OCF "Ipanema" – End-to-End Security (Security Spec) – Security WG CR 1657

Legal Disclaimer

New Norm Refs to Add to Clause 2 "Normative References

OCF Cloud Specification, Information technology – Open Connectivity Foundation (OCF) Specification – Part 8: Cloud Specification
Latest version available at:
https://openconnectivity.org/specs/OCF_Cloud_Specification.pdf

IETF RFC 7252, *The Constrained Application Protocol (CoAP)*, June 2014
https://www.rfc-editor.org/info/rfc7252

IETF RFC 8613, *Object Security for Constrained RESTful Environments (OSCORE)*, July 2019
https://www.rfc-editor.org/info/rfc8613

IETF RFC 8152, *CBOR Object Signing and Encryption (COSE),* July 2017
https://www.rfc-editor.org/info/rfc217

New terms & definitions to Add to Clause 3.1 Terms and Definitions

End-to-End Secure
securely encapsulate information so that OCF Proxies on the end-to-end delivery path do not need to be trusted with the confidentiality, integrity and freshness of that information

End-to-End Security of Unicast Messages
interoperable mechanism which End-to-End Secures the exchange of unicast OCF CRUDN messages

OCF Proxy
functionality which can interpret the OCF compliant URIs of request messages intended for resources on another OCF Server and can route those request messages accordingly

Origin Client
Client which originally generated a request, as opposed to the Client functionality of a Proxy which is forwarding a request from another Device

OSCORE Master Secret
"Master Secret" as defined in clause 3.1 of in IETF RFC 8613

OSCORE Recipient ID
"Recipient ID" as defined in clause 3.1 of in IETF RFC 8613

OSCORE Security Context
"Security Context" as defined in clause 3.1 of in IETF RFC 8613

OSCORE Sender ID
"Sender ID" as defined in clause 3.1 of in IETF RFC 8613

OSCORE Sender Sequence Number
"Sender Sequence Number" as defined in clause 3.1 of in IETF RFC 8613

Target Server
Server to which a request is addressed, as opposed to the Server functionality of a Proxy which receives a request to be forwarded to another Device

New abbreviations to add to Clause 3.2 Abbreviated Terms

AEAD Authenticated Encryption with Authenticated Data

NOTE: Defined in IETF RFC 8152

COSE CBOR Object Signing and Encryption

NOTE: Defined in IETF RFC 8152

OSCORE Object Security for Constrained RESTful Environments

NOTE: Defined in IETF RFC 8613

########Changes to Clause 5.1 Preamble

5.1 Security Model of Operation

This is an informative clause. The goal for the OCF Security architecture is to protect the Resources and all aspects of HW and SW that are used to support the protection of Resource. From OCF perspective, a Device is a logical entity that conforms to the OCF documents. In an interaction between the Devices, the Device acting as the Server holds and controls the Resources and provides the Device acting as a Client with access to those Resources, subject to a set of Security mechanisms. The Platform, hosting the Device may provide Security hardening that will be required for ensuring robustness of the variety of operations described in this document.

The security model of operation for direct Device-to-Device interaction (that is, exchanges which are not facilitated by entities acting as OCF Proxies between the Client and Server) is depicted in Figure 2 and described in the following steps.

