OSCORE-capable Proxies

draft-tiloca-core-oscore-capable-proxies-01

Marco Tiloca, RISE
Rikard Höglund, RISE

IETF 112, CoRE WG, November 8th, 2021

Recap

- A CoAP proxy (P) can be used between client (C) and server (S)
  - A security association might be required between C and P --- examples in next slide
- Good to use OSCORE between C and P
  - Especially, but not only, if C and S already use OSCORE end-to-end
- This is not defined and not admitted in OSCORE (RFC 8613)
  - C and S are the only considered “OSCORE endpoints”
  - It is forbidden to double-protect a message, i.e., both over C ↔ S and over C ↔ P
- This started as an Appendix of *draft-tiloca-core-groupcomm-proxy*
  - Agreed at IETF 110 [1] and at the June CoRE interim [2] to have a separate draft

[1] https://datatracker.ietf.org/doc/minutes-110-core-202103081700/

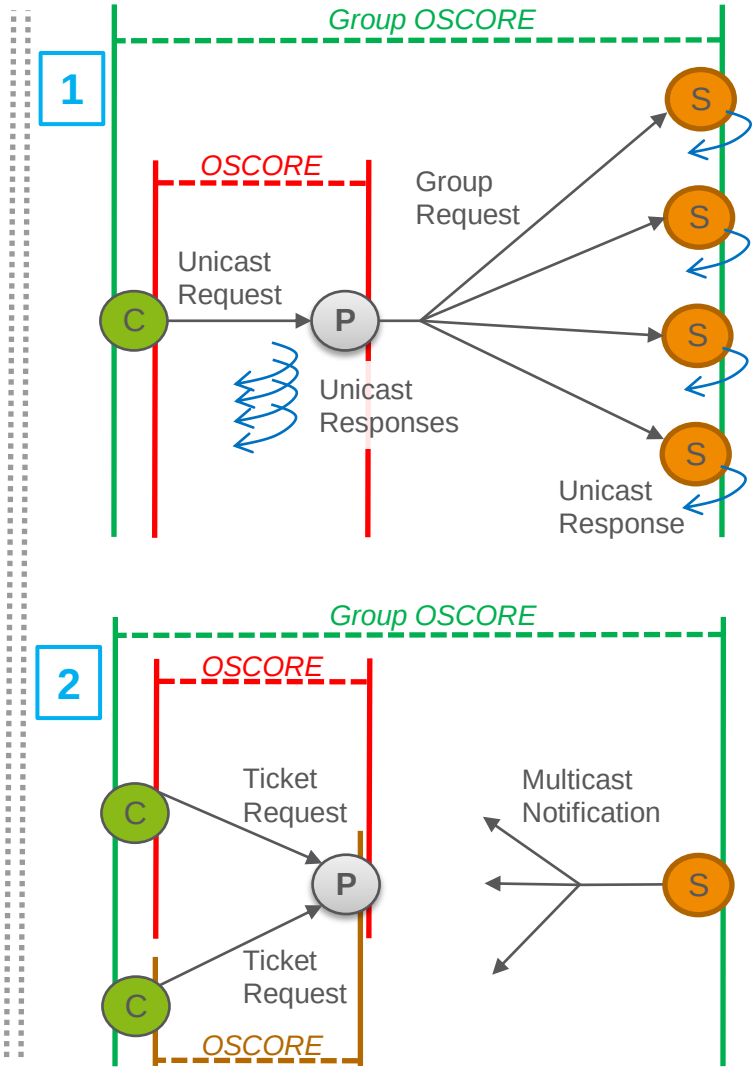
[2] https://datatracker.ietf.org/doc/minutes-interim-2021-core-07-202106091600/

Some use cases

1. **CoAP Group Communication with Proxies**
   - *draft-tiloca-core-groupcomm-proxy*
   - CoAP group communication through a proxy
   - P must identify C through a security association

