CoRE Working Group
Internet-Draft
Updates: 7252, 7641 (if approved)
Intended status: Standards Track
Expires: September 10, 2020

M. Tiloca
R. Hoeglund
RISE AB
C. Amsuess

F. Palombini
Ericsson AB

March 09, 2020

Observe Notifications as CoAP Multicast Responses
draft-tiloca-core-observe-multicast-notifications-02

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.

Table of Contents

1. Introduction 3
1.1. Terminology 4
2. Server-Side Requirements 5
2.1. Request 5
2.2. Informative Response 6
2.3. Notifications 8
2.4. Congestion Control 8
2.5. Cancellation 9
2.5.1. Rough Counting of Clients in the Group Observation . 9
3. Client-Side Requirements 12
3.1. Request 12
3.2. Informative Response 12
3.3. Notifications 13
3.4. Cancellation 13
4. Example 13
5. Protection of Multicast Notifications with Group OSCORE . . . 15
5.1. Signaling the OSCORE Group in the Informative Response . 16
5.2. Server-Side Requirements 17
5.2.1. Registration 18
5.2.2. Informative Response 18
5.2.3. Notifications 18
5.2.4. Cancellation 19
5.3. Client-Side Requirements 19
5.3.1. Informative Response 19
5.3.2. Notifications 20
6. Example with Group OSCORE 20
7. Informative Response Parameters 23
8. Phantom Request Parameters 24
9. Security Considerations 25
10. IANA Considerations 25
10.1. Media Type Registrations 25

10.2. CoAP Content-Formats Registry 26
10.3. Informative Response Parameters Registry 27
10.4. Phantom Request Parameters Registry 27
10.5. CoAP Option Numbers Registry 28
10.6. Expert Review Instructions 28
11. References 29
11.1. Normative References 29
11.2. Informative References 30
11.3. URIs 31
Appendix A. Different Sources for Phantom Requests 32
A.1. PubSub 32
A.2. Sender Introspection 33
Acknowledgments 33
Authors' Addresses 33

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 [I-D.dijk-core-groupcomm-bis]. This includes support for 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 the 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 protect 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 [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.

The server maintains an observer counter for each group observation to a target resource, as a rough estimation of the observers actively taking part in the group observation. The server increments the counter when a new client starts taking part in that group observation. Also, the server should update the counter over time, for instance by using the method described in Section 2.5.1.

2.1. Request

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.

2.2. Informative Response

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. Then, the server considers the N corresponding clients as now taking part in the group observation, of which it increases the corresponding observer counter by N.

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 Content-Format of the informative response is set to application/informative-response+cbor, as defined in Section 10.2. The payload of the informative response is a CBOR map including the following parameters, whose CBOR labels are defined in Section 7.

- 'ph_req', with value the phantom observation request received by the server, encoded as a CBOR map including the following fields, whose CBOR labels are defined in Section 8.

  - 'src_addr', with value the source IP address of the phantom observation request, encoded as a CBOR byte string. This parameter is tagged and identified by the CBOR tag 260 "Network Address (IPv4 or IPv6 or MAC Address)". The specified address is the IP multicast address where the server will send multicast notifications for the target resource.

  * 'src_port', with value the source port number of the phantom observation request, encoded as a CBOR unsigned integer.

  * 'dst_addr', with value the destination IP address of the phantom observation request, encoded as a CBOR byte string. This parameter is tagged and identified by the CBOR tag 260 "Network Address (IPv4 or IPv6 or MAC Address)". The specified address is the IP address of the server hosting the target resource.

  * 'dst_port', with value the destination port number of the phantom observation request, encoded as a CBOR unsigned integer. This is the port number the server hosting the target resource has been listening to.

  * 'coap_msg', with value the byte serialization of the CoAP message sent as phantom observation request, encoded as a CBOR byte string. Specifically, the value of the byte string is the byte serialization of what becomes payload for the transport layer underlying CoAP, such as UDP.

- 'notif_num', specifying a baseline Observe value for the group observation of the target resource, encoded as a CBOR unsigned integer. This parameter specifies either: i) the value of the Observe option in the latest sent multicast notification; or ii) X, where X + 1 will be used as value of the Observe option in the first, yet-to-come multicast notification.

- Optionally, 'res', with value the byte serialization of the current representation of the target resource, encoded as a CBOR byte string.

- Optionally, 'res_ct', with value the format of the current representation of the target resource specified in the 'res' parameter, encoded as a CBOR unsigned integer. This parameter has as value the numeric Content-Format identifier for the representation format of the target resource, taken from the "CoAP Content-Formats" Registry defined in Section 12.3 of [RFC7252]. This parameter MUST be present if the 'res' parameter is present.

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 considers that client as now taking part in the group observation of the target resource, of which it increments the observer counter by 1. Then, the server replies to the client with the same informative response message defined above, which MUST be Confirmable and MUST include also the 'res' and 'res_ct' parameters.

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.

2.3. Notifications

Upon a change of the status of the target resource under group observation, 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 to the observer clients, as value of the 'src_addr' field within the 'ph_req' parameter of the informative response message (see Section 2.2).

- MUST be Non-confirmable.

- 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, as 'coap_msg' field of the 'ph_req' parameter. 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 request associated to the whole set of clients taking part in the group observation of that resource.

2.4. Congestion Control

In order to not cause congestion, the server should conservatively control the sending of multicast notifications. In particular:

- The multicast notifications MUST be Non-confirmable.

- In constrained environments such as low-power, lossy networks (LLNs), the server should only support multicast notifications for resources that are small. Following related guidelines from Section 2.2.4 of [I-D.dijk-core-groupcomm-bis], this can consist, for example, in having the payload of multicast notifications as limited to approximately 5% of the IP Maximum Transmit Unit (MTU) size, so that it fits into a single link-layer frame in case IPv6 over Low-Power Wireless Personal Area Networks (6LoWPAN) (see Section 4 of [RFC4944]) is used.

- The server SHOULD provide multicast notifications with the smallest possible IP multicast scope that fulfills the application

needs. For example, following related guidelines from Section 2.2.4 of [I-D.dijk-core-groupcomm-bis], site-local scope is always preferred over global scope IP multicast, if this fulfills the application needs. Similarly, realm-local scope is always preferred over site-local scope, if this fulfills the application needs.