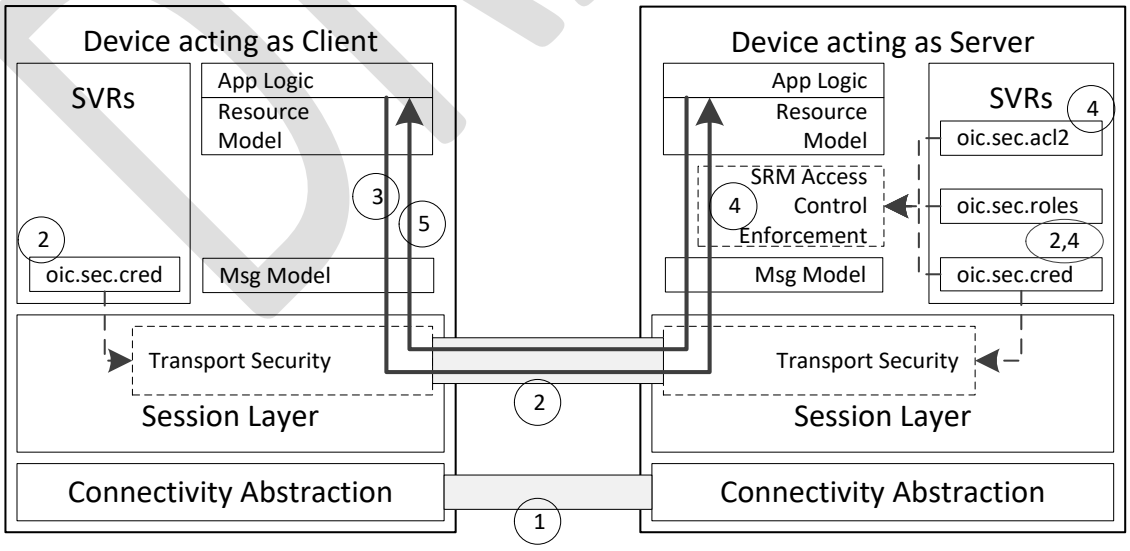

Figure 1 – OCF Layers for direct Device-to-Device interaction

1) The Client establishes a network connection to the Server (Device holding the Resources). The connectivity abstraction layer ensures the Devices are able to connect despite differences in connectivity options.
2) The Devices (e.g. Server and Client) exchange messages either with or without a mutually-authenticated secure channel between the two Devices.
   a) The "/oic/sec/cred" Resource on each Devices holds the credentials used for mutual authentication and (when applicable) certificate validation.
   b) Messages received over a secured channel are associated with a "deviceUUID". In the case of a certificate credential, the "deviceUUID" is in the certificate received from the other Device. In the case of a symmetric key credential, the "deviceUUID" is configured with the credential in the "/oic/sec/cred" Resource.
   c) The Server can associate the Client with any number of roleid. In the case of mutual authentication using a certificate, the roleid (if any) are provided in role certificates; these are configured by the Client to the Server. In the case of a symmetric key, the allowed roleid (if any) are configured with the credential in the "/oic/sec/cred" Resource.
   d) Requests received by a Server over an unsecured channel are treated as anonymous and not associated with any "deviceUUID" or "roleid".
3) The Client submits a request to the Server.
4) The Server receives the request.
   a) If the request is received over an unsecured channel, the Server treats the request as anonymous and no "deviceUUID" or "roleid" are associated with the request.
   b) If the request is received over a secure channel, then the Server associates the "deviceUUID" with the request, and the Server associates all valid roleid of the Client with the request.
   c) The Server then consults the Access Control List (ACL), and looks for an ACL entry matching the following criteria:
      i) The requested Resource matches a Resource reference in the ACE
      ii) The requested operation is permitted by the "permissions" of the ACE, and
      iii) The "subjectUUID" contains either one of a special set of wildcard values or, if the Device is not anonymous, the subject matches the Client Deviceid associated with the request or a valid "roleid" associated with the request. The wildcard values match either all Devices communicating over an authenticated and encrypted session, or all Devices communicating over an unauthenticated and unencrypted session.

      If there is a matching ACE, then access to the Resource is permitted; otherwise access is denied. Access is enforced by the Server's Secure Resource manager (SRM).
5) The Server sends a response back to the Client.

OCF also supports exchange of messages between an Origin Client and Target Server facilitated at one or more entities acting as OCF Proxies.

NOTE 1: Any number of OCF Proxies may be on the path between the Origin Client and Target Server, although this number is expected to be small in practice.

In some scenarios, an OCF Proxy acts as a Server to incoming OCF CRUDN request messages: processing the OCF CRUDN request messages; and then sending appropriate OCF CRUDN request messages onwards towards the Target Server. The OCF Proxy can also process the

corresponding incoming OCF CRUDN response message and send appropriate OCF CRUDN request messages back towards the Origin Client.

This approach implies that the owner of the Security Domain (containing the Origin Client and Target Server) is willing to trust all OCF Proxies on the message delivery path with the confidentiality, integrity and freshness of the OCF CRUDN messages. Alternatively, the Origin Client and Target Server can apply End-to-End Security of Unicast Messages which enables securing the exchange of OCF CRUDN messages so that OCF Proxies do not need to be trusted with the confidentiality and integrity of the OCF CRUDN messages.

The security model of operation when using OCF Proxies without End-to-End Security of Unicast Messages is described in OCF Cloud Specification, OCF Cloud Security Specification, and C2C API.

Figure 2a and Figure 2b depict the security model of operation when using OCF Proxies and End-to-End Security of Messages is applied; see also the following steps. Figure 2a illustrates an example with one OCF Proxy. Figure 2b illustrates a more complex example with two OCF Proxies using C2C API; see notes 1 and 2.

NOTE 2: If the OCF Proxies in Figure 2b are OCF Clouds, OCF Proxy A is the Origin Cloud to which the Origin Client is registered, and OCF Proxy B is the Target Cloud to which the Target Server is registered.

