Oracle® Communications

Cloud Native Core, Certificate Management REST Specification Guide

Release 25.2.200
G48072-01
March 2026

Oracle Communications Cloud Native Core, Certificate Management REST Specification Guide, Release 25.2.200

G48072-01

Contents

Preface

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customer access to and use of Oracle support services will be pursuant to the terms and conditions specified in their Oracle order for the applicable services.

Diversity and Inclusion

Oracle is fully committed to diversity and inclusion. Oracle respects and values having a diverse workforce that increases thought leadership and innovation. As part of our initiative to build a more inclusive culture that positively impacts our employees, customers, and partners, we are working to remove insensitive terms from our products and documentation. We are also mindful of the necessity to maintain compatibility with our customers' existing technologies and the need to ensure continuity of service as Oracle's offerings and industry standards evolve. Because of these technical constraints, our effort to remove insensitive terms is ongoing and will take time and external cooperation.

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

My Oracle Support

My Oracle Support (https://support.oracle.com) is your initial point of contact for all product support and training needs. A representative at Customer Access Support can assist you with My Oracle Support registration.

Call the Customer Access Support main number at 1-800-223-1711 (toll-free in the US), or call the Oracle Support hotline for your local country from the list at http://www.oracle.com/us/support/contact/index.html. When calling, make the selections in the sequence shown below on the Support telephone menu:

1. Select **2** for New Service Request.
2. Select **3** for Hardware, Networking and Solaris Operating System Support.
3. Select one of the following options:
   - For Technical issues such as creating a new Service Request (SR), select **1**.
   - For Non-technical issues such as registration or assistance with My Oracle Support, select **2**.

You are connected to a live agent who can assist you with My Oracle Support registration and opening a support ticket.

My Oracle Support is available 24 hours a day, 7 days a week, 365 days a year.

Acronyms

The following table lists the acronyms and the terminologies used in the document:

Table Acronyms

Acronym	Description
3GPP	3rd Generation Partnership Project
API	Application Programming Interface
CA	Certification Authority is a trusted entity that issues Secure Sockets Layer (SSL) certificates. CAs are also called issuer in this document.
CCA	Client Credentials Assertions
CMP	Certificate Management Protocol
CMP Identity Certificate	Certificate that corresponds to and certifies the CMP Identity Key. It is included in the CMPv2 requests for authentication by CA.
CMP Identity Key	Private Key used by Certificate Management to sign the CMPv2 requests and establish trust between Certificate Management and CA.
CNC	Cloud Native Core
CNC Console	Cloud Native Configuration Console
DNS	Domain Name Server
ECC	Elliptic Curve Cryptography
EE	End Entity
HNC	Heirarchical Namespace Controller
HSM	Hardware Security Module
IDP	Identity Provider
IR	Initialization Requests
OCCM	Oracle Communications Certificate Management
PKI	Public Key Infrastructure
PoP	Proof of Possession
RA	Registration Authority
RSA	Rivest-Shamir-Adleman
SAN	Subject Alternative Name
URI	Uniform Resource Indicator
URN	Uniform Resource Name

What's New in This Guide

This section introduces the documentation updates for release 25.2.2xx.

Release 25.2.200 - G48072-01, March 2026

Added REST Specifications for the Delete Certificate Configuration with Secret functionality.

1 Introduction

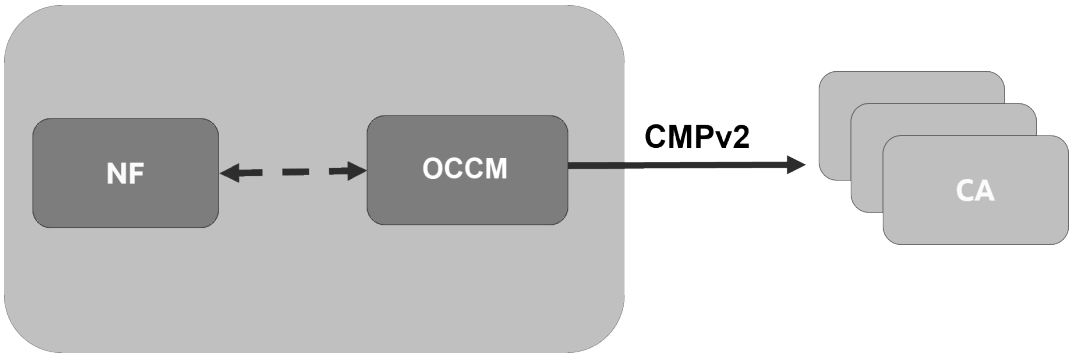
This document provides information about how to configure the services and manageable objects in OCCM using Representational State Transfer Application Program Interfaces (REST APIs).

1.1 Overview

OCCM integrates with the Certificate Authority(s) using Certificate Management Protocol Version 2 (CMPv2) and RFC4210 to facilitate these certificate management operations:

- Operator-initiated certificate creation
- Operator-initiated certificate recreation
- Automatic certificate monitoring and renewal

Figure 1-1 OCCM Integration with CA



OCCM supports transport of CMPv2 messages using HTTP-based protocol.

OCCM provides the following mechanisms to establish initial trust between OCCM and CA(s):

1. Certificate-based message signing
2. Pre-shared key or MAC based authentication

All the subsequent CMPv2 procedures are authenticated using the certificate-based mechanism in compliance with 3GPP TS 33.310.

The keys and X.509 certificates are managed using Kubernetes secrets.

1.2 Reference

Refer to the following documents for more information:

- *Oracle Communications Cloud Native Core, Certificate Management User Guide*
- *Oracle Communications Cloud Native Core, Certificate Management Installation, Upgrade, and Fault Recovery Guide*

- *Oracle Communications Cloud Native Core Security Guide*
- *Oracle Communications Cloud Native Core Solution Upgrade Guide*

2

OCCM Rest Specification

This chapter provides information about REST specifications used in Oracle Communications Cloud Native Core, Certificate Management

OCCM can be configured using Helm configurations, REST APIs, and Cloud Native Configuration Console (CNC Console). REST configurations can also be performed using the Cloud Native Configuration (CNC) Console.

For HELM configurations, see *Oracle Communications Cloud Native Core, Certificate Management Installation, Upgrade, and Fault Recovery Guide*.

For the configurations using CNC Console, see *Oracle Communications Cloud Native Core, Certificate Management User Guide.*

For installing OCCM in an existing NF deployment, see 'Introducing OCCM on an Existing NF Deployment' section in the *Oracle Communications Cloud Native Core, Certificate Management User Guide.*

2.1 OCCM Issuers

OCCM Issuers Data Model

Table 2-1 OCCM Issuers Request Parameters

Field Name	Data Type	Description
name	String	This is a mandatory parameter. Name of CA
server	String	This is a mandatory parameter. Domain URL of CA **Note**: The user must provide the port where the CA server is running, otherwise application will take it as default port 80.

Table 2-1 (Cont.) OCCM Issuers Request Parameters

Field Name	Data Type	Description
recipientDN	String	This is a mandatory parameter. Distinguished name (DN) of the CMP server (usually the addressed CA). Used in the recipient field of CMP request message headers. The argument must be formatted as /type0=value0/type1=value1/type2=.... Special characters may be escaped by \ (backslash); whitespace is retained. Empty values are permitted, but the corresponding type will not be included. Giving a single / will lead to an empty sequence of RDNs (a NULL-DN). Multi-valued RDNs can be formed by placing a + character instead of a / between the AttributeValueAssertions (AVAs) that specify the members of the set. Example: `/DC=org/DC=OpenSSL/DC=users/UID=123456+CN=John Doe`
issuerDN	String	This is an optional parameter. X509 issuer Distinguished Name of the CA server to place in the requested certificate template in IR/KUR. The argument must be formatted as /type0=value0/type1=value1/type2=.... Special characters may be escaped by \ (backslash); whitespace is retained. Empty values are permitted, but the corresponding type will not be included. Giving a single / will lead to an empty sequence of RDNs (a NULL-DN). Multi-valued RDNs can be formed by placing a + character instead of a / between the AttributeValueAssertions (AVAs) that specify the members of the set. Example: `/DC=org/DC=OpenSSL/DC=users/UID=123456+CN=John Doe`

Table 2-1 (Cont.) OCCM Issuers Request Parameters

Field Name	Data Type	Description
totalTimeOut	String	This is a mandatory parameter. Maximum total number of seconds a CMP transaction may take. Default Value: 720 seconds. Max value: 21600 seconds. **Note**: totalTimeOut should always be greater than messageTimeout.
messageTimeout	String	This is a mandatory parameter. Number of seconds a CMP request-response message round trip is allowed to take before a timeout error is returned. Default Value is 120 seconds. Max is 600 seconds. **Note**: messageTimeout should always be less than totalTimeOut.
cmpProtectionOccmCert	Object	This is a mandatory parameter except when OCCM certificate is manually configured. CMP client authentication options for OCCM certificate
cmpProtectionOccmCert.type	Enum	This is an optional parameter MAC or SIGNATURE **Possible values:** MAC SIGNATURE
cmpProtectionOccmCert.digestAlgorithm	Enum	This is a mandatory parameter except when cmpProtectionOccmCert.type selected is MAC. Supported digest to use. Default Value: SHA256 **Possible values:**SHA256, SHA384, SHA512
cmpProtectionOccmCert.macAlgorithm	Enum	This is a mandatory parameter except when cmpProtectionOccmCert.type selected is SIGNATURE) MAC algorithm to use. **Possible values:** HMACSHA256 HMACSHA384 HMACSHA512

Table 2-1 (Cont.) OCCM Issuers Request Parameters

Field Name	Data Type	Description
cmpProtectionOccmCert.macK8sSecretIn	Object	This is a mandatory parameter except when cmpProtectionOccmCert.type selected is SIGNATURE. Kubernetes secret input details for MAC based authentication of OCCM certificate.
cmpProtectionOccmCert.macK8sSecretIn.namespace	String	This is a mandatory parameter. Kubernetes secret namespace where MAC secret is present.
cmpProtectionOccmCert.macK8sSecretIn.namespace	String	This is a mandatory parameter. Name of Kubernetes secret holding MAC secret (pre-shared key) and reference information.
cmpProtectionOccmCert.macK8sSecretIn.passKey	String	This is a mandatory parameter. Kubernetes secret data key against which MAC secret is provided.
cmpProtectionOccmCert.macK8sSecretIn.refKey	String	This is a mandatory parameter. Kubernetes secret data key against which reference string is provided.
cmpProtectionOccmCert.signK8sSecretIn	Object	This is a mandatory parameter except when cmpProtectionOccmCert.type selected is MAC. Kubernetes secret input details for Signature based authentication of OCCM cert.
cmpProtectionOccmCert.signK8sSecretIn.namespace	String	This is a mandatory parameter. Kubernetes secret namespace where OCCM Sign secret is present.
cmpProtectionOccmCert.signK8sSecretIn.name	String	This is a mandatory parameter. Name of Kubernetes secret holding pre-configured private key and certificate.
cmpProtectionOccmCert.signK8sSecretIn.key	String	This is a mandatory parameter. Kubernetes secret data key against which the pre-configured private key file (private key file for the client's current CMP signer certificate) is provided.
cmpProtectionOccmCert.signK8sSecretIn.cert	String	This is a mandatory parameter. Kubernetes secret data key against which the pre-configured certificate (client's current CMP signer certificate) is provided.

