Key Provisioning for Group Communication using ACE

Open points on:

draft-ietf-ace-key-groupcomm-10

Francesca Palombini, Ericsson
Marco Tiloca, RISE

Open point #1 (1/2)

› Scope encoding

The KDC acts also as RS for other resources, accessible via other applications.

C → KDC : POST / authz-info with scope encoded as CBOR byte string

How does the KDC know the format of scope at this point ?

- How does the KDC know how to parse and interpret the scope from the Token?
- How does the KDC know which possible application profile of ACE should be used?
  - Etc: for ace-key-groupcomm CBOR array wrapped in a CBOR byte string
- Arguable workaround: use different values of “audience” as a hint

/!\ General problem for RSs supporting several applications and application profiles /!\

Open point #1 (2/2)

- Scope encoding – Two possible solutions

- Prefix scope with one byte
  - Needs to be agreed between RS and AS (as part of the registration process)
  - If the same scope is reused for several RSs, they need to sync with the AS

- Register CBOR tags – One for each possible format scope
  - Longer than the single-byte prefix (+1?)
  - Useful already in the ACE application profile ace-key-groupcomm-oscore
  - How well does it scale? Can any application or application profile claim for a new tag?

We could describe both: register tags and describe in appendix how either of prefix or tag can be used

From IETF 109: - Ben: “Tags are architecturally cleaner but they don’t say anything about implementation”
- Carsten: “Need to think more; 1-byte CBOR tags registration is restricted”

Open point #2

› From the Requirement REQ7aa:

It is REQUIRED of the application profiles of this specification to define what operations (i.e. CoAP methods) are allowed on each resource, for each role defined in Section 3.1 according to REQ2 (REQ7aa).

Is “CoAP methods” too specific? Other communication protocols can be used to communicate in the groups, such as MQTT. Not even “REST methods” would be a good alternative.

Proposal: replace “*what operations (i.e. CoAP methods) are allowed*” with “*what operations (e.g. CoAP methods) are allowed*”

Objections?

Open point #3 (1/2)

› Possible reason for a group member to observe the resource ace-group/GROUPNAME/nodes/NODENAME at the KDC

If the KDC evicts that node from the group, the KDC can currently inform that node by sending a POST request to the node's resource under 'control_path'.

However, the KDC is also supposed to cancel the node resource ace-group/GROUPNAME/nodes/NODENAME altogether.

If the node was observing that resource, it would receive a 4.01 (Not Found) notification (without an Observe option).

Objections?

Open point #3 (2/2)

› Possible reason for a group member to observe the resource ace-group/GROUPNAME/nodes/NODENAME at the KDC

But if the node observes both this resource and the main group-membership resource at ace-group/GROUPNAME , it would receive notifications from both resources, each time the KDC changes the group key material.

Proposal: the node interested in observing both can especially observe the resource ace-group/GROUPNAME/nodes/NODENAME , by sending a GET request with both Observe:0 and No-Response:2 (Not interested in 2.xx responses).

Objections?

Open point #4

- In the PUT handler of ace-group/GROUPNAME/nodes/NODENAME
  - The payload is expected to be empty. Can the KDC just ignore any non-empty payload, or should it return 4.00 Bad Request? Preferences?
  - The KDC should return 5.03 Service Unavailable if no new individual key material (e.g., OSCORE Sender ID) can be assigned at the moment. Objections?

Open point #5 (1/2)

› In the FETCH handler of ace-group/GROUPNAME/pub-key , the payload format is

{get_pub_keys: [[IDs-filter], [roles-filter]] }

Proposal: we can consider to have instead the format

{get_pub_keys: [inclusion-flag, [IDs-filter], [roles-filter]] }

Open point #5 (2/2)

- In the FETCH handler of ace-group/GROUPNAME/pub-key

  Payload: {get_pub_keys: [inclusion-flag, [IDs-filter], [roles-filter]] }

  inclusion-flag = True

  IDs-filter is used as now: the public keys of those IDs are considered for being returned

  inclusion-flag = False

  IDs-filter is used in a reversed way: the public keys of all group members that don't have one of those IDs are considered for being returned.

  A group member that already has some public keys, can retrieve just possible other ones it is missing (e.g. of recently added group members), instead of retrieving the full set with a GET. Objections?

  The same format would apply to 'get_pub_keys' in the Joining Request, where 'inclusion-flag' would be always set to True.

Open point #6

- In the DELETE handler of ace-group/GROUPNAME/nodes/NODENAME

  Proposal: remove the sentence "*The KDC also verifies that the roles the client is granted in the group allow it to perform this operation on this resource*", since the role should not affect the ability or legitimacy to leave the group.

  As related cleanup, remove the text: " *the handler removes the client from the group identified by "GROUPNAME", for specific roles if roles were specified in the 'scope' field, or for all roles.*"

  Objections?

Open point #7

- The Joining Response and the response from ace-group/GROUPNAME/pub-key include the ‘pub_keys’ parameter, with the members’ public key. Now we say:

  "*... each COSE Key in the COSE_KeySet includes the node identifier of the corresponding group member as value of its ’kid’ key parameter. Alternative specific encodings of this parameter MAY be defined in applications of this specification (OPT1).*"

  So, i) if something different than COSE Keys is used; AND ii) that something does not have parameters to embed node identifiers → we would need an alternative way to provide node identifiers to the requesting client.

  Proposal: add one more optional parameter ‘peer_identifiers’ for these responses, analogous to ‘peer_roles’ but including node identifiers. It is included only if ‘pub_keys’ is present and the used encoding of public keys does not embed node identifiers. Objections?

Minor points

› The 'control_path' parameter in the Joining Request specifies a full URI.

  Then, it's more appropriate to rename it 'control_uri'. Objections?

› Avoid to use “group(s)” , “(set of) groups” or similar. Objections?

Thank you!

Comments/questions?