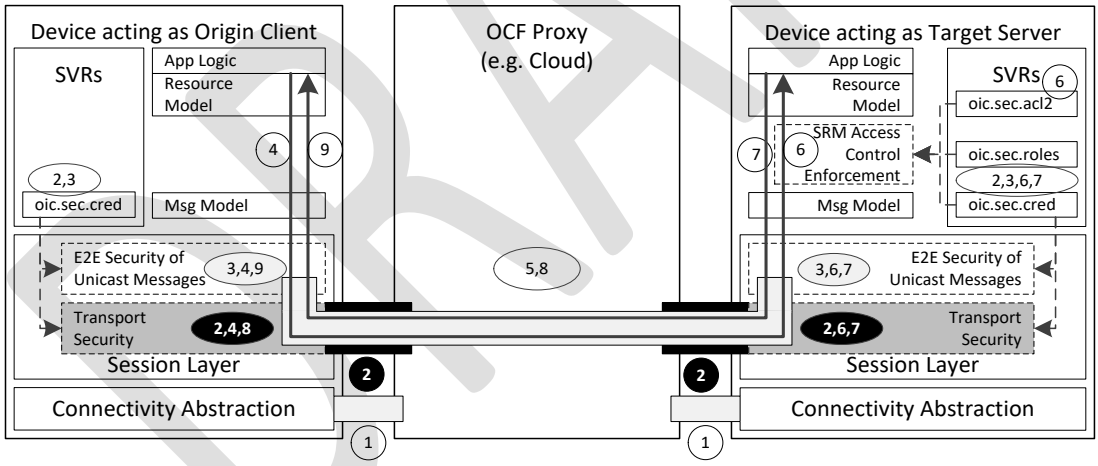

Figure 2a – OCF Layers for interactions via one OCF Proxy

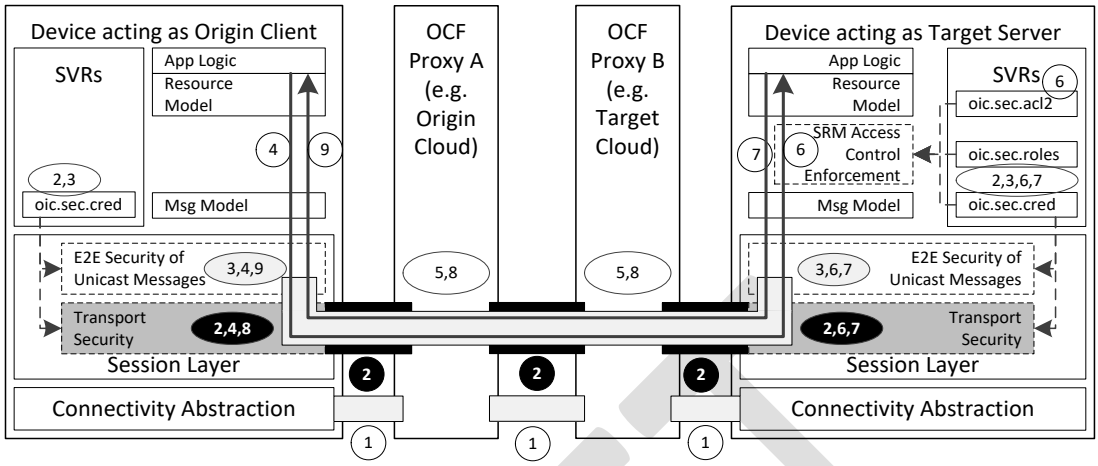

Figure 3a – OCF Layers for interactions via two OCF Proxies

1) Pairwise network connections are established.
2) Messages are exchanged over each network connection either via pairwise mutually-authenticated secure transport connection.
3) The Origin Client and Target Server establish an End-to-End Secured channel which is mutually-authenticated using credentials held in the "/oic/sec/cred" Resources of the Origin Client and Target Server.
4) The Origin Client generates an OCF CRUDN request message to the Target Server. The Origin Client encapsulates the OCF CRUDN request message into an End-to-End Secured request message of the End-to-End Secured channel (established in step 3). Information identifying the Target Server is left un-encrypted in the End-to-End Secured request message, so OCF Proxies can use the identifying information to route the End-to-End Secured request message correctly. The Origin Client sends the End-to-End Secured request message to its OCF Proxy, over the optionally secured transport connection established with that OCF Proxy. See Note 3.
5) Each OCF Proxy on the path extracts the identifying information of the Target Server from the request message and, subject to the OCF Proxy's policies governing End-to-End Secured request messages, forwards the end-to- End-to-End Secured request message towards the Target Server over an optionally secured transport connection. See notes 3, 4 and 5.
6) The Target Server verifies and decrypts the End-to-End Secured request message as a message of the End-to-End Secured channel (established at step 3) to extract the encapsulated OCF CRUDN request message from the Origin Client. The OCF CRUDN request message is treated as being received over an authenticated encrypted ("auth-crypt") connection and associated with a "deviceUUID". The "deviceUUID" is associated with the credential in the "/oic/sec/cred" Resource used to establish the End-to-End Secured channel in step 3.
7) The Target Server determines whether access to the resource is permitted as described in step 4c of the Security model for direct Device-to-Device interaction shown in Figure 2.