- Following related guidelines from Section 4.5.1 of [RFC7641],the server SHOULD NOT send more than one multicast notification every 3 seconds, and SHOULD use an even less aggressive rate when possible (see also Section 3.1.2 of [RFC5405]).

2.5. Cancellation

At any point in time, the server may want to cancel a group observation of a target resource. For instance, the server may realize that no clients or not enough clients are interested in taking part in the group observation anymore. A possible approach that the server can use to assess this is defined in Section 2.5.1.

In order to cancel the group observation, the server sends to itself a phantom cancellation request, i.e. a GET request with an Observe option set to 1 (deregister), without transmitting it on the wire. As per Section 3.6 of [RFC7641], all other options MUST be identical to those in the phantom registration request, except for the set of ETag Options. This request has the same Token value T of the phantom registration request, and is addressed to the resource for which the server wants to end the group observation, as if sent from the group of observers, i.e. with the multicast IP address as source address.

After that, the server sends a multicast response with response code 5.03 (Service Unavailable), signaling that the group observation has been terminated. The response has no payload, and is sent to the same multicast IP address used to send the multicast notifications related to the target resource. As per [RFC7641], this response does not include an Observe option. Finally, the server releases the resources allocated for the group observation, and especially frees up the Token value T used at its endpoint.

2.5.1. Rough Counting of Clients in the Group Observation

To allow the server to keep an estimate of interested clients without creating undue traffic on the network, a new CoAP option is introduced, which SHOULD be supported by clients that listen to multicast responses.

The option is called Multicast-Response-Feedback-Divider, and is only used in responses. As summarized in Figure 1, the option is not critical but proxy-unsafe, and integer valued.

No.	C	U	N	R	Name	Format	Len.	Default
TBD		x			Multicast-Response-Feedback-Divider	uint	0-8 B	(none)

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

Figure 1: Multicast-Response-Feedback-Divider

The Multicast-Response-Feedback-Divider option is of class E for OSCORE [RFC8613][I-D.ietf-core-oscore-groupcomm].

2.5.1.1. Client Processing

Upon receiving a response with a Multicast-Response-Feedback-Divider option, a client SHOULD acknowledge its interest in continuing receiving multicast notifications for the target resource.

To do that, the client picks an integer random number 'c', from 0 inclusive to the number 'q' given in the option exclusive. If 'c' is different than 0, the client takes no further action. Otherwise, the client should wait a random fraction of the Leisure time (see Section 8.2 of [RFC7252]), and then registers a regular unicast observation on the same target resource. To this end, the client essentially follows the steps that got it originally subscribed to group notifications for the target resource.

As the observation registration is only done for its side effect of showing as an attempted observation at the server, the client SHOULD send the unicast request in a non confirmable way, and with the maximum No-Response setting [RFC7967]. The client does not need to wait for responses, and can keep processing further notifications on the same token.

As the Multicast-Response-Feedback-Divider option is unsafe to forward, a proxy needs to answer it on its own, and is later counted as a single client.

2.5.1.2. Client Counting

In order to avoid needless use of network resources, a server SHOULD keep a rough count of the number of clients taking part in the group observation of a target resource. To this end, the server updates the associated observer counter (see Section 2), for instance by using the method described below.

When it wants to obtain a new estimated count, the server picks a number 'm' of confirmations it would like to receive from the clients. It is up to applications to define policies about how the server determines and adjusts the value of 'm'. The following example will be done with m = 5.

Then, the server considers its current estimate of listeners 'n', and divides it by 'm'. The resulting quotient q = ceil(n / m) is set as value in the Multicast-Response-Feedback-Divider option, which is sent within a successful multicast notification. If several multicast notifications are sent in a burst fashion, it is RECOMMENDED for the server to include the Multicast-Response-Feedback-Divider option only in the first one of those notifications.

Later on, the server counts the 'r' attempted unicast observations arriving after the notification, and multiplies that with the last Multicast-Response-Feedback-Divider value 'q', to get an updated client estimate 'n'.

This estimate is skewed by new registrations and by packet loss, but it gives the server a sufficiently good estimation for further counts and for deciding when to cancel the group observation. It is up to applications to define policies about how the server takes the updated value of 'n' into account and determines whether to cancel the group observation.

For example, if the server currently estimates that n = 20 observers are active, it sends a notification out with Multicast-Response-Feedback-Divider: 4. Then, out of 18 actually active clients, 5 send a re-registration request based on their random draw, of which one request gets lost, thus leaving four re-registration requests received by the server. As a consequence, the server updates the observer counter to n = 4 * 4 = 16, and continues sending notifications to the group of observers.

Note that a server can send Multicast-Response-Feedback-Divider: 1 in the last notifications, before cancelling a group observation. This will trigger all the active clients to state their interest in continuing receiving notifications for the target resource.

3. Client-Side Requirements

3.1. Request

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].

3.2. Informative Response

Upon receiving the informative response defined in Section 2.2, the client proceeds as follows.

1. The client configures an observation of the target resource from a CoAP endpoint associated to the IP multicast address, specified by the 'src_addr' field within the 'ph_req' parameter of the informative response.

2. The client retrieves and stores the phantom registration request specified in the 'coap_msg' field within the 'ph_req' parameter of the informative response. The group observation is bound to this 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.

3. The client retrieves and stores the value of the 'notif_num' parameter of the informative response.

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

5. If the informative response from the server includes the parameter 'res' with value the current representation of the target resource, the client considers it as received from an observe notification and processes it as usual.

If any of the expected fields are not present, the client MAY try sending a new registration request to the server (see Section 3.1). Otherwise, the client SHOULD explicitly withdraw from the group observation.

3.3. Notifications

After having successfully processed the informative response as defined in Section 3.2, 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.

The client relies on the value of the Observe option for notification reordering, as defined in Section 3.4 of [RFC7641]. In particular, upon receiving its first multicast notification for the target resource, the client MUST treat it as fresh only if the value of the Observe option is strictly greater than the stored value of the 'notif_num' parameter from the informative response (see Section 2.2).

