OCF “Fargo” – Cloud API for Cloud Services – Core Technology WG

Legal Disclaimer

THIS IS A DRAFT SPECIFICATION DOCUMENT ONLY AND HAS NOT BEEN ADOPTED BY THE OPEN CONNECTIVITY FOUNDATION. THIS DRAFT DOCUMENT MAY NOT BE RELIED UPON FOR ANY PURPOSE OTHER THAN REVIEW OF THE CURRENT STATE OF THE DEVELOPMENT OF THIS DRAFT DOCUMENT. THE OPEN CONNECTIVITY FOUNDATION AND ITS MEMBERS RESERVE THE RIGHT WITHOUT NOTICE TO YOU TO CHANGE ANY OR ALL PORTIONS HEREOF, DELETE PORTIONS HEREOF, MAKE ADDITIONS HERETO, DISCARD THIS DRAFT DOCUMENT IN ITS ENTIRETY OR OTHERWISE MODIFY THIS DRAFT DOCUMENT AT ANY TIME. YOU SHOULD NOT AND MAY NOT RELY UPON THIS DRAFT DOCUMENT IN ANY WAY, INCLUDING BUT NOT LIMITED TO THE DEVELOPMENT OF ANY PRODUCTS OR SERVICES. IMPLEMENTATION OF THIS DRAFT DOCUMENT IS DONE AT YOUR OWN RISK AMEND AND IT IS NOT SUBJECT TO ANY LICENSING GRANTS OR COMMITMENTS UNDER THE OPEN CONNECTIVITY FOUNDATION INTELLECTUAL PROPERTY RIGHTS POLICY OR OTHERWISE. IN CONSIDERATION OF THE OPEN CONNECTIVITY FOUNDATION GRANTING YOU ACCESS TO THIS DRAFT DOCUMENT, YOU DO HEREBY WAIVE ANY AND ALL CLAIMS ASSOCIATED HEREWITH INCLUDING BUT NOT LIMITED TO THOSE CLAIMS DISCUSSED BELOW, AS WELL AS CLAIMS OF DETRIMENTAL RELIANCE.

The OCF logo is a trademark of Open Connectivity Foundation, Inc. in the United States or other countries. *Other names and brands may be claimed as the property of others.

Copyright © 2019 Open Connectivity Foundation, Inc. All rights reserved.

Copying or other form of reproduction and/or distribution of these works are strictly prohibited.

DRAFT

CONTENTS

Copyright Open Connectivity Foundation, Inc. © 2019. All rights Reserved

Figures

Tables

Copyright Open Connectivity Foundation, Inc. © 2019. All rights Reserved

1 Scope

This document defines functional requirements for the OCF Cloud to Cloud Application Programming Interface (API).

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

IETF RFC 2818, *HTTP over TLS,* May 2000
https://tools.ietf.org/html/rfc2818

IETF RFC 6749, *The OAuth 2.0 Authorization Framework,* October 2012
https://tools.ietf.org/html/rfc6749

IETF RFC 6750, *The OAuth 2.0 Authorization Framework: Bearer Token Usage*, October 2012
https://www.rfc-editor.org/info/rfc6750

IETF RFC 7519, *JSON Web Token (JWT),* May 2015,
https://tools.ietf.org/html/rfc7519

IETF RFC 8414, *OAuth 2.0 Authorization Server Metadata,* June 2018
https://tools.ietf.org/html/rfc8414

IETF RFC 5785, *Defining Well-Known Uniform Resource Identifiers (URIs)*, April 2010
https://tools.ietf.org/html/rfc5785

ISO/IEC 30118-1:2018 Information technology -- Open Connectivity Foundation (OCF) Specification -- Part 1: Core specification
https://www.iso.org/standard/53238.html
Latest version available at: https://openconnectivity.org/specs/OCF_Core_Specification.pdf

ISO/IEC 30118-2:2018 Information technology -- Open Connectivity Foundation (OCF) Specification -- Part 2: Security specification
https://www.iso.org/standard/74239.html
Latest version available at: https://openconnectivity.org/specs/OCF_Security_Specification.pdf

OCF Device to Cloud Services Specification, *Open Connectivity Foundation Device to Cloud Services Specification*,
Latest version available at:
https://openconnectivity.org/specs/OCF_Cloud_Specification.pdf

OCF Cloud API for Cloud Services https://github.com/openconnectivityfoundation/core-extensions/blob/ocfcloud-openapi/swagger2.0/oic.r.cloudopenapi.swagger.json

OpenAPI 2.0, *fka Swagger RESTful API Documentation Specification*, Version 2.0
https://github.com/OAI/OpenAPI-Specification/blob/master/versions/2.0.md

DRAFT

3 Terms, definitions, and abbreviated terms

3.1 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/IEC 30118-1:2018 and ISO/IEC 30118-2:2018 and the following apply.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

– ISO Online browsing platform: available at https://www.iso.org/obp

– IEC Electropedia: available at http://www.electropedia.org/

3.2 Abbreviated terms

3.2.1
API
Application Programming Interface

3.2.2
JWT
JSON Web Token

4 Document conventions and organization

4.1 Conventions

In this document a number of terms, conditions, mechanisms, sequences, parameters, events, states, or similar terms are printed with the first letter of each word in uppercase and the rest lowercase (e.g., Network Architecture). Any lowercase uses of these words have the normal technical English meaning.

4.2 Notation

In this document, features are described as required, recommended, allowed or DEPRECATED as follows:

Required (or shall or mandatory)(M).

– These basic features shall be implemented to comply with Core Architecture. The phrases "shall not", and "PROHIBITED" indicate behaviour that is prohibited, i.e. that if performed means the implementation is not in compliance.

Recommended (or should)(S).

– These features add functionality supported by Core Architecture and should be implemented. Recommended features take advantage of the capabilities Core Architecture, usually without imposing major increase of complexity. Notice that for compliance testing, if a recommended feature is implemented, it shall meet the specified requirements to be in compliance with these guidelines. Some recommended features could become requirements in the future. The phrase "should not" indicates behaviour that is permitted but not recommended.

Allowed (may or allowed)(O).

– These features are neither required nor recommended by Core Architecture, but if the feature is implemented, it shall meet the specified requirements to be in compliance with these guidelines.

DEPRECATED.

– Although these features are still described in this document, they should not be implemented except for backward compatibility. The occurrence of a deprecated feature during operation of an implementation compliant with the current document has no effect on the implementation’s operation and does not produce any error conditions. Backward compatibility may require that a feature is implemented and functions as specified but it shall never be used by implementations compliant with this document.

Conditionally allowed (CA)

– The definition or behaviour depends on a condition. If the specified condition is met, then the definition or behaviour is allowed, otherwise it is not allowed.

Conditionally required (CR)

– The definition or behaviour depends on a condition. If the specified condition is met, then the definition or behaviour is required. Otherwise the definition or behaviour is allowed as default unless specifically defined as not allowed.

Strings that are to be taken literally are enclosed in "double quotes".

Words that are emphasized are printed in italic.

5 Overview

5.1 Introduction

This document defines the OCF Cloud API for Cloud Services. In this document "origin Cloud" refers to the OCF Cloud or the 3rd party Cloud through which the user works with his OCF Devices, "target Cloud" refers to the OCF Cloud to which OCF Servers (OCF Devices) are connected which the user wants to control via the "origin Cloud".

An OCF Device is a collection of Resources, each Resource being an OpenAPI 2.0 defined object that represents a physical property or characteristic of the Device (e.g. temperature sensed, light colour, power on switch). The Device itself has an associated Device Type that provides an indication of what the Device is, for example a Light is represented as a Device Type of "oic.d.light".

Please see Figure 1 for a representation of the target architecture.

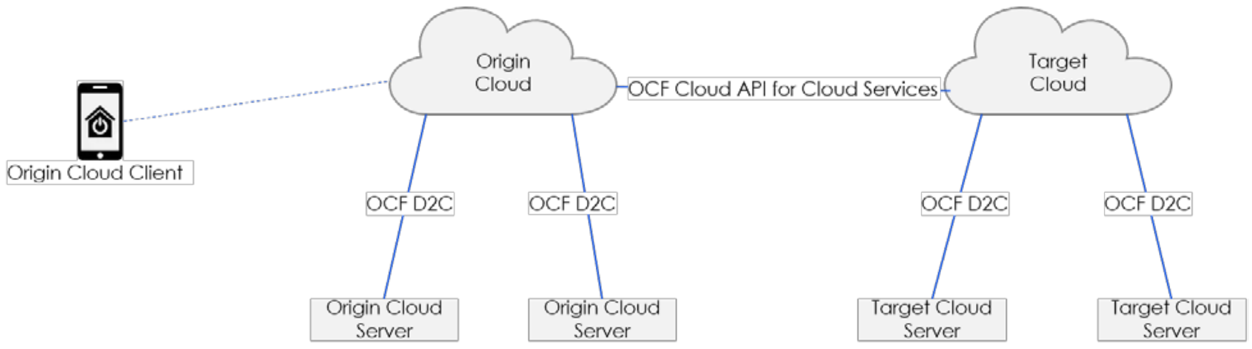


Figure 1 – OCF Cloud Overview

The OCF Cloud API for Cloud Services supports the following cases:

- Account Linking API (clause 7)
  - Initial Account Linking
  - Removal of linked account
- Devices API (clause 8)
  - Retrieval of all Devices associated with a User (clause 8.3)
  - Retrieval of a single Device associated with a User (clause 8.4)
  - Retrieval of a single Resource (clause 8.5)
  - Update of a single Resource (clause 8.6)
- Events API (clause 9)
  - Subscription to an event: establishment of a subscription
  - Notification: event generated on an established subscription

5.2 General OCF Cloud API for Cloud Services Elements

The OCF Cloud API for Cloud Services is a RESTful API over HTTPS (IETF RFC 2818). The API is defined using OpenAPI 2.0.

The ”origin Cloud” communicates with the “target Cloud” using the domain name or URI it has obtained from the initial OAuth 2.0 (IETF RFC 6749) Client Setup, covered in clause 7. Communication between OCF Devices and OCF Clouds is defined in the OCF Device to Cloud Services Specification.

All URIs presented as "href" Properties in a Link in response to a Client or Cloud discovery are in the form "/deviceid/resourceHref".

All requests include the bearer token for the user in question obtained via standard OAuth2.0 (IETF RFC 6749) mechanisms.

Any query parameters received by an "origin Cloud" in a request from an OCF Client are passed through clean (i.e. are part of the URI) in any request that is sent to a target Cloud.

Each request may contain an optional HTTP Correlation-ID header, which carries a unique identifier value that provides a reference to a particular transaction or event chain in the target cloud.

All requests include an HTTP Accept header. All requests or responses that carry content include an HTTP Content-Type header. At a minimum media-types "application/json" and "application/vnd.ocf+cbor" are supported. If the recipient of a request cannot provide a response that is encoded according to the content of the Accept header, then a HTTP 406 (not acceptable) response shall be sent. On reception of a 406 response the originator of the request may re-attempt the request using an alternative Content-Type if supported.

5.3 Cloud to Cloud Operational Overview

5.3.1 Introduction

This clause provides an informative overview of the flows that are enabled by the detailed API defined in clauses 6, 7, 8, and 9. Clause 5.3 provides references to the applicable clauses within this document that define the API specifics.

5.3.2 Conceptual Architecture

Figure 2 describes the overall conceptual architecture.

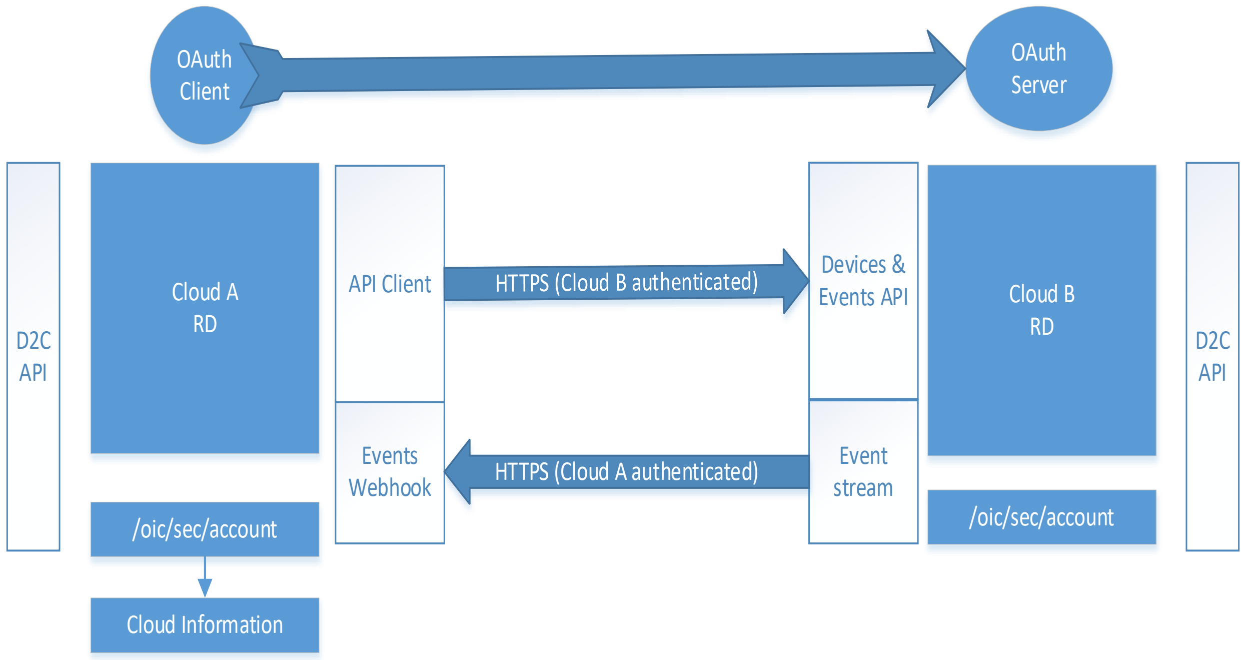


Figure 2 – Conceptual Architecture

5.3.3 Authorizing Cloud Connectivity

Consider a user who has accounts on two distinct, separately owned clouds, and devices associated with each of those accounts on those clouds. The user wants to have a unified view of all of their devices from a single client rather than having a client per cloud. The user via the client they want to use for all devices indicates to the directly connected cloud ("origin Cloud") that they want to link this account with an account on the other cloud ("target Cloud"). This initiates a standard OAuth2.0 Authorization Code Grant Type flow, see IETF RFC 6749, clause 1.3.1. Application of this flow is described in clause 7.

5.3.4 Synchronization of Users set of Devices

After completion of the Authorization Code Grant Type flow from clause 5.3.3 the "origin Cloud" (that is the cloud to which the user is connected) is authorized to use the Device API to obtain on behalf of the user the complete list of devices hosted on the "target Cloud" for which the user has access. The API is described in clause 8, and the flow is further illustrated in clause A.4.

The result of the invocation of the Device API is a complete set of device information that may then be provided in a response to a RETRIEVE on "/oic/res" from the "origin Cloud".

5.3.5 Keeping Up-to-Date: Notifications of changes on other Clouds

Once the set of devices has been obtained, the "origin Cloud" can subscribe to the events to which it is interested across the user's complete device set ("/devices"), or per device in that set ("/devices/{deviceid}"). See clause 9 for details of the API itself.