8) The Target Server generates an OCF CRUDN response message and encapsulates the OCF CRUDN response message into an End-to-End Secured response message of the End-to-End Secured channel (established at step 3). The Target Secure sends the End-to-End Secured response message to its OCF Proxy, over the optionally secured transport connection on which the corresponding request was received. See Note 3.

9) Each OCF Proxy on the path forwards the End-to-End Secured response message towards the Origin Client over the optionally secured transport connection on which the corresponding request message was received. See Note 3.

10) The Origin Client verifies and decrypts the End-to-End Secured response message as a message of the End-to-End Secured channel (established at step 3) to extract the encapsulated OCF CRUDN response message from the Target Server.

NOTE 3: While in transit, the OCF CRUDN message might be secured by up to two independent layers of Security: a layer of End-to-End Security of Unicast Messages (using OSCORE), and an independent layer of transport Security (using DTLS or TLS).

NOTE 4: This document does not address details of how an OCF Proxy determines if its policies permit forwarding the request message towards the identified Target Server. If an OCF Proxy permits forwarding a request message towards a Target Server, then it is assumed that the OCF Proxy also permits forwarding the corresponding response message(s) over the transport connection on which the corresponding request message was received.

NOTE 5: This document does not address how OCF Proxy A determines that OCF Proxy B is the correct OCF Proxy to forward the request message to. The C2C API provides the details for the case where the OCF Proxy A and OCF Proxy B are OCF Clouds.

Resource protection includes protection of data both while at rest and during transit. Aside from access control mechanisms, the OCF Security Specification does not include specification of secure storage of Resources, while stored at Servers. However, at rest protection for Security Resources is expected to be provided through a combination of secure storage and access control. Secure storage can be accomplished through use of hardware Security or encryption of data at rest. The exact implementation of secure storage is subject to a set of hardening requirements that are specified in clause 14 and may be subject to certification guidelines.

Data-in-transit protection, on the other hand, will be specified fully as a normative part of this document. This document supports data-in-transit protection at the transport layer through use of mechanisms such as DTLS, and end-to-end data-in-transit protection through OSCORE.

NOTE 6: DTLS will provide packet by packet protection, rather than protection for the OCF CRUDN message as whole. For instance, if the integrity of the entire OCF CRUDN message as a whole is required, separate end-to-end Security (for example, using OSCORE) should be applied before passing the message down to the transport layer.

Figure 3 depicts OCF Security Enforcement Points.

