CoRE Working Group
Internet-Draft
Updates: 7390, 7641 (if approved)
Intended status: Standards Track
Expires: September 11, 2019

E. Dijk
IoTconsultancy.nl
C. Wang
InterDigital
M. Tiloca
RISE AB

March 10, 2019

Group Communication for the Constrained Application Protocol (CoAP) draft-dijk-core-groupcomm-bis-00

Abstract

This document specifies the use of the Constrained Application Protocol (CoAP) for group communication, using UDP/IP multicast as the underlying data transport. Both unsecured and secured CoAP group communication are specified. Security is achieved by the Group Object Security for Constrained RESTful Environments (Group OSCORE) protocol. The target application area of this specification is any group communication use cases that involve resource-constrained network nodes. The most common of such use cases are listed in this document.

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 September 11, 2019.

Copyright Notice

Table of Contents

1. Introduction 3
1.1. Scope 3
1.2. Terminology 4
2. Use Cases 4
2.1. Discovery 4
2.1.1. Distributed Device Discovery 4
2.1.2. Distributed Service Discovery 5
2.1.3. Directory Discovery 5
2.2. Operational 5
2.2.1. Actuator Group Control 6
2.2.2. Device Group Status Request 6
2.2.3. Network-wide Query 6
2.3. Software Update 6
3. General Group Communication Operation 7
3.1. Group Configuration 7
3.1.1. Group Definition 7
3.1.2. Group Naming (DNS) 7
3.1.3. Group Creation and Membership 8
3.1.4. Group Maintenance 8
3.2. CoAP Usage 9
3.2.1. Request/Response Model 9
3.2.2. Port and URI Path Selection 10
3.2.3. Proxy Operation 11
3.2.4. Congestion Control 11
3.2.5. Observing Resources 11
3.2.6. Block-Wise Transfer 11
3.3. Transport 12
3.3.1. UDP/IPv6 Multicast Transport 12
3.3.2. UDP/IPv4 Multicast Transport 12
3.3.3. 6LoWPAN 12
3.4. Interworking with Other Protocols 12
3.4.1. MLD/MLDv2/IGMP 12
3.4.2. RPL 12
3.4.3. MPL 12
4. Unsecured Group Communication 12
5. Secured Group Communication using Group OSCORE 13
5.1. Secure Group Maintenance 14
6. Security Considerations 15

6.1. CoAP NoSec Mode 15
6.2. Group OSCORE 15
6.3. 6LoWPAN 16
6.4. Wi-Fi 16
6.5. Monitoring 17
7. IANA Considerations 17
8. References 17
8.1. Normative References 17
8.2. Informative References 18
Acknowledgments 19
Authors' Addresses 19

1. Introduction

This document specifies the use of the Constrained Application Protocol (CoAP) [RFC7252] for group communication [RFC7390]. CoAP is a RESTful communication protocol that is suited for usage in resource-constrained nodes, and in resource-constrained networks. This area of use is summarized as Constrained RESTful Environments (CoRE).

One-to-many group communication is achieved in CoAP by using UDP/IP multicast, as the underlying data transport to send multicast request messages. Multiple response messages to a single multicast request message are sent over UDP/IP unicast. Notable CoAP implementations supporting group communication include the framework "Eclipse Californium" 2.0.x [Californium] from the Eclipse Foundation and the "Implementation of CoAP Server & Client in Go" [Go-OCF] from the Open Connectivity Foundation (OCF).

The most common use cases for group communication in resource-constrained networks are listed first, in Section 2. Both unsecured and secured CoAP group communication are specified in this document.

Security is achieved by using Group Object Security for Constrained RESTful Environments (Group OSCORE) [I-D.ietf-core-oscore-groupcomm], which in turn builds on Object Security for Constrained Restful Environments (OSCORE) [I-D.ietf-core-object-security]. This method provides end-to-end application-layer security protection of CoAP messages, by using CBOR Object Signing and Encryption (COSE) [RFC8152] [RFC7049].

1.1. Scope

The guidelines and experimental protocol of [RFC7390] are updated by this document with the abovementioned security solution, and with other recent protocol developments around CoAP such as Observe [RFC7641] and Block-Wise Transfers [RFC7959].