The subscription to "/devices" enables the "origin Cloud" to be notified whenever a new device is added or an existing device removed from the "target Cloud".

The subscription to "/devices/{deviceid}" enables the "origin Cloud" to be notified whenever there is a change in the state of a device (e.g. it has de-registered).

When a new Device registers on the "target Cloud", and a subscription exists, then a notification is sent to the "origin Cloud" with an event_type of "devices_registered" and a payload which contains the "/oic/d" of the newly onboarded device. The "origin Cloud" may then RETRIEVE the Links exposed by the newly added device using "/devices/{deviceid}" where "deviceid" was provided in the payload of the notification. See clause A.10 for a flow illustrating this interaction.

5.3.6 Handling of Requests and Responses for Connected Devices

From the perspective of the Client connected to the "origin Cloud" there is no distinction between devices and their resources hosted by the "origin Cloud" itself and devices and their resources that are hosted by an OCF Cloud API for Cloud Services connected cloud.

Thus all requests for a target resource are formed using the mechanisms described in the OCF Device to Cloud Services Specification.

The "origin Cloud" identifies the host Cloud for the requested Resource via the instance of "/oic/sec/account" for the "deviceid" that is in the request URI. The request is then effectively proxied to the "target Cloud" via the "/devices/{deviceid}/{resourcehref}" API exposed by the "target Cloud" (see clause 8.5 and 8.6). Any query parameters received over the device to cloud connection are included in the URI unaltered. The content-type of the payload in the request or response is honoured. See clauses A.6 and A.7 for illustrative flows of this mechanism for both RETRIEVE and UPDATE cases.

6 Authentication & Authorization

All requests to the “target Cloud” are transferred over an HTTPS connection using server-authenticated TLS1.2 connection and are protected by OAuth2.0 JWT Bearer Tokens, see IETF

RFC 7519. The "origin Cloud" uses the "Bearer" authentication scheme inside the "Authorization" request header field to transmit the access token, as per IETF RFC 6750 clause 2.1. For definition of the "Authorization" request header field, see IETF RFC 2818.

The OAuth2.0 server issuing JWT tokens must also serve the Authorization Server Metadata described in IETF RFC 8414 clause 2, published at the location that is ".well-known" according to IETF RFC 5785. Parsing of the OAuth2.0 JWT token identifies the user identity and the client delegating the request to the "target Cloud".

On the OCF Server side there is no distinction between requests forwarded from the "origin Cloud" and requests coming via the "target Cloud".

7 Account Linking API

7.1 General

The account linking API is the mechanism by which Devices hosted on behalf of a user by the "target Cloud" are linked with a user identity on the "origin Cloud". Account linking is established solely between the "origin Cloud" and the "target Cloud"; devices from the "target Cloud" cannot be proxied further through the OCF Cloud API for Cloud Services.

The OAuth Client of the "origin Cloud" has to be registered with the "target Cloud" as a prerequisite to initiating the Authorization Code Grant Type flow, which allows the user to link his "origin Cloud" account with the "target Cloud". This process is named OAuth Application registration and is beyond the scope of this specification. Successful registration of the OAuth "origin Cloud" Client in the "target Cloud" relies on the two entities establishing trust and obtaining the required client properties (e.g. client id, client secret, allowed redirect URIs). See IETF RFC 6749, clause 2.

The linking is then achieved via the use of an OAuth2.0 Authorization Code Grant Type. Part of the linking process is the end-user consent, which is very important in cross-domain identity federation, ensuring that a malicious OAuth 2.0 Client cannot obtain authorization without the awareness and explicit consent of the resource owner (that is the user) of the "target Cloud". The "target Cloud" should present to the user linking the account the precise scope of authorization information being requested by the Client. Details about scopes are available in clause 7.2. After the user's consent and subsequent authorization code exchange, the JWT bearer access and refresh tokens are obtained from the “target Cloud” by the “origin Cloud”, following the format and Content Type in IETF RFC 6750 clause 4. The JWT bearer access token identifies a user identity on the “target Cloud”.

Presence of the state query parameter, see IETF RFC 6749 clause 4.1.1 is required. State is an opaque value used by the "origin cloud" Client to maintain state between the request and the callback during the account linking process, see clause A.3.

Once such a JWT bearer token has been acquired, the "origin Cloud" links the OAuth2.0 access and refresh token with its known local userid. The user who linked his “target Cloud” account with the “origin Cloud” account is from this moment able to request all his devices through the “origin Cloud”, because the “origin Cloud” can make requests to the “target Cloud” on behalf of the “target Cloud” user account. However, the "origin Cloud" can make requests only in the scope granted by the user during the consent screen.

On initial acquisition of the token the "origin Cloud" may use the Device API to retrieve the Device details for all Devices in scope of the bearer token.

If the "origin Cloud" supports the behaviour defined in the OCF Device to Cloud Services Specification, then once the "origin Cloud" has the set of Devices from the "target Cloud" it creates an instance of "/oic/sec/account" per Device. The optional Property "cloudid" in "/oic/sec/account" is set to the OCF Cloud UUID of the "target Cloud" available in the Common Name field of the End-Entity certificate used to secure the HTTPS OCF Cloud API for Cloud Services Endpoint. If the

Property is missing, empty, or contains the same value as OCF Cloud UUID of the "origin Cloud", then the Device is local to the "origin Cloud".

The “origin Cloud” might use the Events API to establish an "observe" relationship with the Device(s) on the “target Cloud”; such that addition or deletion of Devices on the “target Cloud” can be correctly reflected in the “origin Cloud”. When the Device is unregistered from the "target Cloud", that Device is no longer accessible via the "origin Cloud". When the JWT bearer token obtained from the "target Cloud" expires and the refresh token is still valid, the “origin Cloud” can ask for a new access token through the OAuth2.0 token endpoint of the “target Cloud". Whenever the refresh token expires, is not available, or the access token cannot be obtained, the "origin Cloud" removes all associations with the Devices hosted by the "target Cloud".

It is recommended that the "origin Cloud" establishes an "event stream" for each Device that is hosted on the "target Cloud" by using the subscription mechanism described in clause 9.6.

7.2 OAuth 2.0 OCF Cloud API for Cloud Services Scopes

The OCF Cloud API for Cloud Services defines core set of OAuth Access Token Scopes, see IETF RFC 6749. List of scopes defined in Table 1 must be supported together with its description by each implementation and presented to the user during the account linking process by the OAuth2.0 server of the "target Cloud". The "target Cloud" user shall see an appropriate description on the consent screen and give an explicit consent that the "origin Cloud" requesting the token is authorized to act on behalf of the user in the boundary of obtained scopes.

Table 1 – OCF Cloud API for Cloud Services API OAuth Client Scopes

Scope name	**Scope description** "The application will be able to:"
r:deviceinformation:*	Read basic device information
r:resources:*	Retrieve the data from Resources on your devices
w:resources:*	Update the data in Resources on your devices
w:subscriptions:*	Subscribe to events

Table 2 details the scopes that are applicable per API endpoint.

Table 2 – Required scopes per API endpoint

API Endpoint	**HTTP Request Type**	**Required scopes**
/api/v1/devices	GET	r:deviceinformation:*
/api/v1/devices?content=all	GET	r:deviceinformation:* r:resources:*

/api/v1/devices/{deviceid}	GET	r:deviceinformation:*
/api/v1/devices/{deviceid}?content=all	GET	r:deviceinformation:* r:resources:*
/api/v1/devices/{deviceid}/{resourcehref}	GET	r:resources:*
	POST	w:resources:*
/api/v1/devices/subscriptions	POST	r:deviceinformation:* w:subscriptions:*
	DELETE	r:deviceinformation:* w:subscriptions:*
/api/v1/devices/{deviceid}/subscriptions	POST	r:deviceinformation:* w:subscriptions:*
	DELETE	r:deviceinformation:* w:subscriptions:*
/api/v1/devices/{deviceid}/{resourcehref}/subscriptions	POST	r:resources:* w:subscriptions:*
	DELETE	r:resources:* w:subscriptions:*

List of scopes is not finite; the "target Cloud" can derive from the defined list of scopes following the wildcard scheme. As an example, take the derived scope "w:resources:schedules:*". If the "target Cloud" user give the consent to the "origin Cloud" to "w:resources:schedules:*" - define new schedules for the device, "origin Cloud" can do it automatically on behalf of the user. But if the consent was given only for this scope, "origin Cloud" cannot for example update scenes - "w:resources:scenes:*". In case the user gives consent to the "origin Cloud" to "w:resources:*" which must be by default supported, consent applies to all derived scopes, so the user is able to do both, update scenes and define new schedules for the device.

The request to get the OAuth Access Token shall contain only scopes from Table 1. If needed, the "target Cloud" can extend the OAuth Access Token with pure vendor specific scopes, but they shall not be present in the OAuth Access Token request.

8 Devices API

8.1 Introduction

The Devices API supports the ability to retrieve and interact with the OCF Devices that are within the scope of the provided bearer token.

8.2 Parameters Supported

Table 3 lists the parameters that may be provided within the Device API.

Table 3 – Parameters used in the Device API

Friendly Name	Parameter Name	Location	Description
Accept	Accept	Header	An Accept request HTTP header advertises which content types, expressed as MIME types, the client is able to understand. The resource server then selects one of the proposal and informs the client of its choice with the Content-Type response header.
Correlation ID	Correlation-ID	Header	A Correlation ID, also known as a Transit ID, is a unique identifier value that is attached to requests and messages that allows reference to a particular transaction or event chain.
All Content	content=[base, all]	Query String Parameter	Indicates to the recipient that the response payload shall be the resolved (i.e. resource representation) Link and not the Link itself. Default is base.
Request Payload	payload	Body	Request payload as defined by OCF for the target Resource Type.

8.3 Retrieve All Devices

8.3.1 Summary

This request is sent from the "origin Cloud" to the "target Cloud" in order to obtain information on all the Devices that are registered for the user that is in scope on the "target Cloud".

A request to this API may be triggered on the "origin Cloud" by the completion of account linking. Where the Cloud supports the behaviour defined in the OCF Device to Cloud Services Specification this may also be triggered by reception of a RETRIEVE to "/oic/res" of the Cloud Resource Directory from an OCF Client.

Table 4 provides a summary of the API.

Table 4 – Retrieve All Devices API Summary

HTTP Request Type	URI	Parameters	Response Code	Response Payload
GET	/api/v1/devices	content=[base, all], Correlation-ID, Accept	200	See clause B.1 OpenAPI 2.0 Definition - array of /definitions/Device
			400, 401, 403	Diagnostic payload containing detailed reason.

8.3.2 Request and Response Payload

There is no required payload in the request. The required response payload shall be an array of objects; each object shall contain Properties from "/oic/d" that are relevant for Cloud operation, a status Property ("status") that indicates whether the Device is online or offline, plus an array of Links (as defined for "/oic/res") for the Resources exposed by the specific Device. The minimum set of Resources that are exposed depends on the OCF Device Type of the Device; this shall be in comformance with the requirements for Resource Publication in the OCF Device to Cloud Services Specification.

If the request includes "content=all" (analogous to a batch retrieval of /oic/res in the proximal network) then instead of an array of Links to the hosted Resources, the response payload shall be an array of the representations of the Resources themselves that are exposed for each Device that is available, which includes the Device Information sent in the case where this parameter is not provided. This is illustrated in the examples provided for the Device API in Annex B. See also the definition of a batch response in ISO/IEC 30118-1:2018.

8.3.3 Responses

A 200 response is provided in a success case. The payload contains information for all Devices that are in the scope of the bearer token.

A 400 response indicates that the request was malformed or badly constructed.

A 401 response indicates that the request is unauthorized (likely an invalid bearer token)

A 403 response indicates that the requestor is known however the scope if the request is forbidden.

8.4 Retrieve One Device

8.4.1 Summary

This request is sent from the "origin Cloud" to the "target Cloud" in order to obtain information on a specific Device that is registered for the user that is in scope on the "target Cloud".

A request to this API may be triggered on the "origin Cloud" by the reception of a notification that a new Device has been added to a partner cloud, or as part of the flow following account linking. Where the Cloud supports the OCF Device to Cloud Services Specification, a request to this API may also be triggered by reception of a RETRIEVE to "/oic/res" of the Cloud Resource Directory from an OCF Client with a query parameter that specifies a particular deviceid ("anchor").

Table 5 provides a summary of the API.

Table 5 – Retrieve One Device API Summary

HTTP Request Type	URI	Parameters	Response Code	Response Payload
GET	/api/v1/devices/**{deviceid}**	content=[base, all], Correlation-ID, Accept	200	See B.1 OpenAPI 2.0 Definition - /definitions/Device
			400, 401, 403, 404	Diagnostic payload containing detailed reason.

8.4.2 Request and Response Payload

The deviceid in the URI of the request is the same as the "di" Property from /oic/d of the target OCF device.

The response payload shall be an object containing the mandatory Device information as defined in clause 8.3.2.

If the request includes "content=all" (analogous to a batch retrieval of /oic/res in the proximal network) then instead of an array of Links to the hosted Resources, the response payload shall be an array of the representations of the Resources themselves that are exposed by the Device, which includes the Device Information sent in the case where this parameter is not provided. This is illustrated in the examples provided for the Device API in Annex B. See also the definition of a batch response in ISO/IEC 30118-1:2018.

.

8.4.3 Responses

A 200 response is provided in a success case. The payload contains information for the requested Device.

A 400 response indicates that the request was malformed or badly constructed.

A 401 response indicates that the request is unauthorized (likely an invalid bearer token)

A 403 response indicates that the requestor is known however the scope of the request is forbidden.

A 404 response indicates that the indicated "deviceid" is no longer available on this cloud

8.5 Retrieve Specific Resource

8.5.1 Summary

This request is sent from the "origin Cloud" to the "target Cloud" in order to obtain information on a specific Resource that is exposed by a Device that is registered for the user that is in scope on the "target Cloud".

Where the Cloud supports the OCF Device to Cloud Services Specification this may be triggered by reception of a RETRIEVE to a URI exposed by a Link in the Cloud Resource Directory from an OCF Client.

Table 6 provides a summary of the API.

Table 6 – Retrieve Specific Resource API Summary

HTTP Request Type	**URI**	**Parameters**	**Response Code**	**Response Payload**
GET	/api/v1/devices/**{deviceid}**/**{resourcehref}**	Correlation-ID, Accept	200	Response payload as defined by OCF for the target Resource Type

			400, 401, 403, 404	Diagnostic payload containing detailed reason.
			503	Diagnostic payload containing detailed reason.
			504	Retry-After header and diagnostic payload containing the detailed reason.

8.5.2 Request and Response Payload

The deviceid in the URI in the request is the same as the "di" Property from "/oic/d" of the target OCF device. The "resourcehref" in the URI is the same as the "href" Link Parameter for the target Resource instance.

The response payload shall be as defined by OCF for the Resource being received, or as defined by the vendor if the Resource is a 3rd party Resource.

The content-type of the response payload received from the target server is honoured; that is the content and payload as received by the "target Cloud" is proxied unaltered in the response. Thus for example in the case where the target server is an OCF Device the content type would be "application/vnd.ocf+cbor".

8.5.3 Responses

A 200 response is provided in a success case. The payload in the response is as defined in http://oneiota.org for the target Resource Type.