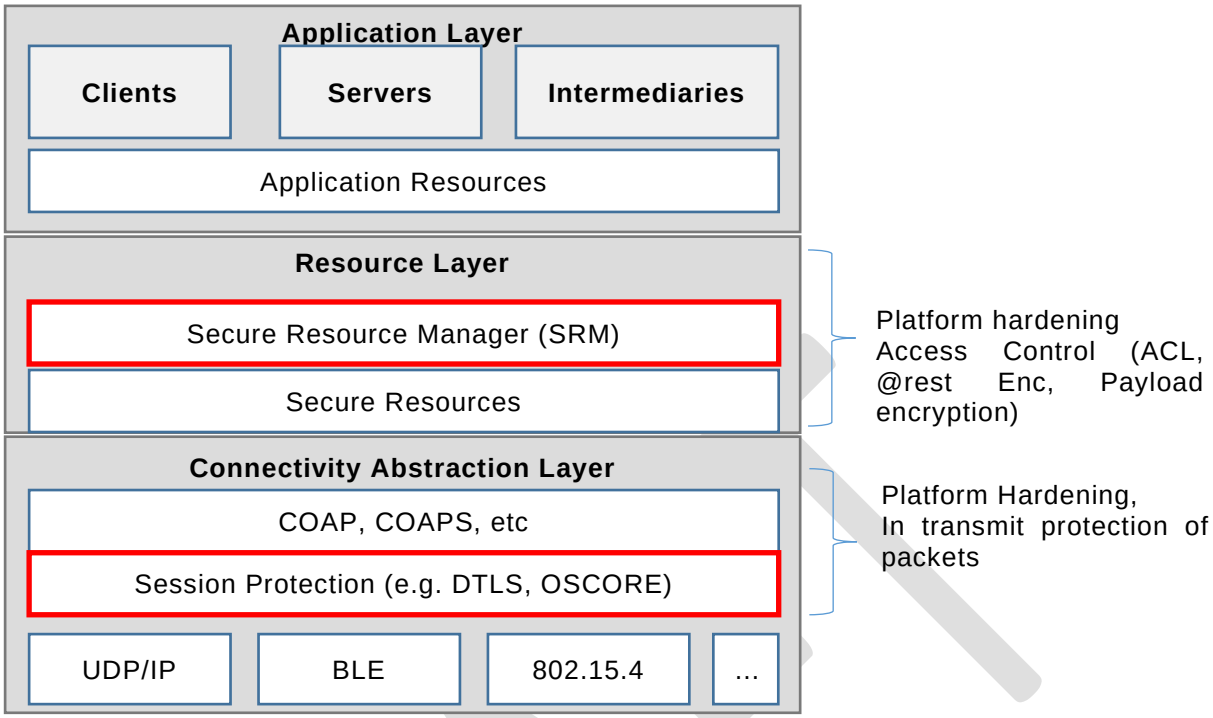

Figure 4 – OCF Security Enforcement Points

########Changes to Clause 5.6

5.6 Credential Overview

Devices may use credentials to prove the identity and role(s) of the parties in bidirectional communication. Credentials can be symmetric or asymmetric. Each device stores secret and public parts of its own credentials where applicable, as well as credentials for other devices that have been provided by the DOTS or a CMS. These credentials are then used in the establishment of secure communication sessions (e.g. using DTLS, TLS or OSCORE) to validate the identities of the participating parties. Role credentials are used once an authenticated session is established, to assert one or more roles for a device.

NEW sub clause of Clause 5

5.X End-to-End Security of Unicast Messages

The Security model for End-to-End Security of Unicast Messages is described in Figure 2a and Figure 2b of clause 5.1 and the accompanying steps.

OCF uses the Object Security for Constrained RESTful Environments (OSCORE) protocol IETF RFC 8613 for End-to-End Security of Unicast Messages. The Origin Client transforms a CoAP-encoded OCF CRUDN request message into an OSCORE request message which can be forwarded towards the Target Server by OCF Proxies; the Target Server then processes the

OSCORE request message to extract the OCF CRUDN request message. Likewise, the Target Server then transforms a CoAP-encoded OCF CRUDN response message into an OSCORE response message which can be forwarded towards the Origin Client by OCF Proxies; the Origin Client then processes the OSCORE response message to extract the OCF CRUDN response message. OSCORE preserves the confidentiality, integrity and freshness of the OCF CRUDN messages while in transit between the Origin Client and the Target Server.

OSCORE specification supports transporting OSCORE messages using the CoAP protocol already used in OCF specifications. The payload of the OSCORE message is a CBOR Object Signature and Encryption (COSE) object (see IETF RFC 8152) in which all elements of the CoAP-encoded OCF CRUDN message, other than those parts which are needed for delivering the message to the receiving Device, are encrypted and integrity protected. OSCORE also includes replay protection.

NEW sub clause of Clause 9.3

9.3.x Credentials for direct provisioning an OSCORE Security Context

A credential entry with the credential type 64 is used for direct provisioning of OSCORE Security Context parameters for use in End-to-End Security of Unicast Messages.

The "privatedata" Property of the credential entry with the credential type 64 in the "/oic/sec/cred" Resource contains the OSCORE Master Key.

A credential entry with the credential type 64 shall expose the OSCORE Configuration ("oscore") Property, which includes:

- The "senderid" Property containing the OSCORE Sender ID parameter.
- The "recipientid" Property containing the OSCORE Recipient ID parameter.
- The "ssn" Property contains a read-only value used to store the OSCORE Sender Sequence Number.

NOTE: values of "senderid" and "recipientid" are expected to be lowercase hexadecimal encoded with "0x" encoding prefix omitted.

See clause Q.2 for description of the OSCORE parameters.

New Clause on Alternative in-transit protection mechanisms (including OSCORE)

Q Alternative in-transit protection mechanisms

Q.1 Introducti

on to in-transit protection mechanisms

In addition to the DTLS protection mechanisms for device-to-device communication specified in clause 10 and clause 11.2, and TLS protection specified in OCF Cloud Security document, OCF supports the following in-transit protection mechanisms:

- End-to-End Security of Unicast Messages using OSCORE, specified in clause Q.2.