3.4. Cancellation

At a certain point in time, a client may become not interested in receiving further multicast notifications about a target resource. When this happens, the client can simply "forget" about being part of the group observation for that target resource, as per Section 3.6 of [RFC7641].

When, later on, the server sends the next multicast notification, the client will not recognize the Token value T in the message. Since the multicast notification is Non-confirmable, it is OPTIONAL for the client to reject the multicast notification with a Reset message, as defined in Section 3.5 of [RFC7641].

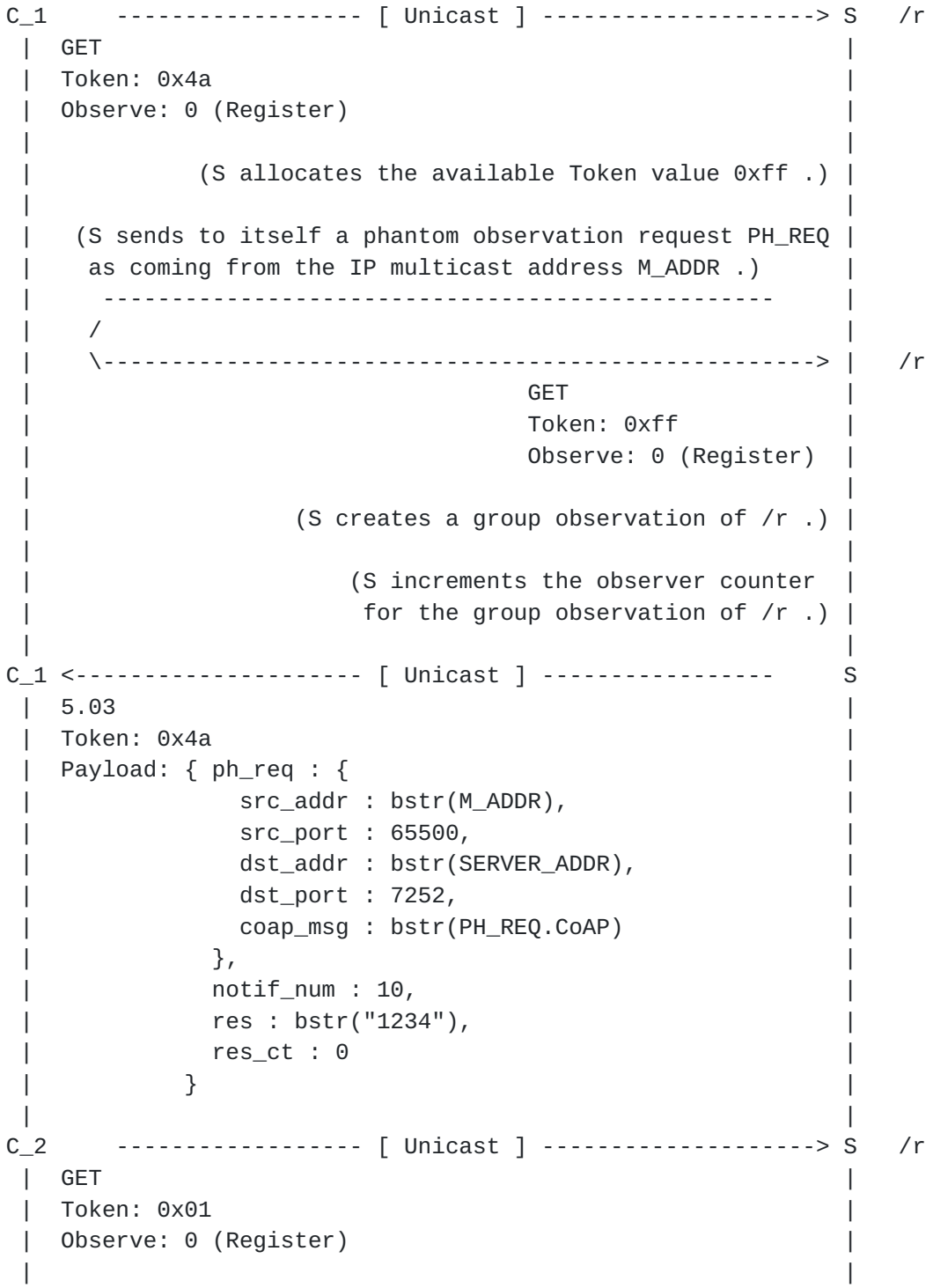
In case the server has cancelled a group observation as defined in Section 2.5, the client simply forgets about the group observation and frees up the used Token value T for that endpoint, upon receiving the multicast error response defined in Section 2.5.