For group communication, only solutions that use CoAP over UDP/ multicast (both IPv6 and IPv4) are considered. Security solutions for group communication other than Group OSCORE are not in scope. General principles for secure group configuration are in scope, however a specific protocol for secure group configuration is not mandated because this is often application-specific.

1.2. 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.

This specification requires readers to be familiar with CoAP [RFC7252] terminology. Section 5 requires readers to be familiar with concepts and terminology from OSCORE [I-D.ietf-core-object-security] and Group OSCORE [I-D.ietf-core-oscore-groupcomm].

2. Use Cases

To illustrate where and how CoAP-based group communication can be used, this section summarizes the most common use cases. These use cases include both secured and non-secured CoAP usage. Each subsection below covers one particular technical category of use cases for CoRE. Within each category, a use case may cover multiple application areas such as home IoT, commercial building IoT (sensing and control), industrial IoT/control, or environmental sensing.

2.1. Discovery

Discovery of physical devices in a network, or discovery of information entities hosted on network devices, are operations that are particularly required in a system during the phases of setup or (re)configuration. When a discovery use case involves devices that need to interact without having been configured previously with a common security context, unsecured CoAP communication is typically used.

2.1.1. Distributed Device Discovery

Device discovery is the discovery and identification of networked devices of a particular class, type, model, or brand. Group communication is used for distributed device discovery, where a central directory service is not used. Typically in distributed service discovery, a multicast request is sent to a particular

address (or address range) and multicast scope of interest, and any devices configured to be discoverable will respond back. For the alternative solution of centralized device discovery a central directory service is accessed through unicast, in which case group communication is not needed.

2.1.2. Distributed Service Discovery

Service discovery is the discovery and identification of particular services hosted on network devices. Services can be identified by one or more parameters such as ID, name, protocol, version and/or type. Distributed service discovery involves group communication to reach individual devices hosting a particular service; with a central directory service not being used. Technically this is similar to the above use case of distributed device discovery (Section 2.1.1). For example, when using CoAP resource discovery (Section 7 of [RFC7252]) there is no technical distinction between doing distributed device discovery and distributed service discovery: both use the same CoAP query interface defined in Section 4 of [RFC6690].

2.1.3. Directory Discovery

This use case is a specific sub-case of Distributed Service Discovery (Section 2.1.2), in which a device needs to identify the location of a Directory on the network to which it can e.g. register its own offered services, or to which it can perform queries to identify and locate other devices/services it needs to access on the network. One particular type of directory is the CoRE Resource Directory [I-D.ietf-core-resource-directory]; and there may be other types of directories that can be discovered using CoAP. Section 3.3 of [RFC7390] shows an example of discovering a CoRE Resource Directory using CoAP group communication. As defined in [I-D.ietf-core-resource-directory], a resource directory is a web entity that stores information about web resources and implements REST interfaces for registration and lookup of those resources. For example, a device can register itself to a resource directory to be looked up by other devices and/or applications.

2.2. Operational

Operational use cases describe those operations that occur most frequently in a networked system, during its operational lifetime and normal usage.

2.2.1. Actuator Group Control

Group communication can be beneficial to control actuators that need to act in synchrony, as a group, with strict timing (latency) requirements. Examples are office lighting, stage lighting, street lighting, or audio alert/Public Address systems. Sections 3.4 and 3.5 of [RFC7390] show examples of lighting control of a group of 6LoWPAN-connected lights.

2.2.2. Device Group Status Request

To properly monitor the status of systems, there may be a need for ad-hoc, unplanned status updates. Group communication can be used to quickly send out a request to a (potentially large) number of devices for specific information. Each device then responds back with the requested data. Those devices that did not respond to the request can optionally be polled again via reliable unicast communication to complete the dataset. The device group may be defined e.g. as "all temperature sensors on floor 3", or "all lights in wing B". For example, it could be a status request for device temperature, most recent sensor event detected, firmware version, network load, and/or battery level.

2.2.3. Network-wide Query

In some cases a whole network or subnet of multiple IP devices needs to be queried for status or other information. This is similar to the previous use case except that the device group is not defined in terms of its function/type but in terms of its network location. Technically this is also similar to distributed service discovery (Section 2.1.2) where a query is processed by all devices on a network - except that the query is not about services offered by the device, but rather specific operational data is requested.