Q.2 End-to-End Security of Unicast Messages using OSCORE

Q.2.1 Introduction to End-to-End Security of Unicast Messages using OSCORE

End-to-End Security of Unicast Messages is accomplished by applying a layer of in-transit protection above the transport layer Security (provided by DTLS or TLS) and below the resource-access authorization layer, using Object Security for Constrained RESTful Environments (OSCORE) IETF RFC 8613.

Relative to an exchange of an OCF CRUDN Request message and OCF CRUDN Response message:

– The Device acting as a Client (that is, sending an OCF CRUDN Request message and receiving the corresponding OCF CRUDN Response message) acts as an OSCORE client. Within the scope of clause Q.2, all Clients are assumed to support OSCORE and perform OSCORE client processing.

– The Device acting as a Server (that is, receiving an OCF CRUDN Request message and sending one or more corresponding OCF CRUDN Response messages) acts as an OSCORE server. Within the scope of clause Q.2, all Servers are assumed to support OSCORE and perform OSCORE server processing.

Clause Q.2.4 specifies the supported mechanism for establishing an OSCORE Security Context between two Devices. For each Device, an authorized Client (e.g. OBT) provisions the OSCORE Security Context parameters to a credential entry of the "/oic/sec/cred" Resource. The "subjectuuid" of that credential entry identifies the other Device that shares that OSCORE Security Context (similar to how a DTLS endpoint associates each DTLS PSK session with the Device UUID of the other DTLS endpoint).

Q.2.2 OSCORE ID Namespace Prefix

Clause Q.2.4 specifies one mechanism for establishing an OSCORE Security Context between two Devices. Different mechanisms have different entities responsible for managing the selection of OSCORE Sender ID and OSCORE Recipient ID. There is value in preventing Devices having multiple OSCORE Security Contexts with identical Recipient IDs: this simplifies processing and avoids inefficiencies.

If a set of one or more coordinated entities (e.g. a group of OBTs) assigns a set of OSCORE Recipient IDs to OSCORE Security Contexts on a Device, then that set of entities is responsible for avoiding duplicate OSCORE Recipient IDs. However, two non-coordinated entities assigning OSCORE Recipient IDs might assign identical OSCORE Recipient IDs if there is no predefined agreement on assignment of OSCORE Recipient IDs.

For this reason, the first byte of the OSCORE Sender ID and OSCORE Recipient ID use a OSCORE Identifier Namespace Prefix. The Table Y is the authoritative definition of the assigned OSCORE Identifier Namespace Prefix values.

Table Y –OSCORE Identifier Namespace Prefix

Value	Interpretation	Applicable clauses
0x00	Reserved for future use	
0x01	Directly provisioned OSCORE Security Context	Q.2.4
0x02-0x0F	Reserved for future use	

Q.2.3 OSCORE protection and verification of unicast OCF CRUDN messages

All OSCORE message processing requirements in clause 8 in IETF RFC 8613 apply.

NOTE 1: Clause 8 in IETF RFC 8613 requires the Client keep the association of the request Token (see IETF RFC 7252) with the Security Context and Partial IV of the request, in order to be able to find the Security Context and compute the OSCORE Additional Authenticated Data when verifying the response.

If a Client has an established OSCORE Security Context associated with a Server, then the following call flow applies whenever the Client sends unicast OCF CRUDN request targeting Resources hosted on the Server. The Client may send multiple OSCORE requests to multiple Servers

1) The Client shall apply the OSCORE request protection processing to OCF CRUDN requests targeting Resources hosted on the Server as specified in Clause 8.1 in IETF RFC 8613, using the OSCORE Security Context. See ISO/IEC 30118-1:2018 for details on setting the Proxy-URI option.

   The Client sends the OSCORE request message to the Server (optionally via OCF Proxies). The OSCORE request message shall be delivered over secure transports: Device-to-Device communication is secured as specified in clause 10; Device to Cloud communication is secured as specified in ISO/IEC 30118-8; and Cloud-to-Cloud communication is secured as specified in Cloud-to-Cloud.