A 400 response indicates that the request was malformed or badly constructed.

A 401 response indicates that the request is unauthorized (likely an invalid bearer token)

A 403 response indicates that the requestor is known however the scope of the request is forbidden.

A 404 response indicates that the indicated "deviceid" is no longer available on this cloud

A 503 response indicates that the service on the "target Cloud" is unavailable for the reason indicated in the diagnostic payload.

A 504 response indicates that the target Device is registered at the "target Cloud", however the Device itself is unavailable, offline, or otherwise unreachable. The response should include a Retry-After header containing the time after which the request may be re-attempted. Additional information is indicated in the diagnostic payload.

8.6 Update a Resource on a Device

8.6.1 Summary

This request is sent from the "origin Cloud" to the "target Cloud" in order to update information contained within a specific Resource exposed by a Device that is registered for the user that is in scope on the "target Cloud".

Where the Cloud supports the OCF Device to Cloud Services Specification a request to this API may be triggered by reception of an UPDATE to a URI exposed by a Link in the Cloud Resource Directory from an OCF Client.

Table 7 provides a summary of the API.

Table 7 – Update Resource API Summary

HTTP Request Type	**URI**	**Parameters**	**Response Code**	**Response Payload**
POST	/api/v1/devices/**{deviceid}**/**{resource href}**	payload, Correlation-ID, Accept	200	Optional resource representation
			400, 401, 403, 404, 415	Diagnostic payload containing detailed reason.
			503	Diagnostic payload containing detailed reason.
			504	Retry-After header and diagnostic payload containing the detailed reason..

8.6.2 Request and Response Payload

The deviceid in the URI in the request is the same as the "di" Property from /oic/d of the target OCF device. The resourcehref in the URI is the same as the "href" Link Parameter for the target Resource instance.

The response payload shall be as defined by OCF for the Resource being received, or as defined by the vendor if the Resource is a 3rd party Resource.

The Content-Type of the request is defined in an HTTP Content-Type header. In the case that the request was initiated by another OCF Device, CoAP content-format header value is mapped to the HTTP Content-Type header to the "target Cloud". If the value is not present, the "target Cloud" will forward the request as-is. Thus for example in the case where the origin client is an OCF Device the CoAP content-format option would be "application/vnd.ocf+cbor", which is passed to the "target Cloud" as an HTTP Content-Type header.

8.6.3 Responses

A 200 response is provided in a success case. The payload may optionally contain the representation of the Resource that was updated.

A 401 response indicates that the request is unauthorized (likely an invalid bearer token)

A 403 response indicates that the requestor is known however the scope if the request is forbidden.

A 415 response indicates unsupported media type specified in the Content-Type header.

A 403 response indicates an invalid bearer token.

A 404 response indicates that the indicated "deviceid" is no longer available on this cloud

A 503 response indicates that the service on the "target Cloud" is unavailable for the reason indicated in the diagnostic payload.

A 504 response indicates that the target Device is registered at the "target Cloud", however the Device itself is unavailable, offline, or otherwise unreachable. The response should include a Retry-After header containing the time after which the request may be re-attempted. Additional information is indicated in the diagnostic payload.

9 Events API

9.1 Introduction

The Events API supports the ability for an interested party to subscribe to events and subsequently receive notifications for those events. The events can be at the Resource level (like a CoAP observe) or at a more system level (such as for a change in the set of known Devices).

The Event's API make use of a webhook mechanism whereby the “target Cloud” can notify the “origin Cloud” when a new event has occurred on the “target Cloud” or any OCF Device linked with the “target Cloud”. This event stream can be started by sending the initial subscription request to the “target Cloud”, on behalf of the user, specifying "eventtypes", "eventsurl" - the endpoint to which events are sent and the "signingSecret" - to confidently verify whether requests from the "target Cloud" are authentic. The mechanism how the "signingSecret" shall be used is specification in the clause 9.2.

Subscription and events are done on behalf of the "target Cloud" user, who linked his account during the account linking process. As a response to the initial subscription request, a Subscription ID identifying this user's event stream is returned. The Subscription ID is the unique string of type UUID, which is created and persisted by the "target Cloud". The created ID is then part of each event sent to the configured "eventsUrl".

The Subscription ID is used also to DELETE this subscription.

After the subscription is successful, the state of the subscription is updated and the first event is sent with the actual state of the model. The subsequent events represent changes which have occurred since the time of the last state update of the model.

Following “origin Cloud's” successful subscription to events of the “target Cloud”, the “target Cloud” can start sending notifications only after it establishes new server-authenticated TLS1.2 connection to a machine hosting "eventsUrl" as specified by “target Cloud”.

The Event stream authorized by the "target Cloud" user sends only events related to devices and system changes he is authorized to see.

Whenever a "target Cloud" with active subscriptions has any device state errors, it shall recover from such errors by sending an event representing the current state of the model to its active subscribers. The subsequent events represent changes which have occurred since the time of the last state update of the model.

The "origin Cloud" is always informed about subscription cancellation through the Event stream as well as through the state available within the GET API Subscription endpoint response. Cancellation can be done by either, the "target Cloud" and the "origin Cloud".

9.2 Events Authentication

HMAC signatures are a way to sign the event data using the "signingSecret" that only the "origin Cloud" and "target Cloud" know. The "signingSecret" shall be created by the "originCloud" and send within the subscription request as defined in the clause 9.4.1. The "target Cloud" shall after successful subscription sign each event using HMAC-SHA256 hashing algorithm, following the formula from the clause 9.2.1. The calculated signature is afterwards attached as the "Event-Signature" header with each event request sent to the "origin Cloud".

The signature is then used by the "origin Cloud" to verify the legitimacy of the source and data itself. When the event is received by the "origin Cloud" it uses its stored secret and the event to generate its own HMAC-SHA256 signature using the formula from the clause 9.2.2 to compare with the value from the "Event-Signature" header.

When the secret and event are the same on both sides then the HMAC signature will match. This match proves the authenticity of the request and data.

Detailed flow overview can be visible on the figures A.8.2, A.9.2, A.10.2 and A.11.2.

9.2.1 Create Event Signature

1) Get the current timestamp in the Unix time format

2) Concatenate the "Content-Type", the "Event-Type", the "Subscription-ID", the "Sequence-Number", the "Event-Timestamp" header values and the event body together, using a colon as a delimiter.

3) Hash the resulting string, using the "signingSecret" as a key using the HMAC-SHA256 hashing algorithm, and taking the hex digest of the hash.

4) Include the resulting signature to the "Event-Signature" header of the event and timestamp to the "Event-Timestamp" header

9.2.2 Verify the Event Signature

1) Extract the timestamp from the "Event-Timestamp" header and make sure that the request occurred recently

2) Concatenate the "Content-Type", the "Event-Type", the "Subscription-ID", the "Sequence-Number", the "Event-Timestamp" header values and the event body together, using a colon as a delimiter.

3) Hash the resulting string, using the "signingSecret" as a key using the HMAC-SHA256 hashing algorithm and take the hex digest of the hash.

4) Compare the resulting signature to the "Event-Signature" header of the received event

9.3 Parameters Supported

Table 8 lists the parameters that may be provided within the Events API.

Table 8 – Parameters used in the Events API

Friendly Name	Parameter Name	Location	Description
Accept	Accept	Header	An Accept request HTTP header advertises which content types, expressed as MIME types, the client is able to understand. The resource server then selects one of the proposal and informs the client of its choice with the Content-Type response header. Each event sent to the defined "eventsUrl" is then using this Accepted content type.
Correlation ID	Correlation-ID	Header	A Correlation ID, also known as a Transit ID, is a unique identifier value that is attached to requests and messages that allows reference to a particular transaction or event chain.
Content Type	Content-Type	Header	The Content-Type header is used to indicate the media type of the resource. In responses, a Content-Type header tells the client what the content type of the returned content actually is.

9.4 Events API subscription and notification request and response payload definitions

9.4.1 Subscription request

B.1 OpenAPI 2.0 Definition (/definitions/SubscribeRequest) provides a definition of the payload contained within the subscription request. The Properties that are contained with the schema definition are further defined in Table 9.

Table 9 – Subscription Request Properties

Property Name	Value type	Mandatory	Description
eventsUrl	URI	Y	URI to which events are to be sent
eventTypes	array of enum	Y	Event type(s) for which the subscription is targeted

signingSecret	String of length 32	Y	Secret used to create HMAC signature for each event

Figure 3 is an example of such a payload.

```
{
"eventsurl": https://mynotificationuri,
"eventtypes": ["resource-contentchanged"],
"signingSecret": "DVDUEBe5nciVSXU85BPxrAjSsHenTzWY"
}
```

Figure 3 – Subscription Request Example

9.4.2 Subscription response

The definition of the response response to a subscription request is in clause B.1 OpenAPI 2.0 Definition (/definitions/SubscribeResponse). The Properties that are contained with the schema definition are further defined in Table 10.

Table 10 – Subscription Response Properties

Property Name	Value type	Mandatory	Description
subscriptionId	uuid	Y	Identity of the subscription. May be mapped from other protocols if a unique identifier exists. Note this cannot be mapped from a CoAP Token as the Token in CoAP is Client-local in scope (i.e. not guaranteed unique beyond the Client issuing the request).

Figure 4 is an example of such a payload.

```
{
"subscriptionId": "1eeb465c-5e8d-4305-a366-bbf035fff671"
}
```

Figure 4 – Subscription Response Example Payload

9.4.3 Notification request population

All events sent to the "eventsUrl" defined during the subscription are populated with event metadata. Event metadata is transferred to the "origin Cloud" as HTTP Headers together with the event payload. All event metadata is required.

Table 11 – Event HTTP Headers description

Event metadata	Description
Correlation-ID	A Correlation ID, also known as a Transit ID, is a unique identifier value that is attached to requests and messages that allows reference to a particular transaction or event chain.
Content-Type	Indicates the media type of the event payload
Event-Type	Type of the event
Subscription-ID	Subscription identifier for which this event is being sent
Sequence-Number	Sequence number of the event; first event starting with number 0
Event-Timestamp	Time when the event occurred in standard Unix time format
Event-Signature	HMAC-SHA256 signature proving the authenticity of the request and data. See 9.2 Events Authentication

The format of the payload in a notification request depends on the event for which the subscription was created.

Table 12 lists the format of the payload per "eventType" that may be received.

Table 12 – Event type to content format map

Event Type	Event API Endpoint	Payload
subscription_canceled	/api/v1/devices/subscriptions /api/v1/devices/{deviceid}/subscriptions	Not present

	/api/v1/devices/{deviceid}/{resourcehref}/subscriptions	
devices_registered	/api/v1/devices/subscriptions	See B.1 OpenAPI 2.0 Definition - /definitions/DevicesRegisteredEvent
devices_unregistered	/api/v1/devices/subscriptions	See B.1 OpenAPI 2.0 Definition - /definitions/DevicesUnregisteredEvent
devices_online	/api/v1/devices/subscriptions	See B.1 OpenAPI 2.0 Definition - /definitions/DevicesOnlineEvent
devices_offline	/api/v1/devices/subscriptions	See B.1 OpenAPI 2.0 Definition - /definitions/DevicesOfflineEvent
resource_contentchanged	/api/v1/devices/{deviceid}/{resourcehref}/subscriptions	Payload as received from the Device providing the representation
resources_published	/api/v1/devices/{deviceid}/subscriptions	See B.1 OpenAPI 2.0 Definition - /definitions/ResourcePublishedEvent
resources_unpublished	/api/v1/devices/{deviceid}/subscriptions	See B.1 OpenAPI 2.0 Definition - /definitions/ResourceUnpublishedEvent

9.5 Subscribe and Unsubscribe to Devices level events, retrieval of the subscription details

9.5.1 Summary

Note: this request is sent from the "origin Cloud" to the "target Cloud". This API is used when changes to the set of Devices that are exposed are required to be notified.

A POST request establishes the subscription; a GET provides details about the subscription; a DELETE request removes the subscription.

Table 13 provides a summary of the API.

Table 13 – Subscription to /devices API Summary

HTTP Request Type	URI	Parameters	Response Code	Response Payload
POST	/api/v1/devices/subscriptions	Correlation-ID, Accept, Content-Type	201	See B.1 OpenAPI 2.0 Definition - /definitions /Subscribe Response
			400, 401, 403	
DELETE	/api/v1/devices/subscriptions**/{subscriptionid}**	Correlation-ID	202	
			400, 401, 403, 404	

9.5.2 Request and Response Payload

The "subscriptionid" in the URI for the DELETE case is the "subscriptionid" that is returned in the response of the subscription POST request.

The response payload for the subscription POST request shall be as defined in clause 9.4.2

9.5.3 Responses

A 201 response is provided in a success case.

A 202 response indicates that the subscription was marked for cancellation.

A 401 response indicates that the request is unauthorized (likely an invalid bearer token).

A 400 response indicates that the request was malformed or badly constructed.

A 403 response indicates an invalid bearer token.

A 404 response indicates the subscription was not found.

9.6 Subscribe and Unsubscribe to Device level events, retrieval of the subscription details

9.6.1 Summary

Note: this request is sent from the "origin Cloud" to the "target Cloud". This API is used when the "origin Cloud" wants to effectively receive an "event stream" of notifications from all observable Resources that exist for a specific Device on the "target Cloud".

A POST request establishes the subscription; a GET provides details about the subscription; a DELETE request removes the subscription

Table 14 provides a summary of the API.

Table 14 – Subscription to Single Device API Summary

HTTP Request Type	URI	Parameters	Response Code	Response Payload
POST	/api/v1/devices/**{deviceid}**/subscriptions	Correlation-ID, Accept, Content-Type	201	See B.1 OpenAPI 2.0 Definition - /definitions /Subscribe Response
			400, 401, 403, 404	
DELETE	/api/v1/devices/**{deviceid}**/subscriptions/**{subscriptionid}**	Correlation-ID	202	
			400, 401, 403, 404	

9.6.2 Request and Response Payload

The “deviceid” in the request URI is the same as the "di" Property from "/oic/d" of the target OCF device.

The "subscriptionid" is the URI in the DELETE case is the "subscriptionid" that is returned in the response of the subscription POST request.

The response payload for the subscription POST request shall be as defined in clause 9.4.2

9.6.3 Responses

A 201 response is provided in a success case.

A 202 response indicates that the subscription was marked for cancellation.

A 400 response indicates that the request was malformed or badly constructed.

A 401 response indicates that the request is unauthorized (likely an invalid bearer token).

A 403 response indicates an invalid bearer token.

A 404 response indicates the device / subscription was not found.

9.7 Subscribe and Unsubscribe to Resource level events, retrieval of the subscription details

9.7.1 Summary

Note: this request is sent from the "origin Cloud" to the "target Cloud". This API is used when the "origin Cloud" wants to receive notifications from a specific Resource that exists on a specific Device on the "target Cloud".

A POST request establishes the subscription; a GET provides details about the subscription; a DELETE request removes the subscription.

Table 15 provides a summary of the API.

