR diagnostic notation):
     {
       "gid": [h'37fc', h'84bd']
     }

   Response:

   Header: Content (Code=2.05)
   Content-Format: "application/ace-groupcomm+cbor"
   Payload (in CBOR diagnostic notation):
     {
       "gid": [h'37fc', h'84bd'],
       "gname": ["g1", "g2"],
       "guri": ["ace-group/g1", "ace-group/g2"]
     }
```

Figure 10: Example of Group Name and URI Retrieval Request-Response

18. Request to Leave the Group

A group member may request to leave the OSCORE group. To this end, the group member sends a Group Leaving Request, as per Section 4.10 of [I-D.ietf-ace-key-groupcomm]. In particular, it sends a CoAP DELETE request to the endpoint /ace-group/GROUPNAME/nodes/NODENAME at the Group Manager.

Upon receiving the Group Leaving Request, the Group Manager processes it as per Section 4.1.6.3 of [I-D.ietf-ace-key-groupcomm].

19. Removal of a Group Member

Other than after a spontaneous request to the Group Manager as described in Section 18, a node may be forcibly removed from the OSCORE group, e.g., due to expired or revoked authorization.

In either case, the Group Manager "forgets" the Birth Gid currently associated to the leaving node in the OSCORE group. This was stored following the Joining Response sent to that node, after its latest (re-)joining of the OSCORE group (see Section 6.4).

If any of the two conditions below holds, the Group Manager MUST inform the leaving node of its eviction as follows. If both conditions hold, the Group Manager MUST inform the leaving node only once, using either of the corresponding methods.

* If, upon joining the group (see Section 6.2), the leaving node specified a URI in the 'control_uri' parameter defined in Section 4.1.2.1 of [I-D.ietf-ace-key-groupcomm], the Group Manager sends a DELETE request targeting the URI specified in the 'control_uri' parameter (OPT9).

* If the leaving node has been observing the associated resource at ace-group/GROUPNAME/nodes/NODENAME, the Group Manager sends an unsolicited 4.04 (Not Found) response to the leaving node, as specified in 4.1.6.2 of [I-D.ietf-ace-key-groupcomm].

If the leaving node has not exclusively the role of monitor, the Group Manager performs the following actions.

* The Group Manager frees the OSCORE Sender ID value of the leaving node. This value MUST NOT become available for possible upcoming joining nodes in the same group, until the group has been rekeyed and assigned a new Group Identifier (Gid).

- The Group Manager MUST add the relinquished Sender ID of the leaving node to the most recent set of stale Sender IDs, in the collection associated to the group (see Section 2.2.1).

- The Group Manager cancels the association between, on one hand, the public key of the leaving node and, on the other hand, the Gid associated to the OSCORE group together with the freed Sender ID value. The Group Manager deletes the public key of the leaving node, if that public key has no remaining association with any pair (Gid, Sender ID).

Then, the Group Manager MUST generate updated security parameters and group keying material, and provide it to the remaining group members (see Section 20). As a consequence, the leaving node is not able to acquire the new security parameters and group keying material distributed after its leaving.

Same considerations in Section 5 of [I-D.ietf-ace-key-groupcomm] apply here as well, considering the Group Manager acting as KDC.

20. Group Rekeying Process

In order to rekey the OSCORE group, the Group Manager distributes a new Group Identifier (Gid), i.e., a new OSCORE ID Context; a new OSCORE Master Secret; and, optionally, a new OSCORE Master Salt for that group. When doing so, the Group Manager MUST increment the version number of the group keying material, before starting its distribution.

Consistently with Section 3.1 of [I-D.ietf-core-oscore-groupcomm]:

- The Group Manager can reassign a Gid to the same group over that group's lifetime, e.g., once the whole space of Gid values has been used for the group in question.

- Before rekeying the group, the Group Manager MUST check if the new Gid to be distributed coincides with the Birth Gid of any of the current group members (see Section 6.4). If any of such "elder members" is found in the group, the Group Manager MUST evict them from the group. That is, the Group Manager MUST terminate their membership and MUST rekey the group in such a way that the new keying material is not provided to those evicted elder members. This also includes adding their relinquished Sender IDs to the most recent set of stale Sender IDs, in the collection associated to the group (see Section 2.2.1), before rekeying the group.

Until a further following group rekeying, the Group Manager MUST store the list of those latest-evicted elder members. If any of those nodes re-joins the group before a further following group rekeying occurs, the Group Manager MUST NOT rekey the group upon their re-joining. When one of those nodes re-joins the group, the Group Manager can rely, e.g., on the ongoing secure communication association to recognize the node as included in the stored list.

Across the rekeying execution, the Group Manager MUST preserve the same unchanged OSCORE Sender IDs for all group members intended to remain in the group. This avoids affecting the retrieval of public keys from the Group Manager and the verification of group messages.

The Group Manager MUST support at least the group rekeying scheme defined in Section 20.1. Future application profiles may define alternative message formats and group rekeying schemes, which MUST comply with the functional steps defined in Section 3.2 of [I-D.ietf-core-oscore-groupcomm].

It is RECOMMENDED that the Group Manager gets confirmation of successful distribution from the group members, and admits a maximum number of individual retransmissions to non-confirming group members. Once completed the group rekeying process, the Group Manager creates a new empty set X' of stale Sender IDs associated to the version of the newly distributed group keying material. Then, the Group Manager MUST add the set X' to the collection of stale Sender IDs associated to the group (see Section 2.2.1).

In case the rekeying terminates and some group members have not received the new keying material, they will not be able to correctly process following secured messages exchanged in the group. These group members will eventually contact the Group Manager, in order to retrieve the current keying material and its version.

Some of these group members may be in multiple groups, each associated to a different Group Manager. When failing to correctly process messages secured with the new keying material, these group members may not have sufficient information to determine which exact Group Manager they should contact, in order to retrieve the current keying material they are missing.

If the Gid is formatted as described in Appendix C of [I-D.ietf-core-oscore-groupcomm], the Group Prefix can be used as a hint to determine the right Group Manager, as long as no collisions among Group Prefixes are experienced. Otherwise, a group member needs to contact the Group Manager of each group, e.g., by first requesting only the version of the current group keying material (see Section 15) and then possibly requesting the current keying material (see Section 8.1).

Furthermore, some of these group members can be in multiple groups, all of which associated to the same Group Manager. In this case, these group members may also not have sufficient information to determine which exact group they should refer to, when contacting the right Group Manager. Hence, they need to contact a Group Manager multiple times, i.e., separately for each group they belong to and associated to that Group Manager.

Finally, Section 20.3 discusses how a group member can realize that it has missed one or more rekeying instances, and the actions it is accordingly required to take.

20.1. Sending Rekeying Messages

The Group Manager MUST support at least the group rekeying scheme defined in this section.

The group rekeying messages MUST have Content-Format set to application/ace-groupcomm+cbor and have the same format used for the Joining Response message in Section 6.4, with the following differences.

* From the Joining Response, only the parameters 'gkty', 'key', 'num', 'exp', and 'ace-groupcomm-profile' are present. In particular, the 'key' parameter includes only the following data.

  - The 'ms' parameter, specifying the new OSCORE Master Secret value. This parameter MUST be present.

  - The 'contextId' parameter, specifying the new Gid to use as OSCORE ID Context value. This parameter MUST be present.

  - The 'salt' value, specifying the new OSCORE Master Salt value. This parameter MAY be present.

- The parameter 'stale_node_ids' MUST also be included, with CBOR label defined in Section 23.3. This parameter is encoded as a CBOR array, where each element is encoded as a CBOR byte string. The CBOR array has to be intended as a set, i.e., the order of its elements is irrelevant. The parameter is populated as follows.
  - The Group Manager creates an empty CBOR array ARRAY.
  - The Group Manager considers the collection of stale Sender IDs associated to the group (see Section 2.2.1), and takes the most recent set X, i.e., the set associated to the current version of the group keying material about to be relinquished.
  - For each Sender ID in X, the Group Manager encodes it as a CBOR byte string and adds the result to ARRAY.
  - The parameter 'stale_node_ids' takes ARRAY as value.

The Group Manager separately sends a group rekeying message formatted as defined above to each group member to be rekeyed.

Each rekeying message MUST be secured with the pairwise secure communication channel between the Group Manager and the group member used during the joining process. In particular, each rekeying message can target the 'control_uri' URI path defined in Section 4.1.2.1 of [I-D.ietf-ace-key-groupcomm] (OPT9), if provided by the intended recipient upon joining the group (see Section 6.2).