2) The Server receives a unicast OSCORE request message. The Server shall apply the OSCORE request verification and decryption processing in clause 8.2 of IETF RFC 8613 with the following clarifications:

   a) At Step 2 in clause 8.2 of IETF RFC 8613

      i) If either the decompression or the COSE message fails to decode, the Server shall respond with error response message (e.g. "Bad Option") including an Outer Max-Age option with value zero.

      ii) The Server attempts to retrieve the OSCORE Security Contexts associated with the Recipient ID in the 'kid' parameter. If the Server fails to retrieve a OSCORE Security Context with OSCORE Recipient ID corresponding to the 'kid' parameter received, then the Server shall respond with an error response message (e.g. "Unauthorized") including an Outer Max-Age option with value zero.

   b) At step 6 in clause 8.2 of IETF RFC 8613, if the decryption failed then the Server shall respond with an error response message (e.g. "Bad Request) including an Outer Max-Age option with value zero.

   c) If a Server exposes one or more observable Resources, then the Server shall support receiving OSCORE request messages using the Observe option.

3) The Server shall process the OCF CRUDN request message (encapsulated in the OSCORE request message) resulting in OCF CRUDN response message(s). The Server shall treat the value of "subjectuuid" in the credential entry which contains the OSCORE Security Context used to verify and decrypt the OSCORE request message in Step 2 as Client's Device UUID for access control processing. The Server shall treat the connection type as "auth-crypt" for access control processing.

NOTE 2: Multiple OCF CRUDN response messages are only sent in scenarios where the OCF CRUDN Request message is an Observe Request message.

4) The Server shall apply the OSCORE response protection processing of Clause 8.3 of IETF RFC 8613 to each OCF CRUDN response message, using the OSCORE Security Context used to successfully decrypt the OSCORE request (in Step 2 of the present clause).

   At Step 3 in Clause 8.3 of IETF RFC 8613, the Server shall compute the AEAD nonce as described in clause 5.2 of RFC8613 by applying the following steps:

      i) Encode the Partial IV (OSCORE Sender Sequence Number in network byte order) and increment the OSCORE Sender Sequence Number by one.

ii) Compute the OSCORE AEAD nonce from the Sender ID, Common IV, and Partial IV.

The Server shall support sending the OCF CRUDN response messages using the Observe option in OSCORE response messages. If an OCF CRUDN response message uses the Observe option, then the OSCORE response message shall include an Outer Max-Age option with value zero. The Server sends the OSCORE response message to the Client (optionally via OCF Proxies). As with the OSCORE request message, the OSCORE response message shall be delivered over secure transports - see Step 1 for details.

The Server shall update the value of the "ssn" Property in the matching credential entry of the "/oic/sec/cred" Resource to reflect the next value of the OSCORE Sender Sequence Number to be sent to a corresponding Endpoint.

Note: if a Client retrieves the "/oic/sec/cred" Resource over the OSCORE channel, the OSCORE Sender Sequence Number in the header of the OSCORE message is expected to match the "ssn" value within the Resource representation.

5) The Client receives the OSCORE response message. The Client uses the Token (see IETF RFC 7252) in this response message to determine the corresponding OCF CRUDN request message, the OSCORE Security Context and Partial IV in Step 1 of the present clause; see Note 1. The Client shall apply OSCORE response protection processing of Clause 8.3 of IETF RFC 8613 using this OSCORE Security Context and Partial IV. The Client should ignore a success response to an OSCORE-protected request if the response is not an OSCORE response message (indicated by the presence of the OSCORE option).

Q.2.4 Direct provisioning of an OSCORE Security Context

This is a mechanism for establishing an OSCORE Security Context for communication between two Endpoints. All configurable parameters of the OSCORE Security Context are either:

– fixed to the OSCORE-specified default value, or

– directly provisioned by an authorized Client (e.g. OBT) to a credential entry of the "/oic/sec/cred" Resource of the two Endpoints.

The following OSCORE Security Context parameters shall use the default values defined in clause 3.2 of IETF RFC 8613 (this information is not configured by the OBT):

– AEAD Algorithm,

– HKDF,

– Replay Window,

– Master Salt,

– ID Context.

The following OSCORE Security Context parameters and associated Device UUID shall be provisioned to a credential entry of "/oic/sec/cred" of the Device:

– The "subjectuuid" shall be set to the deviceUUID of the other Endpoint to be associated with the OSCORE Security Context.

– The "credtype" shall be set to the value specified for a directly provisioned OSCORE Security Context in Table 21, clause 13.1.1.

– The "privatedata" Property of the credential entry shall be set to the 256-bit secret generated by the provisioning client (e.g. OBT). This value shall be used as the OSCORE Master Secret. Two Endpoints provisioned using this mechanism can communicate securely only if provisioned with identical values for the OSCORE Master Secret.