2.3. Software Update

Multicast can be useful to efficiently distribute new software (firmware) to a group of multiple devices. In this case, the group is defined in terms of device type: all devices in the target group are known to be capable of installing and running the new software. The software is distributed as a series of smaller blocks that are collected by all devices and stored in memory. All devices in the target group usually are responsible for integrity verification of the received software; which can be done per-block or for the entire software image once all blocks have been received. Due to the inherent unreliability of CoAP multicast there needs to be a backup mechanism (e.g. implemented using CoAP unicast) by which a device can individually request missing software blocks.

3. General Group Communication Operation

The general operation of group communication, applicable for both unsecured and secured operation, is specified in this section by going through the stack from top to bottom. First, group configuration (e.g. group creation and maintenance which are usually done by an application, user or commissioning entity) is considered in Section 3.1. Then the use of CoAP for group communication including support for protocol extensions (Block-Wise, Observe, PATCH method) follows in Section 3.2. How CoAP group messages are carried over various transport layers is the subject of Section 3.3. Finally, Section 3.4 covers the interworking of CoAP with other protocols at the layers below it.

3.1. Group Configuration

3.1.1. Group Definition

Following [RFC7390], a CoAP group is defined here as a set of CoAP endpoints, where each endpoint is configured to receive CoAP multicast requests that are sent to the group's associated IP multicast address. An endpoint may be a member of multiple groups. Group membership(s) of an endpoint may dynamically change over time. A device sending a CoAP request to a group is not necessarily itself a member of this group: it is only a member if it also has a CoAP server endpoint listening to requests for this group.

A CoAP Group URI has the scheme 'coap' and includes in the authority part either an IP multicast address or a group hostname (e.g., Group Fully Qualified Domain Name (FQDN)) that can be resolved to an IP multicast address. A Group URI also contains an optional UDP port number in the authority part. Group URIs follow the regular CoAP URI syntax (Section 6 of [RFC7252]).

3.1.2. Group Naming (DNS)

For clients, it is RECOMMENDED to use by default an IP multicast address literal in a configured Group URI, instead of a hostname. This is because DNS infrastructure may not be deployed in many constrained networks. In case a group hostname is used in the Group URI, it can be uniquely mapped to an IP multicast address via DNS resolution - if DNS client functionality is available in the clients and the DNS service is supported in the network. Some examples of hierarchical group FQDN naming (and scoping) for a building control application are shown in Section 2.2 of [RFC7390].

3.1.3. Group Creation and Membership

Group membership may be (factory-)preconfigured in devices or dynamically configured in a system on-site.

To create a group, a configuring entity defines an IP multicast address (or hostname) and a UDP port number for the group. Then it configures one or more devices as listeners to that IP multicast address, with a CoAP server listening on the specific port. These devices are the group members. The configuring entity can be a local application with preconfiguration, a user, a cloud service, or a local commissioning tool for example. Also the devices sending requests to the group in the role of CoAP clients need to be configured with the same information, even though they are not necessarily group members. One way to configure a client is to supply it with a CoAP Group URI.

The IETF does not define a mandatory, standardized protocol to accomplish these steps. For secure group communication, the part of the process that involves secure distribution of group keys MAY use standardized communication with a Group Manager as further defined in Section 5. [RFC7390] defines an experimental protocol for configuration of group membership for non-secured group communication, based on JSON-formatted configuration resources.

3.1.4. Group Maintenance

Maintenance of a group includes necessary operations to cope with changes in a system, such as: adding group members, removing group members, reconfiguration of UDP port and/or IP multicast address, reconfiguration of the Group URI, splitting of groups, or merging of groups.

For unsecured group communication, addition/removal of group members is simply done by configuring these devices to start/stop listening to the group IP multicast address, and to start/stop the CoAP server listening to the group IP multicast address and port.

When group communication is secured using Group OSCORE [I-D.ietf-core-oscore-groupcomm] (see Section 5), all CoAP endpoints participating to secure group communication MUST be members of a corresponding OSCORE group, and thus share a common set of cryptographic material. Additional maintenance operations are discussed in Section 5.1.

3.2. CoAP Usage

3.2.1. Request/Response Model

Editor Note, TBD: this section is strongly based on Section 2.5 in [RFC7390]. In case a reference to this section is preferred, we can replace most of the following text in this section by a reference to it.