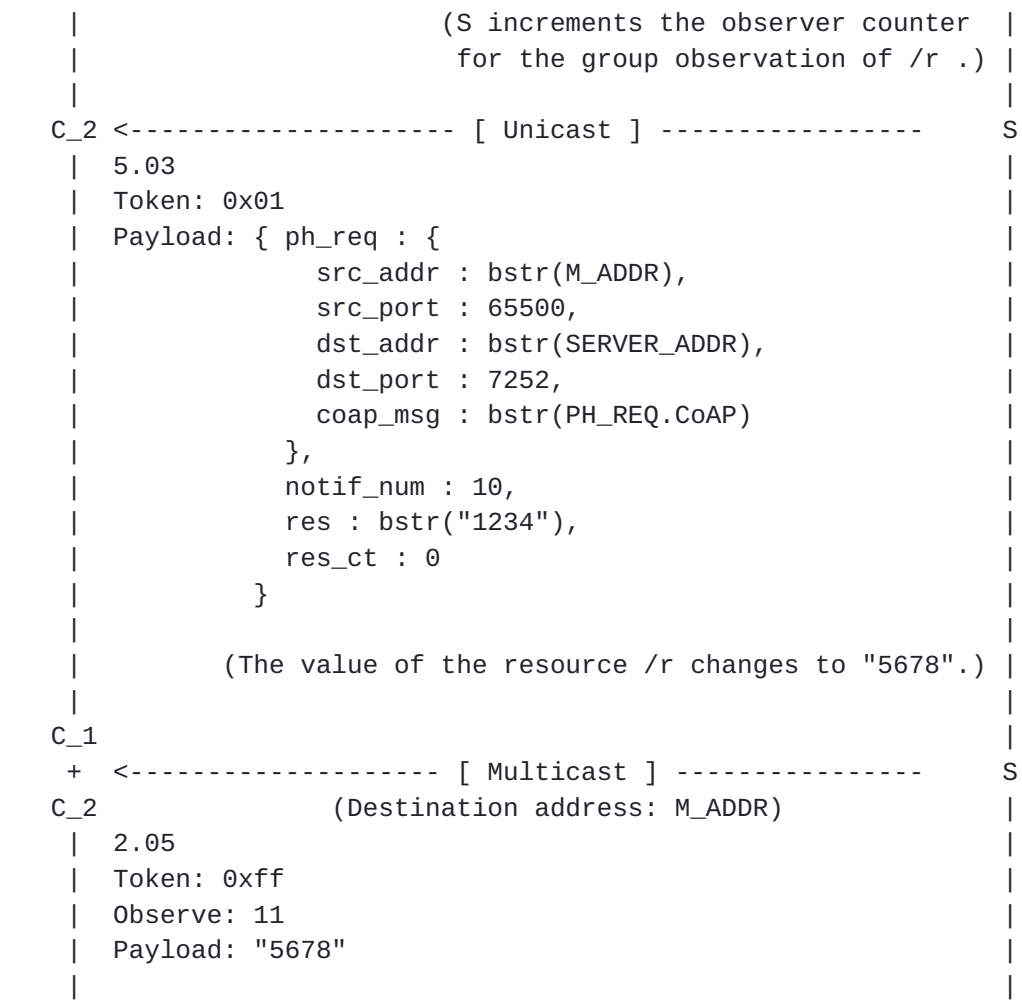
4. Example

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

In the informative responses, 'bstr(X)' denotes a byte string with value the byte serialization of X. Also, the notation Y.CoAP denotes the CoAP-layer part of a message Y, i.e. the part of Y that becomes payload for the transport layer underlying CoAP, such as UDP.

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.





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], 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. Signaling the OSCORE Group in the 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 OPTIONAL to support for the client and server.

Additionally to what defined in Section 2, the CBOR map in the informative response payload contains the following fields, whose CBOR labels are defined in Section 7.

- 'join_uri', with value the URI for joining the OSCORE group at the respective Group Manager, encoded as a CBOR text string. 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.

- 'sec_gp', with value the name of the OSCORE group, encoded as a CBOR text string.

- Optionally, 'as_uri', with value the URI of the Authorization Server associated to the Group Manager for the OSCORE group, encoded as a CBOR text string.

- Optionally, 'cs_alg', with value the algorithm used to countersign messages, encoded as a CBOR text string or integer. The value is taken from the 'Value' column of the "COSE Algorithms" registry defined in [RFC8152].

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

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

- Optionally, 'cs_kenc', with value the encoding of the public keys, encoded as a CBOR integer. The value is taken from the

'Confirmation Key' column of the "CWT Confirmation Method" registry defined in [I-D.ietf-ace-cwt-proof-of-possession]. Future specifications may define additional values for this parameter.

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

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

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 used. 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 3.1). Otherwise, the client SHOULD explicitly withdraw from the group observation.

5.2. Server-Side Requirements

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

5.2.1. Registration

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).

5.2.2. Informative Response

The 'notif_num' parameter of the informative response defined in Section 2 is set as follows.

- If only one multicast notification has been sent for the target resource and it included a Partial IV in its OSCORE option, the 'notif_num' parameter takes the value of that Partial IV.

- If more than one multicast notification have been sent for the target resource, the 'notif_num' parameter takes the value of the Partial IV in the OSCORE option of the latest sent multicast notification.

- In any other case, the 'notif_num' parameter takes the current Sender Sequence Number of the server in the OSCORE group, from its own Sender Context.

5.2.3. Notifications

Upon sending every multicast notification for the target resource, the server protects it with Group OSCORE. In particular, the process described in Section 7.3 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 4.3.1 and 4.3.2 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.

Note that these same values are used to protect each and every multicast notification sent for the target resource.

5.2.4. Cancellation

When cancelling a group observation (see Section 2.5), the phantom cancellation request MUST be secured, by using Group OSCORE.

Like defined in Section 5.2.1 for the phantom registration request, the server protects the phantom cancellation request by using its own Sender Context and consuming its own current Sender Sequence number in the OSCORE group, from its own Sender Context. The following, corresponding multicast error response defined in Section 2.5 is also protected with Group OSCORE, as per Section 7.3 of [I-D.ietf-core-oscore-groupcomm].

Note that, differently from the multicast notifications, this multicast error response will be the only one securely paired with the phantom cancellation request.

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.

5.3.1. Informative Response

Upon receiving the informative response from the server, the client retrieves the phantom registration request specified in the 'coap_msg' field of the 'ph_req' parameter.

Then, the client decrypts and verifies the phantom registration request as defined in Section 7.2 of [I-D.ietf-core-oscore-groupcomm], with the following differences.

- The client MUST NOT perform any replay check. That is, the client skips step 3 in Section 8.2 of [RFC8613].

- If decryption and verification of the phantom registration request succeed:

  * The client MUST NOT update the Replay Window in the Recipient Context associated to the server. That is, the client skips the second bullet of step 6 in Section 8.2 of [RFC8613].

  * The client MUST NOT take any further process as normally expected according to [RFC7252]. That is, the client skips

step 8 in Section 8.2 of [RFC8613]. In particular, the client MUST NOT deliver the phantom registration request to the application, and MUST NOT take any action in the Token space of its own unicast endpoint, where the informative response has been received.

* The client stores the values of the 'kid' and 'piv' fields from the OSCORE option of the phantom registration request.

5.3.2. Notifications

After having successfully processed the informative response as defined in Section 5.3.1, the client will decrypt and verify every multicast notification for the target resource as defined in Section 7.4 of [I-D.ietf-core-oscore-groupcomm], with the following difference.

The client MUST set the two 'external_aad' defined in Sections 4.3.1 and 4.3.2 of [I-D.ietf-core-oscore-groupcomm] as follows. The particular way to achieve this is implementation specific.