This distribution approach requires group members to act (also) as servers, in order to correctly handle unsolicited group rekeying messages from the Group Manager. In particular, if a group member and the Group Manager use OSCORE [RFC8613] to secure their pairwise communications, the group member MUST create a Replay Window in its own Recipient Context upon establishing the OSCORE Security Context with the Group Manager, e.g., by means of the OSCORE profile of ACE [I-D.ietf-ace-oscore-profile].

Group members and the Group Manager SHOULD additionally support alternative distribution approaches that do not require group members to act (also) as servers. A number of such approaches are defined in Section 4.4 of [I-D.ietf-ace-key-groupcomm]. In particular, a group member may subscribe for updates to the group-membership resource of the group, at the endpoint /ace-group/GROUPNAME/ of the Group Manager. This can rely on CoAP Observe [RFC7641] or on a full-fledged Pub-Sub model [I-D.ietf-core-coap-pubsub] with the Group Manager acting as Broker.

20.2. Receiving Rekeying Messages

Once received the new group keying material, a group member proceeds as follows.

The group member considers the stale Sender IDs received from the Group Manager. If the group rekeying scheme defined in Section 20.1 is used, the stale Sender IDs are specified by the 'stale_node_ids' parameter.

After that, as per Section 3.2 of [I-D.ietf-core-oscore-groupcomm], the group member MUST remove every public key associated to a stale Sender ID from its list of group members' public keys used in the group, and MUST delete each of its Recipient Contexts used in the group whose corresponding Recipient ID is a stale Sender ID.

Then, the following cases can occur, based on the version number V' of the new group keying material distributed through the rekeying process. If the group rekeying scheme defined in Section 20.1 is used, this information is specified by the 'num' parameter.

- The group member has not missed any group rekeying. That is, the old keying material owned by the group member has version number V, while the received new keying material has version number V' = (V + 1). In such a case, the group member simply installs the new keying material and derives the corresponding new Security Context.

- The group member has missed one or more group rekeying instances. That is, the old keying material owned by the group member has version number V, while the received new keying material has version number V' > (V + 1). In such a case, the group member MUST proceed as defined in Section 20.3.

- The group member has received keying material not newer than the stored one. That is, the stored keying material owned by the group member has version number V, while the received keying material has version number V' < (V + 1). In such a case, the group member MUST ignore the received rekeying messages and MUST NOT install the received keying material.

20.3. Missed Rekeying Instances

A group member can realize to have missed one or more rekeying instances in one of the ways discussed below. In the following, V denotes the version number of the old keying material stored by the group member, while V' denotes the version number of the latest, possibly just distributed, keying material.

a. The group member has participated to a rekeying process that has distributed new keying material with version number V' > (V + 1), as discussed in Section 20.2.

b. The group member has obtained the latest keying material from the Group Manager, as a response to a Key Distribution Request (see Section 8.1) or to a Joining Request when re-joining the group (see Section 6.2). In particular, V is different than V' specified by the 'num' parameter in the response.

c. The group member has obtained the public keys of other group members, through a Public Key Request-Response exchange with the Group Manager (see Section 10). In particular, V is different than V' specified by the 'num' parameter in the response.

d. The group member has performed a Version Request-Response exchange with the Group Manager (see Section 15). In particular, V is different than V' specified by the 'num' parameter in the response.

In either case, the group member MUST delete the stored keying material with version number V.

If case (a) or case (b) applies, the group member MUST perform the following actions.

1. The group member MUST NOT install the latest keying material yet, in case that was already obtained.

2. The group member sends a Stale Sender IDs Request to the Group Manager (see Section 20.3.1), specifying the version number V as payload of the request.

   If the Stale Sender IDs Response from the Group Manager has no payload, the group member MUST remove all the public keys from its list of group members' public keys used in the group, and MUST delete all its Recipient Contexts used in the group.

   Otherwise, the group member considers the stale Sender IDs specified in the Stale Sender IDs Response from the Group Manager. Then, the group member MUST remove every public key associated to a stale Sender ID from its list of group members' public keys used in the group, and MUST delete each of its Recipient Contexts used in the group whose corresponding Recipient ID is a stale Sender ID.

3. The group member installs the latest keying material with version number V' and derives the corresponding new Security Context.

If case (c) or case (d) applies, the group member SHOULD perform the following actions.

1. The group member sends a Stale Sender IDs Request to the Group Manager (see Section 20.3.1), specifying the version number V as payload of the request.

   If the Stale Sender IDs Response from the Group Manager has no payload, the group member MUST remove all the public keys from its list of group members' public keys used in the group, and MUST delete all its Recipient Contexts used in the group.

   Otherwise, the group member considers the stale Sender IDs specified in the Stale Sender IDs Response from the Group Manager. Then, the group member MUST remove every public key associated to a stale Sender ID from its list of group members' public keys used in the group, and MUST delete each of its Recipient Contexts used in the group whose corresponding Recipient ID is a stale Sender ID.

2. The group member obtains the latest keying material with version number V' from the Group Manager. This can happen by sending a Key Distribution Request to the Group Manager (see Section 8.1), or by re-joining the group (see Section 6.2).

3. The group member installs the latest keying material with version number V' and derives the corresponding new Security Context.

If case (c) or case (d) applies, the group member can alternatively perform the following actions.

1. The group member re-joins the group (see Section 6.2). When doing so, the group member MUST re-join with the same roles it currently has in the group, and MUST request the Group Manager for the public keys of all the current group members. That is, the 'get_pub_keys' parameter of the Joining Request MUST be present and MUST be set to the CBOR simple value Null.

2. When receiving the Joining Response (see Section 6.5 and Section 6.5), the group member retrieves the set Z of public keys specified in the 'pub_keys' parameter.

   Then, the group member MUST remove every public key which is not in Z from its list of group members' public keys used in the group, and MUST delete each of its Recipient Contexts used in the group that does not include any of the public keys in Z.

3. The group member installs the latest keying material with version number V' and derives the corresponding new Security Context.

20.3.1. Retrieval of Stale Sender IDs

When realizing to have missed one or more group rekeying instances (see Section 20.3), a node needs to retrieve from the Group Manager the data required to delete some of its stored group members' public keys and Recipient Contexts (see Section 5.4.1). This data is provided as an aggregated set of stale Sender IDs, which are used as specified in Section 20.3.

In particular, the node sends a CoAP FETCH request to the endpoint /ace-group/GROUPNAME/stale-sids at the Group Manager defined in Section 5.4 of this document, where GROUPNAME is the name of the OSCORE group.

The payload of the Stale Sender IDs Request MUST include a CBOR unsigned integer. This encodes the version number V of the most recent group keying material owned and installed by the requesting client, which is older than the latest, possibly just distributed, keying material with version number V'.

The handler MUST respond with a 4.00 (Bad Request) response, if the request is not formatted correctly. Also, the handler MUST respond with a 4.00 (Bad Request) response, if the specified version number V is greater or equal than the version number V' associated to the latest keying material in the group, i.e., if V >= V'.

Otherwise, the handler responds with a 2.05 (Content) Stale Sender IDs Response. The payload of the response is formatted as defined below, where SKEW = (V' - V + 1).

* The Group Manager considers ITEMS as the current number of sets stored in the collection of stale Sender IDs associated to the group (see Section 2.2.1).

* If SKEW > ITEMS, the Stale Sender IDs Response MUST NOT have a payload.

* Otherwise, the payload of the Stale Sender IDs Response MUST include a CBOR array, where each element is encoded as a CBOR byte string. The CBOR array has to be intended as a set, i.e., the order of its elements is irrelevant. The Group Manager populates the CBOR array as follows.

  - The Group Manager creates an empty CBOR array ARRAY and an empty set X.

- The Group Manager considers the SKEW most recent sets stored in the collection of stale Sender IDs associated to the group. Note that the most recent set is the one associate to the latest version of the group keying material.

- The Group Manager copies all the Sender IDs from the selected sets into X. When doing so, the Group Manager MUST discard duplicates. That is, the same Sender ID MUST NOT be present more than once in the final content of X.

- For each Sender ID in X, the Group Manager encodes it as a CBOR byte string and adds the result to ARRAY.

- Finally, ARRAY is specified as payload of the Stale Sender IDs Response. Note that ARRAY might result in the empty CBOR array.

Figure 11 gives an overview of the exchange described above, while Figure 12 shows an example.