All CoAP requests that are sent via IP multicast MUST be Non-confirmable (Section 8.1 of [RFC7252]). The Message ID in an IP multicast CoAP message is used for optional message deduplication as detailed in Section 4.5 of [RFC7252].

A server MAY send back a unicast response to the CoAP group communication request - whether it does this or not is selected by the server application. The unicast responses received by the CoAP client may be a mixture of success (e.g., 2.05 Content) and failure (e.g., 4.04 Not Found) codes depending on the individual server processing results.

TBD: the CoAP Option for No Server Response [RFC7967] can be used by the client to influence response suppression on the server side. Possibly we can include this information here; it specifically targets use for multicast use cases also.

The client can distinguish the origin of multiple server responses by the source IP address of the UDP message containing the CoAP response or any other available unique identifier (e.g., contained in the CoAP response payload). In case a client has sent multiple group requests with concurrent CoAP transactions ongoing, the responses are as usual matched to a request using the Token.

For multicast CoAP requests, there are additional constraints on the reuse of Token values, compared to the unicast case. In the unicast case, receiving a response effectively frees up its Token value for reuse since no more responses will follow. However, for multicast CoAP, the number of responses is not bounded a priori. Therefore, the reception of a response cannot be used as a trigger to "free up" a Token value for reuse. Reusing a Token value too early could lead to incorrect response/request matching in the client and would be a protocol error. Therefore, the time between reuse of Token values used in multicast requests MUST be greater than:

NON_LIFETIME + MAX_LATENCY + MAX_SERVER_RESPONSE_DELAY

where NON_LIFETIME and MAX_LATENCY are defined in Section 4.8 of [RFC7252]. This specification defines MAX_SERVER_RESPONSE_DELAY as

in [RFC7390], that is: the expected maximum response delay over all servers that the client can send a multicast request to. This delay includes the maximum Leisure time period as defined in Section 8.2 of [RFC7252]. However, CoAP does not define a time limit for the server response delay. Using the default CoAP parameters, the Token reuse time MUST be greater than 250 seconds plus MAX_SERVER_RESPONSE_DELAY. A preferred solution to meet this requirement is to generate a new unique Token for every multicast request, such that a Token value is never reused. If a client has to reuse Token values for some reason, and also MAX_SERVER_RESPONSE_DELAY is unknown, then using MAX_SERVER_RESPONSE_DELAY = 250 seconds is a reasonable guideline. The time between Token reuses is in that case set to a value greater than 500 seconds.

3.2.2. Port and URI Path Selection

A CoAP server that is a member of a group listens for CoAP messages on the group's IP multicast address, usually on the CoAP default UDP port, 5683, or another non-default UDP port if configured. Regardless of the method of selecting the port number, the same port number MUST be used across all CoAP servers that are members of a group and across all CoAP clients performing the group requests to that group. The URI Path used in the request is preferably a path that is known to be supported across all group members. However there are use cases where a request only can be successful for a subset of the group and errors are returned by those group members for which the request was unsuccessful.

Using different ports with the same IP multicast address is an efficient way to create multiple CoAP groups in constrained devices, in case the device's stack only supports a limited number of IP multicast group memberships. However, it must be taken into account that this incurs additional processing overhead on each CoAP server participating in at least one of these groups: messages to groups that are not of interest to the node are only discarded at the higher transport (UDP) layer instead of directly at the network (IP) layer.

Port 5684 is reserved for DTLS-secured CoAP and MUST NOT be used for any CoAP group communication.

For a CoAP server node that supports resource discovery as defined in Section 2.4 of [RFC7252], the default port 5683 MUST be supported (see Section 7.1 of [RFC7252]) for the "All CoAP Nodes" multicast group.

(TBD: consider if we should say that receiving node/server SHOULD NOT send a "ICMP Destination Unreachable - Port Unreachable" in response to such request.)

3.2.3. Proxy Operation

TBD: check if draft-ietf-core-multipart-ct-02 can solve the "multiple answers" case when a Proxy sends back multiple CoAP responses to a multicast request. Possibly a client may support this. Is there a way to signal multipart support by the client? Can the multipart parts signal the origin/IP address of their origin server?

3.2.4. Congestion Control

The measures to reduce network congestion risks are listed in Section 2.8 of [RFC7390], including both mandatory protocol elements as well as guidelines. This specification RECOMMENDS to apply the guidelines specified in that section.