- 'request_kid' takes the value of the 'kid' field from the OSCORE option of the phantom registration request (see Section 5.3.1).

- 'request_piv' takes the value of the 'piv' field from the OSCORE option of the phantom registration request (see Section 5.3.1).

Note that these same values are used to decrypt and verify each and every multicast notification received for the target resource.

The replay protection and checking of multicast notifications is performed as specified in Section 4.1.3.5.2 of [RFC8613], with the following addition. When the client receives its first multicast notification for the target resource and a Partial IV is included in the OSCORE option, the client MUST treat the notification as fresh only if the value of that Partial IV is strictly greater than the stored value of the 'notif_num' parameter from the informative response (see Section 2.2).

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 with address SERVER_ADDR. Before the following exchanges occur, no clients are observing the resource /r , which has value "1234".

In the informative responses, 'bstr(X)' denotes a byte string with value the byte serialization of X. Also, the notation Y.CoAP denotes

the CoAP-layer part of a message Y, i.e. the part of Y that becomes payload for the transport layer underlying CoAP, such as UDP.

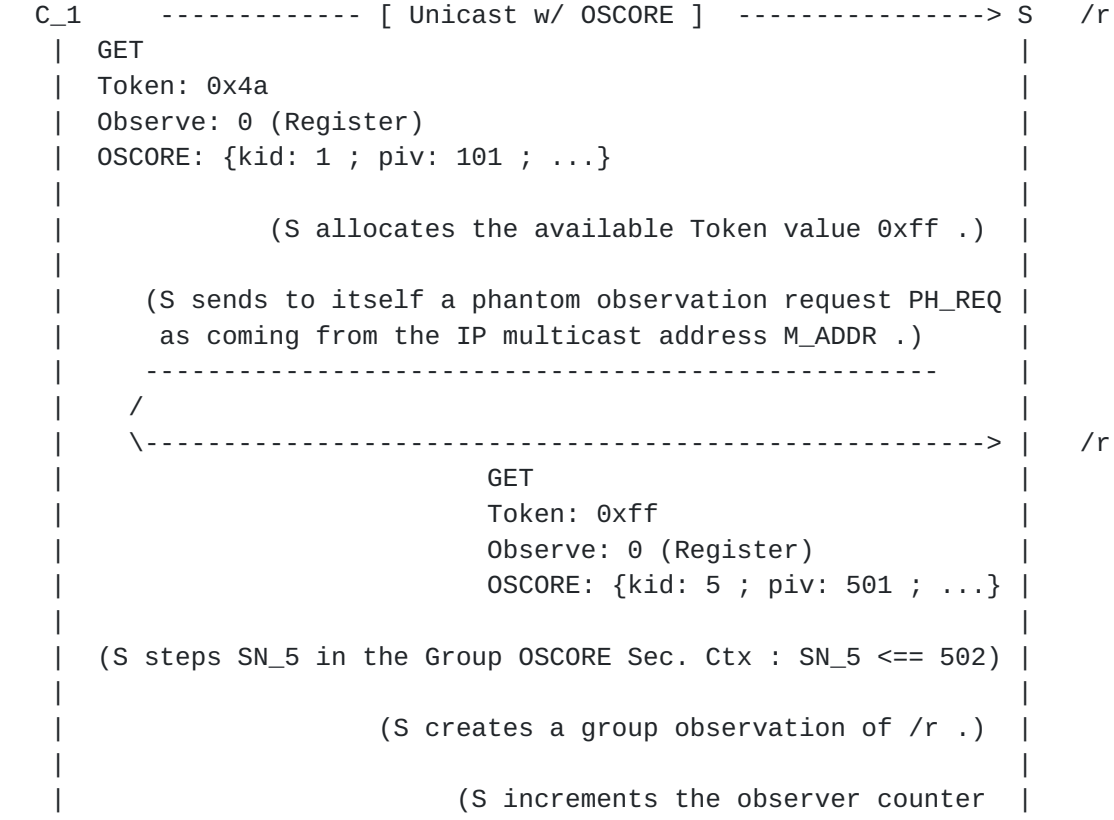
The server S sends multicast notifications to the IP multicast address M_ADDR , and starts the group observation upon receiving a registration request from a first client that wishes to start a traditional observation on the resource /r.