Table 15 – Subscription to Resource API Summary

HTTP Request Type	URI	Parameters	Response Code	Response Payload
POST	/api/v1/devices/**{deviceid}**/**{resourcehref}**/subscriptions	Correlation-ID, Accept, Content-Type	201	See B.1 OpenAPI 2.0 Definition - /definitions /Subscribe Response
			400, 401, 403, 404	
DELETE	/api/v1/devices/**{deviceid}**/**{resourcehref}**/subscriptions/**{subscriptionid}**	Correlation-ID	202	
			400, 401, 403, 404	

9.7.2 Request and Response Payload

The "deviceid" in the URI in the request is the same as the "di" Property from /oic/d of the target OCF device.

The "resourceHref" in the URI is the same as the "href" Link Parameter for the target Resource instance.

The "subscriptionid" is the URI in the DELETE case is the "subscriptionid" that is returned in the response of the subscription POST request.

The response payload for the subscription POST request shall be as defined in clause 9.4.2

9.7.3 Responses

A 201 response is provided in a success case.

A 202 response indicates that the subscription was marked for cancellation.

A 400 response indicates that the request was malformed or badly constructed.

A 401 response indicates that the request is unauthorized (likely an invalid bearer token).

A 403 response indicates an invalid bearer token.

A 404 response indicates the requested device / resource / subscription was not found.

9.8 Notification of /devices level events

9.8.1 Summary

Note: this request is sent from the "target Cloud" to the "origin Cloud".

Table 16 provides a summary of the API.

Table 16 – Notification of /devices API Summary

HTTP Request Type	URI	Parameters	Response Code	Response Payload
POST	/**{eventsUrl}**	Correlation-ID, Content-Type Event-Type, Subscription-ID, Sequence-Number, Event-Signature	200	
			400, 410	

9.8.2 Request and Response Payload

The “eventsUrl” in the URI is the value of the "eventsUrl" Property that was provided in the subscription request.

The payload in the notification request shall be the content of an event in a specific type based on what event the "origin Cloud" subscribes to as defined in clause 9.4.3

9.8.3 Responses

A 200 response is provided in a success case.

A 400 response indicates that the request was malformed or badly constructed.

A 410 response indicates that the subscription identified by the Subscription-ID header is no more in demand and shall be canceled

9.9 Notification of Device level events

9.9.1 Summary

Note: this request is sent from the "target Cloud" to the "origin Cloud".

Table 17 provides a summary of the API.

Table 17 – Notification of Single Device API Summary

HTTP Request Type	URI	Parameters	Response Code	Response Payload
POST	/**{eventsUrl}**	Correlation-ID, Content-Type Event-Type, Subscription-ID, Sequence-Number, Event-Signature	200	
			400, 410	

9.9.2 Request and Response Payload

The “eventsUrl” in the URI is the value of the "eventsUrl" Property that was provided in the subscription request.

The payload in the notification request shall be the content of an event in a specific type based on what event the "origin Cloud" subscribes to as defined in 9.4.3

9.9.3 Responses

A 200 response is provided in a success case.

A 400 response indicates that the request was malformed or badly constructed.

A 410 response indicates that the subscription identified by the Subscription-ID header is no more in demand and shall be canceled

9.10 Notification of Resource level events

9.10.1 Summary

Note: this request is sent from the "target Cloud" to the "origin Cloud".

Table 18 provides a summary of the API.

Table 18 – Notification of Resource API Summary

HTTP Request Type	**URI**	**Parameters**	**Response Code**	**Response Payload**
POST	/**{eventsUrl}**	Correlation-ID, Content-Type Event-Type, Subscription-ID, Sequence-Number, Event-Signature	200	
			400, 410	

9.10.2 Request and Response Payload

The “eventsUrl” in the URI is the value of the "eventsUrl" Property that was provided in the subscription request.

The payload in the notification request shall be the content of an event in a specific type based on what event the "origin Cloud" subscribes to as defined in clause 9.4.3

9.10.3 Responses

A 200 response is provided in a success case.

A 400 response indicates that the request was malformed or badly constructed.

A 410 response indicates that the subscription identified by the Subscription-ID header is no more in demand and shall be canceled

Annex A
Representative Flows

A.1 Introduction

The flows illustrate use of the OCF Cloud API for Cloud Services using OCF Devices as the endpoints and OCF Clouds as the two Clouds that are invoking/acting as endpoints. Note that this is for example use only and the API does not force this setup, which means non-OCF clouds with non-OCF devices can also use the API for interworking with other vendor's clouds.

A.2 OAuth2.0 Application Registration

Figure A.1 provides an example flow showing the registration of the OAuth Client / Application for the "origin Cloud" on the "target Cloud".

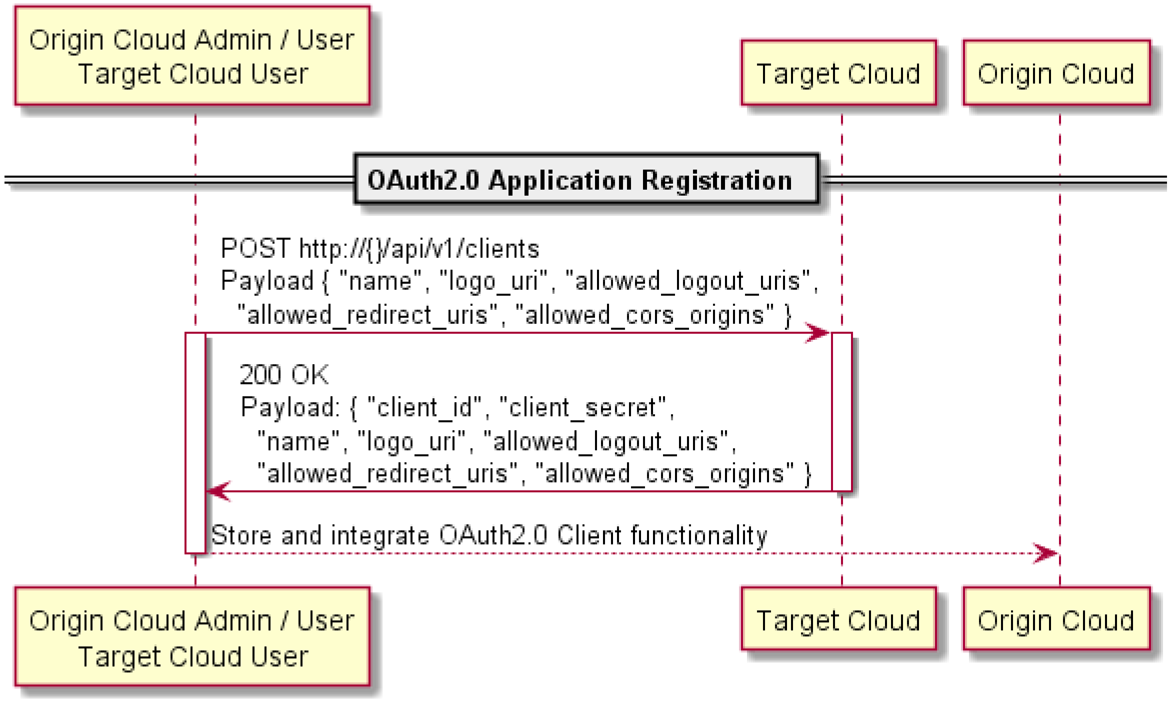

Origin Cloud Admin / User Target Cloud User
Target Cloud
Origin Cloud
OAuth2.0 Application Registration
POST http://{}/api/v1/clients
Payload { "name", "logo_uri", "allowed_logout_uris", "allowed_redirect_uris", "allowed_cors_origins" }
200 OK
Payload: { "client_id", "client_secret", "name", "logo_uri", "allowed_logout_uris", "allowed_redirect_uris", "allowed_cors_origins" }
Store and integrate OAuth2.0 Client functionality
Origin Cloud Admin / User Target Cloud User
Target Cloud
Origin Cloud

Figure A.1 – Establish Business Relationship Example Flow

A.3 Account Linking

Figure A.2 provides an example flow of the account linking for a particular user.

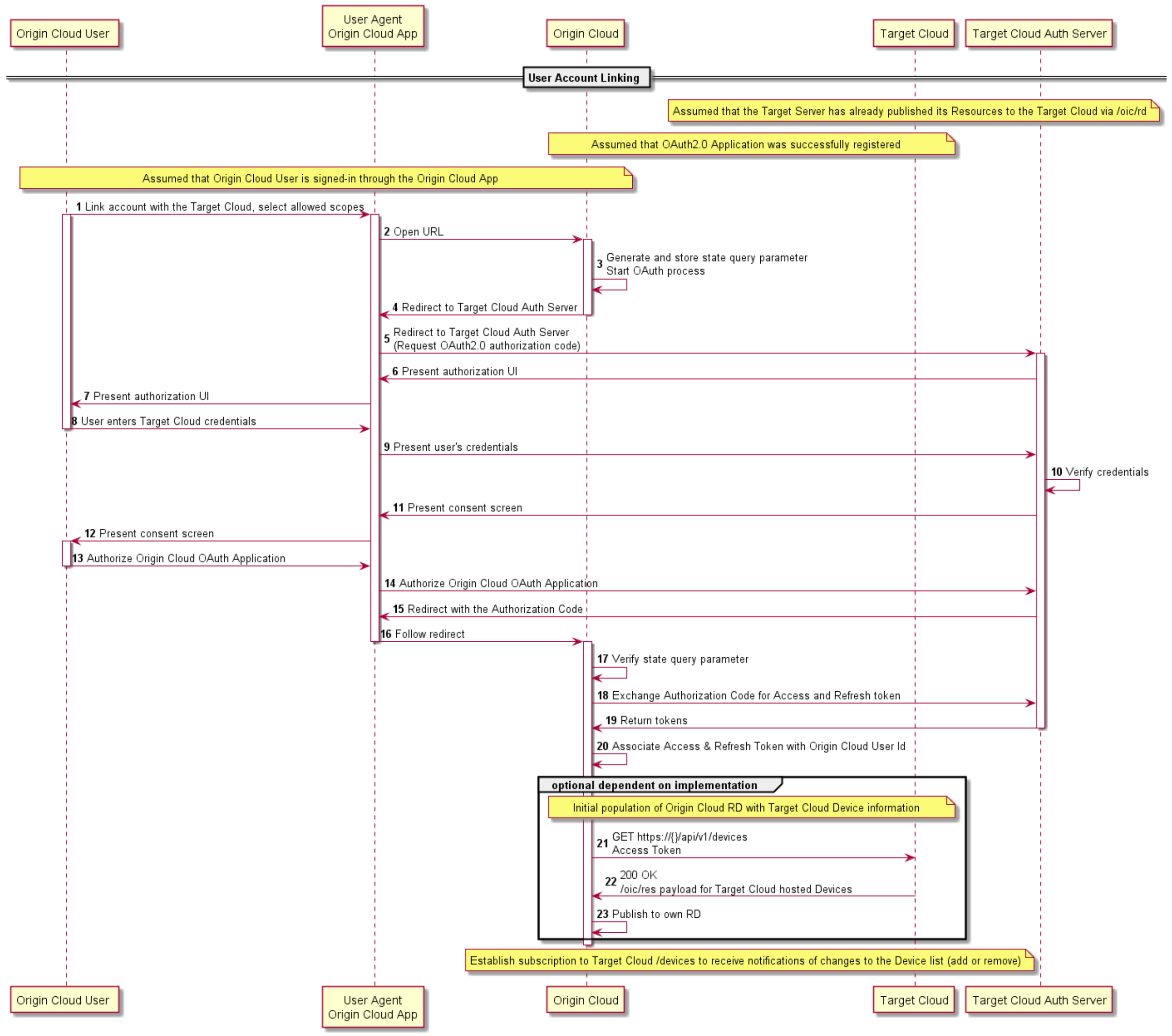

Origin Cloud User
User Agent Origin Cloud App
Origin Cloud
Target Cloud
Target Cloud Auth Server
User Account Linking
Assumed that the Target Server has already published its Resources to the Target Cloud via /oic/rd
Assumed that OAuth2.0 Application was successfully registered
Assumed that Origin Cloud User is signed-in through the Origin Cloud App
1 Link account with the Target Cloud, select allowed scopes
2 Open URL
3 Generate and store state query parameter Start OAuth process
4 Redirect to Target Cloud Auth Server
5 Redirect to Target Cloud Auth Server (Request OAuth2.0 authorization code)
6 Present authorization UI
7 Present authorization UI
8 User enters Target Cloud credentials
9 Present user's credentials
10 Verify credentials
11 Present consent screen
12 Present consent screen
13 Authorize Origin Cloud OAuth Application
14 Authorize Origin Cloud OAuth Application
15 Redirect with the Authorization Code
16 Follow redirect
17 Verify state query parameter
18 Exchange Authorization Code for Access and Refresh token
19 Return tokens
20 Associate Access & Refresh Token with Origin Cloud User Id
optional dependent on implementation
Initial population of Origin Cloud RD with Target Cloud Device information
21 GET https://{}/api/v1/devices Access Token
22 200 OK /oic/res payload for Target Cloud hosted Devices
23 Publish to own RD
Establish subscription to Target Cloud /devices to receive notifications of changes to the Device list (add or remove)
Origin Cloud User
User Agent Origin Cloud App
Origin Cloud
Target Cloud
Target Cloud Auth Server

Figure A.2 – Initial Association Example Flow

A.4 Retrieval of all Devices

A.4.1 Summary

The "origin Cloud" requests all Devices associated with a user (defined by the provided token). This may be invoked following account linking in order to retrieve the set of Devices for the user.

A.4.2 Flow

Figure A.3 provides an example flow for the retrieval of all Devices.

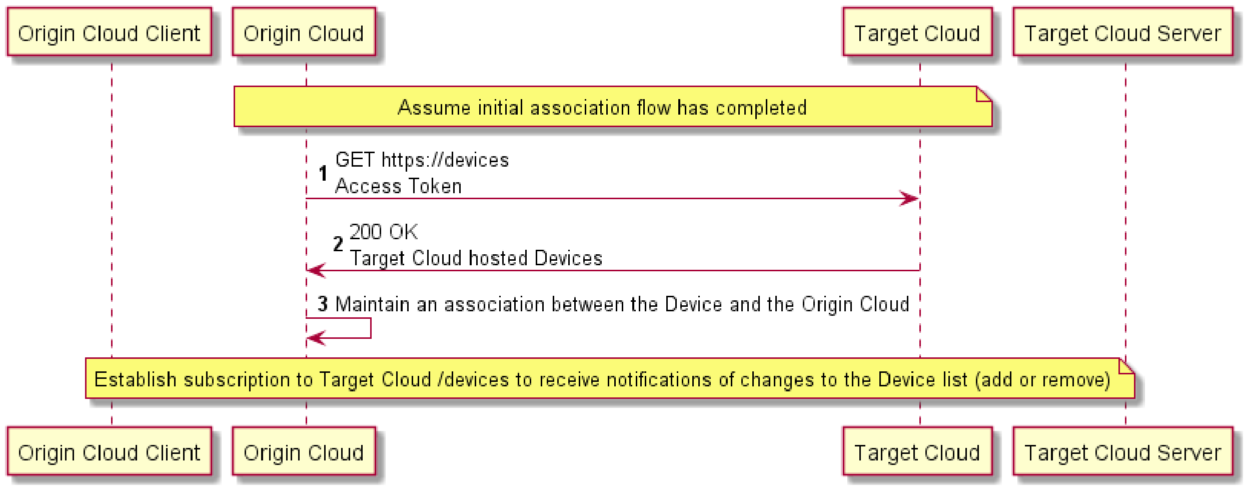

Origin Cloud Client
Origin Cloud
Target Cloud
Target Cloud Server
Assume initial association flow has completed
1 GET https://devices
Access Token
2 200 OK
Target Cloud hosted Devices
3 Maintain an association between the Device and the Origin Cloud
Establish subscription to Target Cloud /devices to receive notifications of changes to the Device list (add or remove)
Origin Cloud Client
Origin Cloud
Target Cloud
Target Cloud Server

Figure A.3 – Retrieve All Devices Example Flow

A.4.3 Flow Description

Table A.1 explains each element in the above sequence diagram

Table A.1 – Retrieve all Devices Flow Summary

Number	Description
1	Cloud requests all Devices given by the scope in the bearer token that was obtained via OAuth.
2	Response is an array of Device information (Properties that are defined in /oic/d that are pertinent to Cloud functionality and Device status).
3	Cloud maintains an association between the Device and the host Cloud.