```
                                                              Group
Node                                                         Manager
  |                                                             |
  |                   Stale Sender IDs Request                  |
  |------------------------------------------------------------>|
  |             FETCH ace-group/GROUPNAME/stale-sids            |
  |                                                             |
  |<---------- Stale Sender IDs Response: 2.05 (Content) -------|
  |                                                             |
```

Figure 11: Message Flow of Stale Sender IDs Request-Response

```
Request:

Header: FETCH (Code=0.05)
Uri-Host: "kdc.example.com"
Uri-Path: "ace-group"
Uri-Path: "g1"
Uri-Path: "stale-sids"
Payload (in CBOR diagnostic notation):
  42

Response:

Header: Content (Code=2.05)
Payload (in CBOR diagnostic notation):
  [h'01', h'fc', h'12ab', h'de44', h'ff']
```

Figure 12: Example of Stale Sender IDs Request-Response

21. Default Values for Group Configuration Parameters

This section defines the default values that the Group Manager assumes for the configuration parameters of an OSCORE group, unless differently specified when creating and configuring the group. This can be achieved as specified in [I-D.ietf-ace-oscore-gm-admin].

21.1. Common

This section always applies, as related to common configuration parameters.

- For the HKDF Algorithm 'hkdf', the Group Manager SHOULD use the same default value defined in Section 3.2 of [RFC8613], i.e., HKDF SHA-256 (COSE algorithm encoding: -10).

- For the format 'pub_key_enc' used to encode the public keys in the group, the Group Manager SHOULD use a CBOR Web Token (CWT)[RFC8392] or an unprotected CWT Claim Set [I-D.ietf-rats-uccs].

  [This is a pending registration requested by draft-ietf-lake-edhoc.]

- For 'max_stale_sets', the Group Manager SHOULD consider N = 3 as the maximum number of stored sets of stale Sender IDs in the collection associated to the group (see Section 2.2.1).

21.2. Group Mode

This section applies if the group uses (also) the group mode of Group OSCORE.

- For the Signature Encryption Algorithm 'sign_enc_alg' used to encrypted messages protected with the group mode, the Group Manager SHOULD use AES-CCM-16-64-128 (COSE algorithm encoding: 10) as default value.

The Group Manager SHOULD use the following default values for the Signature Algorithm 'sign_alg' and related parameters 'sign_params', consistently with the "COSE Algorithms" Registry [COSE.Algorithms], the "COSE Key Types" Registry [COSE.Key.Types] and the "COSE Elliptic Curves" Registry [COSE.Elliptic.Curves].

- For the Signature Algorithm 'sign_alg' used to sign messages protected with the group mode, the signature algorithm EdDSA [RFC8032].

- For the parameters 'sign_params' of the Signature Algorithm:

  - The array [[OKP], [OKP, Ed25519]], in case EdDSA is assumed or specified for 'sign_alg'. In particular, this indicates to use the COSE key type OKP and the elliptic curve Ed25519 [RFC8032].

  - The array [[EC2], [EC2, P-256]], in case ES256 [RFC6979] is specified for 'sign_alg'. In particular, this indicates to use the COSE key type EC2 and the elliptic curve P-256.

  - The array [[EC2], [EC2, P-384]], in case ES384 [RFC6979] is specified for 'sign_alg'. In particular, this indicates to use the COSE key type EC2 and the elliptic curve P-384.

  - The array [[EC2], [EC2, P-521]], in case ES512 [RFC6979] is specified for 'sign_alg'. In particular, this indicates to use the COSE key type EC2 and the elliptic curve P-521.

  - The array [[RSA], [RSA]], in case PS256, PS384 or PS512 [RFC8017] is specified for 'sign_alg'. In particular, this indicates to use the COSE key type RSA.

21.3. Pairwise Mode

This section applies if the group uses (also) the pairwise mode of Group OSCORE.

For the AEAD Algorithm 'alg' used to encrypt messages protected with the pairwise mode, the Group Manager SHOULD use the same default value defined in Section 3.2 of [RFC8613], i.e., AES-CCM-16-64-128 (COSE algorithm encoding: 10).

For the Pairwise Key Agreement Algorithm 'ecdh_alg' and related parameters 'ecdh_params', the Group Manager SHOULD use the following default values, consistently with the "COSE Algorithms" Registry [COSE.Algorithms], the "COSE Key Types" Registry [COSE.Key.Types] and the "COSE Elliptic Curves" Registry [COSE.Elliptic.Curves].

- For the Pairwise Key Agreement Algorithm 'ecdh_alg' used to compute static-static Diffie-Hellman shared secrets, the ECDH algorithm ECDH-SS + HKDF-256 specified in Section 6.3.1 of [I-D.ietf-cose-rfc8152bis-algs].

- For the parameters 'ecdh_params' of the Pairwise Key Agreement Algorithm:
  - The array [[OKP], [OKP, X25519]], in case EdDSA is assumed or specified for 'sign_alg'. In particular, this indicates to use the COSE key type OKP and the elliptic curve X25519 [RFC8032].
  - The array [[EC2], [EC2, P-256]], in case ES256 [RFC6979] is specified for 'sign_alg'. In particular, this indicates to use the COSE key type EC2 and the elliptic curve P-256.
  - The array [[EC2], [EC2, P-384]], in case ES384 [RFC6979] is specified for 'sign_alg'. In particular, this indicates to use the COSE key type EC2 and the elliptic curve P-384.
  - The array [[EC2], [EC2, P-521]], in case ES512 [RFC6979] is specified for 'sign_alg'. In particular, this indicates to use the COSE key type EC2 and the elliptic curve P-521.

22. Security Considerations

Security considerations for this profile are inherited from [I-D.ietf-ace-key-groupcomm], the ACE framework for Authentication and Authorization [I-D.ietf-ace-oauth-authz], and the specific transport profile of ACE signalled by the AS, such as [I-D.ietf-ace-dtls-authorize] and [I-D.ietf-ace-oscore-profile].

The following security considerations also apply for this profile.

22.1. Management of OSCORE Groups

This profile leverages the following management aspects related to OSCORE groups and discussed in the sections of [I-D.ietf-core-oscore-groupcomm] referred below.

- Management of group keying material (see Section 3.1 of [I-D.ietf-core-oscore-groupcomm]). The Group Manager is responsible for the renewal and re-distribution of the keying material in the groups of its competence (rekeying). According to the specific application requirements, this can include rekeying the group upon changes in its membership. In particular, renewing the group keying material is required upon a new node's joining or a current node's leaving, in case backward security and forward security have to be preserved, respectively.

- Provisioning and retrieval of public keys (see Section 2 of [I-D.ietf-core-oscore-groupcomm]). The Group Manager acts as key repository of public keys of group members, and provides them upon request.

- Synchronization of sequence numbers (see Section 6.2 of [I-D.ietf-core-oscore-groupcomm]). This concerns how a responder node that has just joined an OSCORE group can synchronize with the sequence number of requesters in the same group.

Before sending the Joining Response, the Group Manager MUST verify that the joining node actually owns the associated private key. To this end, the Group Manager can rely on the proof-of-possession challenge-response defined in Section 6. Alternatively, the joining node can use its own public key as asymmetric proof-of-possession key to establish a secure channel with the Group Manager, e.g., as in Section 3.2.2 of [I-D.ietf-ace-dtls-authorize]. However, this requires such proof-of-possession key to be compatible with the encoding, as well as with the signature algorithm, and possible associated parameters used in the OSCORE group.

A node may have joined multiple OSCORE groups under different non-synchronized Group Managers. Therefore, it can happen that those OSCORE groups have the same Group Identifier (Gid). It follows that, upon receiving a Group OSCORE message addressed to one of those groups, the node would have multiple Security Contexts matching with the Gid in the incoming message. It is up to the application to decide how to handle such collisions of Group Identifiers, e.g., by trying to process the incoming message using one Security Context at the time until the right one is found.

22.2. Size of Nonces as Proof-of-Possesion Challenge

With reference to the Joining Request message in Section 6.2, the proof-of-possession (PoP) evidence included in 'client_cred_verify' is computed over an input including also N_C | N_S, where | denotes concatenation.

For the N_C challenge, it is RECOMMENDED to use a 8-byte long random nonce. Furthermore, N_C is always conveyed in the 'cnonce' parameter of the Joining Request, which is always sent over the secure communication channel between the joining node and the Group Manager.

As defined in Section 6.2.1, the way the N_S value is computed depends on the particular way the joining node provides the Group Manager with the Access Token, as well as on following interactions between the two.

- If the Access Token is not explicitly posted to the /authz-info endpoint of the Group Manager, then N_S is computed as a 32-byte long challenge. For an example, see point (2) of Section 6.2.1.