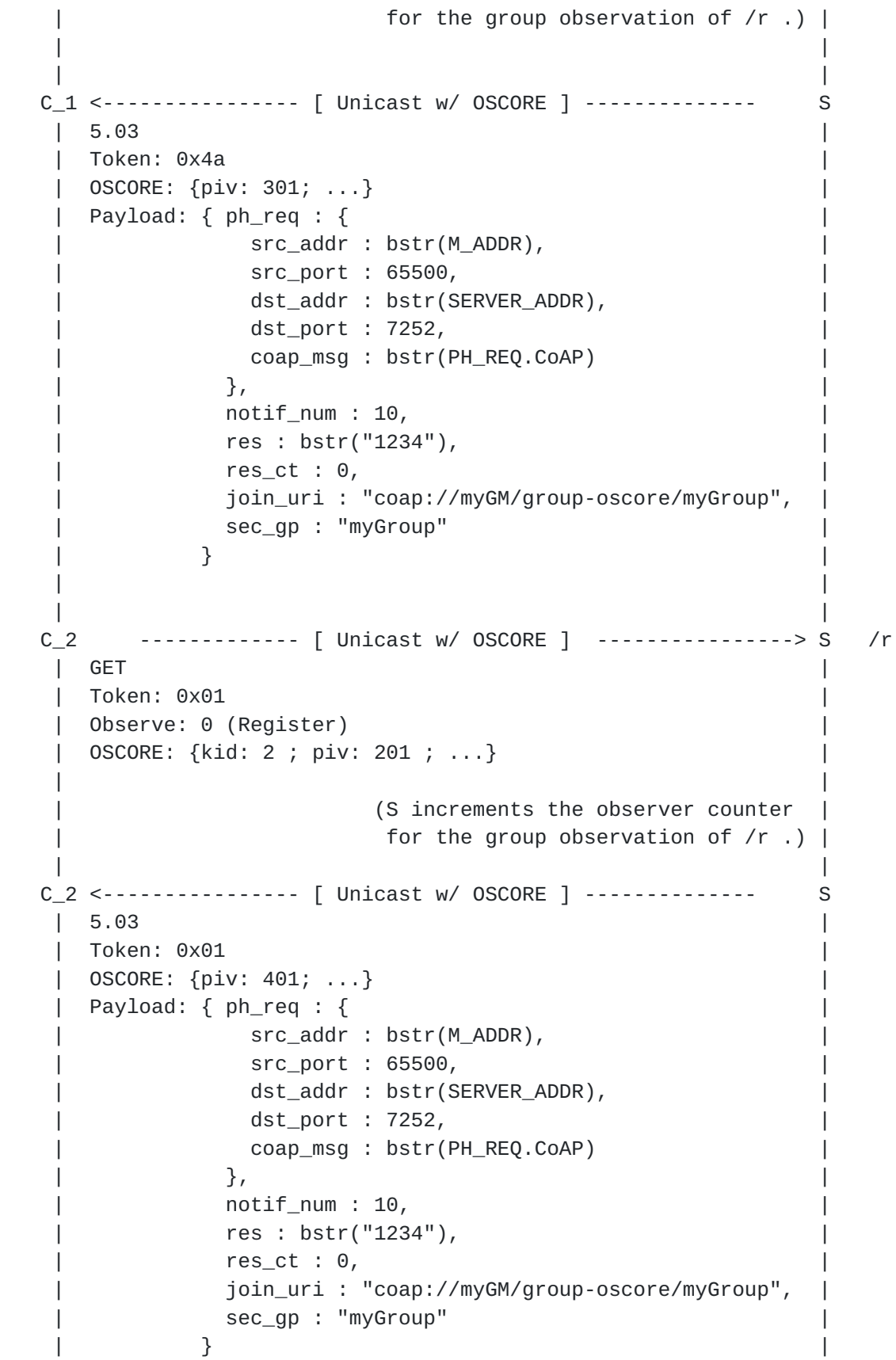
Pairwise communication over unicast are protected with OSCORE, while S protects multicast notifications with Group OSCORE. Specifically:

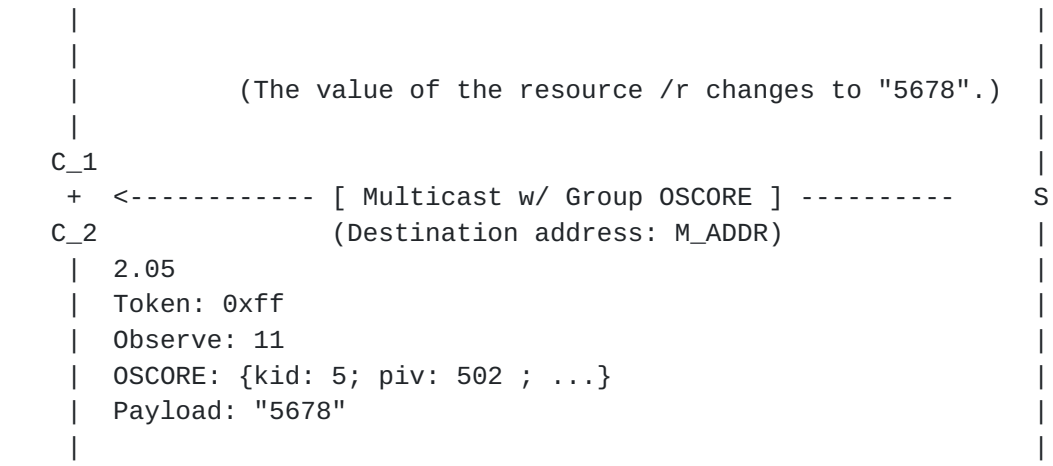
- C_1 and S have a pairwise OSCORE Security Context. In particular, C_1 has 'kid' = 1 as Sender ID, and SN_1 = 101 as Sequence Number. Also, S has 'kid' = 3 as Sender ID, and SN_3 = 301 as Sequence Number.

- 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.




for the group observation of /r .)
C_1 <---------- [Unicast w/ OSCORE] ---------- S
5.03
Token: 0x4a
OSCORE: {piv: 301; ...}
Payload: { ph_req : {
src_addr : bstr(M_ADDR),
src_port : 65500,
dst_addr : bstr(SERVER_ADDR),
dst_port : 7252,
coap_msg : bstr(PH_REQ.CoAP)
},
notif_num : 10,
res : bstr("1234"),
res_ct : 0,
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 increments the observer counter
for the group observation of /r .)
C_2 <---------- [Unicast w/ OSCORE] ---------- S
5.03
Token: 0x01
OSCORE: {piv: 401; ...}
Payload: { ph_req : {
src_addr : bstr(M_ADDR),
src_port : 65500,
dst_addr : bstr(SERVER_ADDR),
dst_port : 7252,
coap_msg : bstr(PH_REQ.CoAP)
},
notif_num : 10,
res : bstr("1234"),
res_ct : 0,
join_uri : "coap://myGM/group-oscore/myGroup",
sec_gp : "myGroup"
}



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 'ph_req' 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. Informative Response Parameters

This specification defines a number of fields used in error messages as informative response defined in Section 2.2 of this specification.

The table below summarizes them, and specifies the CBOR key to use instead of the full descriptive name. Note that the media type application/informative-response+cbor MUST be used when these fields are transported.

Name	CBOR Key	CBOR Type	Reference
ph_req	TBD	map	Section 2.2
notif_num	TBD	unsigned integer	Section 2.2
res	TBD	byte string	Section 2.2
res_ct	TBD	unsigned integer	Section 2.2
join_uri	TBD	text string	Section 5.1
sec_gp	TBD	text string	Section 5.1
as_uri	TBD	text string	Section 5.1
cs_alg	TBD	int / text string	Section 5.1
cs_crv	TBD	int / text string	Section 5.1
cs_kty	TBD	int / text string	Section 5.1
cs_kenc	TBD	int	Section 5.1
alg	TBD	int / text string	Section 5.1
hkdf	TBD	int / text string	Section 5.1

8. Phantom Request Parameters

This specification defines a number of fields for the CBOR map 'ph_req', specifying a phantom request within error messages as informative response defined in Section 2.2 of this specification.