A.5 Retrieval of a single Device

A.5.1 Summary

The "origin Cloud" requests information for a single, specific Device associated with a user (defined by the provided token). This may be invoked by the "origin Cloud" receiving a retrieve request from a connected Client.

A.5.2 Flow

Figure A.4 provides an example flow for the retrieval of a single Device.

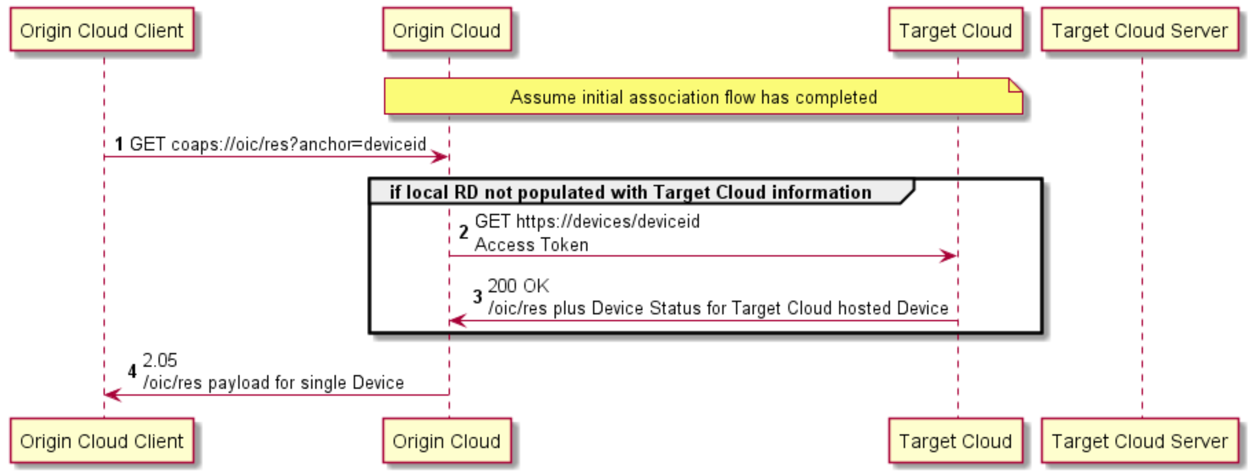

Origin Cloud Client
Origin Cloud
Target Cloud
Target Cloud Server
Assume initial association flow has completed
1 GET coaps://oic/res?anchor=deviceid
if local RD not populated with Target Cloud information
2 GET https://devices/deviceid
Access Token
3 200 OK
/oic/res plus Device Status for Target Cloud hosted Device
4 2.05
/oic/res payload for single Device
Origin Cloud Client
Origin Cloud
Target Cloud
Target Cloud Server

Figure A.4 – Retrieve Single Device Example Flow

A.5.3 Flow Description

Table A.2 explains each element in the above sequence diagram

Table A.2 – Retrieve single Device Flow Summary

Number	Description
1	[OCF Device to Cloud] OCF Client role Device requests /oic/res from the Cloud for a specific anchor (device id).
2	[Assuming that the information hasn't been cached by the Cloud] For the instance of /oic/sec/account that exists for the Device the Cloud does a GET /devices/{deviceid} to the Cloud identified by the cloudid in /oic/sec/account. {deviceid} is also taken from /oic/sec/account.
3	Response is the Device information as well as an array of Links. The "href" in each Link will be of the form "/deviceid/resourcehref".
4	Response payload.

A.6 Retrieval of a single Resource

A.6.1 Summary

The "origin Cloud" requests information for a single, specific Resource exposed by a Device associated with a user (defined by the provided token). This may be invoked by the "origin Cloud" receiving a retrieve request from a connected Client.

A.6.2 Flows

A.6.2.1 Success Path

Figure A.5 provides an example flow for the retrieval of a single Resource.

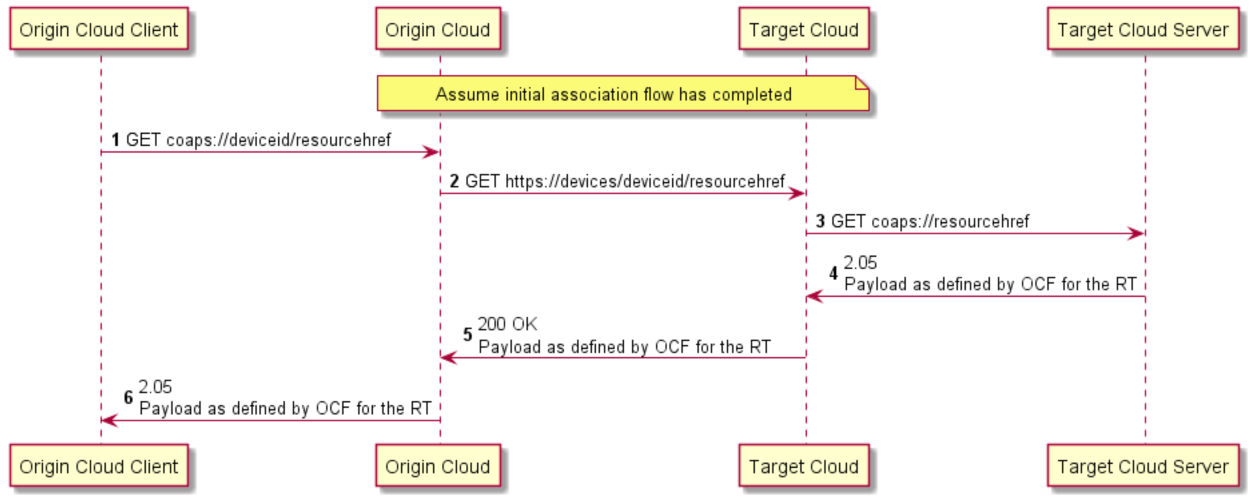

Origin Cloud Client
Origin Cloud
Target Cloud
Target Cloud Server
Assume initial association flow has completed
1 GET coaps://deviceid/resourcehref
2 GET https://devices/deviceid/resourcehref
3 GET coaps://resourcehref
4 2.05
Payload as defined by OCF for the RT
5 200 OK
Payload as defined by OCF for the RT
6 2.05
Payload as defined by OCF for the RT
Origin Cloud Client
Origin Cloud
Target Cloud
Target Cloud Server

Figure A.5 – Retrieve Resource (Success) Example Flow

A.6.2.2 Success Path Flow Description

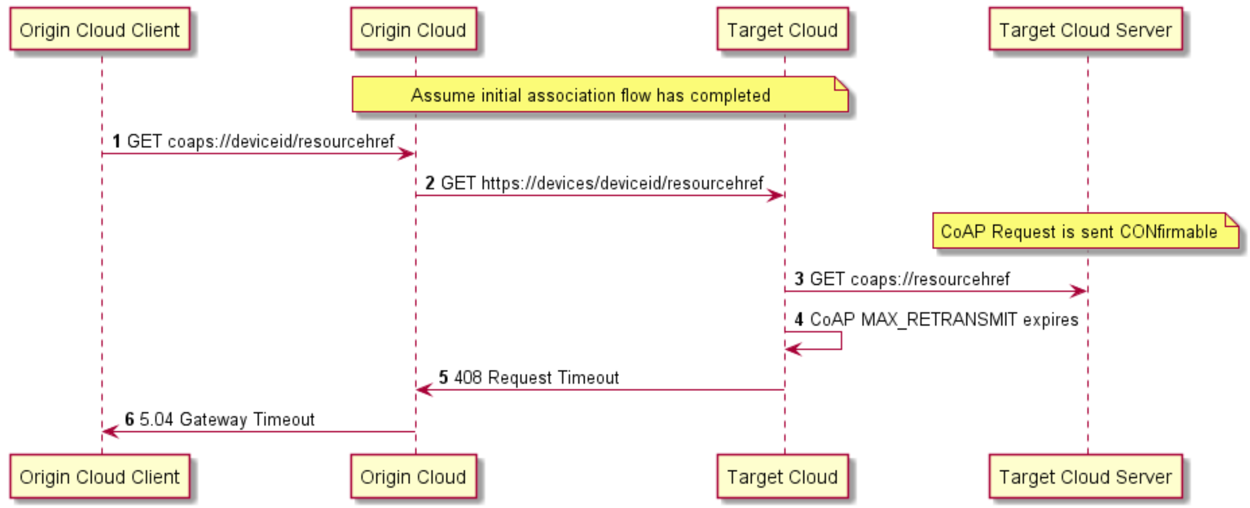
Table A.3 explains each element in the above sequence diagram

Table A.3 – Retrieve single Resource Flow Summary

Number	Description
1	[OCF Device to Cloud] OCF Client role Device requests a Resource from the Cloud using the "href" exposed in the /oic/res response. This will be of the form "/deviceid/resourcehref"
2	[Assuming that the resource representation hasn't been cached by the Cloud] Cloud identifies the host Cloud for the Resource via the instance of /oic/sec/account for the "deviceid". The request is then effectively proxied to the "target Cloud" via a GET /devices/{deviceid}/{resourcehref}. Any query parameters received over CoAP are included in the URI unaltered.
3	[OCF Device to Cloud] "target Cloud" identifies the TLS connection to the end Device via the {deviceid} and proxies the request.
4	Standard OCF response
5	Success path response including the response payload as received for the target Resource
6	Standard OCF response

A.6.2.3 Device is Temporarily Unavailable

Figure A.6 illustrates the case where the Device is temporarily unavailable.



Origin Cloud Client
Origin Cloud
Target Cloud
Target Cloud Server
Assume initial association flow has completed
1 GET coaps://deviceid/resourcehref
2 GET https://devices/deviceid/resourcehref
CoAP Request is sent CONfirmable
3 GET coaps://resourcehref
4 CoAP MAX_RETRANSMIT expires
5 408 Request Timeout
6 5.04 Gateway Timeout

Figure A.6 – Retrieve Resource (Timeout) Example Flow

A.7 Update of a single Resource

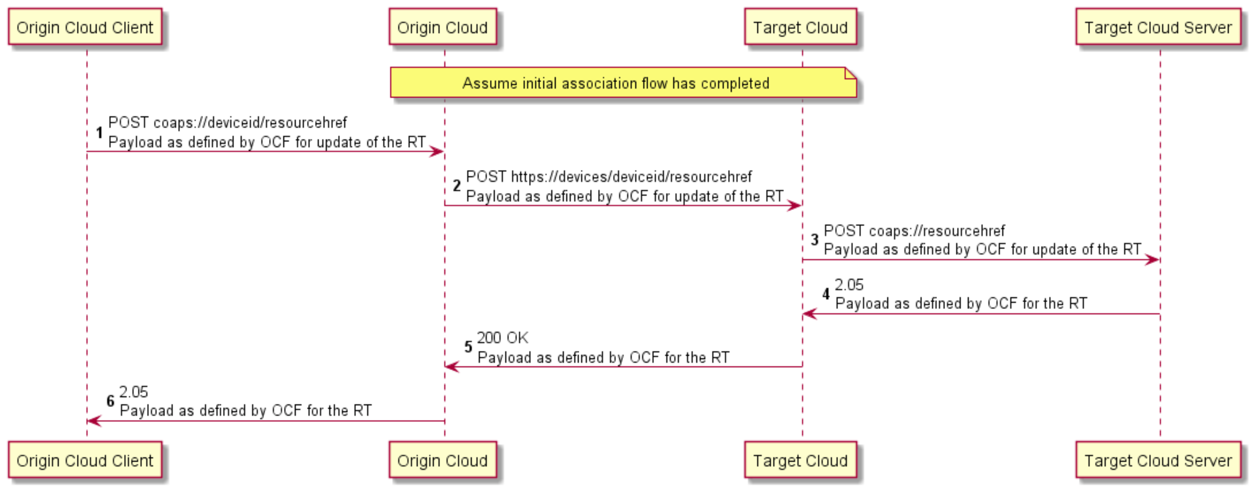
A.7.1 Summary

The "origin Cloud" updates information for a single, specific Device associated with a user (defined by the provided token). This may be invoked by the "origin Cloud" receiving an update request from a connected Client.

A.7.2 Flows

A.7.2.1 Success Path

Figure A.7 provides an example flow for the updating of a single Resource.



Origin Cloud Client
Origin Cloud
Target Cloud
Target Cloud Server
Assume initial association flow has completed
1 POST coaps://deviceid/resourcehref Payload as defined by OCF for update of the RT
2 POST https://devices/deviceid/resourcehref Payload as defined by OCF for update of the RT
3 POST coaps://resourcehref Payload as defined by OCF for update of the RT
4 2.05 Payload as defined by OCF for the RT
5 200 OK Payload as defined by OCF for the RT
6 2.05 Payload as defined by OCF for the RT

Figure A.7 – Update Resource (Success) Example Flow

A.7.2.2 Success Path Flow Description

Table A.4 explains each element in the above sequence diagram

Table A.4 – Update single Resource Flow Summary

Number	Description

1	[OCF Device to Cloud] OCF Client role Device requests a Resource from the Cloud using the "href" exposed in the /oic/res response. This will be of the form "/deviceid/resourcehref"
2	Cloud identifies the host Cloud for the Resource via the instance of /oic/sec/account for the "deviceid". The request is then effectively proxied to the "target Cloud" via a POST /devices/{deviceid}/{resourcehref} including the payload from the original request. Any query parameters received over CoAP are included in the URI unaltered.
3	[OCF Device to Cloud] "target Cloud" identifies the TLS connection to the end Device via the {deviceid} and proxies the request.
4	Standard OCF response
5	Success path response including the response payload as received for the target Resource
6	Standard OCF response

A.7.2.3 Device is Temporarily Unavailable

Figure A.8 illustrates the case where the Device is temporarily unavailable.