- If the Access Token has been explicitly posted to the /authz-info endpoint of the Group Manager, N_S takes the most recent value provided to the client by the Group Manager in the 'kdcchallenge' parameter, as specified in point (1) of Section 6.2.1. This is provided either in the 2.01 response to the Token Post (see Section 6.1), or in a 4.00 response to a following Joining Request (see Section 6.3). In either case, it is RECOMMENDED to use a 8-byte long random challenge as value for N_S.

If we consider both N_C and N_S to take 8-byte long values, the following considerations hold.

- Let us consider both N_C and N_S as taking random values, and the Group Manager to never change the value of the N_S provided to a Client during the lifetime of an Access Token. Then, as per the birthday paradox, the average collision for N_S will happen after 2^32 new posted Access Tokens, while the average collision for N_C will happen after 2^32 new Joining Requests. This amounts to considerably more token provisionings than the expected new joinings of OSCORE groups under a same Group Manager, as well as to considerably more requests to join OSCORE groups from a same Client using a same Access Token under a same Group Manager.

- Section 7 of [I-D.ietf-ace-oscore-profile] as well Appendix B.2 of [RFC8613] recommend the use of 8-byte random values as well. Unlike in those cases, the values of N_C and N_S considered in this document are not used for as sensitive operations as the derivation of a Security Context, and thus do not have possible implications in the security of AEAD ciphers.

22.3. Reusage of Nonces for Proof-of-Possession Input

As long as the Group Manager preserves the same N_S value currently associated to an Access Token, i.e., the latest value provided to a Client in a 'kdcchallenge' parameter, the Client is able to successfully reuse the same proof-of-possession (PoP) input for multiple Joining Requests to that Group Manager.

In particular, the Client can reuse the same N_C value for every Joining Request to the Group Manager, and combine it with the same unchanged N_S value. This results in reusing the same PoP input for producing the PoP evidence to include in the 'client_cred_verify' parameter of the Joining Requests.

Unless the Group Manager maintains a list of N_C values already used by that Client since the latest update to the N_S value associated to the Access Token, the Group Manager can be forced to falsely believe that the Client possesses its own private key at that point in time, upon verifying the PoP evidence in the 'client_cred_verify' parameter.

23. IANA Considerations

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

This document has the following actions for IANA.

23.1. OAuth Parameters Registry

The following registrations are done for the OAuth Parameters Registry following the procedure specified in Section 11.2 of [RFC6749].

* Parameter name: ecdh_info

* Parameter usage location: client-rs request, rs-client response

* Change Controller: IESG

* Specification Document(s): [[This specification]]

* Parameter name: gm_dh_pub_keys

* Parameter usage location: client-rs request, rs-client response

* Change Controller: IESG

* Specification Document(s): [[This specification]]

23.2. OAuth Parameters CBOR Mappings Registry

The following registrations are done for the OAuth Parameters CBOR Mappings Registry following the procedure specified in Section 8.10 of [I-D.ietf-ace-oauth-authz].

* Name: ecdh_info

* CBOR Key: TBD (range -256 to 255)

* Value Type: Simple value Null / array

* Reference: [[This specification]]

* Name: gm_dh_pub_keys

* CBOR Key: TBD (range -256 to 255)

* Value Type: Simple value Null / array

* Reference: [[This specification]]

23.3. ACE Groupcomm Parameters Registry

IANA is asked to register the following entry to the "ACE Groupcomm Parameters" Registry defined in Section 10.5 of [I-D.ietf-ace-key-groupcomm].

* Name: group_senderId

* CBOR Key: TBD

* CBOR Type: Byte string

* Reference: [[This document]] (Section 9)

* Name: kdc_nonce

* CBOR Key: TBD

* CBOR Type: Byte string

* Reference: [[This document]] (Section 6.4)

* Name: kdc_cred

* CBOR Key: TBD

* CBOR Type: Byte string

* Reference: [[This document]] (Section 6.4)

* Name: kdc_cred_verify

- CBOR Key: TBD

- CBOR Type: Byte string

- Reference: [[This document]] (Section 6.4)

- Name: ecdh_info

- CBOR Key: TBD

- CBOR Type: Array

- Reference: [[This document]] (Section 6.3)

- Name: gm_dh_pub_keys

- CBOR Key: TBD

- CBOR Type: Array

- Reference: [[This document]] (Section 6.3)

- Name: group_enc_key

- CBOR Key: TBD

- CBOR Type: Byte String

- Reference: [[This document]] (Section 5.3.1)

- Name: stale_node_ids

- CBOR Key: TBD

- CBOR Type: Array

- Reference: [[This document]] (Section 20)

23.4. ACE Groupcomm Key Registry

IANA is asked to register the following entry to the "ACE Groupcomm Key" Registry defined in Section 10.6 of [I-D.ietf-ace-key-groupcomm].

- Name: Group_OSCORE_Input_Material object
- Key Type Value: GROUPCOMM_KEY_TBD
- Profile: "coap_group_oscore_app", defined in Section 23.5 of this document.
- Description: A Group_OSCORE_Input_Material object encoded as described in Section 6.4 of this document.
- Reference: [[This document]] (Section 6.4)

23.5. ACE Groupcomm Profile Registry

IANA is asked to register the following entry to the "ACE Groupcomm Profile" Registry defined in Section 10.7 of [I-D.ietf-ace-key-groupcomm].

- Name: coap_group_oscore_app
- Description: Application profile to provision keying material for participating in group communication protected with Group OSCORE as per [I-D.ietf-core-oscore-groupcomm].
- CBOR Value: PROFILE_TBD
- Reference: [[This document]] (Section 6.4)

23.6. OSCORE Security Context Parameters Registry

IANA is asked to register the following entries in the "OSCORE Security Context Parameters" Registry defined in Section 9.4 of [I-D.ietf-ace-oscore-profile].

- Name: group_SenderId
- CBOR Label: TBD
- CBOR Type: bstr
- Registry: -
- Description: OSCORE Sender ID assigned to a member of an OSCORE group
- Reference: [[This document]] (Section 6.4)

- Name: pub_key_enc
- CBOR Label: TBD
- CBOR Type: integer
- Registry: COSE Header Parameters
- Description: Encoding of Public Keys to be used in the OSCORE group
- Reference: [[This document]] (Section 6.4)

- Name: sign_enc_alg
- CBOR Label: TBD
- CBOR Type: tstr / int
- Registry: COSE Algorithms
- Description: OSCORE Signature Encryption Algorithm Value
- Reference: [[This document]] (Section 6.4)

- Name: sign_alg
- CBOR Label: TBD
- CBOR Type: tstr / int
- Registry: COSE Algorithms
- Description: OSCORE Signature Algorithm Value
- Reference: [[This document]] (Section 6.4)

- Name: sign_params
- CBOR Label: TBD
- CBOR Type: array
- Registry: COSE Algorithms, COSE Key Types, COSE Elliptic Curves

- Description: OSCORE Signature Algorithm Parameters
- Reference: [[This document]] (Section 6.4)

- Name: ecdh_alg
- CBOR Label: TBD
- CBOR Type: tstr / int
- Registry: COSE Algorithms
- Description: OSCORE Pairwise Key Agreement Algorithm Value
- Reference: [[This document]] (Section 6.4)

- Name: ecdh_params
- CBOR Label: TBD
- CBOR Type: array
- Registry: COSE Algorithms, COSE Key Types, COSE Elliptic Curves
- Description: OSCORE Pairwise Key Agreement Algorithm Parameters
- Reference: [[This document]] (Section 6.4)

23.7. TLS Exporter Label Registry

IANA is asked to register the following entry to the "TLS Exporter Label" Registry defined in Section 6 of [RFC5705] and updated in Section 12 of [RFC8447].

- Value: EXPORTER-ACE-Sign-Challenge-coap-group-oscore-app
- DTLS-OK: Y
- Recommended: N
- Reference: [[This document]] (Section 6.2.1)

23.8. AIF Registry

IANA is asked to register the following entry to the "Toid" sub-registry of the "AIF" Registry defined in Section 5.2 of [I-D.ietf-ace-aif].

* Name: oscore-group-name

* Description/Specification: group name of the OSCORE group, as specified in [[This document]].

IANA is asked to register the following entry to the "Tperm" sub-Registry of the "AIF" Registry defined in Section 5.2 of [I-D.ietf-ace-aif].

* Name: oscore-group-roles

* Description/Specification: role(s) of the member of the OSCORE group, as specified in [[This document]].

23.9. Media Type Registrations

This document registers the 'application/aif-groupcomm-oscore+cbor' media type for the AIF specific data model AIF-OSCORE-GROUPCOMM defined in Section 3 of [[This document]]. This registration follows the procedures specified in [RFC6838].