Table 2-1 (Cont.) OCCM Issuers Request Parameters

Field Name	Data Type	Description
cmpProtectionOccmCert.signK8sSecretIn.extraCerts	Object	This is an optional parameter. List of Kubernetes secret data keys against which the certificates to append in the extraCerts field can be provided. They can be used as the default CMP signer certificate chain to include.
cmpProtectionOtherCert	Object	This is a mandatory parameter. CMP client authentication options for Other(NF) certificate
cmpProtectionOtherCert.type	Enum	This is a mandatory parameter. **Possible Value:** SIGNATURE
cmpProtectionOtherCert.digestAlgorithm	Enum	This is a mandatory parameter. Supported digest to use. Default Value: SHA256 **Possible values:**SHA256, SHA384, SHA512
cmpProtectionOtherCert.signK8sSecretIn	Object	This is a mandatory parameter. Kubernetes secret input details for Signature based authentication of Other (NF) cert.
cmpProtectionOtherCert.signK8sSecretIn.namespace	String	This is a mandatory parameter. Kubernetes secret namespace where NF Sign secret is present.
cmpProtectionOtherCert.signK8sSecretIn.name	String	This is a mandatory parameter. Name of Kubernetes secret holding OCCM key and cert information.
cmpProtectionOtherCert.signK8sSecretIn.key	String	This is a mandatory parameter. Kubernetes secret data key against which OCCM key is provided/created based on whether OCCM cert is created in manual or automatic mode.
cmpProtectionOtherCert.signK8sSecretIn.cert	String	This is a mandatory parameter. Kubernetes secret data key against which OCCM certificate is provided/created based on whether OCCM cert is created in manual or automatic mode.
cmpProtectionOtherCert.signK8sSecretIn.extraCerts	Array	This is an optional parameter. List of Kubernetes secret data keys against which the certificates to append in the extraCerts field can be provided or will be created (if received from CA) along with the OCCM cert, based on whether OCCM cert is created in manual or automatic mode.

Table 2-1 (Cont.) OCCM Issuers Request Parameters

Field Name	Data Type	Description
occmTrustStoreK8sSecretIn	Object	This is a mandatory parameter. Kubernetes secret input which holds OCCM trust store information(CA certificates). Used to validate CMP response messages.
occmTrustStoreK8sSecretIn.namespace	String	This is a mandatory parameter. Kubernetes secret namespace where OCCM trust store secret is present.
occmTrustStoreK8sSecretIn.name	String	This is a mandatory parameter. Name of Kubernetes OCCM trust store secret.
occmTrustStoreK8sSecretIn.rootCACerts	List<String>	This is an optional parameter except if occmTrustStoreK8sSecretIn.serverCert is provided. The certificate(s), typically of root CAs, the client shall use as trust anchors when validating the certificate issued by CA. Note: If server cert is present this is ignored.
occmTrustStoreK8sSecretIn.intCACerts	List<String>	This is an optional parameter. Any non-trusted intermediate CA certificate(s) to use when validating newly enrolled certificates.
occmTrustStoreK8sSecretIn.serverCert	String	This is an optional parameter except if occmTrustStoreK8sSecretIn.rootCACerts is provided. CMP/CA server's certificate to expect and directly trust when validating the certificate issued by CA. Note: If this is present root CA certs will be ignored.
uuid	String	Unique ID for logging and tracking purpose
tlsConfig.enableTLS	boolean	This is an optional parameter. This field when set true "-tls_used" will be included in openssl cmp cmd for TLS communication with CA. Server URL should include https scheme **Possible values:**true, false

Table 2-1 (Cont.) OCCM Issuers Request Parameters

Field Name	Data Type	Description
tlsConfig.tlsTrustStoreK8sSecretItem.namespace	String	This parameter is mandatory when enableTLS is set to true and optional when enableTLS is set to false. Kubernetes secret namespace where TLS trust store secret is present.
tlsConfig.tlsTrustStoreK8sSecretItem.name	String	This parameter is mandatory when enableTLS is set to true and optional when enableTLS is set to false. Name of Kubernetes TLS trust store secret.
tlsConfig.tlsTrustStoreK8sSecretItem.tlsTrustedCerts	List<String>	This parameter is mandatory when enableTLS is set to true and optional when enableTLS is set to false. Certificate(s) used for validating the certificate presented by CA(s) during TLS handshake.

Table 2-2 OCCM Issuers Response Codes

Response codes	Data type	Cardinality	Description
202 Accepted	Object (Issuers)	1	This is a mandatory parameter Object (Issuers)
200 OK	Object (Issuers) Or List(Issuers)	1	This is a mandatory parameter Object (Issuers) Or List(Issuers) matching criteria
400 Bad request	Problem Details	1	This is a mandatory parameter Invalid issuer configuration
500 Internal Server Error	Problem Details	1	This is a mandatory parameter Something went wrong
409 Conflict	Problem Details	1	This is a mandatory parameter Record already exists
404 Not Found	Problem Details	1	This is a mandatory parameter Queried Object(Issuer) not present

> **Note**
>
> OCCM Issuers response body data model varies based on REST operation status.

OCCM Issuers JSON Payload

```
{
    "name": "",
    "server": "",
    "recipientDN": "",
    "issuerDN": "",
    "totalTimeout": "",
    "messageTimeout": "",
    "cmpProtectionOccmCert": {
        "type": null,
        "digestAlgorithm": null,
        "macAlgorithm": null,
        "macK8sSecretIn": {
            "namespace": "",
            "name": "",
            "passKey": "",
            "refKey": ""
        },
        "signK8sSecretIn": {
            "namespace": "",
            "name": "",
            "key": "",
            "cert": "",
            "extraCerts": []
        }
    },
    "cmpProtectionOtherCert": {
        "type": "",
        "digestAlgorithm": "",
        "signK8sSecretIn": {
            "namespace": "",
            "name": "",
            "key": "",
            "cert": "",
            "extraCerts": []
        }
    },
    "occmTrustStoreK8sSecretIn": {
        "namespace": "",
        "name": "",
        "rootCACerts": [],
        "intCACerts": [],
        "serverCert": ""
    },
        "tlsConfig":{
            "enableTLS":true,
                "tlsTrustStoreK8sSecretItem":{
                    "namespace": "",
                    "name": "",
```

```
                    "tlsTrustedCerts":[]
                }
}
```

2.1.1 Fetch All Issuers

OCCM Uses the GET operation to fetch all issuer details.

Resource URI:/occm-config/v1/issuers

Table 2-3 Data structures supported by the GET Response Body on this resource

Response codes	Data type	Cardinality	Description
200 OK	List (Issuers)	1	This is a mandatory parameter. Indicates all the issuer configurations are successfully fetched.
500 Internal Server Error	Problem Details	1	This is a mandatory parameter. Indicates that something went wrong while fetching the issuer configurations.

Sample GET Request:

```
$ curl --location --request GET 'http://{host}:{port}/occm-config/v1/issuers'
\
--header 'Oc-Cncc-Id: Cluster1' \
--header 'Oc-Cncc-Instance-Id: Cluster1-OCCM-instance1' \
--header 'Authorization: Bearer eyJhbGciOiJSUzI1Ni'
```

Sample GET Response:

```
200 OK Response Body: '[
      {
        "uuid": "d692b217-00ca-433b-819d-cb038dba256b",
        "name": "CA1",
        "server": "https://ca1.example.com:8445/cmp/alias",
        "recipientDN": "/CN=CMP Server CA1",
        "issuerDN": "",
        "totalTimeout": "120",
        "messageTimeout": "30",
        "cmpProtectionOccmCert": {
            "type": "MAC",
            "digestAlgorithm": "SHA256",
            "macAlgorithm": "HMACSHA256",
            "macK8sSecretIn": {
                "namespace": "ns1",
                "name": "ca1-mac-secret",
                "passKey": "pwd",
```

```
                "refKey": "ref"
            },
            "signK8sSecretIn": {
                "namespace": "",
                "name": "",
                "key": "",
                "cert": "",
                "extraCerts": []
            }
        },
        "cmpProtectionOtherCert": {
            "type": "SIGNATURE",
            "digestAlgorithm": "SHA384",
            "signK8sSecretIn": {
                "namespace": "ns1",
                "name": "ca1-occm-key-cert-secret",
                "key": "occmkey.pem",
                "cert": "occm.cer",
                "extraCerts": []
            }
        },
        "occmTrustStoreK8sSecretIn": {
            "namespace": "ns1",
            "name": "ca1-occm-trust-store-secret",
            "rootCACerts": [
                "caroot.pem"
            ],
            "intCACerts": [],
            "serverCert": ""
        },
        "tlsConfig": {
            "enableTLS": true,
            "tlsTrustStoreK8sSecretItem": {
                "namespace": "ns1",
                "name": "ca1-tls-trust-store-secret",
                "tlsTrustedCerts": [
                    "tlscaroot.pem"

                ]
            }
        }
    },
    {
        "uuid": "e871a123-11ca-433b-929e-ab049cbd357d",
        "name": "CA2",
        "server": "https://ca2.example.com:8446/cmp/alias",
        "recipientDN": "/CN=CMP Server CA2",
        "issuerDN": "",
        "totalTimeout": "120",
        "messageTimeout": "30",
        "cmpProtectionOccmCert": {
            "type": "MAC",
            "digestAlgorithm": "SHA256",
            "macAlgorithm": "HMACSHA256",
            "macK8sSecretIn": {
                "namespace": "ns1",
```

```
                "name": "ca2-mac-secret",
                "passKey": "pwd",
                "refKey": "ref"
            },
            "signK8sSecretIn": {
                "namespace": "",
                "name": "",
                "key": "",
                "cert": "",
                "extraCerts": []
            }
        },
        "cmpProtectionOtherCert": {
            "type": "SIGNATURE",
            "digestAlgorithm": "SHA384",
            "signK8sSecretIn": {
                "namespace": "ns1",
                "name": "ca2-occm-key-cert-secret",
                "key": "occmkey.pem",
                "cert": "occm.cer",
                "extraCerts": []
            }
        },
        "occmTrustStoreK8sSecretIn": {
            "namespace": "ns1",
            "name": "ca2-occm-trust-store-secret",
            "rootCACerts": [
                "caroot.pem"
            ],
            "intCACerts": [],
            "serverCert": ""
        },
        "tlsConfig": {
            "enableTLS": true,
            "tlsTrustStoreK8sSecretItem": {
                "namespace": "ns1",
                "name": "ca2-tls-trust-store-secret",
                "tlsTrustedCerts": [
                    "tlscaroot.pem"

                ]
            }
        }
    },
    {
        "uuid": "p783a138-02bm-544d-928e-fv059rsa268z",
        "name": "CA3",
        "server": "https://ca3.example.com:8447/cmp/alias",
        "recipientDN": "/CN=CMP Server CA3",
        "issuerDN": "",
        "totalTimeout": "120",
        "messageTimeout": "30",
        "cmpProtectionOccmCert": {
            "type": "MAC",
            "digestAlgorithm": "SHA256",
            "macAlgorithm": "HMACSHA256",
```

```
            "macK8sSecretIn": {
                "namespace": "ns1",
                "name": "ca3-mac-secret",
                "passKey": "pwd",
                "refKey": "ref"
            },
            "signK8sSecretIn": {
                "namespace": "",
                "name": "",
                "key": "",
                "cert": "",
                "extraCerts": []
            }
        },
        "cmpProtectionOtherCert": {
            "type": "SIGNATURE",
            "digestAlgorithm": "SHA384",
            "signK8sSecretIn": {
                "namespace": "ns1",
                "name": "ca3-occm-key-cert-secret",
                "key": "occmkey.pem",
                "cert": "occm.cer",
                "extraCerts": []
            }
        },
        "occmTrustStoreK8sSecretIn": {
            "namespace": "ns1",
            "name": "ca3-occm-trust-store-secret",
            "rootCACerts": [
                "caroot.pem"
            ],
            "intCACerts": [],
            "serverCert": ""
        },
        "tlsConfig": {
            "enableTLS": true,
            "tlsTrustStoreK8sSecretItem": {
                "namespace": "ns1",
                "name": "ca-tls-trust-store-secret",
                "tlsTrustedCerts": [
                    "tlscaroot.pem"

                ]
            }
        }
    }
    ]'
```