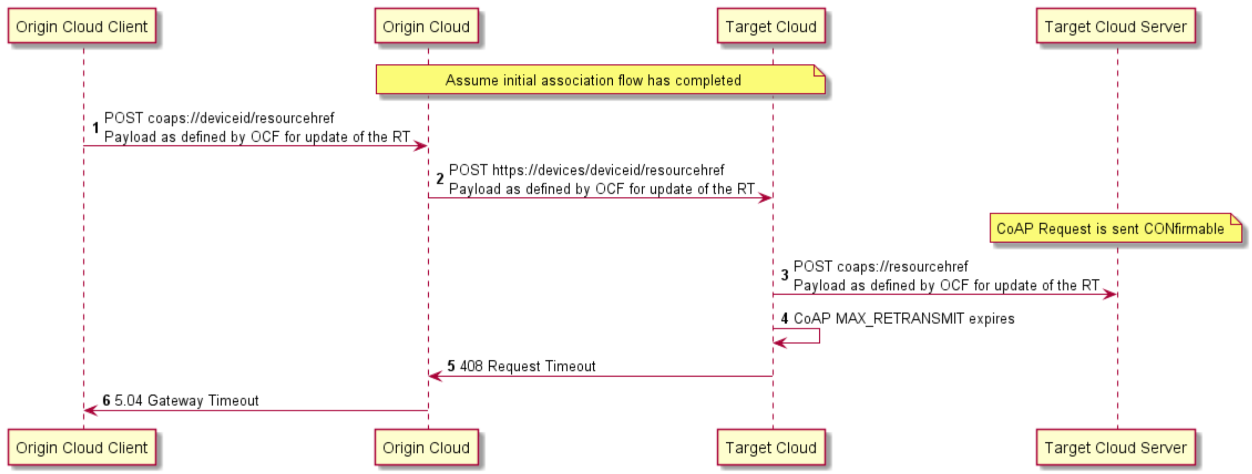

Origin Cloud Client
Origin Cloud
Target Cloud
Target Cloud Server
Assume initial association flow has completed
1 POST coaps://deviceid/resourcehref
Payload as defined by OCF for update of the RT
2 POST https://devices/deviceid/resourcehref
Payload as defined by OCF for update of the RT
CoAP Request is sent CONfirmable
3 POST coaps://resourcehref
Payload as defined by OCF for update of the RT
4 CoAP MAX_RETRANSMIT expires
5 408 Request Timeout
6 5.04 Gateway Timeout
Origin Cloud Client
Origin Cloud
Target Cloud
Target Cloud Server

Figure A.8 – Update Resource (Timeout) Example Flow

A.8 Establishment of new subscription request

A.8.1 Summary

The "origin Cloud" requests the establishment of an observe relationship with a single, specific Resource on a Device associated with a user (defined by the provided token). This may be invoked by "origin Cloud" receiving a retrieve request containing an observe option from a connected Client.

A.8.2 Flows

Figure A.9 provides an example flow for the establishment of a subscription to the "resource_contentchanged" event for a specific Resource.

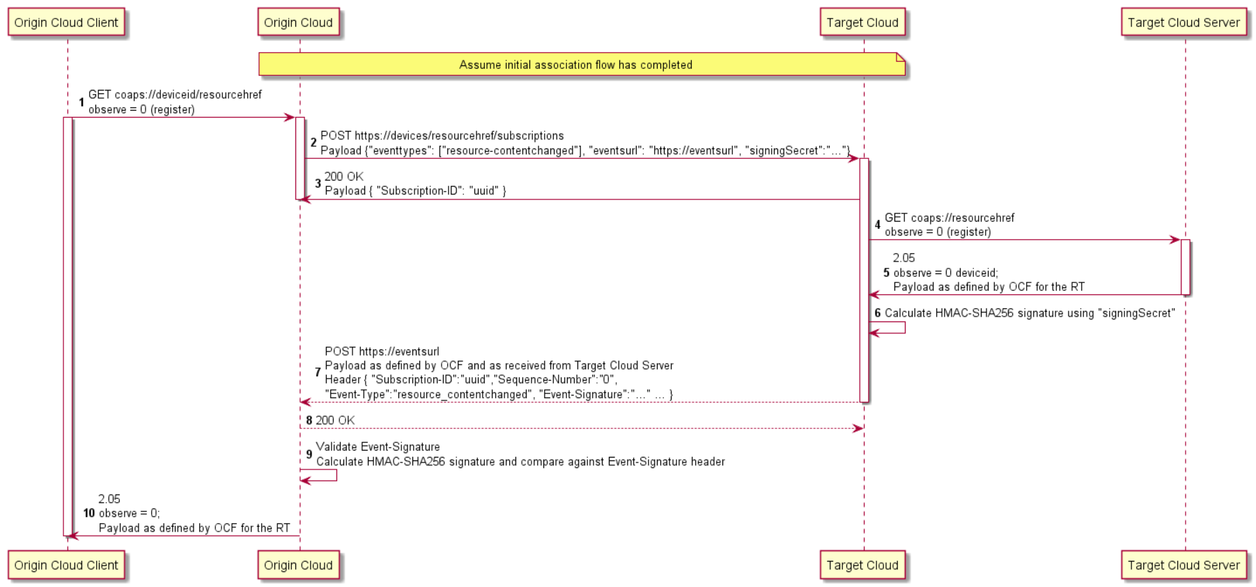

Origin Cloud Client
Origin Cloud
Target Cloud
Target Cloud Server
Assume initial association flow has completed
1 GET coaps://deviceid/resourcehref observe = 0 (register)
2 POST https://devices/resourcehref/subscriptions Payload {"eventtypes": ["resource-contentchanged"], "eventsurl": "https://eventsurl", "signingSecret":"…"}
3 200 OK Payload { "Subscription-ID": "uuid" }
4 GET coaps://resourcehref observe = 0 (register)
5 2.05 observe = 0 deviceid; Payload as defined by OCF for the RT
6 Calculate HMAC-SHA256 signature using "signingSecret"
7 POST https://eventsurl Payload as defined by OCF and as received from Target Cloud Server Header { "Subscription-ID":"uuid","Sequence-Number":"0", "Event-Type":"resource_contentchanged", "Event-Signature":"…" … }
8 200 OK
9 Validate Event-Signature Calculate HMAC-SHA256 signature and compare against Event-Signature header
10 2.05 observe = 0; Payload as defined by OCF for the RT
Origin Cloud Client
Origin Cloud
Target Cloud
Target Cloud Server

Figure A.9 – Observe Establishment Example Flow

A.9 Event generated for a subscription

A.9.1 Summary

An event occurs for a Resource with which the "origin Cloud" has established a subscription/event relationship. This may be invoked by the target end Device being updated.

A.9.2 Flows

Figure A.10 provides an example flow for the handling of a generated "resource_contentchanged" event.

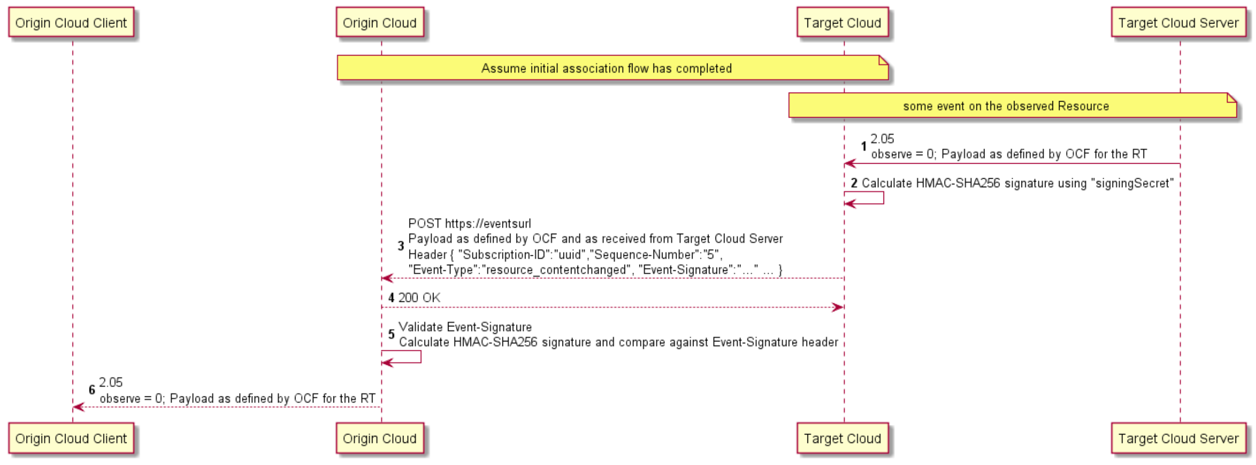

Origin Cloud Client
Origin Cloud
Target Cloud
Target Cloud Server
Assume initial association flow has completed
some event on the observed Resource
1 2.05 observe = 0; Payload as defined by OCF for the RT
2 Calculate HMAC-SHA256 signature using "signingSecret"
3 POST https://eventsurl Payload as defined by OCF and as received from Target Cloud Server Header { "Subscription-ID":"uuid","Sequence-Number":"5", "Event-Type":"resource_contentchanged", "Event-Signature":"…" … }
4 200 OK
5 Validate Event-Signature Calculate HMAC-SHA256 signature and compare against Event-Signature header
6 2.05 observe = 0; Payload as defined by OCF for the RT
Origin Cloud Client
Origin Cloud
Target Cloud
Target Cloud Server

Figure A.10 – "resource_contentchanged" Event Example Flow

A.10 Addition of new registration

A.10.1 Summary

The "origin Cloud" has a priori established a subscription/event relationship with the set of Devices associated with a user exposed by "target Cloud". The user then registers a new Device with "target Cloud".

A.10.2 Flows

Figure A.11 provides an example flow for the generation of a notification (event) when a new Device is registered.

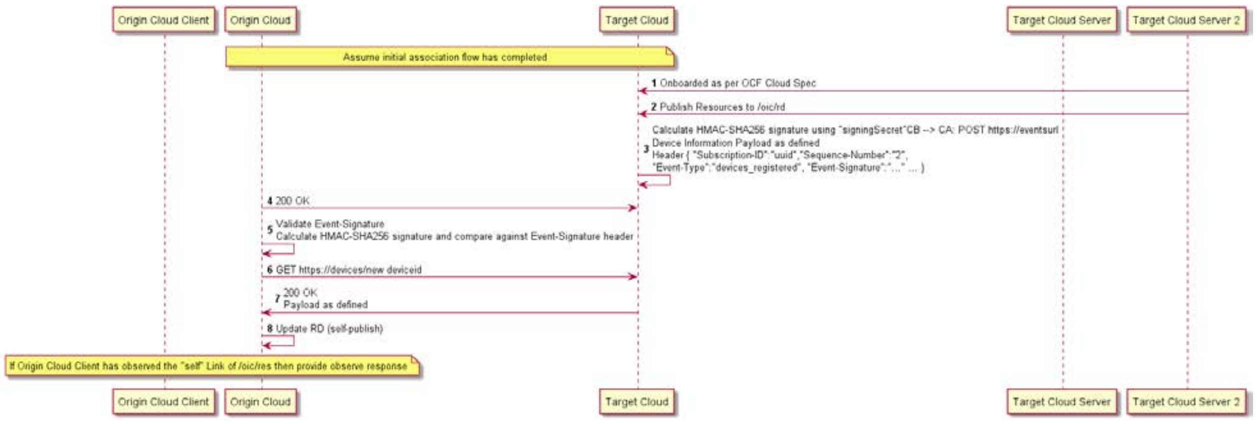

Origin Cloud Client
Origin Cloud
Target Cloud
Target Cloud Server
Target Cloud Server 2
Assume initial association flow has completed
1 Onboarded as per OCF Cloud Spec
2 Publish Resources to /oic/rd
3 Calculate HMAC-SHA256 signature using "signingSecret"CB --> CA: POST https://eventsurl
Device Information Payload as defined
Header { "Subscription-ID":"uuid","Sequence-Number":"2",
"Event-Type":"devices_registered", "Event-Signature":"..." ... }
4 200 OK
5 Validate Event-Signature
Calculate HMAC-SHA256 signature and compare against Event-Signature header
6 GET https://devices/new deviceid
7 200 OK
Payload as defined
8 Update RD (self-publish)
If Origin Cloud Client has observed the "self" Link of /oic/res then provide observe response

Figure A.11 – Addition of new registered Device example flow

A.11 Removal of existing device registration

A.11.1 Summary

The "origin Cloud" has a priori established a subscription/event relationship with the set of Devices associated with a user exposed by "target Cloud". The user then removes a Device from "target Cloud".

A.11.2 Flows

Figure A.12 provides an example flow for the generation of a notification (event) when a Device is removed.

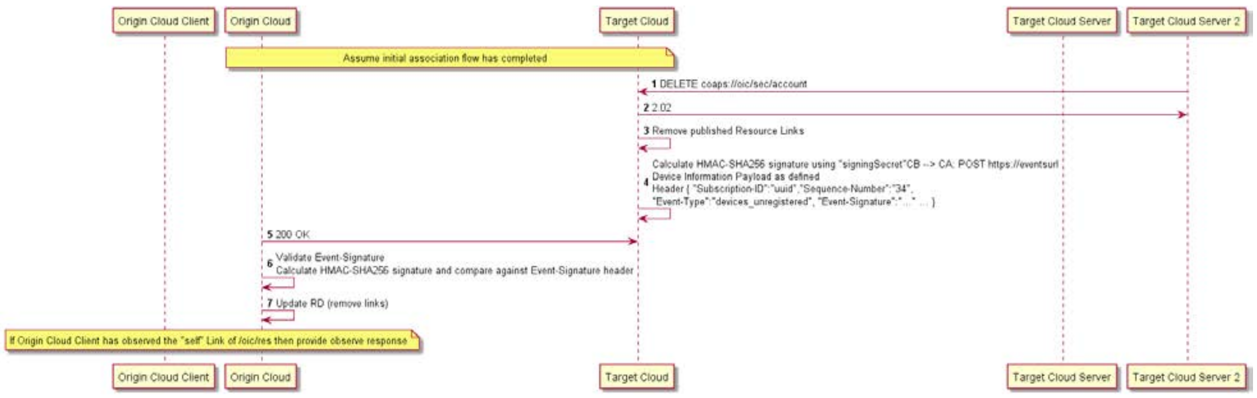

Origin Cloud Client
Origin Cloud
Target Cloud
Target Cloud Server
Target Cloud Server 2
Assume initial association flow has completed
1 DELETE coaps://oic/sec/account
2 2.02
3 Remove published Resource Links
4 Calculate HMAC-SHA256 signature using "signingSecret"CB --> CA: POST https://eventsurl
Device Information Payload as defined
Header { "Subscription-ID":"uuid","Sequence-Number":"34",
"Event-Type":"devices_unregistered", "Event-Signature":"..." ... }
5 200 OK
6 Validate Event-Signature
Calculate HMAC-SHA256 signature and compare against Event-Signature header
7 Update RD (remove links)
If Origin Cloud Client has observed the "self" Link of /oic/res then provide observe response

Figure A.12 – Removal of existing registration example flow

Annex B
Open API Definition

B.1 OCF Cloud API for Cloud Services

B.1.1 Supported APIs

B.1.1.1 /api/v1/devices/subscriptions

Subscribe to devices events by providing `eventTypes` you're interested in and `eventsUrl` endpoint where events will be sent to as defined. Successful response contains `subscriptionId` which identifies registered subscription and is part of each event. First event for each registered event type is received immediately after subscription and contains actual state of the resource, followed by new events in case of any change.

Supported events and required scopes
- `devices_registered`: `r:deviceinformation:*`
- `devices_unregistered`: `r:deviceinformation:*`
- `devices_online`: `r:deviceinformation:*`
- `devices_offline`: `r:deviceinformation:*`

B.1.1.2 /api/v1/devices

Get all devices which are signed up to the OCF Cloud - either `online` or `offline`. Devices which are `online` are signed in to the system and are accessible. Offline devices are signed up to the system, but currently disconnected.