The table below summarizes them, and specifies the CBOR key to use instead of the full descriptive name. Note that the media type application/informative-response+cbor MUST be used when these fields are transported.

Name	CBOR Key	CBOR Type	Reference
src_addr	TBD	byte string	Section 2.2
src_port	TBD	unsigned integer	Section 2.2
dst_addr	TBD	byte string	Section 2.2
dst_port	TBD	unsigned integer	Section 2.2
coap_msg	TBD	byte string	Section 2.2

9. Security Considerations

The same security considerations from [RFC7252][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.

10. IANA Considerations

This document has the following actions for IANA.

10.1. Media Type Registrations

This specification registers the media type 'application/informative-response+cbor' for error messages as informative response defined in Section 2.2 of this specification, when carrying parameters encoded in CBOR. This registration follows the procedures specified in [RFC6838].

- Type name: application

- Subtype name: informative-response+cbor

- Required parameters: none
- Optional parameters: none
- Encoding considerations: Must be encoded as a CBOR map containing the parameters defined in Section 2.2 of [this document].
- Security considerations: See Section 9 of [this document].
- Interoperability considerations: n/a
- Published specification: [this document]
- Applications that use this media type: The type is used by CoAP servers and clients that support error messages as informative response defined in Section 2.2 of [this document].
- Additional information:
  - Magic number(s): n/a
  - File extension(s): .informative-response
  - Macintosh file type code(s): n/a
- Person & email address to contact for further information: iesg@ietf.org [1]
- Intended usage: COMMON
- Restrictions on usage: None
- Author: Marco Tiloca marco.tiloca@ri.se [2]
- Change controller: IESG
- Provisional registration? (standards tree only): No

10.2. CoAP Content-Formats Registry

IANA is asked to add the following entry to the "CoAP Content-Formats" subregistry defined in Section 12.3 of [RFC7252], within the "Constrained RESTful Environments (CoRE) Parameters" registry.

Media Type: application/informative-response+cbor

Encoding: -

ID: TBD

Reference: [this document]

10.3. Informative Response Parameters Registry

This specification establishes the "Informative Response Parameters" IANA Registry. The Registry has been created to use the "Expert Review Required" registration procedure [RFC8126]. Expert review guidelines are provided in Section 10.6.

The columns of this Registry are:

- Name: This is a descriptive name that enables easier reference to the item. The name MUST be unique. It is not used in the encoding.

- CBOR Key: This is the value used as CBOR key of the item. These values MUST be unique. The value can be a positive integer, a negative integer, or a string.

- CBOR Type: This contains the CBOR type of the item, or a pointer to the registry that defines its type, when that depends on another item.

- Reference: This contains a pointer to the public specification for the item.

This Registry has been initially populated by the values in Section 7. The "Reference" column for all of these entries refers to sections of this document.

10.4. Phantom Request Parameters Registry

This specification establishes the "Phantom Request Parameters" IANA Registry. The Registry has been created to use the "Expert Review Required" registration procedure [RFC8126]. Expert review guidelines are provided in Section 10.6.

The columns of this Registry are:

- Name: This is a descriptive name that enables easier reference to the item. The name MUST be unique. It is not used in the encoding.

- CBOR Key: This is the value used as CBOR key of the item. These values MUST be unique. The value can be a positive integer, a negative integer, or a string.

- CBOR Type: This contains the CBOR type of the item, or a pointer to the registry that defines its type, when that depends on another item.

- Reference: This contains a pointer to the public specification for the item.

This Registry has been initially populated by the values in Section 8. The "Reference" column for all of these entries refers to sections of this document.

10.5. CoAP Option Numbers Registry

IANA is asked to enter the following option numbers to the "CoAP Option Numbers" registry defined in [RFC7252] within the "CoRE Parameters" registry.

Number	Name	Reference
TBD	Multicast-Response-Feedback-Divider	[[this document]]

10.6. Expert Review Instructions

The IANA Registries established in this document are defined as expert review. This section gives some general guidelines for what the experts should be looking for, but they are being designated as experts for a reason so they should be given substantial latitude.

Expert reviewers should take into consideration the following points:

- Point squatting should be discouraged. Reviewers are encouraged to get sufficient information for registration requests to ensure that the usage is not going to duplicate one that is already registered and that the point is likely to be used in deployments. The zones tagged as private use are intended for testing purposes and closed environments, code points in other ranges should not be assigned for testing.

- Specifications are required for the standards track range of point assignment. Specifications should exist for specification required ranges, but early assignment before a specification is available is considered to be permissible. Specifications are needed for the first-come, first-serve range if they are expected to be used outside of closed environments in an interoperable way. When specifications are not provided, the description provided

needs to have sufficient information to identify what the point is being used for.

- Experts should take into account the expected usage of fields when approving point assignment. The fact that there is a range for standards track documents does not mean that a standards track document cannot have points assigned outside of that range. The length of the encoded value should be weighed against how many code points of that length are left, the size of device it will be used on, and the number of code points left that encode to that size.

11. References

11.1. Normative References

[I-D.dijk-core-groupcomm-bis]
Dijk, E., Wang, C., and M. Tiloca, "Group Communication for the Constrained Application Protocol (CoAP)", draft-dijk-core-groupcomm-bis-03 (work in progress), March 2020.

[I-D.ietf-core-oscore-groupcomm]
Tiloca, M., Selander, G., Palombini, F., and J. Park, "Group OSCORE - Secure Group Communication for CoAP", draft-ietf-core-oscore-groupcomm-07 (work in progress), November 2020.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC4944] Montenegro, G., Kushalnagar, N., Hui, J., and D. Culler, "Transmission of IPv6 Packets over IEEE 802.15.4 Networks", RFC 4944, DOI 10.17487/RFC4944, September 2007, <https://www.rfc-editor.org/info/rfc4944>.

[RFC5405] Eggert, L. and G. Fairhurst, "Unicast UDP Usage Guidelines for Application Designers", RFC 5405, DOI 10.17487/RFC5405, November 2008, <https://www.rfc-editor.org/info/rfc5405>.