2.1.2 Fetch Issuers by UUID

OCCM uses the GET operation to fetch issuer configuration by UUID.

Resource URI:/occm-config/v1/issuers

Table 2-4 Path Parameter

Field Name	Data Type	Mandatory (M) or Optional (O)	Description
uuid	String	M	Indicates the issuer UUID.

Data structures supported by the GET Response Body on this resource

Response codes	Data type	Cardinality	Description
200 OK	Issuers	1	This is a mandatory parameter. Indicates that the issuer configuration identified by the given UUID is successfully fetched.
404 NOT FOUND	Problem Details	1	This is a mandatory parameter. Indicates that there is no issuer configuration with the given UUID.
500 Internal Server Error	Problem Details	1	This is a mandatory parameter. Indicates that something went wrong while fetching the issuer configuration.

Sample GET Request:

```
$ curl --location --request GET 'http://{host}:{port}/occm-config/v1/issuers/
4c5b4025-6c63-438c-bcd7-27b5bf8da4fd' \
--header 'Oc-Cncc-Id: Cluster1' \
--header 'Oc-Cncc-Instance-Id: Cluster1-OCCM-instance1' \
--header 'Authorization: Bearer eyJhbGciOiJSUzI1NiIs'
```

Sample GET Response:

```
200 OK Response Body: '{
        "uuid": "4c5b4025-6c63-438c-bcd7-27b5bf8da4fd",
        "name": "CA1",
        "server": "http://ca1-openssl-mock.ns1.svc.local:8080",
        "recipientDN": "/CN=svc.local",
        "issuerDN": "/CN=svc.local",
        "totalTimeout": "720",
        "messageTimeout": "120",
        "cmpProtectionOccmCert": {
            "type": null,
            "digestAlgorithm": null,
            "macAlgorithm": null,
            "macK8sSecretIn": {
```

```
                "namespace": "",
                "name": "",
                "passKey": "",
                "refKey": ""
            },
            "signK8sSecretIn": {
                "namespace": "",
                "name": "",
                "key": "",
                "cert": "",
                "extraCerts": []
            }
        },
        "cmpProtectionOtherCert": {
            "type": "SIGNATURE",
            "digestAlgorithm": "SHA256",
            "signK8sSecretIn": {
                "namespace": "ns1",
                "name": "ca1-cmp-identity-secret",
                "key": "cmpkey.pem",
                "cert": "cmpcert.pem",
                "extraCerts": []
            }
        },
        "occmTrustStoreK8sSecretIn": {
            "namespace": "ns1",
            "name": "ca1-occm-trust-store-secret",
            "rootCACerts": [
                "caroot.cer"
            ],
            "intCACerts": [
                "intca.cer"
            ],
            "serverCert": ""
        },
        "tlsConfig":{
            "enableTLS":false,
                "tlsTrustStoreK8sSecretItem":{
                    "namespace": "",
                    "name": "",
                    "tlsTrustedCerts":[""]
                }
        }
}'
```

2.1.3 Add Issuer Configurations

OCCM Uses the POST operation to create a new issuer using the configuration in the request.

Resource URI: /occm-config/v1/issuers

Table 2-5 Data structures supported by the POST Response Body on this resource

Response codes	Data type	Cardinality	Description
201 CREATED	Issuers	1	This is a mandatory parameter. Indicates that the issuer configuration is created successfully.
400 BAD REQUEST	Problem Details	1	This is a mandatory parameter. Indicates invalid request data
409 CONFLICT	Problem Details	1	This is a mandatory parameter. Indicates that the issuer already exists with the name provided in the request.
500 Internal Server Error	Problem Details	1	This is a mandatory parameter. Indicates that something went wrong while creating the issuer configuration.

Sample POST Request when Enable TLS is False:

```
$ curl --location --request POST 'http://{host}:{port}/occm-config/v1/
issuers' \
--header 'Oc-Cncc-Id: Cluster1' \
--header 'Oc-Cncc-Instance-Id: Cluster1-OCCM-instance1' \
--header 'Authorization: Bearer LxuLeX9dihXDUcoFwDw' \
--header 'Content-Type: application/json' \
--data-raw '{
    "name": "CA1",
    "server": "http://ca1-openssl-mock.ns1.svc.local:8080",
    "recipientDN": "/CN=svc.local",
    "issuerDN": "/CN=svc.local",
    "totalTimeout": "60",
    "messageTimeout": "30",
    "cmpProtectionOccmCert": {
        "type": null,
        "digestAlgorithm": null,
        "macAlgorithm": null,
        "macK8sSecretIn": {
            "namespace": "",
            "name": "",
            "passKey": "",
            "refKey": ""
        },
        "signK8sSecretIn": {
            "namespace": "",
```

```
            "name": "",
            "key": "",
            "cert": "",
            "extraCerts": []
        }
    },
    "cmpProtectionOtherCert": {
        "type": "SIGNATURE",
        "digestAlgorithm": "SHA256",
        "signK8sSecretIn": {
            "namespace": "ns1",
            "name": "ca1-occm-key-cert-secret",
            "key": "occmkey.pem",
            "cert": "occm.cer",
            "extraCerts": []
        }
    },
    "occmTrustStoreK8sSecretIn": {
        "namespace": "ns1",
        "name": "ca1-occm-trust-store-secret",
        "rootCACerts": ["caroot.cer"],
        "intCACerts": ["intca.cer"],
        "serverCert": ""
    },
        "tlsConfig":{
            "enableTLS":false,
                "tlsTrustStoreK8sSecretItem":{
                    "namespace": "",
                    "name": "",
                    "tlsTrustedCerts":[""]
                }
        }
}'
```

Sample POST Response:

```
 202 Accepted Response Body: '{
    "uuid": "4c5b4025-6c63-438c-bcd7-27b5bf8da4fd",
    "name": "CA1",
    "server": "http://ca1-openssl-mock.ns1.svc.local:8080",
    "recipientDN": "/CN=svc.local",
    "issuerDN": "/CN=svc.local",
    "totalTimeout": "60",
    "messageTimeout": "30",
    "cmpProtectionOccmCert": {
        "type": null,
        "digestAlgorithm": null,
        "macAlgorithm": null,
        "macK8sSecretIn": {
            "namespace": "",
            "name": "",
            "passKey": "",
            "refKey": ""
        },
        "signK8sSecretIn": {
```

```
                "namespace": "",
                "name": "",
                "key": "",
                "cert": "",
                "extraCerts": []
            }
        },
        "cmpProtectionOtherCert": {
            "type": "SIGNATURE",
            "digestAlgorithm": "SHA256",
            "signK8sSecretIn": {
                "namespace": "ns1",
                "name": "ca1-occm-key-cert-secret",
                "key": "occmkey.pem",
                "cert": "occm.cer",
                "extraCerts": []
            }
        },
        "occmTrustStoreK8sSecretIn": {
            "namespace": "ns1",
            "name": "ca1-occm-trust-store-secret",
            "rootCACerts": ["caroot.cer"],
            "intCACerts": ["intca.cer"],
            "serverCert": ""
        },
            "tlsConfig":{
                "enableTLS":false,
                    "tlsTrustStoreK8sSecretItem":{
                        "namespace": "",
                        "name": "",
                        "tlsTrustedCerts":[""]
                    }
            }
}'
```

Sample POST request when Enable TLS is True (HTTPS):

```
$ curl --location --request POST 'http://{host}:{port}/occm-config/v1/
issuers' \
--header 'Oc-Cncc-Id: Cluster1' \
--header 'Oc-Cncc-Instance-Id: Cluster1-OCCM-instance1' \
--header 'Authorization: Bearer LxuLeX9dihXDUcoFwDw' \
--header 'Content-Type: application/json' \
--data-raw '{
    "name": "CA-TLS-1",
    "server": "https://ca1.example.com:8443/cmp/occmalias",
    "recipientDN": "/CN=svc.local",
    "issuerDN": "",
    "totalTimeout": "60",
    "messageTimeout": "30",
    "cmpProtectionOccmCert": {
        "type": null,
        "digestAlgorithm": null,
        "macAlgorithm": null,
        "macK8sSecretIn": {
```

```
                "namespace": "",
                "name": "",
                "passKey": "",
                "refKey": ""
            },
            "signK8sSecretIn": {
                "namespace": "",
                "name": "",
                "key": "",
                "cert": "",
                "extraCerts": []
            }
        },
        "cmpProtectionOtherCert": {
            "type": "SIGNATURE",
            "digestAlgorithm": "SHA256",
            "signK8sSecretIn": {
                "namespace": "ns1",
                "name": "ca1-occm-key-cert-secret",
                "key": "occmkey.pem",
                "cert": "occm.cer",
                "extraCerts": []
            }
        },
        "occmTrustStoreK8sSecretIn": {
            "namespace": "ns1",
            "name": "ca1-occm-trust-store-secret",
            "rootCACerts": ["caroot.cer"],
            "intCACerts": ["intca.cer"],
            "serverCert": ""
        },
            "tlsConfig":{
                "enableTLS":true,
                    "tlsTrustStoreK8sSecretItem":{
                        "namespace": "ns1",
                        "name": "ca1-tls-trust-store-secret",
                        "tlsTrustedCerts":["tlscaroot.pem"]
                    }
            }
}'
```

Sample POST Response:

```
200 Success Response Body: '{
    "uuid": "4c5b4025-6c63-438c-bcd7-27b5bf8da4fd",
    "name": "CA-TLS-1",
  "server": "https://ca1.example.com:8443/cmp/occmalias",
    "recipientDN": "/CN=svc.local",
    "issuerDN": "",
    "totalTimeout": "60",
    "messageTimeout": "30",
    "cmpProtectionOccmCert": {
        "type": null,
        "digestAlgorithm": null,
        "macAlgorithm": null,
```

```
            "macK8sSecretIn": {
                "namespace": "",
                "name": "",
                "passKey": "",
                "refKey": ""
            },
            "signK8sSecretIn": {
                "namespace": "",
                "name": "",
                "key": "",
                "cert": "",
                "extraCerts": []
            }
        },
        "cmpProtectionOtherCert": {
            "type": "SIGNATURE",
            "digestAlgorithm": "SHA256",
            "signK8sSecretIn": {
                "namespace": "ns1",
                "name": "ca1-occm-key-cert-secret",
                "key": "occmkey.pem",
                "cert": "occm.cer",
                "extraCerts": []
            }
        },
        "occmTrustStoreK8sSecretIn": {
            "namespace": "ns1",
            "name": "ca1-occm-trust-store-secret",
            "rootCACerts": ["caroot.cer"],
            "intCACerts": ["intca.cer"],
            "serverCert": ""
        },
            "tlsConfig":{
                "enableTLS":true,
                    "tlsTrustStoreK8sSecretItem":{
                        "namespace": "ns1",
                        "name": "ca1-tls-trust-store-secret",
                        "tlsTrustedCerts":["tlscaroot.pem"]
                    }
            }
}'
```

2.1.4 Update Issuer Configurations

OCCM uses the PUT operation to update the existing issuer configuration identified by its UUID.