These media type has parameters for specifying the object identifier ("Toid") and set of permissions ("Tperm") defined for the AIF-generic model in [I-D.ietf-ace-aif]; default values are the values "oscore-group-name" for "Toid" and "oscore-group-roles" for "Tperm".

Type name: application

Subtype name: aif-groupcomm-oscore+cbor

Required parameters: "Toid", "Tperm"

Optional parameters: N/A

Encoding considerations: Must be encoded as a CBOR array, each element of which is an array [Toid, Tperm] as defined in Section 3 of [[This document]].

Security considerations: See Section 22 of [[This document]].

Interoperability considerations: N/A

Published specification: [[This document]]

Applications that use this media type: The type is used by applications that want to express authorization information about joining OSCORE groups, as specified in [[This document]].

Fragment identifier considerations: N/A

Additional information: N/A

Person & email address to contact for further information: iesg@ietf.org (mailto:iesg@ietf.org)

Intended usage: COMMON

Restrictions on usage: None

Author: Marco Tiloca marco.tiloca@ri.se (mailto:marco.tiloca@ri.se)

Change controller: IESG

Provisional registration? No

23.10. CoAP Content-Format Registry

IANA is asked to register the following entry to the "CoAP Content-Formats" registry, within the "CoRE Parameters" registry:

Media Type: application/aif-groupcomm-oscore+cbor;Toid="oscore-group-name",Tperm"oscore-group-roles"

Encoding: -

ID: TBD

Reference: [[This document]]

23.11. Group OSCORE Roles Registry

This document establishes the IANA "Group OSCORE Roles" Registry. The Registry has been created to use the "Expert Review" registration procedure [RFC8126]. Expert review guidelines are provided in Section 23.15.

This registry includes the possible roles that nodes can take in an OSCORE group, each in combination with a numeric identifier. These numeric identifiers are used to express authorization information about joining OSCORE groups, as specified in Section 3 of [[This document]].

The columns of this registry are:

* Name: A value that can be used in documents for easier comprehension, to identify a possible role that nodes can take in an OSCORE group.

* Value: The numeric identifier for this role. Integer values greater than 65535 are marked as "Private Use", all other values use the registration policy "Expert Review" [RFC8126].

* Description: This field contains a brief description of the role.

* Reference: This contains a pointer to the public specification for the role.

This registry will be initially populated by the values in Figure 1.

The Reference column for all of these entries will be [[This document]].

23.12. CoRE Resource Type Registry

IANA is asked to register a new Resource Type (rt=) Link Target Attribute in the "Resource Type (rt=) Link Target Attribute Values" subregistry under the "Constrained Restful Environments (CoRE) Parameters" [CORE.Parameters] registry.

* Value: "core.osc.gm"

* Description: Group-membership resource of an OSCORE Group Manager.

* Reference: [[This document]]

23.13. ACE Scope Semantics Registry

IANA is asked to register the following entry in the "ACE Scope Semantics" registry defined in Section 10.12 of [I-D.ietf-ace-key-groupcomm].

* Value: SEM_ID_TBD

- Description: Access to OSCORE groups through the ACE Group Manager.

- Reference: [[This document]]

23.14. ACE Groupcomm Errors

IANA is asked to register the following entry in the "ACE Groupcomm Errors" registry defined in Section 10.13 of [I-D.ietf-ace-key-groupcomm].

- Value: 7

- Description: Signatures not used in the group.

- Reference: [[This document]]

- Value: 8

- Description: Operation permitted only to signature verifiers.

- Reference: [[This document]]

- Value: 9

- Description: Group currently not active.

- Reference: [[This document]]

23.15. Expert Review Instructions

The IANA Registry established in this document is defined as "Expert Review". This section gives some general guidelines for what the experts should be looking for, but they are being designated as experts for a reason so they should be given substantial latitude.

Expert reviewers should take into consideration the following points:

- Clarity and correctness of registrations. Experts are expected to check the clarity of purpose and use of the requested entries. Experts should inspect the entry for the considered role, to verify the correctness of its description against the role as intended in the specification that defined it. Expert should consider requesting an opinion on the correctness of registered parameters from the Authentication and Authorization for Constrained Environments (ACE) Working Group and the Constrained RESTful Environments (CoRE) Working Group.

  Entries that do not meet these objective of clarity and completeness should not be registered.

- Duplicated registration and point squatting should be discouraged. Reviewers are encouraged to get sufficient information for registration requests to ensure that the usage is not going to duplicate one that is already registered and that the point is likely to be used in deployments.

- Experts should take into account the expected usage of roles when approving point assignment. Given a 'Value' V as code point, the length of the encoding of (2^(V+1) - 1) should be weighed against the usage of the entry, considering the resources and capabilities of devices it will be used on. Additionally, given a 'Value' V as code point, the length of the encoding of (2^(V+1) - 1) should be weighed against how many code points resulting in that encoding length are left, and the resources and capabilities of devices it will be used on.

- Specifications are recommended. When specifications are not provided, the description provided needs to have sufficient information to verify the points above.

24. References

24.1. Normative References

[CORE.Parameters]
IANA, "Constrained RESTful Environments (CoRE) Parameters", <https://www.iana.org/assignments/core-parameters/core-parameters.xhtml>.

[COSE.Algorithms]
IANA, "COSE Algorithms", <https://www.iana.org/assignments/cose/cose.xhtml#algorithms>.

[COSE.Elliptic.Curves]
IANA, "COSE Elliptic Curves", <https://www.iana.org/assignments/cose/cose.xhtml#elliptic-curves>.

[COSE.Header.Parameters]
IANA, "COSE Header Parameters", <https://www.iana.org/assignments/cose/cose.xhtml#header-parameters>.

[COSE.Key.Types]
IANA, "COSE Key Types", <https://www.iana.org/assignments/cose/cose.xhtml#key-type>.

[I-D.ietf-ace-aif]
Bormann, C., "An Authorization Information Format (AIF) for ACE", Work in Progress, Internet-Draft, draft-ietf-ace-aif-03, 24 June 2021, <https://www.ietf.org/archive/id/draft-ietf-ace-aif-03.txt>.

[I-D.ietf-ace-key-groupcomm]
Palombini, F. and M. Tiloca, "Key Provisioning for Group Communication using ACE", Work in Progress, Internet-Draft, draft-ietf-ace-key-groupcomm-13, 12 July 2021, <https://www.ietf.org/archive/id/draft-ietf-ace-key-groupcomm-13.txt>.

[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)", Work in Progress, Internet-Draft, draft-ietf-ace-oauth-authz-43, 10 July 2021, <https://www.ietf.org/archive/id/draft-ietf-ace-oauth-authz-43.txt>.

[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", Work in Progress, Internet-Draft, draft-ietf-ace-oscore-profile-19, 6 May 2021, <https://www.ietf.org/archive/id/draft-ietf-ace-oscore-profile-19.txt>.

[I-D.ietf-core-oscore-groupcomm]
Tiloca, M., Selander, G., Palombini, F., Mattsson, J. P., and J. Park, "Group OSCORE - Secure Group Communication for CoAP", Work in Progress, Internet-Draft, draft-ietf-core-oscore-groupcomm-12, 12 July 2021, <https://www.ietf.org/archive/id/draft-ietf-core-oscore-groupcomm-12.txt>.

[I-D.ietf-cose-rfc8152bis-algs]
Schaad, J., "CBOR Object Signing and Encryption (COSE): Initial Algorithms", Work in Progress, Internet-Draft, draft-ietf-cose-rfc8152bis-algs-12, 24 September 2020, <https://www.ietf.org/archive/id/draft-ietf-cose-rfc8152bis-algs-12.txt>.

[I-D.ietf-cose-rfc8152bis-struct]
Schaad, J., "CBOR Object Signing and Encryption (COSE): Structures and Process", Work in Progress, Internet-Draft, draft-ietf-cose-rfc8152bis-struct-15, 1 February 2021, <https://www.ietf.org/archive/id/draft-ietf-cose-rfc8152bis-struct-15.txt>.

[NIST-800-56A]
Barker, E., Chen, L., Roginsky, A., Vassilev, A., and R. Davis, "Recommendation for Pair-Wise Key-Establishment Schemes Using Discrete Logarithm Cryptography - NIST Special Publication 800-56A, Revision 3", April 2018, <https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-56Ar3.pdf>.

[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>.

