Key Provisioning for Group Communication using ACE

draft-ietf-ace-key-groupcomm-06

Francesca Palombini, Ericsson
Marco Tiloca, RISE

ACE WG, Interim, May 18, 2020

Quick Recap

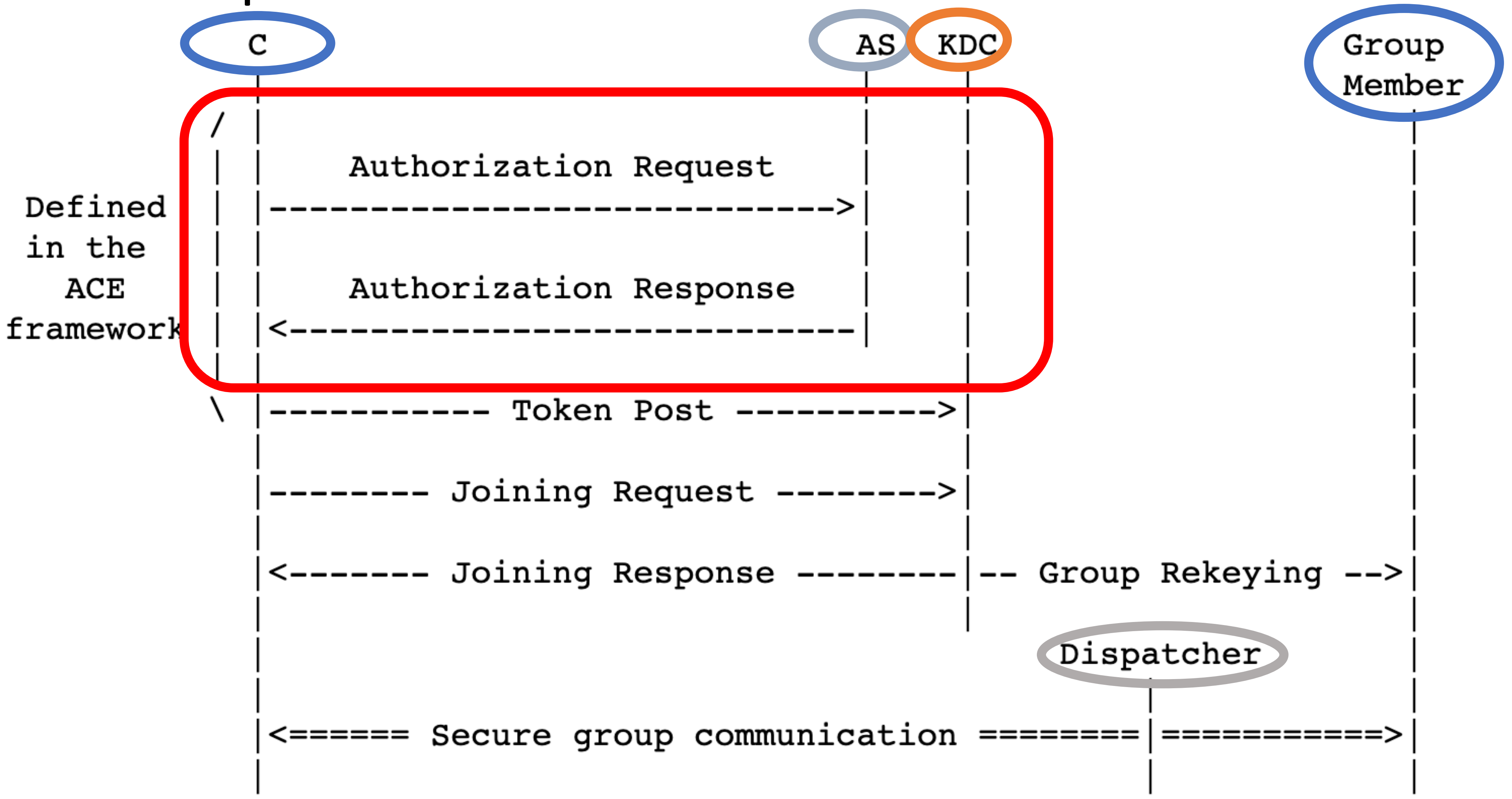
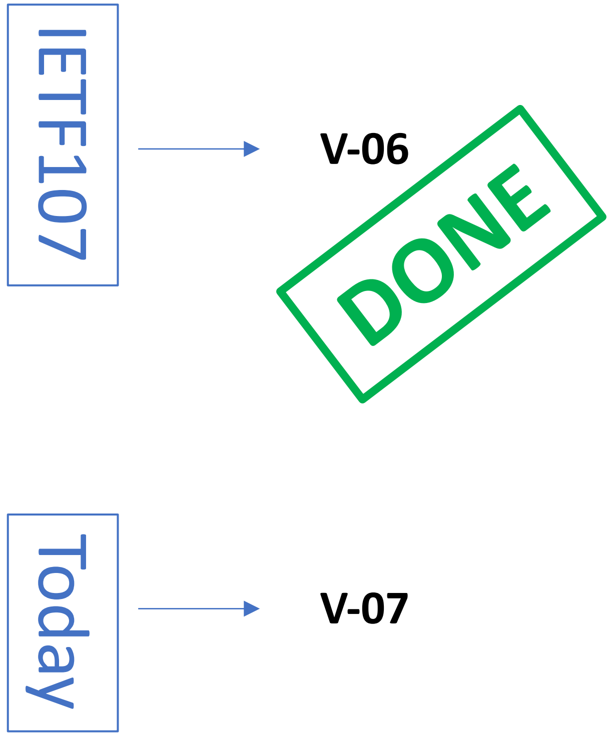


Figure 2: Message Flow Upon New Node's Joining

What’s left?