Resource URI: /occm-config/v1/issuers/{uuid}

All fields of the issuer configuration can be edited, if no certificate configuration is attached to it.

However, if any certification configuration is mapped to the given issuer, only the HTTP scheme can be updated (HTTP to HTTPS and vice versa).

Table 2-6 Path Parameter

Name	Data Type	Mandatory (M) or Optional (O)	Cardinality	Description
uuid	String	M	1	Indicates the issuer UUID

Table 2-7 Data structures supported by the PUT Response Body on this resource

Response codes	Data type	Cardinality	Description
200 OK	Issuers	1	This is a mandatory parameter. Indicates that the issuer identified by the given UUID is updated successfully.
400 BAD REQUEST	Problem Details	1	This is a mandatory parameter. Indicates invalid request data.
404 NOT FOUND	Problem Details	1	This is a mandatory parameter. Indicates that there is no issuer with the given UUID.
500 Internal Server Error	Problem Details	1	This is a mandatory parameter. Indicates that something went wrong while updating the issuer configuration.

Sample PUT Request when Enable TLS is False:

```
$ curl --location --request PUT 'http://{host}:{port}/occm-config/v1/issuers/
4c5b4025-6c63-438c-bcd7-27b5bf8da4fd' \
--header 'Oc-Cncc-Id: Cluster1' \
--header 'Oc-Cncc-Instance-Id: Cluster1-OCCM-instance1' \
--header 'Authorization: Bearer LxuLeX9dihXDUcoFwDw' \
--header 'Content-Type: application/json' \
--data-raw '{
    "uuid": "4c5b4025-6c63-438c-bcd7-27b5bf8da4fd",
    "name": "CA1",
    "server": "http://ca1-openssl-mock.ns1.svc.local:8080",
    "recipientDN": "/CN=svc.local",
    "issuerDN": "/CN=svc.local",
    "totalTimeout": "60",
    "messageTimeout": "30",
    "cmpProtectionOccmCert": {
        "type": null,
        "digestAlgorithm": null,
        "macAlgorithm": null,
```

```
            "macK8sSecretIn": {
                "namespace": "",
                "name": "",
                "passKey": "",
                "refKey": ""
            },
            "signK8sSecretIn": {
                "namespace": "",
                "name": "",
                "key": "",
                "cert": "",
                "extraCerts": []
            }
        },
        "cmpProtectionOtherCert": {
            "type": "SIGNATURE",
            "digestAlgorithm": "SHA256",
            "signK8sSecretIn": {
                "namespace": "ns1",
                "name": "ca1-occm-key-cert-secret",
                "key": "occmkey.pem",
                "cert": "occm.cer",
                "extraCerts": []
            }
        },
        "occmTrustStoreK8sSecretIn": {
            "namespace": "ns1",
            "name": "ca1-occm-trust-store-secret",
            "rootCACerts": ["caroot.cer"],
            "intCACerts": ["intca.cer"],
            "serverCert": ""
        },
            "tlsConfig":{
                "enableTLS":false,
                    "tlsTrustStoreK8sSecretItem":{
                        "namespace": "",
                        "name": "",
                        "tlsTrustedCerts":[""]
                    }
            }
}'
```

Sample PUT Response:

```
200 Success Response Body: '{
    "uuid": "4c5b4025-6c63-438c-bcd7-27b5bf8da4fd",
    "name": "CA1",
    "server": "http://ca1-openssl-mock.ns1.svc.local:8080",
    "recipientDN": "/CN=svc.local",
    "issuerDN": "/CN=svc.local",
    "totalTimeout": "60",
    "messageTimeout": "30",
    "cmpProtectionOccmCert": {
        "type": null,
        "digestAlgorithm": null,
```

```
            "macAlgorithm": null,
            "macK8sSecretIn": {
                "namespace": "",
                "name": "",
                "passKey": "",
                "refKey": ""
            },
            "signK8sSecretIn": {
                "namespace": "",
                "name": "",
                "key": "",
                "cert": "",
                "extraCerts": []
            }
        },
        "cmpProtectionOtherCert": {
            "type": "SIGNATURE",
            "digestAlgorithm": "SHA256",
            "signK8sSecretIn": {
                "namespace": "ns1",
                "name": "ca1-occm-key-cert-secret",
                "key": "occmkey.pem",
                "cert": "occm.cer",
                "extraCerts": []
            }
        },
        "occmTrustStoreK8sSecretIn": {
            "namespace": "ns1",
            "name": "ca1-occm-trust-store-secret",
            "rootCACerts": ["caroot.cer"],
            "intCACerts": ["intca.cer"],
            "serverCert": ""
        },
            "tlsConfig":{
                "enableTLS":false,
                    "tlsTrustStoreK8sSecretItem":{
                        "namespace": "",
                        "name": "",
                        "tlsTrustedCerts":[""]
                    }
            }
}'
```

Sample PUT Request when Enable TLS is true:

```
curl --location --request PUT 'http://{host}:{port}/occm-config/v1/issuers/
4c5b4025-6c63-438c-bcd7-27b5bf8da4fd' \
--header 'Oc-Cncc-Id: Cluster1' \
--header 'Oc-Cncc-Instance-Id: Cluster1-OCCM-instance1' \
--header 'Authorization: Bearer LxuLeX9dihXDUcoFwDw' \
--header 'Content-Type: application/json' \
--data-raw '{
    "uuid": "4c5b4025-6c63-438c-bcd7-27b5bf8da4fd",
    "name": "CA-TLS-1",
    "server": "https://ca1.example.com:8443/cmp/occmalias",
```

```
    "recipientDN": "/CN=svc.local",
    "issuerDN": "",
    "totalTimeout": "60",
    "messageTimeout": "30",
    "cmpProtectionOccmCert": {
        "type": null,
        "digestAlgorithm": null,
        "macAlgorithm": null,
        "macK8sSecretIn": {
            "namespace": "",
            "name": "",
            "passKey": "",
            "refKey": ""
        },
        "signK8sSecretIn": {
            "namespace": "",
            "name": "",
            "key": "",
            "cert": "",
            "extraCerts": []
        }
    },
    "cmpProtectionOtherCert": {
        "type": "SIGNATURE",
        "digestAlgorithm": "SHA256",
        "signK8sSecretIn": {
            "namespace": "ns1",
            "name": "ca1-occm-key-cert-secret",
            "key": "occmkey.pem",
            "cert": "occm.cer",
            "extraCerts": []
        }
    },
    "occmTrustStoreK8sSecretIn": {
        "namespace": "ns1",
        "name": "ca1-occm-trust-store-secret",
        "rootCACerts": ["caroot.cer"],
        "intCACerts": ["intca.cer"],
        "serverCert": ""
    },
        "tlsConfig":{
            "enableTLS":true,
                "tlsTrustStoreK8sSecretItem":{
                    "namespace": "ns1",
                    "name": "occm-tls-trust-store-secret",
                    "tlsTrustedCerts":["tlscaroot.pem"]
                }
        }
}'
```

Sample PUT Response:

```
200 Success Response Body: '{
    "uuid": "4c5b4025-6c63-438c-bcd7-27b5bf8da4fd",
    "name": "CA-TLS-1",
```

```
        "server": "https://ca1.example.com:8443/cmp/occmalias",
        "recipientDN": "/CN=svc.local",
        "issuerDN": "",
        "totalTimeout": "60",
        "messageTimeout": "30",
        "cmpProtectionOccmCert": {
            "type": null,
            "digestAlgorithm": null,
            "macAlgorithm": null,
            "macK8sSecretIn": {
                "namespace": "",
                "name": "",
                "passKey": "",
                "refKey": ""
            },
            "signK8sSecretIn": {
                "namespace": "",
                "name": "",
                "key": "",
                "cert": "",
                "extraCerts": []
            }
        },
        "cmpProtectionOtherCert": {
            "type": "SIGNATURE",
            "digestAlgorithm": "SHA256",
            "signK8sSecretIn": {
                "namespace": "ns1",
                "name": "ca1-occm-key-cert-secret",
                "key": "occmkey.pem",
                "cert": "occm.cer",
                "extraCerts": []
            }
        },
        "occmTrustStoreK8sSecretIn": {
            "namespace": "ns1",
            "name": "ca1-occm-trust-store-secret",
            "rootCACerts": ["caroot.cer"],
            "intCACerts": ["intca.cer"],
            "serverCert": ""
        },
            "tlsConfig":{
                "enableTLS":true,
                    "tlsTrustStoreK8sSecretItem":{
                        "namespace": "ns1",
                        "name": "occm-tls-trust-store-secret",
                        "tlsTrustedCerts":["tlscaroot.pem"]
                    }
            }
    }'
```

2.1.5 Delete Issuers Data

OCCM uses the DELETE operation to delete the issuer configuration based on the UUID.

> **Note**
>
> An issuer can only be deleted if there are no certificates referring to this issuer entry.

Resource URI: /occm-config/v1/issuers/{uuid}

Table 2-8 Path Parameter

Name	Data Type	Mandatory (M) or Optional (O)	Cardinality	Description
uuid	String	M	1	Issuer uuid

Table 2-9 Data structures supported by the DELETE Response Body on this resource

Response codes	Data type	Cardinality	Description
200 OK	Issuers	1	This is a mandatory parameter. Indicates that the issuer identified by the given UUID is deleted successfully.
400 Bad Request	Problem Details	1	This is a mandatory parameter. Indicates invalid request data.
404 NOT FOUND	Problem Details	1	This is a mandatory parameter. Indicates that there is no issuer with the given UUID.
500 Internal Server Error	Problem Details	1	This is a mandatory parameter. Indicates that something went wrong while deleting the issuer configuration.