[RFC5705] Rescorla, E., "Keying Material Exporters for Transport Layer Security (TLS)", RFC 5705, DOI 10.17487/RFC5705, March 2010, <https://www.rfc-editor.org/info/rfc5705>.

[RFC6838] Freed, N., Klensin, J., and T. Hansen, "Media Type Specifications and Registration Procedures", BCP 13, RFC 6838, DOI 10.17487/RFC6838, January 2013, <https://www.rfc-editor.org/info/rfc6838>.

[RFC6979] Pornin, T., "Deterministic Usage of the Digital Signature Algorithm (DSA) and Elliptic Curve Digital Signature Algorithm (ECDSA)", RFC 6979, DOI 10.17487/RFC6979, August 2013, <https://www.rfc-editor.org/info/rfc6979>.

[RFC7252] Shelby, Z., Hartke, K., and C. Bormann, "The Constrained Application Protocol (CoAP)", RFC 7252, DOI 10.17487/RFC7252, June 2014, <https://www.rfc-editor.org/info/rfc7252>.

[RFC7748] Langley, A., Hamburg, M., and S. Turner, "Elliptic Curves for Security", RFC 7748, DOI 10.17487/RFC7748, January 2016, <https://www.rfc-editor.org/info/rfc7748>.

[RFC8017] Moriarty, K., Ed., Kaliski, B., Jonsson, J., and A. Rusch, "PKCS #1: RSA Cryptography Specifications Version 2.2", RFC 8017, DOI 10.17487/RFC8017, November 2016, <https://www.rfc-editor.org/info/rfc8017>.

[RFC8032] Josefsson, S. and I. Liusvaara, "Edwards-Curve Digital Signature Algorithm (EdDSA)", RFC 8032, DOI 10.17487/RFC8032, January 2017, <https://www.rfc-editor.org/info/rfc8032>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/info/rfc8126>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

[RFC8447] Salowey, J. and S. Turner, "IANA Registry Updates for TLS and DTLS", RFC 8447, DOI 10.17487/RFC8447, August 2018, <https://www.rfc-editor.org/info/rfc8447>.

[RFC8610] Birkholz, H., Vigano, C., and C. Bormann, "Concise Data Definition Language (CDDL): A Notational Convention to Express Concise Binary Object Representation (CBOR) and JSON Data Structures", RFC 8610, DOI 10.17487/RFC8610, June 2019, <https://www.rfc-editor.org/info/rfc8610>.

[RFC8613] Selander, G., Mattsson, J., Palombini, F., and L. Seitz, "Object Security for Constrained RESTful Environments (OSCORE)", RFC 8613, DOI 10.17487/RFC8613, July 2019, <https://www.rfc-editor.org/info/rfc8613>.

[RFC8742] Bormann, C., "Concise Binary Object Representation (CBOR) Sequences", RFC 8742, DOI 10.17487/RFC8742, February 2020, <https://www.rfc-editor.org/info/rfc8742>.

[RFC8949] Bormann, C. and P. Hoffman, "Concise Binary Object Representation (CBOR)", STD 94, RFC 8949, DOI 10.17487/RFC8949, December 2020, <https://www.rfc-editor.org/info/rfc8949>.

24.2. Informative References

[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)", Work in Progress, Internet-Draft, draft-ietf-ace-dtls-authorize-18, 4 June 2021, <https://www.ietf.org/archive/id/draft-ietf-ace-dtls-authorize-18.txt>.

[I-D.ietf-ace-oscore-gm-admin]
Tiloca, M., Hoeglund, R., Stok, P. V. D., Palombini, F., and K. Hartke, "Admin Interface for the OSCORE Group Manager", Work in Progress, Internet-Draft, draft-ietf-ace-oscore-gm-admin-03, 12 July 2021, <https://www.ietf.org/archive/id/draft-ietf-ace-oscore-gm-admin-03.txt>.

[I-D.ietf-core-coap-pubsub]
Koster, M., Keranen, A., and J. Jimenez, "Publish-Subscribe Broker for the Constrained Application Protocol (CoAP)", Work in Progress, Internet-Draft, draft-ietf-core-coap-pubsub-09, 30 September 2019, <https://www.ietf.org/archive/id/draft-ietf-core-coap-pubsub-09.txt>.

[I-D.ietf-core-groupcomm-bis]
Dijk, E., Wang, C., and M. Tiloca, "Group Communication for the Constrained Application Protocol (CoAP)", Work in Progress, Internet-Draft, draft-ietf-core-groupcomm-bis-04, 12 July 2021, <https://www.ietf.org/archive/id/draft-ietf-core-groupcomm-bis-04.txt>.

[I-D.ietf-cose-cbor-encoded-cert]
Raza, S., Hoeglund, J., Selander, G., Mattsson, J. P., and M. Furuhed, "CBOR Encoded X.509 Certificates (C509 Certificates)", Work in Progress, Internet-Draft, draft-ietf-cose-cbor-encoded-cert-01, 25 May 2021, <https://www.ietf.org/archive/id/draft-ietf-cose-cbor-encoded-cert-01.txt>.

[I-D.ietf-rats-uccs]
Birkholz, H., O'Donoghue, J., Cam-Winget, N., and C. Bormann, "A CBOR Tag for Unprotected CWT Claims Sets", Work in Progress, Internet-Draft, draft-ietf-rats-uccs-00, 19 May 2021, <https://www.ietf.org/archive/id/draft-ietf-rats-uccs-00.txt>.

[I-D.tiloca-core-oscore-discovery]
Tiloca, M., Amsuess, C., and P. V. D. Stok, "Discovery of OSCORE Groups with the CoRE Resource Directory", Work in Progress, Internet-Draft, draft-tiloca-core-oscore-discovery-09, 12 July 2021, <https://www.ietf.org/archive/id/draft-tiloca-core-oscore-discovery-09.txt>.

[RFC5869] Krawczyk, H. and P. Eronen, "HMAC-based Extract-and-Expand Key Derivation Function (HKDF)", RFC 5869, DOI 10.17487/RFC5869, May 2010, <https://www.rfc-editor.org/info/rfc5869>.

[RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, DOI 10.17487/RFC6347, January 2012, <https://www.rfc-editor.org/info/rfc6347>.

[RFC6690] Shelby, Z., "Constrained RESTful Environments (CoRE) Link Format", RFC 6690, DOI 10.17487/RFC6690, August 2012, <https://www.rfc-editor.org/info/rfc6690>.

[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>.

[RFC7641] Hartke, K., "Observing Resources in the Constrained Application Protocol (CoAP)", RFC 7641, DOI 10.17487/RFC7641, September 2015, <https://www.rfc-editor.org/info/rfc7641>.

[RFC7925] Tschofenig, H., Ed. and T. Fossati, "Transport Layer Security (TLS) / Datagram Transport Layer Security (DTLS) Profiles for the Internet of Things", RFC 7925, DOI 10.17487/RFC7925, July 2016, <https://www.rfc-editor.org/info/rfc7925>.

[RFC8392] Jones, M., Wahlstroem, E., Erdtman, S., and H. Tschofenig, "CBOR Web Token (CWT)", RFC 8392, DOI 10.17487/RFC8392, May 2018, <https://www.rfc-editor.org/info/rfc8392>.

Appendix A. Profile Requirements

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

- REQ1 - If the value of the GROUPNAME URI path and the group name in the Access Token scope (gname in Section 3.1 of [I-D.ietf-ace-key-groupcomm]) do not match, specify the mechanism to map the GROUPNAME value in the URI to the group name: not applicable, since a match is required.

- REQ2 - Specify the encoding and value of roles, for scope entries of 'scope': see Section 3 and Section 4.1.

- REQ3 - if used, specify the acceptable values for 'sign_alg': values from the "Value" column of the "COSE Algorithms" Registry [COSE.Algorithms].

- REQ4 - If used, specify the acceptable values for 'sign_parameters': format and values from the COSE algorithm capabilities as specified in the "COSE Algorithms" Registry [COSE.Algorithms].

- REQ5 - If used, specify the acceptable values for 'sign_key_parameters': format and values from the COSE key type capabilities as specified in the "COSE Key Types" Registry [COSE.Key.Types].

- REQ6 - Specify the acceptable formats for encoding public keys and, if used, the acceptable values for 'pub_key_enc': acceptable formats explicitly provide the full set of information related to the public key algorithm (see Section 6.1 and Section 6.4). Consistent acceptable values for 'pub_key_enc' are taken from the "Label" column of the "COSE Header Parameters" Registry [COSE.Header.Parameters].

- REQ7 - Register a Resource Type for the root url-path, which is used to discover the correct url to access at the KDC (see Section 4.1 of [I-D.ietf-ace-key-groupcomm]): the Resource Type (rt=) Link Target Attribute value "core.osc.gm" is registered in Section 23.12.

