Observe Notifications as CoAP Multicast Responses
draft-tiloca-core-observe-multicast-notifications-01

Abstract

The Constrained Application Protocol (CoAP) allows clients to "observe" resources at a server, and receive notifications as unicast responses upon changes of the resource state. In some use cases, such as based on publish-subscribe, it would be convenient for the server to send a single notification to all the clients observing a same target resource. This document defines how a CoAP server sends observe notifications as response messages over multicast, by synchronizing all the observers of a same resource on a same shared Token value. Besides, this document defines how Group OSCORE can be used to protect multicast notifications end-to-end from the CoAP server to the multiple observer clients.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on May 7, 2020.
1. Introduction

The Constrained Application Protocol (CoAP) [RFC7252] has been extended with a number of mechanisms, including resource Observation [RFC7641]. This enables CoAP clients to register at a CoAP server as "observers" of a resource, and hence being automatically notified with an unsolicited response upon changes of the resource state.

CoAP supports group communication over IP multicast [RFC7390], and [I-D.dijk-core-groupcomm-bis] has been enabling Observe registration requests over multicast, in order for clients to efficiently register as observers of a resource hosted at multiple servers.
However, in a number of use cases, using multicast messages for responses would also be desirable. That is, it would be useful that a server sends observe notifications for a same target resource to multiple observers as responses over IP multicast.

For instance, in CoAP publish-subscribe [I-D.ietf-core-coap-pubsub], multiple clients can subscribe to a topic, by observing the related resource hosted at the responsible broker. When a new value is published on that topic, it would be convenient for the broker to send a single multicast notification at once, to all the subscriber clients observing that topic.

A different use case concerns clients observing a same registration resource at the CoRE Resource Directory [I-D.ietf-core-resource-directory]. For example, multiple clients can benefit of observation for discovering (to-be-created) OSCORE groups [I-D.ietf-core-oscore-groupcomm], by retrieving from the Resource Directory updated links and descriptions to join them through the respective Group Manager [I-D.tiloca-core-oscore-discovery].

More in general, multicast notifications would be beneficial whenever several CoAP clients observe a same target resource at a CoAP server, and can be all notified at once by means of a single response message. However, CoAP does not currently define response messages over IP multicast. This specification fills this gap and provides the following twofold contribution.

First, it defines a method to deliver Observe notifications as CoAP responses over IP multicast. In the proposed method, the group of potential observers entrusts the server to manage the Token space for multicast notifications. By doing so, the server provides all the observers of a target resource with the same Token value to bind to their own observation. That Token value is then used in every multicast notification for that target resource. This is achieved by means of an informative unicast response sent by the server to each observer client.

Second, this specification defines how to use Group OSCORE [I-D.ietf-core-oscore-groupcomm] to protect multicast notifications end-to-end between the server and the observer clients. This is also achieved by means of the informative unicast response mentioned above, which additionally includes parameter values used by the server to secure every multicast notification for the target resource by using Group OSCORE. This provides a secure binding between each of such notifications and the observation of each of the clients.
1.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

Readers are expected to be familiar with terms and concepts described in CoAP [RFC7252], group communication for CoAP [RFC7390][I-D.dijk-core-groupcomm-bis], Observe [RFC7641], CBOR [RFC7049], OSCORE [RFC8613], and Group OSCORE [I-D.ietf-core-oscore-groupcomm].

This specification additionally defines the following terminology.

- Traditional observation. A resource observation associated to a single observer client, as defined in [RFC7641].

- Group observation. A resource observation associated to a group of clients. The server sends notifications for the group-observed resource over IP multicast to all the observer clients.

- Phantom request. The CoAP request message that the server would have received to generate a group observation on one of its resources. The phantom request is generated inside the server and does not hit the wire.

- Informative response. A CoAP response message that the server sends to a given client via unicast, providing the client with information on a group observation.

2. Server-Side Requirements

The server can, at any time, start a group observation on one of its resources. Practically, the server may want to do that under the following circumstances.

- In the absence of observations for the target resource, the server receives a registration request from a first client wishing to start a traditional observation on that resource.

- When a certain amount of traditional observations has been established on the target resource, the server decides to make those clients part of a group observation on that resource.
When it wants to start a group observation on one of its resources, and assuming it knows the multicast IP address to use to send multicast notifications to, the server proceeds as follows.