Sample DELETE Request:

```
$ curl --location --request DELETE 'http://{host}:{port}/occm-config/v1/
issuers/4c5b4025-6c63-438c-bcd7-27b5bf8da4fd' \
--header 'oc-cncc-id: Cluster1' \
--header oc-cncc-instance-id: Cluster1-OCCM-instance1' \
--header 'Authorization: Bearer eyJhbGciOiJSUzI1NiIs.…...'
```

Sample DELETE Response:

```
200 OK Response Body: '{
    "uuid": "4c5b4025-6c63-438c-bcd7-27b5bf8da4fd",         {
    "name": "CA1",
    "server": "http://ca1-openssl-mock.ns1.svc.local:8080",
```

```
        "recipientDN": "/CN=svc.local",
        "issuerDN": "/CN=svc.local",
        "totalTimeout": "60",
        "messageTimeout": "30",
        "cmpProtectionOccmCert": {
            "type": null,
            "digestAlgorithm": null,
            "macAlgorithm": null,
            "macK8sSecretIn": {
                "namespace": "",
                "name": "",
                "passKey": "",
                "refKey": ""
            },
            "signK8sSecretIn": {
                "namespace": "",
                "name": "",
                "key": "",
                "cert": "",
                "extraCerts": []
            }
        },
        "cmpProtectionOtherCert": {
            "type": "SIGNATURE",
            "digestAlgorithm": "SHA256",
            "signK8sSecretIn": {
                "namespace": "ns1",
                "name": "ca1-occm-key-cert-secret",
                "key": "occmkey.pem",
                "cert": "occm.cer",
                "extraCerts": []
            }
        },
        "occmTrustStoreK8sSecretIn": {
            "namespace": "ns1",
            "name": "ca1-occm-trust-store-secret",
            "rootCACerts": ["caroot.cer"],
            "intCACerts": ["intca.cer"],
            "serverCert": ""
        },
            "tlsConfig":{
                "enableTLS":false,
                    "tlsTrustStoreK8sSecretItem":{
                        "namespace": "",
                        "name": "",
                        "tlsTrustedCerts":[""]
                    }
            }
    }'
```

2.2 OCCM Certificates

OCCM Certificates Data Model

Table 2-10 OCCM Certificates Request Parameters

Field Name	DataType	Description
name	String	This is a mandatory parameter. Name of the certificate
lcmType	Enum	This is a mandatory parameter. **Possible Values:** AUTOMATIC, MANUAL
certType	Enum	This is a mandatory parameter. **Possible Values:** OCCM, OTHER
renewBefore	String	This is an optional parameter. Number of days before the certificate expiry, when the certificate will be renewed. Default Value 14 Days Min: 1 days Max: [(validity that is, csr.days)-1] days
certPurpose	String	This is an optional parameter. Purpose of certificate creation
issuer	String	This is a mandatory parameter. Name of CA
privateKey	Object	This is a mandatory parameter. Private key details like algorithm, key size and key encoding
privateKey.keyAlgo	Enum	This is a mandatory parameter. Private key algorithm to be used. Supported values: RSA and EC **Possible values:**RSA, EC
privateKey.keySize	Enum	This is an optional parameter. The number of bits in the generated key. Need to select a bit length of at least 2048 when using RSA and 256 when using ECDSA. These are the smallest key sizes allowed for SSL certificates. **Possible values**: KEYSIZE_2048, KEYSIZE_4096 **Default Value for RSA Key**: 2048 bits
privateKey.keyEncoding	Enum	This is a mandatory parameter. The output format of a private key input source. **Default Value:** PEM **Possible values:**PEM, DER

Table 2-10 (Cont.) OCCM Certificates Request Parameters

Field Name	DataType	Description
privateKey.ecCurve	Enum	This is an optional parameter. The EC curve to use if the key algorithm selected is EC. **Default Value:** SECP384r1 **Possible values:** SECP256r1 SECP384r1
privateKey.keyFormat	String	This is a mandatory parameter. The output format of a private key input source. Default Value: PEM
privateKey.privateKeyK8sSecretOut	Object	This is a mandatory parameter except in case of CMP Identity (OCCM) certificates, this field is optional since it is auto-populated from issuer Private key output location
privateKey.privateKeyK8sSecretOut.namespace	String	This is a mandatory parameter except in case of CMP Identity (OCCM) certificates, this field is optional since it is auto-populated from issuer. Kubernetes namespace
privateKey.privateKeyK8sSecretOut.name	String	This is a mandatory parameter except in case of CMP Identity (OCCM) certificates, this field is optional since it is auto-populated from issuer. Kubernetes secret name
privateKey.privateKeyK8sSecretOut.key	String	This is a mandatory parameter except in case of CMP Identity (OCCM) certificates, this field is optional since it is auto-populated from issuer. Kubernetes secret key against which the key-pair will be stored.
csr	Object	This is a mandatory parameter. Certificate Signing Request data
csr.extendedKeyUsage	Object	This is a mandatory parameter. A multi-valued certificate extension containing a list of values indicating purposes for which the certificate public key can be used

Table 2-10 (Cont.) OCCM Certificates Request Parameters

Field Name	DataType	Description
csr.extendedKeyUsage.critical	Boolean	This is an optional parameter. When set to true, extended key usage extension will be marked as critical. **Default Value:** false **Possible values:** true false
csr.extendedKeyUsage.extendedKeyUsageValues	List<Enum>	This is an optional parameter. List of extendedKeyUsage values Default Value(s):CLIENT_AUTH, SERVER_AUTH **Possible values:** CLIENT_AUTH, SERVER_AUTH
csr.keyUsage	Object	This is a mandatory parameter. A multi-valued certificate extension containing a list of names of the permitted key usages.
csr.keyUsage.critical	Boolean	This is an optional parameter. When set to true, key usage extension will be marked as critical. **Possible values:** true false
csr.keyUsage.keyUsageValues	List<Enum>	This is an optional parameter. List of keyUsage values **Default Value**: DIGITAL_SIGNATURE **Possible values:**DIGITAL_SIGNATURE, KEY_ENCIPHERMENT, NON_REPUDIATION
csr.basicConstraints	Object	This is an optional parameter. This is a multi-valued extension which indicates whether a certificate is a CA certificate. The first value is CA followed by TRUE or FALSE
csr.basicConstraints.critical	Boolean	This is an optional parameter. When set to true, basicConstraints extension will be marked as critical. **Possible values:** true false

Table 2-10 (Cont.) OCCM Certificates Request Parameters

Field Name	DataType	Description
csr.basicConstraints.basicConstraintsValue	Enum	This is an optional parameter. BasicConstraints value **Default Value:** END_ENTITY **Possible values:** END_ENTITY
csr.subject	Object	This is a mandatory parameter. Information about company
csr.subject.country	String	This is an optional parameter. Country code where company is legally located.
csr.subject.state	String	This is an optional parameter. State where company is legally located.
csr.subject.location	String	This is an optional parameter. The city or town where company is legally located.
csr.subject.organization	String	This is an optional parameter. Your company's legally registered name.
csr.subject.organizationUnit	String	This is an optional parameter. Name of your department within the organization.
csr.subject.commonName	String	This is an optional parameter. The **Common Name** (AKA **CN**) represents the server name to be protected by the SSL certificate. The certificate is valid only if the request hostname matches the certificate common name.
csr.days.	String	This is an optional parameter. Requested validity for the certificate i.e. Number of days requested for which the certificate will be valid. Default Value :365 Days Max Value: 1096 Min Value: 2 Days
csr.subjectAltName	Object	This is an optional parameter. A multi-valued extension indicating all of the domain names, IP addresses, URIs etc that are secured by the certificate.

Table 2-10 (Cont.) OCCM Certificates Request Parameters

Field Name	DataType	Description
csr.subjectAltName.critical	Boolean	This is an optional parameter. When set to true, subjectAltName extension will be marked as critical. **Default Value:** true **Possible values:** true false
csr.subjectAltName.ipAddress	List<String>	This is an optional parameter. List of IP addresses.
csr.subjectAltName.dns	List<String>	This is an optional parameter. List of domain names
csr.subjectAltName.uriIdUrn	List<String>	This is an optional parameter. List of URI ID (URN of the NFInstanceId)
csr.subjectAltName.uriIdApiRoot	List<String>	This is an optional parameter. List of uniform resource locater IDs
csr.certK8sSecretOut	Object	This is a mandatory parameter except in case of CMP Identity (OCCM) certificate, this field is optional since it is auto-populated from issuer. Certificate output location.
csr.certK8sSecretOut.namespace	String	This is a mandatory parameter except in case of CMP Identity (OCCM) certificate, this field is optional since it is auto-populated from issuer. Kubernetes secret namespace
csr.certK8sSecretOut.name	String	This is a mandatory parameter except in case of CMP Identity (OCCM) certificate, this field is optional since it is auto-populated from issuer.. Kubernetes secret name
csr.certK8sSecretOut.key	String	This is a mandatory parameter except in case of CMP Identity (OCCM) certificate, this field is optional since it is auto-populated from issuer.. Kubernetes secret key against which the certificate will be stored.
csr.certChainK8sSecretOut	Object	This is a mandatory parameter. Certificate Chain output location.
csr.certChainK8sSecretOut.namespace	String	This is an optional parameter. Kubernetes secret namespace

Table 2-10 (Cont.) OCCM Certificates Request Parameters

Field Name	DataType	Description
csr.certChainK8sSecretOut.name	String	This is a mandatory parameter. Kubernetes secret name
csr.certChainK8sSecretOut.key	String	This is an optional parameter. Kubernetes secret key against which the certificate chain will be stored.
csr.mergeCertAndChain	boolean	This is an optional parameter. When set to true, the complete chain containing the leaf certificate and the intermediate CA certificates obtained from CA will be written in the Kubernetes secret against the configured key. **Default Value:** false
nf	String	This is a mandatory parameter. NF name
uuid	String	Unique id for logging and tracking purpose
overrideSecret	boolean	This is an optional parameter. This flag is used to override the Kubernetes secret with new certificate. **Default Value:** false **Possible values:** true false
caBundleK8sSecretIn	Object	This is an optional parameter. CA bundle secret input details. Used to trust peer entities.
caBundleK8sSecretIn.namespace	String	This is an optional parameter. Kubernetes secret namespace
caBundleK8sSecretIn.name	String	This is an optional parameter. Kubernetes secret name
caBundleK8sSecretIn.key	String	This is an optional parameter. Kubernetes secret key against which CA bundle certificate(s) will be stored.
namespace	String	Indicates the namespace where the certificate is stored.By default, this field is left blank during creation.