- High level status update (and where to find in-detail updates)
- TODOs left from February’s interim
- Updates that need ok from wg (already implemented)
  - “Scope question”
  - “Jim’s proposal on legal requestor”
- Open github issues (including open points from latest review)

- Open points from mailing list posts
  - “Keeping the same key identifier for groups”
  - “Considerations on congestion control”
  - “When does a KDC need to roll the keys over”
- Couple of open points to check from previous review
- Open points from Peter’s review



What happened since IETF 107 – status update

- Version -06 was submitted May 11th 2020
  - Based on review from Jim: https://mailarchive.ietf.org/arch/msg/ace/gsF02hHymShiYkr1pxvqWTNUcP4/
  - Discussion at the ACE IETF 107 virtual
  - PR v-06 was merged – all issues closed

TODO’s list

- Jim’s question 6 - Keeping the same key identifier for groups
- Jim’s question 5 - Congestion control needs to be included
- Jim’s question 4 - Missing Role - DONE
- Jim review -05 - DONE + TO ANSWER
- Jim Github Issue 59 - Multiple joins - DONE
- Jim Github Issue 60 - When are the public key and proof of possession required to be present - DONE
- Ace Interim Jan 2020 - DONE
- Jim review -04 → 4 open points
- Peter review -03
- Marco’s open points - DONE
- Jim review -03 - DONE + TO ANSWER
- Ludwig review -02 - DONE
- Jim’s comment to Ludwig’s review - DONE
- Jim’s question 3 – map points for join - DONE
- Jim’s question 2 – get as join - DONE + TO ANSWER
- Jim’s question 1 - When does a KDC need to roll the keys over
- Daniel review -02 - DONE

Jim review -04 - Jan 30, 2020 https://mailarchive.ietf.org/arch/msg/ace/mR-qwWEhmmOFIVL2ToH5pc2GXqs

Section 3.1 - Should we try and harmonize the scope format here with either the format in MQTT or with the one that Carsten once proposed. The multiplicity of scope formats is going to become a problem at some point.

Section 4.1 - How does one enforce the MUST NOT for the resource name of /ace-group?

```
Note that the root url-path "ace-group" given here are default names:
implementations are not required to use these names, and can define their own
instead. […]
```

```
* /ace-group : this resource is fixed and indicates that this specification is
used. Other applications that run on a KDC implementing this specification MUST
NOT use this same resource.
```

Jim review -04 - Jan 30, 2020
https://mailarchive.ietf.org/arch/msg/ace/mR-qwWEhmmOFIVL2ToH5pc2GXqs

Section 4.1 - I am not sure, but I think that I might want to suggest putting a node in the path between guid and the node id for a single end point. That is /ace-group/gid/nodes/node. This allows for one to not deal with potential name collisions with the other nodes under gid.

Should a client be able to ask for a previous set of keying material? Consider the case of a client having key material n, missing update n+1 and getting update n+2. The client then gets a message encrypted to update n+1. It never saw the material and thus cannot decrypt the message.

Jim 1: When does a KDC need to roll the keys over https://mailarchive.ietf.org/arch/msg/ace/bDuQvYeqks78UB_DDKmzFXWROMI/

[…] The one case that I have not been able to get a good handle on is as follows:

The KDC persists keys and key ids in a database.

The KDC at some point crashes and then restarts. Should the KDC roll the epoch and the key material on re-start or should it just load the current key material and continue with it?

A client that tries to JOIN would be unable to do so because the KDC does not respond so that would not force a roll over.

A client that tries to do a RENEWAL because of IV exhaustion would be unable to do so and would have to go quiet until the KDC because alive and that would not force a roll over.

A client that tries to do a LEAVE would not be able to tell the KDC that.

This is the one case that having the KDC do the roll over of the keys would make sense. However the KDC would be unable to tell anybody of that decision as all of the observe relationships would have been lost at the point so that in some respects the damage of allowing an entity that left to continue reading messages would continue.

Jim 5: Congestion control needs to be included https://mailarchive.ietf.org/arch/msg/ace/0YwfcQ6DEjtwdblMKPeMzCsJzZk/

I had a weird weekend trying to get coverage testing up for my Observe implementation and in the process found out that it had not implemented the required congestion control. As part of this I had to go back and do a careful read of RFC 7641 to get things right in my code and following that I thought that this document really needs to have a discussion of congestion control as well. Part of this can be a reference to section 4.5.1 of RFC 7641 where we are using observe but we need to go through the document and potentially look at some other places where we need to discuss congestion as well.

I will also note that observe does not guarantee that all messages will be sent to a client, just that after a while it will have the most current version of the content. This means that there is a high probability that clients will not get every update of key material if turn over is being done at all quickly.

Jim 6: Keeping the same key identifier for groups https://mailarchive.ietf.org/arch/msg/ace/I1603SP6I2MmPZNb55QxGgNWIyU/

As Ludwig pointed out during the F2F, it makes far more sense to try and keep an entity using the same key identifier for as long as possible. This is in part to make sure that signing keys do not need to be retrieved if they can be easily cached. In looking at this deeper during my implementation I ended up with the following question:

The way that I have set things up in my implementation it is simple to ensure that the same kid value is going to be used with the same CWT, however it might make more sense to use the signing key as the continuity identifier instead. The issue that arises in this case is that there might be two different active CWT objects that are associated with the same signing key. That is there are two CWTs but the same signing key was used while doing a join operation. I already do some matching between different CWTs by assuming that if the bearer key in the CWT is the same then they are sufficiently equivalent to threat them as the same. This lead to some interesting discussions in Montreal about if this meant just the "secret" or if it meant all of the elements provided by the AS which are used in the key derivation process. (I have gone back and forth on this and currently am sitting on the "just the secret" side of the fence.)

Plan forward

- Agree on these points
- Submit v-07