1. The server builds a phantom observation request, i.e. a GET request with an Observe option set to 0 (register).

2. The server selects a currently available value T, from the token space used for messages from the chosen multicast IP address to the server address intended for accessing the target resource. That token space is under exclusive control of the server.

3. The server processes the phantom observation request above, without transmitting it on the wire. The request is addressed to the resource for which the server wants to start the group observation, as if sent from the group of observers, i.e. with the multicast IP address as source address.

4. Upon processing the self-generated phantom request, the server interprets it as an observe registration received from the group of potential observer clients. In particular, from then on, the server MUST use T as its own local Token value associated to that observation, with respect to the (next hop towards) the clients.

5. The server does not immediately respond to the phantom observation request with a multicast notification. The server stores the phantom observation request as is, throughout the lifetime of the group observation.

After having started a group observation on a target resource, the server proceeds as follows.

- For each traditional observation ongoing on the target resource, the server MAY cancel that observation, and then adds the corresponding client to the list of observers taking part in the group observation. The server sends to each of such clients an informative response message, encoded as a unicast response with response code 5.03 (Service Unavailable). As per [RFC7641], such a response does not include an Observe option. The response MUST be Confirmable and MUST NOT encode link-local addresses. The payload of the response is a CBOR map including the following fields.

  * The IP multicast address where the server will hereafter send multicast notifications for the target resource, encoded as a CBOR byte string, with CBOR label "address".
* The byte serialization of the phantom observation request received by the server, encoded as a CBOR byte string, with CBOR label "registr".

* Optionally, the current representation of the target resource, encoded as a CBOR byte string, with CBOR label "res".

- Upon receiving a registration request to observe the target resource, the server does not create a corresponding individual observation for the requesting client. Instead, the server adds that client to the list of observers taking part in the group observation of the target resource, and replies to the client with the same informative response message defined above, which MUST be Confirmable and MUST include the current representation of the target resource. Note that this also applies when, with no ongoing traditional observations on the target resource, the server receives a registration request from a first client and decides to start a group observation on the target resource.

- Upon a change of the status of the target resource, the server sends a multicast notification, intended to all the clients taking part in the group observation of that resource. In particular, each of such multicast notifications:
  
  * MUST be sent to the IP multicast address indicated in the informative response message to the observer clients.
  
  * MUST include an Observe option, as per [RFC7641].
  
  * MUST have the same Token value T of the phantom registration request that started the group observation, also included in the informative response message to the observer clients. That is, every multicast notification for a target resource is not bound to the observation requests from the different clients, but rather to the phantom registration associated to the whole set of clients currently taking part in the group observation of that resource.

3. Client-Side Requirements

A client sends an observation request to the server as described in [RFC7641], i.e. a GET request with an Observe option set to 0 (register). The request MUST NOT encode link-local addresses. If the server is not configured to accept registrations on that target resource with a group observation, this would still result in a positive notification response to the client as described in [RFC7641].
Upon receiving the informative response defined in Section 2, the client proceeds as follows.

1. The client configures an observation of the target resource from a CoAP endpoint associated to the multicast IP address, and stores the phantom registration request specified in the informative response from the server. The group observation is bound to the phantom registration request. In particular, the client MUST use its Token value T as its own local Token value associated to that group observation, with respect to the (next hop towards the) server. The particular way to achieve this is implementation specific.

2. If a traditional observation to the target resource is ongoing, the client MAY silently cancel it without notifying the server.

3. If the informative response from the server includes the current representation of the target resource, the client considers it as received from an observe notification and processes it as usual.

From then on, the client will receive, accept and process multicast notifications about the state of the target resource from the server, as responses to the phantom registration request and with Token value T.

4. Example

The following example refers to two clients $C_1$ and $C_2$ that register to observe a resource $/r$ at a Server $S$. Before the following exchanges occur, no clients are observing the resource $/r$, which has value “1234”.

The server $S$ sends multicast notifications to the IP multicast address $M_{addr}$, and starts the group observation upon receiving a registration request from a first client that wishes to start a traditional observation on the resource $/r$.

```
C_1 ---------------- [ Unicast ] ----------------------> S /r
    | GET
    | Token: 0x4a
    | Observe: 0 (Register)
    |
    (S allocates the available Token value 0xff.)
```
(S sends to itself a phantom observation request Ph_req as coming from the IP multicast address M_addr.)

(S creates a group observation of /r.)

(S adds C_1 to the list of observers taking part in the group observation of /r.)