Table 2-11 OCCM Certificate Response Codes

Response codes	Data type	Cardinality	Description
200 OK	Object (Certs) Or List(Certs)	1	This is a mandatory parameter. Object Certs Or List (CertConfig) matching criteria
202 Accepted	String	1	This is a mandatory parameter. Return uuid
400 Bad request	Problem Details	1	This is a mandatory parameter. Input does not match to process request
404 Not Found	Problem Details	1	This is a mandatory parameter. Certificate not found
500 Internal Server Error	Problem Details	1	This is a mandatory parameter. Something went wrong
409 Conflict	Problem Details	1	This is a mandatory parameter. Record already exists

> **Note**
>
> OCCM Certificates response body data model varies based on REST operation status.

OCCM Certificate JSON payload

```
{
     "name": "",
     "lcmType": "",
     "certType": "",
     "renewBefore": "",
     "certPurpose": "",
     "issuer": "",
     "privateKey": {
         "keyAlgo": "",
         "keySize": "",
         "keyEncoding": "",
         "ecCurve":"",
         "privateKeyK8sSecretOut": {
             "namespace": "",
             "name": "",
             "key": ""
     }
     },
```

```
    "csr": {
         "extendedKeyUsage": {
             "critical" : "",
             "extendedKeyUsageValues" : []
         },
        "keyUsage": {
             "critical" : "",
             "keyUsageValues" : []

    },
         "basicConstraints": {
             "critical" : true,
             "basicConstraintsValue" : ""
         },
         "subject": {
             "country": "",
             "state": "",
             "location": "",
             "organization": "",
             "organizationUnit": "",
             "commonName": ""
         },
         "days": "",
         "subjectAltName": {
             "critical" : "",
             "ipAddress": [],
             "dns": [],
             "uriIdUrn": [],
             "uriIdApiRoot": []
         },
      "certK8sSecretOut": {
         "namespace": "",
         "name": "",
         "key": ""
     },
     "certChainK8sSecretOut": {
         "namespace": "",
         "name": "",
         "key": ""
     },
         "mergeCertAndChain" : false
     },

     "caBundleK8sSecretIn": {
         "namespace": "",
         "name": "",
         "key": ""
     },

     "nf": "",
     "overrideSecret": false
}
```

2.2.1 Fetch all Certificate Configurations

OCCM Uses the GET operation to fetch all the certificate configurations.

Resource URI: /occm-config/v1/certs

Table 2-12 Data structures supported by the GET Response Body on this resource

Response codes	Data type	Cardinality	Description
200 OK	Object (Certs) Or List(Certs)	1	This is a mandatory parameter. Indicates all the certificate configurations are successfully fetched.
500 Internal Server Error	Problem Details	1	This is a mandatory parameter. Indicates that something went wrong while fetching all the certificate configurations.

Sample GET Request:

```
\$ curl --location --request GET 'http://{host}:{port}/occm-config/v1/certs' \
--header 'Oc-Cncc-Id: Cluster1' \
--header 'Oc-Cncc-Instance-Id: Cluster1-OCCM-instance1' \
--header 'Authorization: Bearer eyJhbGciOiJSU.…...'
```

Sample GET Response:

```
200 OK Response Body:  '[
    {
    "uuid": "b4d896ac-689d-4e12-a76c-54c8de4ffe52",
    "name": "NRFTLS12",
     "lcmType": "AUTOMATIC",
     "certType": "OTHER",
     "renewBefore": "7",
     "certPurpose": "NRF SBI",
     "issuer": "CA21",
     "namespace": "occm",
     "privateKey": {
         "keyAlgo": "RSA",
         "keySize": "KEYSIZE_2048",
         "keyEncoding": "PEM",
         "ecCurve":"",
         "privateKeyK8sSecretOut": {
             "namespace": "occm",
             "name": "nrf-tls-secret-54",
             "key": "nrf.pem"
     }
     },
    "csr": {
```

```
        "extendedKeyUsage": {
            "critical" :  false,
            "extendedKeyUsageValues" : [
                "CLIENT_AUTH",
                "SERVER_AUTH"
        ]
        },
       "keyUsage": {
            "critical" : true,
            "keyUsageValues" : [
               "DIGITAL_SIGNATURE"

            ]

    },
        "basicConstraints": {
            "critical" : true,
            "basicConstraintsValue" : "END_ENTITY"
        },
        "subject": {
            "country": "IN",
            "state": "KA",
            "location": "BLR",
            "organization": "Oracle",
            "organizationUnit": "OracleBU",
            "commonName": "some.example.com"
        },
        "days": "365",
        "subjectAltName": {
            "critical" : null,
            "ipAddress": [
                "10.10.10.13",
                "10.10.10.14"
            ],
            "dns": [
                "a.example.com",
                "b.example.com"
            ],
            "uriIdUrn": [
                "urn:uuid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6"
            ],
            "uriIdApiRoot": [
            ]
        },
      "certK8sSecretOut": {
         "namespace": "occm",
         "name": "nrf-tls-secret-54",
         "key": "nrf.cer"
     },
     "certChainK8sSecretOut": {
         "namespace": "occm",
         "name": "nrf-tls-secret-54",
         "key": "nrfcertchain.cer"
     },
         "mergeCertAndChain" : false
     },
```

```
    "caBundleK8sSecretIn": {
        "namespace": "occm",
        "name": "ca-bundle-secret",
        "key": "cabundle.cer"
    },

    "nf": "NRF",
    "overrideSecret": false
},

 {
   "uuid": "a5d906ad-789p-5f13-b86n-54p9fg5iif82",
   "name": "NRFTLS13",
    "lcmType": "AUTOMATIC",
    "certType": "OTHER",
    "renewBefore": "7",
    "certPurpose": "NRF SBI",
    "issuer": "CA21",
    "namespace": "occm",
    "privateKey": {
        "keyAlgo": "RSA",
        "keySize": "KEYSIZE_2048",
        "keyEncoding": "PEM",
        "ecCurve":"",
        "privateKeyK8sSecretOut": {
            "namespace": "occm",
            "name": "nrf-tls-secret-55",
            "key": "nrf.pem"
    }
    },
   "csr": {
        "extendedKeyUsage": {
            "critical" :  false,
            "extendedKeyUsageValues" : [
                "CLIENT_AUTH",
                "SERVER_AUTH"
        ]
        },
       "keyUsage": {
            "critical" : true,
            "keyUsageValues" : [
               "DIGITAL_SIGNATURE"

            ]

   },
        "basicConstraints": {
            "critical" : true,
            "basicConstraintsValue" : "END_ENTITY"
        },
        "subject": {
            "country": "IN",
            "state": "KA",
            "location": "BLR",
            "organization": "Oracle",
```

```
                "organizationUnit": "OracleBU",
                "commonName": "some.example.com"
            },
            "days": "365",
            "subjectAltName": {
                "critical" : null,
                "ipAddress": [
                    "10.10.10.13",
                    "10.10.10.14"
                ],
                "dns": [
                    "a.example.com",
                    "b.example.com"
                ],
                "uriIdUrn": [
                    "urn:uuid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6"
                ],
                "uriIdApiRoot": [
                ]
            },
         "certK8sSecretOut": {
            "namespace": "occm",
            "name": "nrf-tls-secret-55",
            "key": "nrf.cer"
        },
        "certChainK8sSecretOut": {
            "namespace": "occm",
            "name": "nrf-tls-secret-55",
            "key": "nrfcertchain.cer"
        },
            "mergeCertAndChain" : false
        },

        "caBundleK8sSecretIn": {
            "namespace": "occm",
            "name": "ca-bundle-secret",
            "key": "cabundle.cer"
        },

        "nf": "NRF",
        "overrideSecret": false
     }

    ]'
```

2.2.2 Fetch Certificate Configurations by UUID

OCCM Uses the GET operation to fetch the certificate configuration based on the UUID.

Resource URI: /occm-config/v1/certs/{uuid}

Table 2-13 Path Parameter

Name	Data Type	Mandatory (M) or Optional (O)	Cardinality	Description
uuid	String	M	1	Fetches Certificate configuration by UUID

Table 2-14 Data structures supported by the GET Response Body on this resource

Response codes	Data type	Cardinality	Description
200 OK	Object (Certs)	1	This is a mandatory parameter. Indicates that the certificate configuration identified by the given UUID is successfully fetched.
400 Bad Request	Problem Details	1	This is a mandatory parameter. wrong Input
404 Not Found	Problem Details	1	This is a mandatory parameter. Indicates that there is no certificate configuration with the given UUID.
500 Internal Server Error	Problem Details	1	This is a mandatory parameter. Indicates that something went wrong while fetching the certificate configuration.

Sample Get Request:

```
$ curl --location --request GET 'http://{host}:{port}/occm-config/v1/certs/
b4d896ac-689d-4e12-a76c-54c8de4ffe52' \
--header 'Oc-Cncc-Id: Cluster1' \
--header 'Oc-Cncc-Instance-Id: Cluster1-OCCM-instance1' \
--header 'Authorization: Bearer eyJhbGciOiJSU.…'
```

Sample Get Response:

```
200 OK Response Body: '{
    "uuid": "b4d896ac-689d-4e12-a76c-54c8de4ffe52",
    "name": "NRFTLS12",
     "lcmType": "AUTOMATIC",
     "certType": "OTHER",
     "renewBefore": "7",
     "certPurpose": "NRF SBI",
     "issuer": "CA21",
     "namespace": "occm",
```

```
 "privateKey": {
     "keyAlgo": "RSA",
     "keySize": "KEYSIZE_2048",
     "keyEncoding": "PEM",
     "ecCurve":"",
     "privateKeyK8sSecretOut": {
         "namespace": "occm",
         "name": "nrf-tls-secret-54",
         "key": "nrf.pem"
 }
 },
"csr": {
     "extendedKeyUsage": {
         "critical" :  false,
         "extendedKeyUsageValues" : [
             "CLIENT_AUTH",
             "SERVER_AUTH"
     ]
     },
    "keyUsage": {
         "critical" : true,
         "keyUsageValues" : [
        "DIGITAL_SIGNATURE"

         ]

},
     "basicConstraints": {
         "critical" : true,
         "basicConstraintsValue" : "END_ENTITY"
     },
     "subject": {
         "country": "IN",
         "state": "KA",
         "location": "BLR",
         "organization": "Oracle",
         "organizationUnit": "OracleBU",
         "commonName": "some.example.com"
     },
     "days": "365",
     "subjectAltName": {
         "critical" : null,
         "ipAddress": [
             "10.10.10.13",
             "10.10.10.14"
         ],
         "dns": [
             "a.example.com",
             "b.example.com"
         ],
         "uriIdUrn": [
             "urn:uuid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6"
         ],
         "uriIdApiRoot": [
         ]
     },
```

```
 "certK8sSecretOut": {
    "namespace": "occm",
    "name": "nrf-tls-secret-54",
    "key": "nrf.cer"
},
"certChainK8sSecretOut": {
    "namespace": "occm",
    "name": "nrf-tls-secret-54",
    "key": "nrfcertchain.cer"
} ,
    "mergeCertAndChain" : false
},

"caBundleK8sSecretIn": {
    "namespace": "occm",
    "name": "nrf-cabu",
    "key": "cabundle.cer"
},

"nf": "NRF",
"overrideSecret": false
}'
```

2.2.3 Add Certificate Configurations

OCCM uses the POST operation to create a new certificate using the configuration in the request.