3.2.5. Observing Resources

The CoAP Observe Option [RFC7641] is a protocol extension of CoAP, that allows a CoAP client to retrieve a representation of a resource and automatically keep this representation up-to-date over a longer period of time. The client gets notified when the representation has changed. [RFC7641] does not mention whether the Observe Option can be combined with CoAP multicast.

Using the Observe Option in a CoAP multicast GET request is explicitly specified here as allowed; it is useful as a means to start observing a particular resource on all members of a (multicast) group at the same time. Group members that do not have this resource or do not allow the GET method on it will respond with the usual 4.04 Not Found or 4.05 Method Not Allowed, respectively.

A client sending a multicast GET with Observe MAY repeat this request using the same Token Option and Observe Option value, in order to ensure that enough (or all) group members have been reached with the request.

3.2.6. Block-Wise Transfer

Section 2.8 of [RFC7959] specifies how a server (group member), responding to a multicast request with a large resource, can use Block-Wise transfer to limit the size of the initial response.

TBD: investigate use of Block-Wise for PUT/POST/PATCH/iPATCH operations e.g. to be used for distributing software blocks over multicast. We can specify its use, or remark that this is undefined.

3.3. Transport

TBD: Mark [RFC8323] (TCP, TLS, WebSockets) as not applicable for this form of groupcomm, as well as CoAP-over-SMS.

3.3.1. UDP/IPv6 Multicast Transport

TBD: include the "Exceptions" cases here of RFC 7390 Section 2.10. State that IPv6 multicast is prerequisite. Also mention the All-CoAP-nodes IPv6 addresses.

3.3.2. UDP/IPv4 Multicast Transport

TBD: includes the "Exceptions" cases here of RFC 7390 2.10. State that IPv4 multicast is prerequisite. mention All-CoAP-nodes IPv4 addresses and the like

3.3.3. 6LoWPAN

TBD: 6lowpan-specific considerations to go here. Specifically, a multicast request should preferably fit in one L2 frame to avoid the strong performance drop that comes with 6LoWPAN-fragmentation and reassembly. Also reference [RFC7346] for the realm-local scope.

3.4. Interworking with Other Protocols

3.4.1. MLD/MLDv2/IGMP

TBD: see Section 4.2 of [RFC7390] and include the content here or refer to it.

3.4.2. RPL

TBD: see Section 4.3 of [RFC7390] and include the content here or refer to it.

3.4.3. MPL

TBD: see Section 4.4. [RFC7390] and include the content here or refer to it.

4. Unsecured Group Communication

CoAP group communication can operate in CoAP NoSec (No Security) mode, without using application-layer and transport-layer security mechanisms. The NoSec mode uses the "coap" scheme, and is defined in Section 9 of [RFC7252]. Before using this mode of operation, the security implications (Section 6.1) must be well understood.

5. Secured Group Communication using Group OSCORE

The application-layer protocol Object Security for Constrained RESTful Environments (OSCORE) [I-D.ietf-core-object-security] provides end-to-end encryption, integrity and replay protection of CoAP messages exchanged between two CoAP endpoints. These can act both as CoAP Client as well as CoAP Server, and share an OSCORE Security Context used to protect and verify exchanged messages. The use of OSCORE does not affect the URI scheme and OSCORE can therefore be used with any URI scheme defined for CoAP.

OSCORE uses COSE [RFC8152] to perform encryption, signing and Message Authentication Code operations, and to efficiently encode the result as a COSE object. In particular, OSCORE takes as input an unprotected CoAP message and transforms it into a protected CoAP message, by using Authenticated Encryption Algorithms with Additional Data (AEAD).

OSCORE makes it possible to selectively protect different parts of a CoAP message in different ways, so still allowing intermediaries (e.g., CoAP proxies) to perform their intended funtionalities. That is, some message parts are encrypted and integrity protected; other parts only integrity protected to be accessible to, but not modifiable by, proxies; and some parts are kept as plain content to be both accessible to and modifiable by proxies. Such differences especially concern the CoAP options included in the unprotected message.

Group OSCORE [I-D.ietf-core-oscore-groupcomm] builds on OSCORE, and provides end-to-end security of CoAP messages exchanged between members of an OSCORE group, while fulfilling the same security requirements.

