Proxy Operations for CoAP Group Communication

draft-tiloca-core-groupcomm-proxy-01

Marco Tiloca, RISE
Esko Dijk, IoTconsultancy.nl

IETF 108, CoRE WG, July 31st, 2020

Recap

- CoAP supports group communication over IP multicast
  - *draft-ietf-core-groupcomm-bis*

- Issues when using proxies
  - Clients to be allow-listed and authenticated on the proxy
  - The client may receive multiple responses to a single *unicast* request
  - The client may not be able to distinguish responses and origin servers
  - The proxy does not know when to stop handling responses

- Possible approaches for proxy to handle the responses
  - Individually forwarded back to the client
  - Forwarded back to the client as a single aggregated response

Contribution

- Description of proxy operations for CoAP group communication
  - Addressed all issues in *draft-ietf-core-groupcomm-bis*
  - Signaling protocol with two new CoAP options
  - Responses individually forwarded back to the client

- The proxy is explicitly configured to support group communication
  - Clients are allowed-listed on the proxy, and identified by the proxy

- Version -01 addresses Christian's review [1] – Thanks!
  - Revised properties and usage of the two CoAP options
  - "Nested OSCORE" (Appendix A), if OSCORE is used between Client and Proxy

[1] https://mailarchive.ietf.org/arch/msg/core/AwYqnQu703V5RGR43JQxRslkYsw/

Rationale

- In the request addressed to the proxy, the client indicates:
  - To be interested in and capable of handling multiple responses
  - For how long the proxy should collect and forward back responses

- In a response to a group request, the proxy includes the server address
  - The client can distinguish the responses and the different servers
  - The client can contact an individual server (directly, or via the proxy)

- Group OSCORE for e2e security between client and servers

Multicast-Signaling option

No.	C	U	N	R	Name	Format	Length	Default
TBD1		x	-		Multicast-Signaling	uint	1-5 B	(none)

C=Critical, U=Unsafe, N=NoCacheKey, R=Repeatable

- Used only in requests
  - Presence: explicit claim of support and interest from the client
  - Value: indication to the proxy on how long to handle unicast responses
- The proxy removes the option, before forwarding the request

Response-Forwarding option

No.	C	U	N	R	Name	Format	Length	Default
TBD2			-		Response-Forwarding	string	8-20 B	(none)

C=Critical, U=Unsafe, N=NoCacheKey, R=Repeatable

- Used only in responses
  - Presence: allows the client to distinguish responses and originator servers
  - Value: absolute URI of the server (address and port from the response)
- The proxy adds the option, before forwarding the response to the client

Workflow: C -> P

- C prepares a request addressed to P
  - The group URI is included in the Proxi-Uri option or the URI-* options
- C chooses T seconds, as token retention time
  - T < Tr , with Tr = token reuse time
  - T considers processing at the proxy and involved RTTs
- C includes the Multicast-Signaling option, with value T′ < T
- C sends the request to P via unicast
  - C retains the token beyond the reception of a first matching response

Workflow: P -> S

- P identifies C and verifies it is allowed-listed
- P verifies the presence of the Multicast-Signaling option
  - P extracts the timeout value T′
  - P removes the Multicast-Signaling option
- P forwards the request to the group of servers, over IP multicast
- P will handle responses for the following T′ seconds
  - Observe notifications are an exception – they are handled until the Observe client state is cleared.

Workflow: S -> P

- S processes the request and sends the response to P

- P includes the Response-Forwarding option in the response
  - The option value is absolute URI of the server
  - IP address: source address of the response
  - Port number: source port number of the response

Workflow: P -> C

- P forwards responses back to C, individually as they come
- P frees-up its token towards the group of servers after T′ seconds
  - Later responses will not match and not be forwarded to C
  - Observe notifications are the exception
- C retrieves the Response-Forwarding option
  - C distinguishes different responses from different origin servers
  - C is able to later contact a server individually (directly or via the proxy)
- C frees-up its token towards the proxy after T seconds
  - Observe notifications are the exception

“Nested OSCORE”

- P has to authenticate C
  - A DTLS session would work
  - If Group OSCORE is used with the servers
    - P can check the counter signature in the group request
    - P needs to store the clients’ public keys used in the OSCORE group
    - P may be induced to forward replayed group requests to the servers

- Appendix A – OSCORE between C and P
  - If Group OSCORE is also used between C and the servers
    1. Protect the group request with Group OSCORE (C<->Servers context)
    2. Protect the result with OSCORE (C<->P context)
       - Some class U options are processed as class E options
    3. Reverse processing for responses

Summary

- Proxy operations for CoAP group communication
  - Embedded signaling protocol, using two new CoAP options
  - The proxy forwards individual responses to the client for a signaled time
  - The client can distinguish the origin servers and corresponding responses
- Next steps
  - Cover the case with a chain of proxies
  - Define HTTP headers for Cross-Proxies
- Need for reviews

Thank you!

Comments/questions?

https://gitlab.com/crimson84/draft-tiloca-core-groupcomm-proxy