Resource URI: /occm-config/v1/certs

Table 2-15 Data structures supported by the POST Response Body on this resource

Response codes	Data type	Cardinality	Description
202 Accepted	Object (Certs)	1	This is a mandatory parameter. Indicates that the request to create a new certificate is accepted successfully and will be processed asynchronously.
400 BAD REQUEST	Problem Details	1	This is a mandatory parameter. Indicates invalid request data.
409 CONFLICT	Problem Details	1	This is a mandatory parameter. Indicates invalid request data.

Table 2-15 (Cont.) Data structures supported by the POST Response Body on this resource

Response codes	Data type	Cardinality	Description
500 Internal Server Error	Problem Details	1	This is a mandatory parameter. Indicates that something went wrong while fetching the certificate configuration.

Sample POST request:

```
$ curl --location --request POST 'http://{host}:{port}/occm-config/v1/certs' \
--header 'Oc-Cncc-Id: Cluster1' \
--header 'Oc-Cncc-Instance-Id: Cluster1-OCCM-instance1' \
--header 'Authorization: Bearer eyJhbGciOiJS.....9IIGc4g' \
--header 'Content-Type: application/json' \
--data-raw '{
     "name": "NRF-TLS",
     "lcmType": "AUTOMATIC",
     "certType": "OTHER",
     "renewBefore": "14",
     "certPurpose": "NRF SBI",
     "issuer": "CA1",
     "privateKey": {
         "keyAlgo": "RSA",
         "keySize": "KEYSIZE_2048",
         "keyEncoding": "PEM",
         "ecCurve":"",
         "privateKeyK8sSecretOut": {
             "namespace": "occm",
             "name": "nrf-tls-secret",
             "key": "nrfkey.pem"
     }
     },
    "csr": {
         "extendedKeyUsage": {
             "critical" :  false,
             "extendedKeyUsageValues" : [
                 "CLIENT_AUTH",
                 "SERVER_AUTH"
         ]
         },
        "keyUsage": {
             "critical" : true,
             "keyUsageValues" : [
           "DIGITAL_SIGNATURE"
        ]

    },
         "basicConstraints": {
             "critical" : true,
```

```
                "basicConstraintsValue" : "END_ENTITY"
            },
            "subject": {
                "country": "IN",
                "state": "KA",
                "location": "BLR",
                "organization": "Oracle",
                "organizationUnit": "OracleBU",
                "commonName": "xyz.example.com"
            },
            "days": "365",
            "subjectAltName": {
                "critical" : null,
                "ipAddress": [
                    "10.10.10.13"
                ],
                "dns": [
                   x.company.com
                ],
                "uriIdUrn": [
                    "urn:uuid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6"
                ],
                "uriIdApiRoot": [
                ]
            },
         "certK8sSecretOut": {
            "namespace": "occm",
            "name": "nrf-tls-secret",
            "key": "nrfcert.pem"
        },
        "certChainK8sSecretOut": {
            "namespace": "occm",
            "name": "nrf-tls-secret",
            "key": "nrfcertchain.pem"
        } ,
            "mergeCertAndChain" : false
        },

        "caBundleK8sSecretIn": {
            "namespace": "occm",
            "name": "ca-bundle-secret",
            "key": "caroot.pem"
        },

        "nf": "NRF",
        "overrideSecret": false
    }'
```

Sample POST Response

```
$ 202 Accepted  Response Body: '{
    "uuid": "b4d896ac-689d-4e12-a76c-54c8de4ffe52",
      "name": "NRF-TLS",
     "lcmType": "AUTOMATIC",
     "certType": "OTHER",
```

```
 "renewBefore": "14",
 "certPurpose": "NRF SBI",
 "issuer": "CA1",
 "privateKey": {
     "keyAlgo": "RSA",
     "keySize": "KEYSIZE_2048",
     "keyEncoding": "PEM",
     "ecCurve":"",
     "privateKeyK8sSecretOut": {
         "namespace": "occm",
         "name": "nrf-tls-secret",
         "key": "nrfkey.pem"
 }
 },
"csr": {
     "extendedKeyUsage": {
         "critical" :  false,
         "extendedKeyUsageValues" : [
             "CLIENT_AUTH",
             "SERVER_AUTH"
     ]
     },
    "keyUsage": {
         "critical" : true,
         "keyUsageValues" : [
       "DIGITAL_SIGNATURE"
    ]

},
     "basicConstraints": {
         "critical" : true,
         "basicConstraintsValue" : "END_ENTITY"
     },
     "subject": {
         "country": "IN",
         "state": "KA",
         "location": "BLR",
         "organization": "Oracle",
         "organizationUnit": "OracleBU",
         "commonName": "xyz.example.com"
     },
     "days": "365",
     "subjectAltName": {
         "critical" : null,
         "ipAddress": [
             "10.10.10.13"
         ],
         "dns": [
            x.company.com
         ],
         "uriIdUrn": [
             "urn:uuid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6"
         ],
         "uriIdApiRoot": [
         ]
     },
```

```
     "certK8sSecretOut": {
        "namespace": "occm",
        "name": "nrf-tls-secret",
        "key": "nrfcert.pem"
    },
    "certChainK8sSecretOut": {
        "namespace": "occm",
        "name": "nrf-tls-secret",
        "key": "nrfcertchain.pem"
    } ,
        "mergeCertAndChain" : false
    },

    "caBundleK8sSecretIn": {
        "namespace": "occm",
        "name": "ca-bundle-secret",
        "key": "caroot.pem"
    },

    "nf": "NRF",
    "overrideSecret": false
}'
```

2.2.4 Recreate Certificates

OCCM uses the PUT operation to recreate the existing certificate identified by its UUID.. You can recreate any certificate that was created successfully and whose status is READY, EXPIRED, or FAILED. To recreate a certificate, the certificate configuration must exist in OCCM. This enhances OCCM's usablity in managing certificate lifecycle operations. For example, if a certificate has been deleted, revoked or has expired, the operator can recreate it using existing configurations. The certificate configuration must exist in OCCM while triggering recreate request.

Resource URI: /occm-config/v1/certs/{uuid}/recreate

Table 2-16 Path Parameter

Name	Data type	Mandatory (M) or Optional (O)	Cardinality	Description
uuid	String	O	1	Indicates the certificate configuration UUID

Table 2-17 Data structures supported by the PUT Response Body on this resource

Response codes	Data Type	Cardinality	Description
202 Accepted	Object (Certs)	1	This is a mandatory parameter. Indicates that the request to create a new certificate is accepted successfully and will be processed asynchronously.
404 Not Found	Problem Details	1	This is a mandatory parameter. Indicates that no certificate exists with the given UUID.
409 CONFLICT	Problem Details	1	This is a mandatory parameter. Indicates invalid request data.
500 Internal Server Error	Problem Details	1	This is a mandatory parameter. Indicates that something went wrong while fetching the certificate configuration.

Sample PUT Request:

```
curl --location --request PUT 'http://{host}:{port}/occm-config/v1/certs/
9983d728-c618-41c3-b7f9-530d00fb7ab1/recreate' \
--header 'oc-cncc-id: Cluster1' \
--header 'oc-cncc-instance-id: Cluster1-occm-instance1' \
--header 'Authorization: Bearer eyJhbGciO........95adDrLUfcAQ'
```

Sample Response:

```
202 Accepted Response Body: '{
    "uuid": "9983d728-c618-41c3-b7f9-530d00fb7ab1",
    "name": "NRF-TLS-1",
    "lcmType": "AUTOMATIC",
    "certType": "OTHER",
    "renewBefore": "15",
    "certPurpose": "NRF SBI",
    "issuer": "CA1",
    "namespace": "occm",
    "privateKey": {
        "keyAlgo": "EC",
        "keySize": null,
        "keyEncoding": "DER",
        "ecCurve": "SECP256r1",
        "keyFormat": null,
        "privateKeyK8sSecretOut": {
```

```
            "namespace": "ns1",
            "name": "nrf-tls-secret",
            "key": "nrfkey.pem"
        }
    },
    "csr": {
        "extendedKeyUsage": {
            "critical": false,
            "extendedKeyUsageValues": [
                "CLIENT_AUTH",
                "SERVER_AUTH"
            ]
        },
        "keyUsage": {
            "critical": false,
            "keyUsageValues": [
                "DIGITAL_SIGNATURE",
                "KEY_ENCIPHERMENT"
            ]
        },
        "basicConstraints": {
            "critical": false,
            "basicConstraintsValue": "END_ENTITY"
        },
        "subject": {
            "country": "IN",
            "state": "KA",
            "location": "BLR",
            "organization": "Oracle",
            "organizationUnit": "CGBU",
            "commonName": "occm"
        },
        "days": "365",
        "subjectAltName": {
            "critical": false,
            "ipAddress": [
                "10.10.10.13",
                "10.10.10.14"
            ],
            "dns": [
                "x.example.com",
                "y.example.com"
            ],
            "uriIdApiRoot": null,
            "uriIdUrn": [
                "urn:uuid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6"
            ]
        },
        "certK8sSecretOut": {
            "namespace": "ns1",
            "name": "nrf-tls-secret",
            "key": "nrf.cer"
        },
        "certChainK8sSecretOut": {
            "namespace": "ns1",
            "name": "nrf-tls-secret",
```

```
            "key": "nrfcertchain.cer"
        } ,
         "mergeCertAndChain" : false

    },
    "caBundleK8sSecretIn": {
        "namespace": "",
        "name": "",
        "key": ""
    },
    "nf": "NRF",
    "overrideSecret": false
}'
```

2.2.5 Edit Certificates

OCCM uses the PUT operation to update the existing certificates identified by certificate UUID. It is only supported for end entity certificates.

Resource URI: /occm-config/v1/certs/{uuid}

Table 2-18 Path Parameter

Name	Data type	Mandatory (M) or Optional (O)	Cardinality	Description
uuid	String	M	1	Indicates the certificate configuration UUID

Data structures supported by the PUT Response Body in this resource

Response codes	Data type	Cardinality	Description
202 Accepted	Object (Certs)	1	This is a mandatory parameter. Indicates that the request to create a new certificate is accepted successfully and will be processed asynchronously.
400 BAD REQUEST	Problem Details	1	This is a mandatory parameter. Indicates invalid request data.
404 Not Found	Problem Details	1	This is a mandatory parameter. Indicates that no certificate exists with the given UUID.
409 CONFLICT	Problem Details	1	This is a mandatory parameter. Indicates invalid request data.

Response codes	Data type	Cardinality	Description
500 Internal Server Error	Problem Details	1	This is a mandatory parameter. Indicates that something went wrong while fetching the certificate configuration.