In particular, Group OSCORE protects CoAP requests sent over IP multicast by a CoAP client, as well as multiple corresponding CoAP responses sent over IP unicast by different CoAP servers. However, the same keying material can also be used to protect CoAP requests sent over IP unicast to a single CoAP server in the OSCORE group, as well as the corresponding responses.

Group OSCORE uses digital signatures to ensure source authentication of all messages exchanged within the OSCORE group. That is, sender devices sign their outgoing messages by means of their own private key, and embed the signature in the protected CoAP message.

A Group Manager is responsible for one or multiple OSCORE groups. In particular, the Group Manager acts as repository of public keys of

group members; manages, renews and provides keying material in the group; and drives the join process for new group members.

As RECOMMENDED in [I-D.ietf-core-oscore-groupcomm], a CoAP endpoint can join an OSCORE group by using the method 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].

A CoAP endpoint can discover OSCORE groups and retrieve information to join them through their Group Managers by using the method described in [I-D.tiloca-core-oscore-discovery] and based on the CoRE Resource Directory [I-D.ietf-core-resource-directory].

If security is required, CoAP group communication as described in this specification MUST use Group OSCORE. In particular, a CoAP group as defined in Section 3.1.1 and using secure group communication is associated to an OSCORE group, which includes:

- All members of the CoAP group, i.e. the CoAP endpoints configured (also) as CoAP servers and listening to the group's multicast IP address.

- All further CoAP endpoints configured only as CoAP clients, that send (multicast) CoAP requests to the CoAP group.

5.1. Secure Group Maintenance

Additional key management operations on the OSCORE group are required, depending also on the security requirements of the application (see Section 6.2). That is:

- Adding new members to a CoAP group or enabling new client-only endpoints to interact with that group require also that each of such members/endpoints join the corresponding OSCORE group. By doing so, they are securely provided with the necessary cryptographic material. In case backward security is needed, this also requires to first renew such material and distribute it to the current members/endpoints, before new ones are added and join the OSCORE group.

- In case forward security is needed, removing members from a CoAP group or stopping client-only endpoints from interacting with that group requires removing such members/endpoints from the corresponding OSCORE group. To this end, new cryptographic material is generated and securely distributed only to the remaining members/endpoints. This ensures that only the members/endpoints intended to remain are able to continue participating to

secure group communication, while the evicted ones are not able to.

The key management operations mentioned above are entrusted to the Group Manager responsible for the OSCORE group [I-D.ietf-core-oscore-groupcomm], and it is RECOMMENDED to perform them according to the approach described in [I-D.ietf-ace-key-groupcomm-oscore].

6. Security Considerations

This section provides security considerations for CoAP group communication using IP multicast.

6.1. CoAP NoSec Mode

CoAP group communication, if not protected, is vulnerable to all the attacks mentioned in Section 11 of [RFC7252] for IP multicast.

Thus, for sensitive and mission-critical applications (e.g., health monitoring systems and alarm monitoring systems), it is NOT RECOMMENDED to deploy CoAP group communication in NoSec mode.

Without application-layer security, CoAP group communication SHOULD only be deployed in non-critical applications (e.g., read-only temperature sensors where the client reading out the values does not use the data to control actuators or to base an important decision on).

Discovery of devices and resources is a typical use case where NoSec mode is applied, since the devices involved do not have yet configured any mutual security relations at the time the discovery takes place.

6.2. Group OSCORE

Group OSCORE provides end-to-end application-level security. This has many desirable properties, including maintaining security properties while forwarding traffic through intermediaries (proxies). Application-level security also tends to more cleanly separate security from the dynamics of group membership (e.g., the problem of distributing security keys across large groups with many members that come and go).

For sensitive and mission-critical applications, CoAP group communication MUST be protected by using Group OSCORE as specified in [I-D.ietf-core-oscore-groupcomm]. The same security considerations

from Section 8 of [I-D.ietf-core-oscore-groupcomm] hold for this specification.

A key management scheme for secure revocation and renewal of group keying material should be adopted in OSCORE groups. Also, the key management scheme should preserve backward and forward security in the OSCORE group, if the application requires so (see Section 2.1 of [I-D.ietf-core-oscore-groupcomm]).

CoAP endpoints using Group OSCORE countersign their outgoing messages, by means of the countersignature algorithm used in the OSCORE group. This ensures source authentication of messages exchanged by CoAP endpoints through CoAP group communication. In fact, it allows to verify that a received message has actually been originated by a specific and identified member of the OSCORE group.