– The OSCORE Configuration parameters ("oscore") Property shall be present, and shall include the following Properties:

- The OSCORE Sender ID of the OSCORE Security Context is in the "senderid" Property. That value shall be set to the hexadecimal representation of a 56-bit value selected by the provisioning Client (e.g. OBT). When using the mechanism described in the present clause, the first byte of this value is expected to have the value assigned in Table Y for a directly provisioned OSCORE Security Context.
- The OSCORE Recipient ID of the OSCORE Security Context is in the "recipientid" Property. That value shall be set to the hexadecimal representation of a 56-bit value selected by the provisioning Client (e.g. OBT). The first byte of this value is expected to have the value assigned in Table Y for a directly provisioned OSCORE Security Context.

NOTE 2: The values for the OSCORE Sender ID and OSCORE Recipient ID of the OSCORE Security Context for one Device are provisioned as the values for the OSCORE Recipient ID and OSCORE Sender ID of the OSCORE Security Context for the other Device respectively.

On Device powering down, for each such credential entry, the Device shall write the value of corresponding OSCORE Sender Sequence Number as "ssn" Property to non-volatile memory. In event of a crash, devices should apply Appendix B.1.1 of IETF RFC 8613.

########Changes to Clause 13.3.1, Table 21

Table 1 – Properties of the "oic.sec.creds" Property

Property Title	Property Name	Value Type	Value Rule	Mandatory	Access Mode	Device State	Description
Credential ID	credid	UINT16	0 – 64K-1	Yes	RW		Short credential ID for local references from other Resource
Subject UUID	subjectuuid	String	uuid	Yes	RW		A uuid that identifies the subject to which this credential applies or "*" if any identity is acceptable
Role ID	roleid	oic.sec.roletype	-	No	RW		Identifies the role(s) the subject is authorized to assert.
Credential Type	credtype	oic.sec.credtype	bitmask	Yes	RW		Represents this credential's type. 0 – Used for testing 1 – Symmetric pair-wise key 2 – Symmetric group key 4 – Asymmetric signing key 8 – Asymmetric signing key with certificate 16 – PIN or password 32 – Asymmetric encryption key 64 - Directly Provisioned OSCORE Security Context
Credential Usage	credusage	oic.sec.credusagetype	String	No	RW		Used to resolve undecidability of the credential. Provides indication for how/where the cred is used "oic.sec.cred.trustca": certificate trust anchor "oic.sec.cred.cert": identity certificate "oic.sec.cred.rolecert": role certificate "oic.sec.cred.mfgtrustca": manufacturer certificate trust anchor "oic.sec.cred.mfgcert": manufacturer certificate
Public Data	publicdata	oic.sec.pubdatatype	-	No	RW		Credential Type dependent. Public credential information 1:2: ticket, public SKDC values 4, 32: Public key value 8: A chain of one or more certificate
Private Data	privatedata	oic.sec.privdatatype	-	No	-	RESET	Server shall set to manufacturer default
					RW	RFOTM	Set by DOTS after successful OTM
					W	RFPRO	Set by authenticated DOTS or CMS
					-	RFNOP	Not writable during normal operation.
					W	SRESET	DOTS may modify to enable transition to RFPRO.
Optional Data	optionaldata	oic.sec.optdatatype	-	No	RW		Credential Type dependent. Credential revocation status information 1, 2, 4, 32, 64: revocation status information 8: Revocation information

Period	period	String	-	No	RW		Period as defined by IETF RFC 5545. The credential should not be used if the current time is outside the Period window.
Credential Refresh Method	crms	oic.sec.crmtype	array	No	RW		Credentials with a Period Property are refreshed using the credential refresh method (crm) according to the type definitions for "oic.sec.crm".
OSCORE Configuration	oscore	oic.sec.oscoretype	-	No	RW		Contains parameters for use with credentials intended for use with OSCORE. See type definition for "oic.sec.oscoretype".

########Insert this new table in Clause 13.3.1

Table X defines the Properties of "oic.sec.oscoretype".

Table 2 – Properties of the "oic.sec. oscoretype" Property

Property Title	Property Name	Value Type	Value Rule	Access Mode	Mandatory	Description
OSCORE Sender ID	senderid	String	Hexadecimal encoding	RW	No	OSCORE Sender ID for this OSCORE Security Context.
OSCORE Recipient ID	recipientid	String	Hexadecimal encoding	RW	No	OSCORE Recipient ID for this OSCORE Security Context.
OSCORE Sender Sequence Number 1	ssn	Integer		R	No	OSCORE Sender Sequence Number being stored in non volatile memory to handle the loss of mutable security context parameters. See clause Q.2.4
OSCORE Security Context Description	desc	String		RW	No	Description of the usage of this OSCOE Security Context.