B.1.1.3 /api/v1/devices/{deviceId}/subscriptions/{subscriptionId}

Cancel subscription identified by the id returned in a response for subscription.

B.1.1.4 /{eventsUrl}

Events endpoint provided during subscription where events specified in the subscription will be sent to as defined per event type. Confirmation of each event sent to the `eventsUrl` endpoint is required with `2xx` success code. Events you will receive based on event type you're subscribed to are:
- `subscription_canceled`: `SubscriptionCanceledEvent`
- `devices_registered`: `DevicesRegisteredEvent`
- `devices_unregistered`: `DevicesUnregisteredEvent`
- `resources_published`: `ResourcesPublishedEvent`
- `resources_unpublished`: `ResourcesUnpublishedEvent`
- `devices_online`: `DevicesOnlineEvent`
- `devices_offline`: `DevicesOfflineEvent`
- `resource_contentchanged`: `ResourceContentChangedEvent`

B.1.1.5 /api/v1/devices/{deviceId}/{resourceLinkHref}/subscriptions/{subscriptionId}

Cancel subscription identified by the id returned in a response for subscription.

B.1.1.6 /api/v1/devices/subscriptions/{subscriptionId}

Cancel subscription identified by the id returned in a response for subscription.

B.1.1.7 /api/v1/devices/{deviceId}/subscriptions

Subscribe to device events by providing `eventTypes` you're interested in and `eventsUrl` endpoint where events will be sent to as defined. Successful response contains `subscriptionId` which identifies registered subscription and is part of each event. First event for each registered event type is received immediately after subscription and contains actual state of the resource, followed by new events in case of any change.

Supported events and required scopes
- `resources_published`: `r:deviceinformation:*`
- `resources_unpublished`: `r:deviceinformation:*`

B.1.1.8 /api/v1/devices/{deviceId}/{resourceLinkHref}

#/responses/GatewayTimeout

B.1.1.9 /api/v1/devices/{deviceId}

Device requested with content=all query parameter

B.1.1.10 /api/v1/devices/{deviceId}/{resourceLinkHref}/subscriptions

Subscribe to resource events by providing `eventTypes` you're interested in and `eventsUrl` endpoint where events will be sent to as defined. Successful response contains `subscriptionId` which identifies registered subscription and is part of each event. First event for each registered event type is received immediately after subscription and contains actual state of the resource, followed by new events in case of any change.

Supported events and required scopes
- `resource_contentchanged`: `r:resources:*`