- REQ8 - Define what operations (e.g., CoAP methods) are allowed on each resource, for each role defined in REQ2: see Section 5.5.

- REQ9 - Specify the exact encoding of group identifier (see Section 4.1.1.1 of [I-D.ietf-ace-key-groupcomm]): CBOR byte string (see Section 17).

- REQ10 - Format of the 'key' value: see Section 6.4.

- REQ11 - Acceptable values of 'gkty': Group_OSCORE_Input_Material object (see Section 6.4).

- REQ12 - Specify the format of the identifiers of group members: the Sender ID used in the OSCORE group (see Section 6.4 and Section 10).

- REQ13 - Specify the communication protocol that the members of the group must use: CoAP [RFC7252], possibly over IP multicast [I-D.ietf-core-groupcomm-bis].

- REQ14 - Specify the security protocols that the group members must use to protect their communication: Group OSCORE [I-D.ietf-core-oscore-groupcomm].

- REQ15 - Specify and register the application profile identifier: coap_group_oscore_app (see Section 23.5).

- REQ16 - Specify policies at the KDC to handle member ids that are not included in 'get_pub_keys': see Section 10.

- REQ17 - If used, specify the format and content of 'group_policies' and its entries: see Section 6.4.

- REQ18 - Specify the format of newly-generated individual keying material for group members, or of the information to derive it, and corresponding CBOR label: see Section 9.

- REQ19 - Specify how the communication is secured between the Client and KDC: by means of any transport profile of ACE [I-D.ietf-ace-oauth-authz] between Client and Group Manager that complies with the requirements in Appendix C of [I-D.ietf-ace-oauth-authz].

- REQ20 - Specify the exact approaches used to compute and verify the PoP evidence to include in 'client_cred_verify', and which of those approaches is used in which case: see Section 6.2 and Section 6.3.

- REQ21 - Specify how the nonce N_S is generated, if the token is not being posted (e.g., if it is used directly to validate TLS instead): see Section 6.2.1.

- REQ22 - Specify if 'mgt_key_material' is used, and if yes specify its format and content: not used in this version of the profile.

- REQ23 - Define the initial value of the 'num' parameter: The initial value MUST be set to 0 when creating the OSCORE group, e.g., as in [I-D.ietf-ace-oscore-gm-admin].

- REQ24 - Specify and register the identifier of newly defined semantics for binary scopes: see Section 23.13.

- OPT1 (Optional) - Specify the negotiation of parameter values for signature algorithm and signature keys, if 'sign_info' is not used: possible early discovery by using the approach based on the CoRE Resource Directory described in [I-D.tiloca-core-oscore-discovery].

- OPT2 (Optional) - Specify additional parameters used in the Token Post exchange: 'ecdh_info', to negotiate the ECDH algorithm, ECDH algorithm parameters, ECDH key parameters and exact encoding of public keys used in the group, in case the joining node supports the pairwise mode of Group OSCORE.

- OPT3 (Optional) - Specify the encoding of 'pub_keys_repos' if the default is not used: no.

- OPT4 (Optional) - 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: no.

- OPT5 (Optional) - Specify possible or required payload formats for specific error cases: send a 4.00 (Bad Request) response to a Joining Request (see Section 6.3).

- OPT6 (Optional) - Specify the behavior of the handler in case of failure to retrieve a public key for the specific node: send a 4.00 (Bad Request) response to a Joining Request (see Section 6.3).

- OPT7 (Optional) - Specify CBOR values to use for abbreviating identifiers of roles in the group or topic: see Section 4.1.

- OPT8 (Optional) - Specify for the KDC to perform group rekeying (together or instead of renewing individual keying material) when receiving a Key Renewal Request: the Group Manager SHOULD NOT perform a group rekeying, unless already scheduled to occur shortly (see Section 9).

- OPT9 (Optional) - Specify the functionalities implemented at the 'control_uri' resource hosted at the Client, including message exchange encoding and other details (see Section 4.1.2.1 of [I-D.ietf-ace-key-groupcomm]): see Section 19 for the eviction of a group member; see Section 20 for the group rekeying process.

- OPT10 (Optional) - Specify how the identifier of the sender's public key is included in the group request: no.

- OPT11 (Optional) - Specify additional identifiers of error types, as values of the 'error' field in an error response from the KDC: see Section 23.14.

Appendix B. Extensibility for Future COSE Algorithms

As defined in Section 8.1 of [I-D.ietf-cose-rfc8152bis-algs], future algorithms can be registered in the "COSE Algorithms" Registry [COSE.Algorithms] as specifying none or multiple COSE capabilities.

To enable the seamless use of such future registered algorithms, this section defines a general, agile format for:

- Each 'ecdh_info_entry' of the 'ecdh_info' parameter in the Token Post response, see Section 6.1 and Section 6.1.1;

- The 'sign_params' and 'ecdh_params' parameters within the 'key' parameter, see Section 6.4, as part of the response payloads used in Section 6.4, Section 8.1, Section 8.2 and Section 20.

Appendix B of [I-D.ietf-ace-key-groupcomm] describes the analogous general format for 'sign_info_entry' of the 'sign_info' parameter in the Token Post response, see Section 6.1.

If any of the currently registered COSE algorithms is considered, using this general format yields the same structure defined in this document for the items above, thus ensuring retro-compatibility.

B.1. Format of 'ecdh_info_entry'

The format of each 'ecdh_info_entry' (see Section 6.1 and Section 6.1.1) is generalized as follows. Given N the number of elements of the 'ecdh_parameters' array, i.e., the number of COSE capabilities of the ECDH algorithm, then:

- 'ecdh_key_parameters' is replaced by N elements 'ecdh_capab_i', each of which is a CBOR array.
- The i-th array following 'ecdh_parameters', i.e., 'ecdh_capab_i' (i = 0, ..., N-1), is the array of COSE capabilities for the algorithm capability specified in 'ecdh_parameters'[i].

```
ecdh_info_entry =
[
  id : gname / [ + gname ],
  ecdh_alg : int / tstr,
  ecdh_parameters : [ alg_capab_1 : any,
                      alg_capab_2 : any,
                      ...,
                      alg_capab_N : any],
  ecdh_capab_1 : [ any ],
  ecdh_capab_2 : [ any ],
  ...,
  ecdh_capab_N : [ any ],
  pub_key_enc = int / nil
]

gname = tstr
```

Figure 13: 'ecdh_info_entry' with general format

B.2. Format of 'key'

The format of 'key' (see Section 6.4) is generalized as follows.

- The 'sign_params' array includes N+1 elements, whose exact structure and value depend on the value of the signature algorithm specified in 'sign_alg'.
  - The first element, i.e., 'sign_params'[0], is the array of the N COSE capabilities for the signature algorithm, as specified for that algorithm in the "Capabilities" column of the "COSE Algorithms" Registry [COSE.Algorithms] (see Section 8.1 of [I-D.ietf-cose-rfc8152bis-algs]).

  - Each following element 'sign_params'[i], i.e., with index i > 0, is the array of COSE capabilities for the algorithm capability specified in 'sign_params'[0][i-1].

  For example, if 'sign_params'[0][0] specifies the key type as capability of the algorithm, then 'sign_params'[1] is the array of COSE capabilities for the COSE key type associated to the signature algorithm, as specified for that key type in the "Capabilities" column of the "COSE Key Types" Registry [COSE.Key.Types] (see Section 8.2 of [I-D.ietf-cose-rfc8152bis-algs]).

* The 'ecdh_params' array includes M+1 elements, whose exact structure and value depend on the value of the ECDH algorithm specified in 'ecdh_alg'.

  - The first element, i.e., 'ecdh_params'[0], is the array of the M COSE capabilities for the ECDH algorithm, as specified for that algorithm in the "Capabilities" column of the "COSE Algorithms" Registry [COSE.Algorithms] (see Section 8.1 of [I-D.ietf-cose-rfc8152bis-algs]).

  - Each following element 'ecdh_params'[i], i.e., with index i > 0, is the array of COSE capabilities for the algorithm capability specified in 'ecdh_params'[0][i-1].

  For example, if 'ecdh_params'[0][0] specifies the key type as capability of the algorithm, then 'ecdh_params'[1] is the array of COSE capabilities for the COSE key type associated to the ECDH algorithm, as specified for that key type in the "Capabilities" column of the "COSE Key Types" Registry [COSE.Key.Types] (see Section 8.2 of [I-D.ietf-cose-rfc8152bis-algs]).

Appendix C. Document Updates

RFC EDITOR: PLEASE REMOVE THIS SECTION.