(C_1 <--------------------- [ Unicast ] ----------------- S)

5.03
Token: 0x4a
Payload: { address: M_addr ,
           registr: Ph_req ,
           res: "1234" }

(C_2 ------------------ [ Unicast ] --------------------> S)

GET
Token: 0x01
Observe: 0 (Register)

(S adds C_2 to the list of observers taking part in the group observation of /r.)

(C_2 <--------------------- [ Unicast ] ----------------- S)

5.03
Token: 0x01
Payload: { address: M_addr ,
           registr: Ph_req ,
           res: "1234" }

(The value of the resource /r changes to "5678".)

C_1 + <--------------------- [ Multicast ] ----------------- S

(C_2 (Destination address: M_addr))

2.05
Token: 0xff
Observe: 11
Payload: "5678"
5. Protection of Multicast Notifications with Group OSCORE

A server can protect multicast notifications by using Group OSCORE [I-D.ietf-core-oscore-groupcomm]. In such a case, both the server and the clients interested in receiving multicast notifications from that server have to be members of the same OSCORE group.

Clients MAY discover the OSCORE group to refer to by using the method in [I-D.tiloca-core-oscore-discovery], also based on the CoRE Resource Directory (RD) [I-D.ietf-core-resource-directory]. Alternatively, the server MAY communicate to the client what OSCORE group to join, as described in Section 5.1. Furthermore, both the clients and the server MAY join the OSCORE group by using the approach described in [I-D.ietf-ace-key-groupcomm-oscore] and based on the ACE framework for Authentication and Authorization in constrained environments [I-D.ietf-ace-oauth-authz]. Further details on how to discover the OSCORE group and join it are out of the scope of this specification.

Alternative security protocols than Group OSCORE, such as OSCORE [RFC8613] and/or DTLS [RFC6347] [I-D.ietf-tls-dtls13], can be used to protect other exchanges via unicast between the server and each client, including the original client registration (see Section 3).

5.1. OSCORE Group in Informative Response

This section describes a mechanism for the server to communicate to the client what OSCORE group to join in order to decrypt and verify the multicast notifications protected with group OSCORE. The client MAY use the information provided by the server to start the ACE joining procedure described in [I-D.ietf-ace-key-groupcomm-oscore]. This mechanism is OPTIONALLY supported by client and server.

Additionally to what defined in Section 2, the CBOR map in the informative response payload contains the following fields:

- The URI for joining the OSCORE group at the respective Group Manager, encoded as a CBOR text string, with CBOR label "join-uri". If the procedure described in [I-D.ietf-ace-key-groupcomm-oscore] is used for joining, this field specifically indicates the URI of the group-membership resource at the Group Manager.

- The name of the OSCORE group, encoded as a CBOR text string, with CBOR label "sec-gp".
- Optionally, the URI of the Authorization Server associated to the Group Manager for the OSCORE group, encoded as a CBOR text string, with CBOR label "as-uri".

- Optionally, the algorithm used to countersign messages, encoded as a text string or an int, with value taken from the 'Value' column of the "COSE Algorithms" registry defined in [RFC8152], with CBOR label "cs-alg".

- Optionally, the elliptic curve for the algorithm used to countersign messages, encoded as a text string or an int, with value taken from the 'Value' column of the "COSE Elliptic Curve" registry defined in [RFC8152], with CBOR label "cs-crv".

- Optionally, the key type of countersignature keys used to countersign messages, encoded as a text string or an int, with value taken from the 'Value' column of the "COSE Key Types" registry defined in [RFC8152], with CBOR label "cs-kty".

- Optionally, the encoding of the public keys, encoded as an int, with value taken from the 'Confirmation Key' column of the "CWT Confirmation Method" registry defined in [I-D.ietf-ace-cwt-proof-of-possession], with CBOR label "cs-kenc". Future specifications may define additional values for this parameter.

- Optionally, the AEAD algorithm, encoded as a text string or an int, with value taken from the 'Value' column of the "COSE Algorithms" registry defined in [RFC8152], with CBOR label "alg".

- Optionally, the HKDF algorithm, encoded as a text string or an int, with value taken from the 'Value' column of the "COSE Algorithms" registry defined in [RFC8152], with CBOR label "hkdf".

The values of 'cs_alg', 'cs_crv', 'cs_kty' and 'cs_kenc' provide an early knowledge of the format and encoding of public keys used in the OSCORE group. Thus, the client does not need to ask the Group Manager for this information as a preliminary step before the (ACE) join process, or to perform a trial-and-error exchange with the Group Manager upon joining the group. Hence, the client is able to provide the Group Manager with its own public key in the correct expected format and encoding, at the very first step of the (ACE) join process.