Appendix F of [I-D.ietf-core-oscore-groupcomm] discusses a number of cases where a recipient CoAP endpoint may skip the verification of countersignatures, possibly on a per-message basis. However, this is NOT RECOMMENDED. That is, a CoAP endpoint receiving a message secured with Group OSCORE SHOULD always verify the countersignature.

Group OSCORE addresses security attacks mentioned in Sections 11.2-11.6 of [RFC7252], with particular reference to their execution over IP multicast. That is: it provides confidentiality and integrity of request/response data through proxies also in multicast settings; it prevents amplification attacks carried out through responses to injected requests over IP multicast; it limits the impact of attacks based on IP spoofing; it prevents cross-protocol attacks; it derives the group key material from, among other things, a Master Secret securely generated by the Group Manager and provided to CoAP endpoints upon their joining of the OSCORE group; countersignatures assure source authentication of exchanged CoAP messages, and hence prevent a group member to be used for subverting security in the whole group.

6.3. 6LoWPAN

Editor Note, TBD: identify if multi-fragment multicast requests have a negative effect on security and, if so, advice here on trying to avoid such requests. Also an attacker could use multi-fragment to occupy reassembly buffers of many routing 6LoWPAN nodes.

6.4. Wi-Fi

TBD: Wi-Fi specific security considerations; see also Section 5.3.1 of [RFC7390].

6.5. Monitoring

TBD: see Section 5.4 of [RFC7390].

7. IANA Considerations

This document has no actions for IANA.

8. References

8.1. Normative References

[I-D.ietf-core-object-security]
Selander, G., Mattsson, J., Palombini, F., and L. Seitz, "Object Security for Constrained RESTful Environments (OSCORE)", draft-ietf-core-object-security-16 (work in progress), March 2019.

[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-04 (work in progress), March 2019.

[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>.

[RFC6690] Shelby, Z., "Constrained RESTful Environments (CoRE) Link Format", RFC 6690, DOI 10.17487/RFC6690, August 2012, <https://www.rfc-editor.org/info/rfc6690>.

[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>.

[RFC7959] Bormann, C. and Z. Shelby, Ed., "Block-Wise Transfers in the Constrained Application Protocol (CoAP)", RFC 7959, DOI 10.17487/RFC7959, August 2016, <https://www.rfc-editor.org/info/rfc7959>.

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

[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>.

[RFC8323] Bormann, C., Lemay, S., Tschofenig, H., Hartke, K., Silverajan, B., and B. Raymor, Ed., "CoAP (Constrained Application Protocol) over TCP, TLS, and WebSockets", RFC 8323, DOI 10.17487/RFC8323, February 2018, <https://www.rfc-editor.org/info/rfc8323>.

8.2. Informative References

[Californium] Eclipse Foundation, "Eclipse Californium", March 2019, <https://github.com/eclipse/californium/tree/2.0.x/californium-core/src/main/java/org/eclipse/californium/core>.

[Go-OCF] Open Connectivity Foundation (OCF), "Implementation of CoAP Server & Client in Go", March 2019, <https://github.com/go-ocf/go-coap>.

[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-01 (work in progress), March 2019.

[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-22 (work in progress), March 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-19 (work in progress), January 2019.

[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-01 (work in progress), January 2019.

[RFC7346] Droms, R., "IPv6 Multicast Address Scopes", RFC 7346, DOI 10.17487/RFC7346, August 2014, <https://www.rfc-editor.org/info/rfc7346>.

[RFC7390] Rahman, A., Ed. and E. Dijk, Ed., "Group Communication for the Constrained Application Protocol (CoAP)", RFC 7390, DOI 10.17487/RFC7390, October 2014, <https://www.rfc-editor.org/info/rfc7390>.

[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>.

Acknowledgments

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

Authors' Addresses

Esko Dijk
IoTconsultancy.nl

Utrecht
The Netherlands

Email: esko.dijk@iotconsultancy.nl

Chonggang Wang
InterDigital
1001 E Hector St, Suite 300
Conshohocken PA 19428
United States

Email: Chonggang.Wang@InterDigital.com

Marco Tiloca
RISE AB
Isafjordsgatan 22
Kista SE-16440 Stockholm
Sweden

Email: marco.tiloca@ri.se