C.1. Version -10 to -11

* Removed redundancy of key type capabilities, from 'sign_info', 'ecdh_info' and 'key'.

* New resource to retrieve the Group Manager's public key.

* New resource to retrieve material for Signature Verifiers.

* New parameter 'sign_enc_alg' related to the group mode.

- 'pub_key_enc' takes value from the COSE Header Parameters registry.

- Improved alignment of the Joining Response payload with the Group OSCORE Security Context parameters.

- Recycling Group IDs by tracking "Birth GIDs".

- Error handling in case of non available Sender IDs upon joining.

- Error handling in case EdDSA public keys with invalid Y coordinate when the pairwise mode of Group OSCORE is supported.

- Generalized proof-of-possession (PoP) for the joining node's private key; defined Diffie-Hellman based PoP for OSCORE groups using only the pairwise mode.

- Proof-of-possession of the Group Manager's private key in the Joining Response.

- Always use 'peer_identifiers' to convey Sender IDs as node identifiers.

- Stale Sender IDs provided when rekeying the group.

- New resource for late retrieval of stale Sender IDs.

- Added examples of message exchanges.

- Revised default values of group configuration parameters.

- Fixes to IANA registrations.

- General format of parameters related to COSE capabilities, supporting future registered COSE algorithms (new Appendix).

C.2. Version -09 to -10

- Updated non-recycling policy of Sender IDs.

- Removed policies about Sender Sequence Number synchronization.

- 'control_path' renamed to 'control_uri'.

- Format of 'get_pub_keys' aligned with draft-ietf-ace-key-groupcomm.

- Additional way to inform of group eviction.

- Registered semantics identifier for extended scope format.
- Extended error handling, with error type specified in some error responses.
- Renumbered requirements.

C.3. Version -08 to -09

- The url-path "ace-group" is used.
- Added overview of admitted methods on the Group Manager resources.
- Added exchange of parameters relevant for the pairwise mode of Group OSCORE.
- The signed value for 'client_cred_verify' includes also the scope.
- Renamed the key material object as Group_OSCORE_Input_Material object.
- Replaced 'clientId' with 'group_SenderId'.
- Added message exchange for Group Names request-response.
- No reassignment of Sender ID and Gid in the same OSCORE group.
- Updates on group rekeying contextual with request of new Sender ID.
- Signature verifiers can also retrieve Group Names and URIs.
- Removed group policy about supporting Group OSCORE in pairwise mode.
- Registration of the resource type rt="core.osc.gm".
- Update list of requirements.
- Clarifications and editorial revision.

C.4. Version -07 to -08

- AIF specific data model to express scope entries.
- A set of roles is checked as valid when processing the Joining Request.

- Updated format of 'get_pub_keys' in the Joining Request.

- Payload format and default values of group policies in the Joining Response.

- Updated payload format of the FETCH request to retrieve public keys.

- Default values for group configuration parameters.

- IANA registrations to support the AIF specific data model.

C.5. Version -06 to -07

- Alignments with draft-ietf-core-oscore-groupcomm.

- New format of 'sign_info', using the COSE capabilities.

- New format of Joining Response parameters, using the COSE capabilities.

- Considerations on group rekeying.

- Editorial revision.

C.6. Version -05 to -06

- Added role of external signature verifier.

- Parameter 'rsnonce' renamed to 'kdcchallenge'.

- Parameter 'kdcchallenge' may be omitted in some cases.

- Clarified difference between group name and OSCORE Gid.

- Removed the role combination ["requester", "monitor"].

- Admit implicit scope and audience in the Authorization Request.

- New format for the 'sign_info' parameter.

- Scope not mandatory to include in the Joining Request.

- Group policy about supporting Group OSCORE in pairwise mode.

- Possible individual rekeying of a single requesting node combined with a group rekeying.

- Security considerations on reusage of signature challenges.

- Addressing optional requirement OPT8 from draft-ietf-ace-key-groupcomm

- Editorial improvements.

C.7. Version -04 to -05

- Nonce N_S also in error responses to the Joining Requests.

- Supporting single Access Token for multiple groups/topics.

- Supporting legal requesters/responders using the 'peer_roles' parameter.

- Registered and used dedicated label for TLS Exporter.

- Added method for uploading a new public key to the Group Manager.

- Added resource and method for retrieving the current group status.

- Fixed inconsistency in retrieving group keying material only.

- Clarified retrieval of keying material for monitor-only members.

- Clarification on incrementing version number when rekeying the group.

- Clarification on what is re-distributed with the group rekeying.

- Security considerations on the size of the nonces used for the signature challenge.

- Added CBOR values to abbreviate role identifiers in the group.

C.8. Version -03 to -04

- New abstract.

- Moved general content to draft-ietf-ace-key-groupcomm

- Terminology: node name; node resource.

- Creation and pointing at node resource.

- Updated Group Manager API (REST methods and offered services).

- Size of challenges 'cnonce' and 'rsnonce'.

- Value of 'rsnonce' for reused or non-traditionally-posted tokens.

- Removed reference to RFC 7390.

- New requirements from draft-ietf-ace-key-groupcomm

- Editorial improvements.

C.9. Version -02 to -03

- New sections, aligned with the interface of ace-key-groupcomm .

- Exchange of information on the signature algorithm and related parameters, during the Token POST (Section 4.1).

- Nonce 'rsnonce' from the Group Manager to the Client (Section 4.1).

- Client PoP signature in the Key Distribution Request upon joining (Section 4.2).

- Local actions on the Group Manager, upon a new node's joining (Section 4.2).

- Local actions on the Group Manager, upon a node's leaving (Section 12).

- IANA registration in ACE Groupcomm Parameters Registry.

- More fulfilled profile requirements (Appendix A).

C.10. Version -01 to -02

- Editorial fixes.

- Changed: "listener" to "responder"; "pure listener" to "monitor".

- Changed profile name to "coap_group_oscore_app", to reflect it is an application profile.

- Added the 'type' parameter for all requests to a Join Resource.

- Added parameters to indicate the encoding of public keys.

- Challenge-response for proof-of-possession of signature keys (Section 4).

- Renamed 'key_info' parameter to 'sign_info'; updated its format; extended to include also parameters of the signature key (Section 4.1).

- Code 4.00 (Bad request), in responses to joining nodes providing an invalid public key (Section 4.3).

- Clarifications on provisioning and checking of public keys (Sections 4 and 6).

- Extended discussion on group rekeying and possible different approaches (Section 7).

- Extended security considerations: proof-of-possession of signature keys; collision of OSCORE Group Identifiers (Section 8).

- Registered three entries in the IANA Registry "Sequence Number Synchronization Method Registry" (Section 9).

- Registered one public key encoding in the "ACE Public Key Encoding" IANA Registry (Section 9).

C.11. Version -00 to -01

- Changed name of 'req_aud' to 'audience' in the Authorization Request (Section 3.1).

- Added negotiation of signature algorithm/parameters between Client and Group Manager (Section 4).

- Updated format of the Key Distribution Response as a whole (Section 4.3).

- Added parameter 'cs_params' in the 'key' parameter of the Key Distribution Response (Section 4.3).

- New IANA registrations in the "ACE Authorization Server Request Creation Hints" Registry, "ACE Groupcomm Key" Registry, "OSCORE Security Context Parameters" Registry and "ACE Groupcomm Profile" Registry (Section 9).

Acknowledgments

The authors sincerely thank Christian Amsuess, Santiago Aragon, Stefan Beck, Carsten Bormann, Martin Gunnarsson, Rikard Hoeglund, Daniel Migault, Jim Schaad, Ludwig Seitz, Goeran Selander and Peter van der Stok for their comments and feedback.

The work on this document has been partly supported by VINNOVA and the Celtic-Next project CRITISEC; by the H2020 project SIFIS-Home (Grant agreement 952652); and by the EIT-Digital High Impact Initiative ACTIVE.

Authors' Addresses

Marco Tiloca
RISE AB
Isafjordsgatan 22
SE-164 29 Stockholm Kista
Sweden

Email: marco.tiloca@ri.se

Jiye Park
Universitaet Duisburg-Essen
Schuetzenbahn 70
45127 Essen
Germany

Email: ji-ye.park@uni-due.de

Francesca Palombini
Ericsson AB
Torshamnsgatan 23
SE-16440 Stockholm Kista
Sweden

Email: francesca.palombini@ericsson.com