B.1.2 OpenAPI 2.0 definition

```
{
  "swagger": "2.0",
  "info": {
    "title": "OCF Cloud API for Cloud Services",
    "version": "0.0.3-20190828",
    "license": {
      "name": "Copyright 2019 Open Connectivity Foundation, Inc. All rights reserved.",
      "x-description": "Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions are met:\n        1.
Redistributions of source code must retain the above copyright notice, this list of conditions and
the following disclaimer.\n        2.  Redistributions in binary form must reproduce the above
copyright notice, this list of conditions and the following disclaimer in the documentation and/or
other materials provided with the distribution.\n\n        THIS SOFTWARE IS PROVIDED BY THE Open
Connectivity Foundation, INC. \"AS IS\" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE OR
WARRANTIES OF NON-INFRINGEMENT, ARE DISCLAIMED.\n        IN NO EVENT SHALL THE Open Connectivity
Foundation, INC. OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY,
OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)\n        HOWEVER CAUSED AND ON
ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR
OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
SUCH DAMAGE.\n"
    }
  },
  "host": "api.example.com",
  "schemes": [
    "https"
  ],
  "tags": [
    {
      "name": "Devices",
      "description": "Basic information about devices"
    },
    {
      "name": "Resources",
      "description": "Read or change the configuration of the device"
    },
    {
      "name": "Events",
      "description": "Be notified about changes occuring on the device"
    }
  ],
  "paths": {
    "/api/v1/devices": {
      "parameters": [
        {
          "$ref": "#/parameters/CorrelationId"
        },
        {
          "$ref": "#/parameters/Accept"
        },
        {
          "$ref": "#/parameters/BatchFormat"
        }
      ],
      "get": {
        "tags": [
          "Devices"
        ],
        "summary": "Get all devices",
        "description": "Get all devices which are signed up to the OCF Cloud - either `online` or
`offline`. Devices which are `online` are signed in to the system and are accessible. Offline
devices are signed up to the system, but currently disconnected.\n",
        "produces": [
          "application/json"
        ],
        "responses": {
          "200": {
            "description": "An array of devices",
            "schema": {
              "type": "array",
              "items": {
                "$ref": "#/definitions/Device"
              }
            }
          },
          "400": {
            "$ref": "#/responses/BadRequest"
          },
          "401": {
            "$ref": "#/responses/Unauthorized"
          },
          "403": {
            "$ref": "#/responses/Forbidden"
          }
        },
        "security": [
          {
            "oauth2": [
              "r:deviceinformation:*"
            ]
          }
        ]
      }
    },
    "/api/v1/devices/subscriptions": {
      "parameters": [
        {
          "$ref": "#/parameters/CorrelationId"
        },
        {
          "$ref": "#/parameters/Accept"
        },
        {
          "$ref": "#/parameters/ContentType"
        }
      ],
      "post": {
        "tags": [
          "Events"
        ],
        "summary": "Subscribe to events against the set of devices",
        "description": "Subscribe to devices events by providing `eventTypes` you're interested in and `eventsUrl` endpoint where events will be sent to as defined. Successful response contains `subscriptionId` which identifies registered subscription and is part of each event. First event for each registered event type is received immediately after subscription and contains actual state of the resource, followed by new events in case of any change.\n\n**Supported events** and required scopes\n- `devices_registered`: `r:deviceinformation:*`\n- `devices_unregistered`: `r:deviceinformation:*`\n- `devices_online`: `r:deviceinformation:*`\n- `devices_offline`: `r:deviceinformation:*`\n",
        "parameters": [
          {
            "$ref": "#/parameters/SubscribeRequest"
          }
        ],
        "consumes": [
          "application/json"
        ],
        "produces": [
          "application/json"
        ],
        "responses": {
          "201": {
            "$ref": "#/definitions/SubscribeResponse"
          },
          "400": {
            "$ref": "#/responses/BadRequest"
          },
          "401": {
            "$ref": "#/responses/Unauthorized"
          },
          "403": {
            "$ref": "#/responses/Forbidden"
          }
        },
        "security": [
          {
            "oauth2": [
              "w:subscriptions:*",
              "r:deviceinformation:*"
            ]
          }
        ]
      }
    },
    "/api/v1/devices/subscriptions/{subscriptionId}": {
      "parameters": [
        {
          "$ref": "#/parameters/CorrelationId"
        },
        {
          "$ref": "#/parameters/SubscriptionIdPath"
        }
      ],
      "delete": {
        "tags": [
          "Events"
        ],
        "summary": "Unsubscribe from events against the set of devices",
        "description": "Cancel subscription identified by the id returned in a response for subscription.\n",
        "responses": {
          "202": {
            "description": "Subscription was marked for cancellation"
          },
          "400": {
            "$ref": "#/responses/BadRequest"
          },
          "401": {
            "$ref": "#/responses/Unauthorized"
          },
          "403": {
            "$ref": "#/responses/Forbidden"
          },
          "404": {
            "$ref": "#/responses/NotFound"
          }
        },
        "security": [
          {
            "oauth2": [
              "w:subscriptions:*",
              "r:deviceinformation:*"
            ]
          }
        ]
      }
    },
    "/api/v1/devices/{deviceId}": {
      "parameters": [
        {
          "$ref": "#/parameters/CorrelationId"
        },
        {
          "$ref": "#/parameters/Accept"
        },
        {
          "$ref": "#/parameters/DeviceId"
        },
        {
          "$ref": "#/parameters/BatchFormat"
        }
      ],
      "get": {
        "tags": [
          "Devices"
        ],
        "summary": "Get the device by ID",
        "consumes": [
          "application/json"
        ],
        "produces": [
          "application/json"
        ],
        "responses": {
          "200": {
            "description": "Device requested with content=all query parameter",
            "schema": {
              "$ref": "#/definitions/DeviceContentAll"
            }
          },
          "400": {
            "$ref": "#/responses/BadRequest"
          },
          "401": {
            "$ref": "#/responses/Unauthorized"
          },
          "403": {
            "$ref": "#/responses/Forbidden"
          },
          "404": {
            "$ref": "#/responses/NotFound"
          }
        },
        "security": [
            {
              "oauth2": [
                "r:deviceinformation:*"
              ]
            }
          ]
        }
      },
      "/api/v1/devices/{deviceId}/subscriptions": {
        "parameters": [
          {
            "$ref": "#/parameters/CorrelationId"
          },
          {
            "$ref": "#/parameters/DeviceId"
          },
          {
            "$ref": "#/parameters/Accept"
          },
          {
            "$ref": "#/parameters/ContentType"
          }
        ],
        "post": {
          "tags": [
            "Events"
          ],
          "summary": "Subscribe to events against a specific device",
          "description": "Subscribe to device events by providing `eventTypes` you're interested in
and `eventsUrl` endpoint where events will be sent to as defined. Successful response contains
`subscriptionId` which identifies registered subscription and is part of each event. First event for
each registered event type is received immediately after subscription and contains actual state of
the resource, followed by new events in case of any change.\n\n**Supported events** and required
scopes\n- `resources_published`: `r:deviceinformation:*`\n- `resources_unpublished`:
`r:deviceinformation:*`\n",
          "parameters": [
            {
              "$ref": "#/parameters/SubscribeRequest"
            }
          ],
          "consumes": [
            "application/json"
          ],
          "produces": [
            "application/json"
          ],
          "responses": {
            "201": {
              "$ref": "#/definitions/SubscribeResponse"
            },
            "400": {
              "$ref": "#/responses/BadRequest"
            },
            "401": {
              "$ref": "#/responses/Unauthorized"
            },
            "403": {
              "$ref": "#/responses/Forbidden"
            },
            "404": {
              "$ref": "#/responses/NotFound"
            }
          },
          "security": [
            {
              "oauth2": [
                "w:subscriptions:*",
                "r:deviceinformation:*"
              ]
            }
          ]
      }
    },
    "/api/v1/devices/{deviceId}/subscriptions/{subscriptionId}": {
      "parameters": [
        {
          "$ref": "#/parameters/CorrelationId"
        },
        {
          "$ref": "#/parameters/DeviceId"
        },
        {
          "$ref": "#/parameters/SubscriptionIdPath"
        }
      ],
      "delete": {
        "tags": [
          "Events"
        ],
        "summary": "Unsubscribe from events against a specific device",
        "description": "Cancel subscription identified by the id returned in a response for
subscription.\n",
        "responses": {
          "202": {
            "description": "Subscription was marked for cancellation"
          },
          "400": {
            "$ref": "#/responses/BadRequest"
          },
          "401": {
            "$ref": "#/responses/Unauthorized"
          },
          "403": {
            "$ref": "#/responses/Forbidden"
          },
          "404": {
            "$ref": "#/responses/NotFound"
          }
        },
        "security": [
          {
            "oauth2": [
              "w:subscriptions:*",
              "r:deviceinformation:*"
            ]
          }
        ]
      }
    },
    "/api/v1/devices/{deviceId}/{resourceLinkHref}": {
      "parameters": [
        {
          "$ref": "#/parameters/CorrelationId"
        },
        {
          "$ref": "#/parameters/DeviceId"
        },
        {
          "$ref": "#/parameters/ResourceLinkHref"
        },
        {
          "$ref": "#/parameters/Accept"
        }
      ],
      "get": {
        "tags": [
          "Resources"
        ],
        "summary": "Retrieve resource values",
        "consumes": [
          "application/json",
          "application/vnd.ocf+cbor"
        ],
        "produces": [
          "application/json",
          "application/vnd.ocf+cbor"
        ],
        "responses": {
          "200": {
            "$ref": "#/definitions/ResourceRetrieveResponse"
          },
          "400": {
            "$ref": "#/responses/BadRequest"
          },
          "401": {
            "$ref": "#/responses/Unauthorized"
          },
          "403": {
            "$ref": "#/responses/Forbidden"
          },
          "404": {
            "$ref": "#/responses/NotFound"
          },
          "503": {
            "description": "#/responses/ServiceUnavailable"
          },
          "504": {
            "description": "#/responses/GatewayTimeout"
          }
        },
        "security": [
          {
            "oauth2": [
              "r:resources:*"
            ]
          }
        ]
      },
      "post": {
        "tags": [
          "Resources"
        ],
        "summary": "Update resource values",
        "parameters": [
          {
            "$ref": "#/parameters/ResourceUpdateRequest"
          },
          {
            "$ref": "#/parameters/ContentType"
          }
        ],
        "consumes": [
          "application/json",
          "application/vnd.ocf+cbor"
        ],
        "produces": [
          "application/json",
          "application/vnd.ocf+cbor"
        ],
        "responses": {
          "200": {
            "$ref": "#/definitions/ResourceRetrieveResponse"
          },
          "400": {
            "$ref": "#/responses/BadRequest"
          },
          "401": {
            "$ref": "#/responses/Unauthorized"
          },
          "403": {
            "$ref": "#/responses/Forbidden"
          },
          "404": {
            "$ref": "#/responses/NotFound"
          },
          "415": {
            "description": "Unsupported media type specified in the Content-Type header"
          },
          "503": {
            "description": "#/responses/ServiceUnavailable"
          },
          "504": {
            "description": "#/responses/GatewayTimeout"
          }
        },
        "security": [
          {
            "oauth2": [
              "w:resources:*"
            ]
          }
        ]
      }
    },
    "/api/v1/devices/{deviceId}/{resourceLinkHref}/subscriptions": {
      "parameters": [
        {
          "$ref": "#/parameters/CorrelationId"
        },
        {
          "$ref": "#/parameters/DeviceId"
        },
        {
          "$ref": "#/parameters/ResourceLinkHref"
        },
        {
          "$ref": "#/parameters/Accept"
        },
        {
          "$ref": "#/parameters/ContentType"
        }
      ],
      "post": {
        "tags": [
          "Events"
        ],
        "summary": "Subscribe to events against a specific resource",
        "description": "Subscribe to resource events by providing `eventTypes` you're interested in
and `eventsUrl` endpoint where events will be sent to as defined. Successful response contains
`subscriptionId` which identifies registered subscription and is part of each event. First event for
each registered event type is received immediately after subscription and contains actual state of
the resource, followed by new events in case of any change.\n \n**Supported events** and required
scopes\n- `resource_contentchanged`: `r:resources:*`",
        "parameters": [
          {
            "$ref": "#/parameters/SubscribeRequest"
          }
        ],
        "consumes": [
          "application/json"
        ],
        "produces": [
          "application/json"
        ],
        "responses": {
          "201": {
            "$ref": "#/definitions/SubscribeResponse"
          },
          "400": {
            "$ref": "#/responses/BadRequest"
          },
          "401": {
            "$ref": "#/responses/Unauthorized"
          },
          "403": {
            "$ref": "#/responses/Forbidden"
          },
          "404": {
            "$ref": "#/responses/NotFound"
          }
        },
        "security": [
          {
            "oauth2": [
              "w:subscriptions:*",
              "r:resources:*"
            ]
          }
        ]
      }
    },
    "/api/v1/devices/{deviceId}/{resourceLinkHref}/subscriptions/{subscriptionId}": {
      "parameters": [
        {
          "$ref": "#/parameters/CorrelationId"
        },
        {
          "$ref": "#/parameters/DeviceId"
        },
        {
          "$ref": "#/parameters/ResourceLinkHref"
        },
        {
          "$ref": "#/parameters/SubscriptionIdPath"
        }
      ],
      "delete": {
        "tags": [
          "Events"
        ],
        "summary": "Unsubscribe from events against a specific resource",
        "description": "Cancel subscription identified by the id returned in a response for
subscription.\n",
        "responses": {
          "202": {
            "description": "Subscription was marked for cancellation"
          },
          "400": {
            "$ref": "#/responses/BadRequest"
          },
          "401": {
            "$ref": "#/responses/Unauthorized"
          },
          "403": {
            "$ref": "#/responses/Forbidden"
          },
          "404": {
            "$ref": "#/responses/NotFound"
          }
        },
        "security": [
          {
            "oauth2": [
              "w:subscriptions:*",
              "r:resources:*"
            ]
          }
        ]
      }
    },
    "/{eventsUrl}": {
      "post": {
        "tags": [
          "Events"
        ],
        "summary": "Events endpoint provided by the subscriber, where events are delivered",
        "description": "Events endpoint provided during subscription where events specified in the
subscription will be sent to  as defined per event type. Confirmation of each event sent to the
`eventsUrl` endpoint is required with `2xx` success code. Events you will receive based on event
type you're subscribed to are:\n - `subscription_canceled`: `SubscriptionCanceledEvent`\n -
`devices_registered`: `DevicesRegisteredEvent`\n - `devices_unregistered`:
`DevicesUnregisteredEvent`\n - `resources_published`: `ResourcesPublishedEvent`\n -
`resources_unpublished`: `ResourcesUnpublishedEvent`\n - `devices_online`: `DevicesOnlineEvent`\n -
`devices_offline`: `DevicesOfflineEvent`\n - `resource_contentchanged`:
`ResourceContentChangedEvent`",
        "parameters": [
          {
            "$ref": "#/parameters/CorrelationId"
          },
          {
            "$ref": "#/parameters/ContentType"
          },
          {
            "$ref": "#/parameters/EventType"
          },
          {
            "$ref": "#/parameters/SubscriptionId"
          },
          {
            "$ref": "#/parameters/SequenceNumber"
          },
          {
            "$ref": "#/parameters/EventSignature"
          },
          {
            "$ref": "#/parameters/EventTimestamp"
          },
          {
            "$ref": "#/parameters/EventsUrl"
          },
          {
            "$ref": "#/parameters/Event"
          }
        ],
        "consumes": [
          "application/json",
          "application/vnd.ocf+cbor"
        ],
        "responses": {
          "200": {
            "description": "Event successfully recieved"
          },
          "400": {
            "$ref": "#/responses/BadRequest"
          },
          "410": {
            "description": "The subscription identified by the Subscription-ID header is no more in
demand and shall be canceled"
          }
        }
      }
    }
  },
  "securityDefinitions": {
    "oauth2": {
      "type": "oauth2",
      "flow": "accessCode",
      "authorizationUrl": "https://example.com/api/oauth/dialog",
      "tokenUrl": "https://example.com/api/oauth/token",
      "scopes": {
        "r:deviceinformation:*": "Read basic device information",
        "r:resources:*": "Read content of published resource",
        "w:resources:*": "Update content of published resource",
        "w:subscriptions:*": "Create subscriptions"
      }
    }
  },
  "parameters": {
    "CorrelationId": {
      "name": "Correlation-ID",
      "in": "header",
      "type": "string",
      "format": "uuid",
      "description": "A Correlation ID, also known as a Transit ID, is a unique identifier value
that is attached to requests and messages that allow reference to a particular transaction or event
chain.\n"
    },
    "ContentType": {
      "name": "Content-Type",
      "in": "header",
      "type": "string",
      "enum": [
        "application/json",
        "application/vnd.ocf+cbor"
      ],
      "required": true,
      "description": "The Content-Type header is used to indicate the media type of the resource. In
responses, a Content-Type header tells the client what the content type of the returned content
actually is. In requests, (such as POST), the client tells the server what type of data is actually
sent.\n"
    },
    "Accept": {
      "name": "Accept",
      "in": "header",
      "type": "string",
      "enum": [
        "application/json",
        "application/vnd.ocf+cbor"
      ],
      "description": "The Accept request header can be used to specify certain media types which are
acceptable for the response. Accept headers can be used to indicate that the request is specifically
limited to a small set of desired types.\n"
    },
    "SubscriptionId": {
      "name": "Subscription-ID",
      "in": "header",
      "description": "Unique id of the subscription",
      "type": "string",
      "format": "uuid",
      "required": true
    },
    "SequenceNumber": {
      "name": "Sequence-Number",
      "in": "header",
      "description": "Sequence number of the event; first event starting with number 0",
      "type": "string",
      "required": true
    },
    "EventSignature": {
      "name": "Event-Signature",
      "in": "header",
      "description": "The signature created by combining the `signingSecret` from the subscription
request, headers and the body of the request using a stanard HMAC-SHA256 keyed hash.",
      "type": "string",
      "required": true
    },
    "EventTimestamp": {
      "name": "Event-Timestamp",
      "in": "header",
      "description": "Time when the event occurred in standard Unix time format",
      "type": "string",
      "required": true
    },
    "EventType": {
      "name": "Event-Type",
      "in": "header",
      "type": "string",
      "enum": [
        "subscription_canceled",
        "devices_registered",
        "devices_unregistered",
        "resource_contentchanged",
        "resources_published",
        "resources_unpublished",
        "devices_online",
        "devices_offline"
      ],
      "required": true
    },
    "DeviceType": {
      "description": "Filter devices by device type",
      "name": "rt",
      "in": "query",
      "type": "array",
      "items": {
        "type": "string"
      }
    },
    "ResourceLinkHref": {
      "description": "Path to resource",
      "name": "resourceLinkHref",
      "in": "path",
      "type": "string",
      "required": true
    },
    "DeviceId": {
      "description": "Id of the device",
      "name": "deviceId",
      "in": "path",
      "type": "string",
      "format": "uuid",
      "required": true
    },
    "SubscriptionIdPath": {
      "name": "subscriptionId",
      "in": "path",
      "type": "string",
      "format": "uuid",
      "required": true
    },
    "BatchFormat": {
      "name": "content",
      "in": "query",
      "description": "Indicates to the recipient that the response payload shall be the resolved
(i.e. resource representation) Link and not the Link itself. Default is `base`. When requesting
`all`, additional scope `r:resources:*:*` is required",
      "type": "string",
      "enum": [
        "base",
        "all"
      ]
    },
    "EventsUrl": {
      "name": "eventsUrl",
      "type": "string",
      "in": "path",
      "required": true
    },
    "ResourceUpdateRequest": {
      "description": "Map of resource values to be updated",
      "name": "content",
      "in": "body",
      "schema": {
        "$ref": "#/definitions/ResourceUpdateRequest"
      },
      "required": true
    },
    "SubscribeRequest": {
      "name": "content",
      "in": "body",
      "schema": {
        "$ref": "#/definitions/SubscribeRequest"
      },
      "required": true
    },
    "Event": {
      "description": "Event of a specific type, based on what you are subscribed to",
      "name": "content",
      "in": "body",
      "schema": {
        "$ref": "#/definitions/ResourceContentChangedEvent"
      },
      "required": true
    }
  },
  "responses": {
    "Unauthorized": {
      "description": "Unauthorized"
    },
    "NotFound": {
      "description": "Not found"
    },
    "SubscriptionCancellationPending": {
      "description": "Subscription was marked for cancellation"
    },
    "Forbidden": {
      "description": "Insufficient permissions"
    },
    "BadRequest": {
      "description": "The request was malformed or badly constructed"
    },
    "ServiceUnavailable": {
      "description": "The service on the Target Cloud is unavailable for the reason indicated in the
diagnostic payload"
    },
    "GatewayTimeout": {
      "description": "The target Device is registered at the target Cloud, however the Device itself
is unavailable, offline, or otherwise unreachable. The response should include a Retry-After header
containing the time after which the request may be re-attempted. Additional information is indicated
in the diagnostic payload."
    }
  },
  "definitions": {
    "Device": {
      "type": "object",
      "properties": {
        "device": {
          "$ref": "https://raw.githubusercontent.com/ondrejtomcik/coreocf/Bugzilla-
2709/swagger2.0/oic.wk.d.swagger.json#/definitions/Device"
        },
        "status": {
          "$ref": "#/definitions/DeviceStatus"
        },
        "links": {
          "type": "array",
          "items": {
            "$ref": "https://raw.githubusercontent.com/ondrejtomcik/coreocf/Bugzilla-
2709/swagger2.0/oic.wk.res.swagger.json#/definitions/oic.oic-link"
          }
        }
      },
      "example": {
        "device": {
          "dmn": "Open Connectivity Foundation",
          "n": "Food safety sensor",
          "di": "53080a4f-5e3e-4291-802f-3436238232d2"
        },
        "status": "online",
        "links": [
          {
            "href": "/53080a4f-5e3e-4291-802f-3436238232d2/humidity",
            "rt": [
              "oic.r.humidity"
            ]
          },
          {
            "href": "/53080a4f-5e3e-4291-802f-3436238232d2/temperature",
            "rt": [
              "oic.r.temperature"
            ]
          }
        ]
      }
    },
    "DeviceContentAll": {
      "type": "object",
      "properties": {
        "device": {
          "$ref": "https://raw.githubusercontent.com/ondrejtomcik/coreocf/Bugzilla-
2709/swagger2.0/oic.wk.d.swagger.json#/definitions/Device"
        },
        "status": {
          "$ref": "#/definitions/DeviceStatus"
        },
        "links": {
          "type": "array",
          "items": {
            "type": "object",
            "properties": {
              "href": {
                "type": "string"
              },
              "rep": {
                "type": "object"
              }
            }
          }
        }
      },
      "example": {
        "device": {
          "dmn": "Open Connectivity Foundation",
          "n": "Food safety sensor",
          "di": "53080a4f-5e3e-4291-802f-3436238232d2"
        },
        "status": "online",
        "links": [
          {
            "href": "/humidity",
            "rep": {
              "humidity": 62,
              "desiredHumidity": 65
            }
          },
          {
            "href": "/temperature",
            "rep": {
              "temperature": 21,
              "units": "C"
            }
          }
        ]
      }
    },
    "DeviceStatus": {
      "description": "Device status available from the OCF Cloud, which tracks if the device has
opened TCP connection and is signed in",
      "type": "string",
      "enum": [
        "online",
        "offline"
      ]
    },
    "ResourceUpdateRequest": {
      "type": "string",
      "description": "Desired content of the resource",
      "example": "o29kZXNpcmVkSHVtaWRpdHkYPGV0eXBlc4Fub2ljLnIuaHVtaWRpdHloaHVtaWRpdHkYKA=="
    },
    "ResourceRetrieveResponse": {
      "type": "string",
      "description": "Content of the resource returned from the device",
      "example": "o29kZXNpcmVkSHVtaWRpdHkYPGV0eXBlc4Fub2ljLnIuaHVtaWRpdHloaHVtaWRpdHkYKA=="
    },
    "EventType": {
      "type": "string",
      "enum": [
        "subscription_canceled",
        "devices_registered",
        "devices_unregistered",
        "resource_contentchanged",
        "resources_published",
        "resources_unpublished",
        "devices_online",
        "devices_offline"
      ]
    },
    "SubscriptionId": {
      "description": "Unique id of the subscription",
      "type": "string",
      "format": "uuid"
    },
    "SubscribeRequest": {
      "type": "object",
      "properties": {
        "eventsUrl": {
          "$ref": "#/definitions/EventsUrl"
        },
        "eventTypes": {
          "type": "array",
          "items": {
            "$ref": "#/definitions/EventType"
          }
        },
        "signingSecret": {
          "type": "string",
          "maxLength": 32,
          "minLength": 32
        }
      },
      "required": [
        "eventsUrl",
        "eventTypes",
        "signingSecret"
      ],
      "example": {
        "eventsUrl": "https://events.example.com/",
        "eventTypes": [
          "devices_registered",
          "devices_unregistered"
        ]
      }
    },
    "SubscribeResponse": {
      "description": "Subscription was registered, waiting for verification",
      "type": "object",
      "properties": {
        "subscriptionId": {
          "$ref": "#/definitions/SubscriptionId"
        }
      },
      "required": [
        "subscriptionId"
      ],
      "example": {
        "subscriptionId": "1eeb465c-5e8d-4305-a366-bbf035fff671"
      }
    },
    "EventsUrl": {
      "type": "string",
      "format": "url",
      "example": "https://events.exaple.com/"
    },
    "SubscriptionCanceledEvent": {
      "type": "object",
      "description": "Subscription with provided id was canceled"
    },
    "DevicesRegisteredEvent": {
      "description": "Device was successfully signed up to the OCF Cloud, as defined in the 
`oic.sec.account`",
      "type": "object",
      "properties": {
        "content": {
          "type": "array",
          "items": {
            "properties": {
              "di": {
                "type": "string",
                "format": "uuid"
              }
            }
          }
        }
      }
    },
    "DevicesUnregisteredEvent": {
      "description": "Device was successfully signed off from the OCF Cloud, as defined in the 
`oic.sec.account`",
      "type": "object",
      "properties": {
        "content": {
          "type": "array",
          "items": {
            "properties": {
              "di": {
                "type": "string",
                "format": "uuid"
              }
            }
          }
        }
      }
    },
    "ResourcesPublishedEvent": {
      "type": "object",
      "properties": {
        "content": {
          "type": "array",
          "items": {
            "$ref": "https://raw.githubusercontent.com/ondrejtomcik/coreocf/Bugzilla-
2709/swagger2.0/oic.wk.res.swagger.json#/definitions/oic.oic-link"
          }
        }
      }
    },
    "ResourcesUnpublishedEvent": {
      "type": "object",
      "properties": {
        "content": {
          "type": "array",
          "items": {
            "$ref": "https://raw.githubusercontent.com/ondrejtomcik/coreocf/Bugzilla-
2709/swagger2.0/oic.wk.res.swagger.json#/definitions/oic.oic-link"
```

```
          }
        }
      }
    },
    "DevicesOnlineEvent": {
      "type": "object",
      "properties": {
        "content": {
          "type": "array",
          "items": {
            "properties": {
              "di": {
                "type": "string",
                "format": "uuid"
              }
            }
          }
        }
      }
    },
    "DevicesOfflineEvent": {
      "type": "object",
      "properties": {
        "content": {
          "type": "array",
          "items": {
            "properties": {
              "di": {
                "type": "string",
                "format": "uuid"
              }
            }
          }
        }
      }
    },
    "ResourceContentChangedEvent": {
      "type": "string",
      "description": "New Content of the resource returned from the device",
      "example": "o29kZXNpcmVkSHVtaWRpdHkYPGV0eXBlc4Fub2ljLnIuaHVtaWRpdHloaHVtaWRpdHkYKA=="
    }
  }
}
```