Group OSCORE - Secure Group Communication for CoAP

draft-ietf-core-oscore-groupcomm-10

Marco Tiloca, RISE
Göran Selander, Ericsson
Francesca Palombini, Ericsson
Jiye Park, Universität Duisburg-Essen

IETF 109, CoRE WG, November 17th, 2020

Update since the July meeting

- Version -10 submitted before the cut-off
  - Addressed WGLC comments [1][2]
  - Addressed more points discussed around IETF 108

- 3rd interop during this Hackathon
  - Rikard Höglund, Peter van der Stok, Christian Amsüss
  - The pairwise mode was also successfully tested

- New review of -10 from Christian [3] – Thanks!

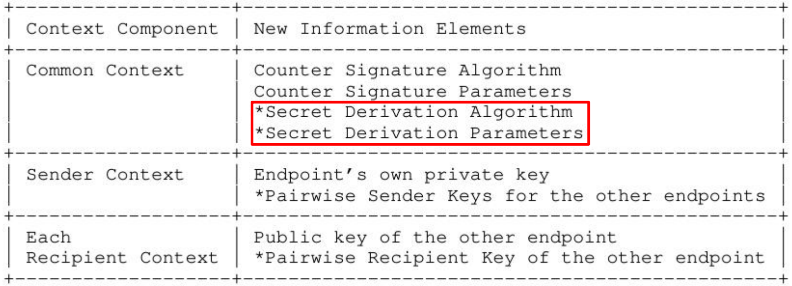
[1] https://mailarchive.ietf.org/arch/msg/core/VMhrAPEt4TE8jahatVd1EoDzdMI/

[2] https://mailarchive.ietf.org/arch/msg/core/tOHaMpTrWJ2CfsX2E5IGS8qpt-U/

[3] https://mailarchive.ietf.org/arch/msg/core/pXEyxhbf-s2wgGDzrDhUNPsHZZc/

Main updates in -10

- Common Security Context
  - – Removed “Counter Signature Key Parameters”
  - – Added parameters for the pairwise mode



Context Component	New Information Elements
Common Context	Counter Signature Algorithm Counter Signature Parameters *Secret Derivation Algorithm *Secret Derivation Parameters
Sender Context	Endpoint’s own private key *Pairwise Sender Keys for the other endpoints
Each Recipient Context	Public key of the other endpoint *Pairwise Recipient Key of the other endpoint

- A server may respond with 5.03
  - – Not having the public key of the client yet
  - – Not possible to retrieve it right away

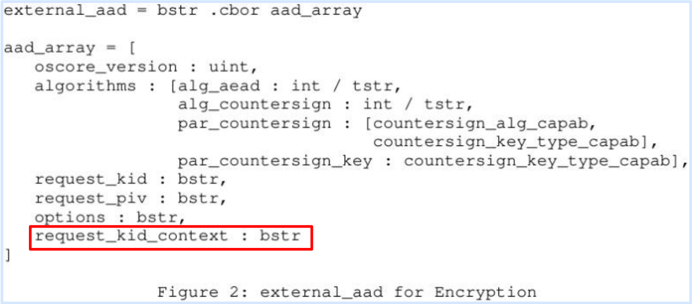
- Non-recycling policies for the Group Manager
  - – Don’t reassign the same Sender ID in the same group
    - Open point about slightly relaxing it
  - – Don’t reassign the same Group ID to the same group

Main updates in -10

- Sender Sequence Number (SSN)
  - Keep one shared space, for group mode and pairwise mode
  - Reset to 0 when establishing a new context
    - Got a new Sender ID; or whole group rekeying

- Request protected with Ctx_old , response protected with Ctx_new
  - The server MUST use its SSN as Partial IV of that response

- Added ‘request_kid_context’ to the external_aad
  - Support observations beyond a group rekeying
  - Required now that the SSN is reset upon rekeying
  - A notification can’t match with 2 registration requests



```
external_aad = bstr .cbor aad_array

aad_array = [
   oscore_version : uint,
   algorithms : [alg_aead : int / tstr,
                 alg_countersign : int / tstr,
                 par_countersign : [countersign_alg_capab,
                                    countersign_key_type_capab],
                 par_countersign_key : countersign_key_type_capab],
   request_kid : bstr,
   request_piv : bstr,
   options : bstr,
   request_kid_context : bstr
]
```

Figure 2: external_aad for Encryption

Main updates in -10

- More on supporting Observation

- The **client and server** store the ‘kid’ and ‘kid context’ from the registration request
  - Used to correctly build the external_aad of notifications

- The **client** stores ‘kid’ and ‘kid context’ from the registration request
  - Only if actually interested in continuing the observation beyond a group rekeying

- The **client** stores an invariant identifier of the group
  - Unchanged over group rekeyings, e.g. the “group name” of *ace-key-groupcomm-oscore*
  - Simpler to get updated key material from the Group Manager, if a rekeying was missed
  - Only if actually interested in continuing the observation beyond a group rekeying

From Christian’s review

- Improve distinction between anti-replay and freshness
  - Clarify server “synchronization” with a client, as related to freshness