2. **CoAP Observe Notifications over Multicast**
   - *draft-ietf-core-observe-multicast-notifications*
   - If Group OSCORE is used for e2e security …
   - … C provides P with a Ticket Request obtained from S
   - That provisioning should be protected over C ↔ P

3. **LwM2M Client and External Application Server**
   - The LwM2M Client may communicate with an External Application Server, also using OSCORE
   - The LwM2M Server would act as CoAP proxy, forwarding outside the LwM2M domain



Contribution

› Twofold update to RFC 8613

1. Define the use of OSCORE in a communication leg including a proxy
   - › Between origin client/server and a proxy; or between two proxies in a chain
   - › Not only an origin client/server, but also an intermediary can be an “OSCORE endpoint”

2. Explicitly admit nested OSCORE protection – “OSCORE-in-OSCORE”
   - – E.g., first protect end-to-end over C ↔ S, then further protect the result over C ↔ P
   - – Typically, at most 2 OSCORE “layers” for the same message
     - › 1 end-to-end + 1 between two adjacent hops
   - – Possible to seamlessly apply >2 OSCORE layers to the same message

› Focus on OSCORE, but the same applies “as is” to Group OSCORE

Updates since v -00

- Version -00 and planned updates presented at the September interim meeting [3]

- Latest version -01 addresses comments from Göran and Christian – Thanks!
  - Suggestions for more uses case to mention
  - Lift the limit of 2 OSCORE layers applied to the same message
  - Main feedback: the original presentation of message processing was too complicated

- Added more use cases, now in a new Section 2.4
  - Cross-proxy, as third party service to indicate transports available at the server [4][5]
  - Proxy as an entry point in a firewalled network, accessible only by authenticated clients
  - Privacy-oriented scenarios, with chain of proxies and >2 OSCORE layers per message

[3] https://datatracker.ietf.org/doc/minutes-interim-2021-core-10-202109151600/
[4] https://datatracker.ietf.org/doc/draft-amsuess-core-transport-indication/
[5] https://mailarchive.ietf.org/arch/msg/core/RZH8pgyksEwtMYVE1MrPkj9opyg/

Updates since v -00

- Revised presentation of message processing
  - Now much shorter and simpler
  - High-level general algorithm, fitting a client, proxy or server as a message processor
  - Now clearly said: no need for an explicit signaling method to guide the message processing

- Unlike RFC 8613, protect also these CoAP options when appling an OSCORE layer
  - An OSCORE Option, when present as the result of the immediately previous OSCORE layer
  - Options intended to the other OSCORE endpoint X, e.g., proxy related options when X is proxy

- Processing of an outgoing request
  - More options are protected (see above)
  - The origin client uses the Security Context shared with the origin server as first one

Updates since v -00

- Processing of an incoming request REQ, based on what it includes
  - **Case A** – Proxy-related options: included
    - Forward to the next hop, possibly adding a further OSCORE layer
  - **Case B** – Proxy-related options: not included; OSCORE option: not included
    - Deliver to the application, if any
  - **Case C** – Proxy-related options: not included; OSCORE option: included
    - Decrypt REQ using the Security Context retrieved through the OSCORE option
    - Repeat the (A/B/C) condition assessment over the decrypted request

  Error handling is also documented in the draft

Updates since v -00

- Processing of an outgoing response
  - More options are protected (see previous slide)
  - The origin server uses the Security Context shared with the origin client as first one
  - Apply the same OSCORE layers removed from the request
    - In the reverse order than the one they were removed
    - Only the successfully removed layers, if it is an error response

- Processing of an incoming response
  - Remove the same OSCORE layers added to the request
    - In the reverse order than the one they were added
  - The layers to remove are at most as many as the added ones

Summary and next steps

- Proposed update to RFC 8613
  - Define the use of OSCORE in a communication leg including a proxy
  - Explicitly admit nested OSCORE protection – “OSCORE-in-OSCORE”
- Main update in v -01
  - Message processing simplified and generalized to >2 OSCORE layers
  - Removed detailed breakdown and heavy notation → document much shorter and simpler
- Next steps
  - Add examples
  - Discuss caching of responses, building on *draft-amsuess-core-cachable-oscore*
  - Elaborate on applying >2 OSCORE layers to a same message
  - Look into CoAP header compression from RFC 8824. Use as is? Need for adaptations?
- More comments and input are welcome!