The values of 'cs_alg', 'alg' and 'hkdf' provide an early knowledge of the algorithms used in the OSCORE group. Thus, the client is able to decide whether to actually proceed with the (ACE) join process, depending on its support for the indicated algorithms.
As mentioned above, since this mechanism is OPTIONAL, all the fields are OPTIONAL in the informative response. However, the "join-uri" and "sec-gp" fields MUST be present if the mechanism is implemented and run. If any of the fields are present without the "join-uri" and "sec-gp" fields present, the client MUST ignore these fields, since they would not be sufficient to start the (ACE) join procedure. When this happens, the client MAY try sending a new registration request to the server (see Section 3). Otherwise, the client SHOULD explicitly withdraw from the group observation, as defined in Section 3.

5.2. Server-Side Requirements

When using Group OSCORE to protect multicast notifications, the server performs as described in Section 2, with the following differences.

- The phantom registration request MUST be secured, by using Group OSCORE. To this end, the server protects the phantom registration request as if it was the actual sender, i.e. by using its own Sender Context. As a consequence, the server consumes the current value of its own Sender Sequence Number SN in the OSCORE group, and hence updates it to $SN^* = (SN + 1)$. Consistently, the OSCORE option in the phantom registration request includes:
  - As 'kid', the Sender ID of the server in the OSCORE group.
  - As 'piv', the previously consumed sender sequence number value $SN$ of the server in the OSCORE group, i.e. $(SN^* - 1)$.

- Upon sending every multicast notification for the target resource, the server protects it with Group OSCORE. In particular, the process described in Section 6.3 of [I-D.ietf-core-oscore-groupcomm] applies, with the following additions when building the two OSCORE ‘external_aad’ to encrypt and countersign the multicast notification (see Sections 3.1 and 3.2 of [I-D.ietf-core-oscore-groupcomm]).
  - The ‘request_kid’ is the ‘kid’ value in the OSCORE option of the phantom registration request, i.e. the Sender ID of the server.
  - The ‘request_piv’ is the ‘piv’ value in the OSCORE option of the phantom registration request, i.e. the consumed sender sequence number SN of the server.
5.3. Client-Side Requirements

When using Group OSCORE to protect multicast notifications, the client performs as described in Section 3, with the following differences.

- Upon receiving the informative response from the server, the client retrieves the specified phantom registration request, and extracts the ‘kid’ and ‘piv’ fields of its OSCORE option.

- From then on, when verifying multicast notifications for that target resource as described in Section 6.4 of [I-D.ietf-core-oscore-groupcomm], the client MUST use the extracted ‘kid’ as ‘request_kid’ and the extracted ‘piv’ as ‘request_piv’, in the two ‘external_aad’ for decrypting and verifying every multicast notification from the server for the target resource (see Sections 3.1 and 3.2 of [I-D.ietf-core-oscore-groupcomm]). The particular way to achieve this is implementation specific.

6. Example with Group OSCORE

The following example refers to two clients C_1 and C_2 that register to observe a resource /r at a Server S. Before the following exchanges occur, no clients are observing the resource /r, which has value "1234".

The server S sends multicast notifications to the IP multicast address M_addr, and starts the group observation upon receiving a registration request from a first client that wishes to start a traditional observation on the resource /r.

Pairwise communication over unicast are protected with OSCORE, while S protects multicast notifications with Group OSCORE. Specifically:

- C_1 and S have a pairwise OSCORE Security Context. In particular, C_1 has ‘kid’ = 1 as Sender ID, and SN_1 = 101 as Sequence Number. Also, S has ‘kid’ = 3 as Sender ID, and SN_3 = 301 as Sequence Number.

- C_2 and S have a pairwise OSCORE Security Context. In particular, C_2 has ‘kid’ = 2 as Sender ID, and SN_2 = 201 as Sequence Number. Also, S has ‘kid’ = 4 as Sender ID, and SN_4 = 401 as Sequence Number.

- S is a member of the OSCORE group with name "myGroup", and ‘kid_context’ = "feedca57ab2e" as Group ID. In the OSCORE group, S has ‘kid’ = 5 as Sender ID, and SN_5 = 501 as Sequence Number.
C_1     ------------ [ Unicast w/ OSCORE ] -------------> S   /r
GET
Token: 0x4a
Observe: 0 (Register)
OSCORE: {kid: 1 ; piv: 101 ; ...}

(S allocates the available Token value 0xff.)