- Methods in Appendix E
  - E.1 “Best effort” and E.2 “Baseline” are not significant and can be removed
  - E.3 using Echo makes a Replay Window valid and brings freshness

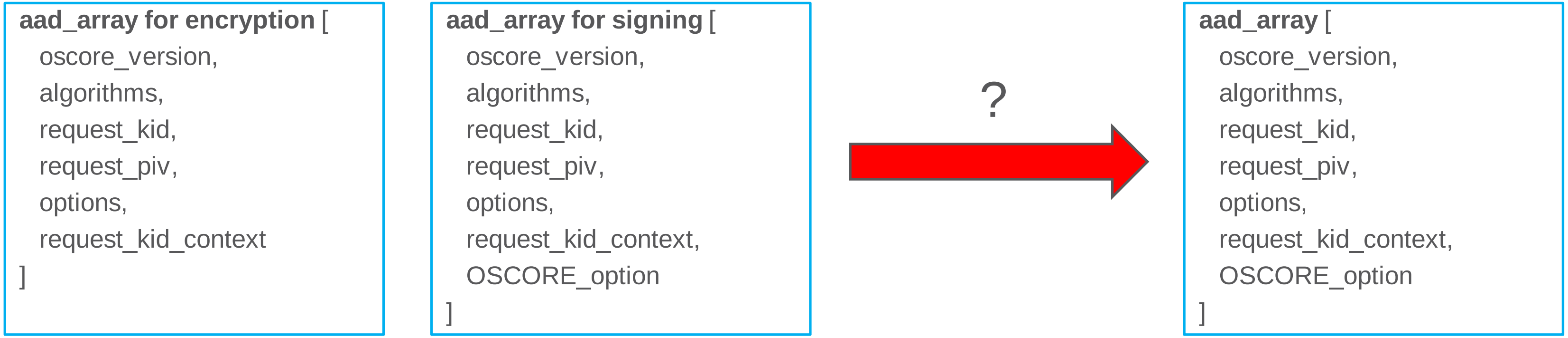
- More reasons to lose part of the Security Context
  - Reached the limit of Recipient Contexts, due to memory availability
  - Delete a current Recipient Context, to make room for a new one
  - Hereafter, each new Recipient Context starts with an invalid Replay Window

- Get rekeyed by the Group Manager or run Echo (achieving also freshness)

From Christian’s review

- Relax non-recycling of Sender IDs in the same group
  - Now: never-ever recycle → eventually leads to large KID sizes, with no way back
  - Proposal: never recycle under the same GID value. Issues with that?

- Converge to a single external_aad format ?
  - We have added ‘request_kid_context’
  - Now both external_aad structures deviate from RFC 8613 anyway



```
aad_array for encryption [
  oscore_version,
  algorithms,
  request_kid,
  request_piv,
  options,
  request_kid_context
]
```

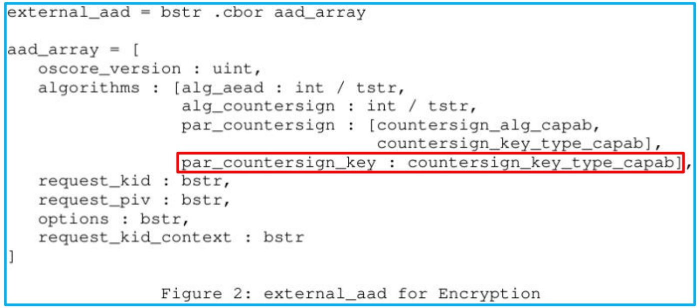
```
aad_array for signing [
  oscore_version,
  algorithms,
  request_kid,
  request_piv,
  options,
  request_kid_context,
  OSCORE_option
]
```

```
aad_array [
  oscore_version,
  algorithms,
  request_kid,
  request_piv,
  options,
  request_kid_context,
  OSCORE_option
]
```

From Christian’s review

- More on the external_aad

- Can we remove ‘par_countersign_key’ ?
  - It’s repeating what in ‘par_countersign’
  - Redundancy removed from the Common Context



```
external_aad = bstr .cbor aad_array

aad_array = [
   oscore_version : uint,
   algorithms : [alg_aead : int / tstr,
                 alg_countersign : int / tstr,
                 par_countersign : [countersign_alg_capab,
                                    countersign_key_type_capab],
                 par_countersign_key : countersign_key_type_capab],
   request_kid : bstr,
   request_piv : bstr,
   options : bstr,
   request_kid_context : bstr
]
```

Figure 2: external_aad for Encryption

- Can we further generalize ‘par_countersign’ ?
  - Today, algorithms have only “Key Type” as capability
  - COSE admits algorithms with 0 or 2+ capabilities
  - Possible future-friendly format

par_countersign [
  countersign_alg_capab [C1, C2, …, CN],
  countersign_C1_capab [C1, …],
  countersign_C2_capab [C2, …],
  …
  countersign_CN_capab [CN, …]
]

Summary and next steps

- Addressed comments from WGLC and IETF 108

- Successful tests at the Hackathon
  - Message exchange in group mode and pairwise mode

- Next steps
  - Submit version -11 addressing Christian’s review
  - More interop tests, covering also error cases

Thank you!

Comments/questions?

https://github.com/core-wg/oscore-groupcomm