[RFC6838] Freed, N., Klensin, J., and T. Hansen, "Media Type Specifications and Registration Procedures", BCP 13, RFC 6838, DOI 10.17487/RFC6838, January 2013, <https://www.rfc-editor.org/info/rfc6838>.

[RFC7049] Bormann, C. and P. Hoffman, "Concise Binary Object Representation (CBOR)", RFC 7049, DOI 10.17487/RFC7049, October 2013, <https://www.rfc-editor.org/info/rfc7049>.

[RFC7252] Shelby, Z., Hartke, K., and C. Bormann, "The Constrained Application Protocol (CoAP)", RFC 7252, DOI 10.17487/RFC7252, June 2014, <https://www.rfc-editor.org/info/rfc7252>.

[RFC7641] Hartke, K., "Observing Resources in the Constrained Application Protocol (CoAP)", RFC 7641, DOI 10.17487/RFC7641, September 2015, <https://www.rfc-editor.org/info/rfc7641>.

[RFC7967] Bhattacharyya, A., Bandyopadhyay, S., Pal, A., and T. Bose, "Constrained Application Protocol (CoAP) Option for No Server Response", RFC 7967, DOI 10.17487/RFC7967, August 2016, <https://www.rfc-editor.org/info/rfc7967>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/info/rfc8126>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8613] Selander, G., Mattsson, J., Palombini, F., and L. Seitz, "Object Security for Constrained RESTful Environments (OSCORE)", RFC 8613, DOI 10.17487/RFC8613, July 2019, <https://www.rfc-editor.org/info/rfc8613>.

11.2. Informative References

[I-D.ietf-ace-cwt-proof-of-possession]
Jones, M., Seitz, L., Selander, G., Erdtman, S., and H. Tschofenig, "Proof-of-Possession Key Semantics for CBOR Web Tokens (CWTs)", draft-ietf-ace-cwt-proof-of-possession-11 (work in progress), October 2019.

[I-D.ietf-ace-key-groupcomm-oscore]
Tiloca, M., Park, J., and F. Palombini, "Key Management for OSCORE Groups in ACE", draft-ietf-ace-key-groupcomm-oscore-05 (work in progress), March 2020.

[I-D.ietf-ace-oauth-authz]
Seitz, L., Selander, G., Wahlstroem, E., Erdtman, S., and H. Tschofenig, "Authentication and Authorization for Constrained Environments (ACE) using the OAuth 2.0 Framework (ACE-OAuth)", draft-ietf-ace-oauth-authz-33 (work in progress), February 2020.

[I-D.ietf-core-coap-pubsub]
Koster, M., Keranen, A., and J. Jimenez, "Publish-Subscribe Broker for the Constrained Application Protocol (CoAP)", draft-ietf-core-coap-pubsub-09 (work in progress), September 2019.

[I-D.ietf-core-resource-directory]
Shelby, Z., Koster, M., Bormann, C., Stok, P., and C. Amsuess, "CoRE Resource Directory", draft-ietf-core-resource-directory-23 (work in progress), July 2019.

[I-D.ietf-tls-dtls13]
Rescorla, E., Tschofenig, H., and N. Modadugu, "The Datagram Transport Layer Security (DTLS) Protocol Version 1.3", draft-ietf-tls-dtls13-37 (work in progress), March 2020.

[I-D.tiloca-core-oscore-discovery]
Tiloca, M., Amsuess, C., and P. Stok, "Discovery of OSCORE Groups with the CoRE Resource Directory", draft-tiloca-core-oscore-discovery-05 (work in progress), March 2020.

[RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, DOI 10.17487/RFC6347, January 2012, <https://www.rfc-editor.org/info/rfc6347>.

[RFC8152] Schaad, J., "CBOR Object Signing and Encryption (COSE)", RFC 8152, DOI 10.17487/RFC8152, July 2017, <https://www.rfc-editor.org/info/rfc8152>.

11.3. URIs

[1] mailto:iesg@ietf.org

[2] mailto:marco.tiloca@ri.se

Appendix A. Different Sources for Phantom Requests

While the clients usually receive the phantom request and related information through an Informative Response, the same data can be made available through different services, such as the following ones.

A.1. PubSub

In a pubsub case ([I-D.ietf-core-coap-pubsub]), a phantom request can be discovered, along with topic metadata. For instance, a discovery step can make available the following metadata.

This examples assumes a CoRAL namespace that contains properties analogous to those in the content-format application/informative-response+cbor.

Request:

```
GET </ps/topics?rt=oic.r.temperature>
Accept: CoRAL
```

Response:

```
2.05 Content
Content-Format: CoRAL

rdf:type <http://example.org/pubsub/topic-list>
topic    </ps/topics/1234> {
    dst_addr h"ff35003020010db8..1234"
    src_port 5683
    dst_addr h"20010db80100..0001"
    dst_port 5683
    coap_msg h"120100006464b431323334"
}
```

With this information from the topic discovery step, the client can already set up its multicast address and start receiving multicast notifications.

In heavily asymmetric networks like municipal notification services, discovery and notifications do not necessarily need to use the same network link. For example, a departure monitor could use its (costly and usually-off) cellular uplink to discover the topics it needs to update its display to, and then listen on a LoRA-WAN interface for receiving the actual multicast notifications.

A.2. Sender Introspection

For network debugging purposes, it can be useful to query a server that sends multicast messages for phantom requests.

Such an interface is left for other documents to specify on demand, but could look like this:

Request:

```
GET </.well-known/core/mc-sender?token=6464>
```

Response:

```
2.05 Content
Content-Format: application/informative-response+cbor

{'ph_req': {
    'dst_addr': h"ff35003020010db8..1234"
    'src_port': 5683
    'dst_addr': h"20010db80100..0001"
    'dst_port': 5683
    'coap_msg': h"120100006464b431323334"
}}
```

For example, a network sniffer could offer sending such a request when unknown multicast notifications are seen on a network. Consequently, it can associate those notifications with a URI, or decrypt them, if member of the correct OSCORE group.

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
Torshamnsgatan 23
Kista SE-16440 Stockholm
Sweden

Email: francesca.palombini@ericsson.com