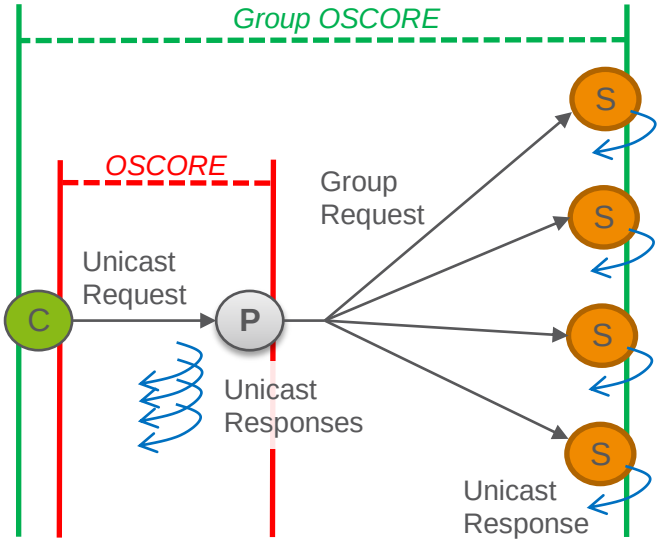
Thank you!

Comments/questions?

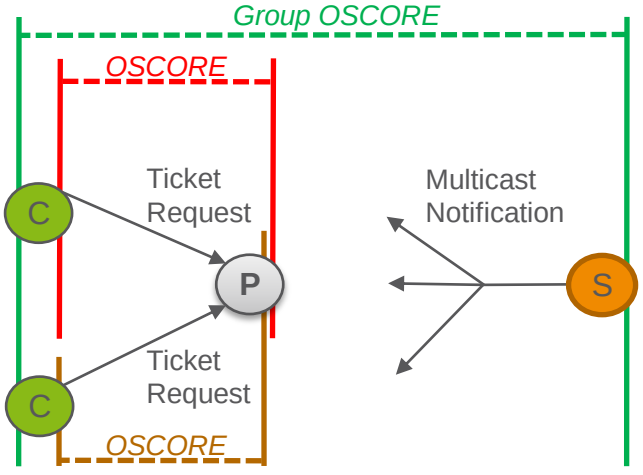
https://gitlab.com/crimson84/draft-tiloca-core-oscore-to-proxies

Some use cases

- **CoAP Group Communication with Proxies**
  - *draft-tiloca-core-groupcomm-proxy*
  - CoAP group communication through a proxy
  - Possible e2e security with Group OSCORE
  - P must identify C through a security association before forwarding a request to the group



- **CoAP Observe Notifications over Multicast, with Group OSCORE for e2e security**
  - *draft-ietf-core-observe-multicast-notifications*
  - C provides P with a Ticket Request obtained from S
  - This allows P to correctly listen to multicast notifications sent by S
  - The provisioning of the Ticket Request to P should be protected over C ↔ P



Some use cases

- **OMA LwM2M Client and External Application Server**
  - *Lightweight Machine to Machine Technical Specification – Transport Binding*

> *OSCORE MAY also be used between LwM2M endpoint and non-LwM2M endpoint, e.g., between an Application Server and a LwM2M Client via a LwM2M server. Both the LwM2M endpoint and non-LwM2M endpoint MUST implement OSCORE and be provisioned with an OSCORE Security Context.*

  - The LwM2M Client may register to and communicate with the LwM2M Server using OSCORE
  - The LwM2M Client may communicate with an External Application Server, also using OSCORE
  - The LwM2M Server would act as CoAP proxy, forwarding outside the LwM2M domain