(S sends to itself a phantom observation request Ph_req as coming from the IP multicast address M_addr.)

(S steps SN_5 in the Group OSCORE Sec. Ctx : SN_5 <= 502)

(S creates a group observation of /r.)

(S adds C_1 to the list of observers taking part in the group observation of /r.)

C_1 <---------------- [ Unicast w/ OSCORE ] -------------     S
5.03
Token: 0x4a
OSCORE: {piv: 301; ...}
Payload: { address  : M_addr ,
           registr  : Ph_req ,
           res  : "1234" ,
           join-uri : coap://myGM/group-oscore/myGroup
           sec-gp : "myGroup" }

C_2     ------------ [ Unicast w/ OSCORE ] --------------> S   /r
GET
Token: 0x01
Observe: 0 (Register)
OSCORE: {kid: 2 ; piv: 201 ; ...}

(S adds C_2 to the list of observers taking part in the group observation of /r.)

C_2 <---------------- [ Unicast w/ OSCORE ] --------------     S
5.03
Token: 0x01
OSCORE: {piv: 401; ...}
Payload: { address : M_addr,
          registr : Ph_req,
          res : "1234",
          join-uri : coap://myGM/group-oscore/myGroup
          sec-gp : "myGroup" }

(The value of the resource /r changes to "5678".)

C_1
+ <------------ [ Multicast w/ Group OSCORE ] ---------- S
C_2
  2.05
  Token: 0xff
  Observe: 11
  OSCORE: {kid: 5; piv: 502 ; ...}
  Payload: "5678"

The two external_aad used to encrypt and countersign the multicast notification above have ‘req_kid’ = 5 and ‘req_iv’ = 501. These are indicated in the ‘kid’ and ‘iv’ field of the OSCORE option of the phantom observation request, which is included in the ‘registr’ parameter of the unicast informative response to the two clients. Thus, the two clients can build the two same external_aad for decrypting and verifying this multicast notification and the following ones.

7. Security Considerations

The same security considerations from [RFC7252][RFC7390][RFC7641][I-D.dijk-core-groupcomm-bis][RFC8613][I-D.ietf-core-oscore-groupcomm] hold for this document.

If multicast notifications are protected using Group OSCORE, the original registration requests and related unicast (notification) responses MUST also be secured, including and especially the informative responses from the server. This prevents on-path active adversaries from altering the conveyed IP multicast address and serialized phantom request. Thus, it ensures secure binding between every multicast notification for a same observed resource and the phantom request that started the group observation of that resource.
To this end, clients and servers SHOULD use OSCORE or Group OSCORE, so ensuring that the secure binding above is enforced end-to-end between the server and each observing client.

8. IANA Considerations

This document has the following actions for IANA.

TODO: registry for labels of the map in the informative response.

9. References

9.1. Normative References

[I-D.dijk-core-groupcomm-bis]

[I-D.ietf-core-oscore-groupcomm]


9.2. Informative References

[I-D.ietf-ace-cwt-proof-of-possession]

[I-D.ietf-ace-key-groupcomm-oscore]

[I-D.ietf-ace-oauth-authz]

[I-D.ietf-core-coap-pubsub]

[I-D.ietf-core-resource-directory]

[I-D.ietf-tls-dtls13]

[I-D.tiloca-core-oscore-discovery]


Acknowledgments

The authors sincerely thank Carsten Bormann, Klaus Hartke, John Mattsson, Ludwig Seitz, Jim Schaad and Goeran Selander for their comments and feedback.

The work on this document has been partly supported by VINNOVA and the Celtic-Next project CRITISEC.

Authors’ Addresses

Marco Tiloca
RISE AB
Isafjordsgatan 22
Kista SE-16440 Stockholm
Sweden
Email: marco.tiloca@ri.se

Rikard Hoeglund
RISE AB
Isafjordsgatan 22
Kista SE-16440 Stockholm
Sweden
Email: rikard.hoglund@ri.se

Christian Amsuess
Hollandstr. 12/4
Vienna 1020
Austria
Email: christian@amsuess.com
Francesca Palombini
Ericsson AB
Torshamngatan 23
Kista SE-16440 Stockholm
Sweden

Email: francesca.palombini@ericsson.com