Sample PUT Request: When DNS is updated

```
curl --location --request PUT 'http://{host}:{port}/occm-config/v1/certs/
9983d728-c618-41c3-b7f9-530d00fb7ab1' \
--header 'oc-cncc-id: Cluster1' \
--header 'oc-cncc-instance-id: Cluster1-occm-instance1' \
--header 'Authorization: Bearer eyJhbGciO........95adDrLUfcAQ' \
--header 'Content-Type: application/json' \
--data-raw '{
    "uuid": "9983d728-c618-41c3-b7f9-530d00fb7ab1",
    "name": "NRF-TLS-1",
    "lcmType": "AUTOMATIC",
    "certType": "OTHER",
    "renewBefore": "15",
    "certPurpose": "NRF SBI",
    "issuer": "CA1",
    "namespace": "occm",
    "privateKey": {
        "keyAlgo": "EC",
        "keySize": null,
        "keyEncoding": "DER",
        "ecCurve": "SECP256r1",
        "keyFormat": null,
        "privateKeyK8sSecretOut": {
            "namespace": "ns1",
            "name": "nrf-tls-secret",
            "key": "nrfkey.pem"
        }
    },
    "csr": {
        "extendedKeyUsage": {
            "critical": false,
            "extendedKeyUsageValues": [
                "CLIENT_AUTH",
                "SERVER_AUTH"
            ]
        },
        "keyUsage": {
            "critical": false,
            "keyUsageValues": [
                "DIGITAL_SIGNATURE",
                "KEY_ENCIPHERMENT"
            ]
        },
        "basicConstraints": {
```

```
                "critical": false,
                "basicConstraintsValue": "END_ENTITY"
            },
            "subject": {
                "country": "IN",
                "state": "KA",
                "location": "BLR",
                "organization": "Oracle",
                "organizationUnit": "CGBU",
                "commonName": "occm"
            },
            "days": "365",
            "subjectAltName": {
                "critical": false,
                "ipAddress": [
                    "10.10.10.13",
                    "10.10.10.14"
                ],
                "dns": [
                ],
                "uriIdApiRoot": null,
                "uriIdUrn": [
                    "urn:uuid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6"
                ]
            },
            "certK8sSecretOut": {
                "namespace": "ns1",
                "name": "nrf-tls-secret",
                "key": "nrf.cer"
            },
            "certChainK8sSecretOut": {
                "namespace": "ns1",
                "name": "nrf-tls-secret",
                "key": "nrfcertchain.cer"
            },
             "mergeCertAndChain" : false
        },
        "caBundleK8sSecretIn": {
            "namespace": "",
            "name": "",
            "key": ""
        },
        "nf": "NRF",
        "overrideSecret": false
    }'
```

Sample Response:

```
202 Accepted Response Body: '{
    "uuid": "9983d728-c618-41c3-b7f9-530d00fb7ab1",
    "name": "NRF-TLS-1",
    "lcmType": "AUTOMATIC",
    "certType": "OTHER",
    "renewBefore": "15",
    "certPurpose": "NRF SBI",
```

```
"issuer": "CA1",
"namespace": "occm",
"privateKey": {
    "keyAlgo": "EC",
    "keySize": null,
    "keyEncoding": "DER",
    "ecCurve": "SECP256r1",
    "keyFormat": null,
    "privateKeyK8sSecretOut": {
        "namespace": "ns1",
        "name": "nrf-tls-secret",
        "key": "nrfkey.pem"
    }
},
"csr": {
    "extendedKeyUsage": {
        "critical": false,
        "extendedKeyUsageValues": [
            "CLIENT_AUTH",
            "SERVER_AUTH"
        ]
    },
    "keyUsage": {
        "critical": false,
        "keyUsageValues": [
            "DIGITAL_SIGNATURE",
            "KEY_ENCIPHERMENT"
        ]
    },
    "basicConstraints": {
        "critical": false,
        "basicConstraintsValue": "END_ENTITY"
    },
    "subject": {
        "country": "IN",
        "state": "KA",
        "location": "BLR",
        "organization": "Oracle",
        "organizationUnit": "CGBU",
        "commonName": "occm"
    },
    "days": "365",
    "subjectAltName": {
        "critical": false,
        "ipAddress": [
            "10.10.10.13",
            "10.10.10.14"
        ],
        "dns": [
            "x.example.com",
            "y.example.com"
        ],
        "uriIdApiRoot": null,
        "uriIdUrn": [
            "urn:uuid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6"
        ]
```

```
        },
        "certK8sSecretOut": {
            "namespace": "ns1",
            "name": "nrf-tls-secret",
            "key": "nrf.cer"
        },
        "certChainK8sSecretOut": {
            "namespace": "ns1",
            "name": "nrf-tls-secret",
            "key": "nrfcertchain.cer"
        } ,
         "mergeCertAndChain" : false

    },
    "caBundleK8sSecretIn": {
        "namespace": "",
        "name": "",
        "key": ""
    },
    "nf": "NRF",
    "overrideSecret": false
}'
```

2.2.6 Delete Certificate Configuration Data

OCCM uses the DELETE operation to delete the certificate configuration identified by its UUID.

Resource URI: /occm-config/v1/certs/{uuid}

Table 2-19 Path Parameter

Name	Data type	Mandatory (M) or Optional (O)	Cardinality	Description
uuid	String	O	1	Indicates the certificate configuration UUID

Table 2-20 Data structures supported by the DELETE Response Body on this resource

Response codes	Data type	Cardinality	Description
200 OK	Certs	1	This is a mandatory parameter. Indicates that the certificate configuration identified by the given UUID is deleted successfully.
400 BAD REQUEST	Problem Details	1	This is a mandatory parameter. Indicates invalid request data.

Table 2-20 (Cont.) Data structures supported by the DELETE Response Body on this resource

Response codes	Data type	Cardinality	Description
404 Not Found	Problem Details	1	This is a mandatory parameter. Indicates that no certificate exists with the given UUID.
500 Internal Server Error	Problem Details	1	This is a mandatory parameter. Indicates that something went wrong while deleting the certificate configuration.

Sample DELETE Request:

```
$ curl --location --request DELETE 'http://{host}:{port}/occm-config/v1/certs/
b4d896ac-689d-4e12-a76c-54c8de4ffe52' \
--header 'Oc-Cncc-Id: Cluster1' \
--header 'Oc-Cncc-Instance-Id: Cluster1-OCCM-instance1' \
--header 'Authorization: Bearer eyJhbGciOiJSUzI1Ni.…'
```

Sample Response:

```
200 OK Response Body: '{
    "uuid": "b4d896ac-689d-4e12-a76c-54c8de4ffe52",
    "name": "NRFTLS12",
     "lcmType": "AUTOMATIC",
     "certType": "OTHER",
     "renewBefore": "7",
     "certPurpose": "NRF SBI",
     "issuer": "CA21",
     "namespace": "occm",
     "privateKey": {
         "keyAlgo": "RSA",
         "keySize": "KEYSIZE_2048",
         "keyEncoding": "PEM",
         "ecCurve":"",
         "privateKeyK8sSecretOut": {
             "namespace": "occm",
             "name": "nrf-tls-secret-54",
             "key": "nrf.pem"
     }
     },
    "csr": {
         "extendedKeyUsage": {
             "critical" :  false,
             "extendedKeyUsageValues" : [
                 "CLIENT_AUTH",
                 "SERVER_AUTH"
         ]
```

```
        },
       "keyUsage": {
            "critical" : true,
            "keyUsageValues" : [
             "DIGITAL_SIGNATURE"
           ]

    },
        "basicConstraints": {
            "critical" : true,
            "basicConstraintsValue" : "END_ENTITY"
        },
        "subject": {
            "country": "IN",
            "state": "KA",
            "location": "BLR",
            "organization": "Oracle",
            "organizationUnit": "OracleBU",
            "commonName": "some.example.com"
        },
        "days": "365",
        "subjectAltName": {
            "critical" : null,
            "ipAddress": [
                "10.10.10.13",
                "10.10.10.14"
            ],
            "dns": [
                "centos8-2.example.com",
                "centos8-3.example.com"
            ],
            "uriIdUrn": [
                "urn:uuid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6"
            ],
            "uriIdApiRoot": [
            ]
        },
     "certK8sSecretOut": {
        "namespace": "occm",
        "name": "nrf-tls-secret-54",
        "key": "nrf.cer"
    },
    "certChainK8sSecretOut": {
        "namespace": "occm",
        "name": "nrf-tls-secret-54",
        "key": "nrfcertchain.cer"
    } ,
        "mergeCertAndChain" : false
    },

    "caBundleK8sSecretIn": {
        "namespace": "occm",
        "name": "ca-bundle-secret",
        "key": "cabundle.cer"
    },
```

```
    "nf": "NRF",
    "overrideSecret": false
}'
```

2.2.7 Delete Certificate Configuration with Secret

OCCM uses the DELETE operation to delete the certificate configuration along with its secret identified by its UUID.

Resource URI: /occm-config/v1/certs/{uuid}/secret

Table 2-21 Path Parameter

Name	Data type	Mandatory (M) or Optional (O)	Cardinality	Description
uuid	String	O	1	Indicates the certificate configuration UUID

Table 2-22 Data structures supported by the DELETE Response Body on this resource

Response codes	Data type	Cardinality	Description
200 OK	Certs	1	This is a mandatory parameter. Indicates that the certificate configuration identified by the given UUID is deleted successfully.
400 BAD REQUEST	Problem Details	1	This is a mandatory parameter. Indicates invalid request data.
404 Not Found	Problem Details	1	This is a mandatory parameter. Indicates that no certificate exists with the given UUID.
500 Internal Server Error	Problem Details	1	This is a mandatory parameter. Indicates that something went wrong while deleting the certificate configuration.

Sample DELETE Request:

```
$ curl --location --request DELETE 'http://{host}:{port}/occm-config/v1/certs/
b4d896ac-689d-4e12-a76c-54c8de4ffe52/secret' \
--header 'oc-cncc-Id: Cluster1' \
--header 'oc-cncc-Instance-Id: Cluster1-OCCM-instance1' \
--header 'Authorization: Bearer eyJhbGciOiJSUz......oJHvivVcGRoA'
```

Sample Response:

```
200 OK Response Body: '{
    "uuid": "b4d896ac-689d-4e12-a76c-54c8de4ffe52",
    "name": "NRFTLS12",
     "lcmType": "AUTOMATIC",
     "certType": "OTHER",
     "renewBefore": "7",
     "certPurpose": "NRF SBI",
     "issuer": "CA21",
     "namespace": "occm",
     "privateKey": {
         "keyAlgo": "RSA",
         "keySize": "KEYSIZE_2048",
         "keyEncoding": "PEM",
         "ecCurve":"",
         "privateKeyK8sSecretOut": {
             "namespace": "occm",
             "name": "nrf-tls-secret-54",
             "key": "nrf.pem"
     }
     },
    "csr": {
         "extendedKeyUsage": {
             "critical" :  false,
             "extendedKeyUsageValues" : [
                 "CLIENT_AUTH",
                 "SERVER_AUTH"
         ]
         },
        "keyUsage": {
             "critical" : true,
             "keyUsageValues" : [
              "DIGITAL_SIGNATURE"
            ]

    },
         "basicConstraints": {
             "critical" : true,
             "basicConstraintsValue" : "END_ENTITY"
         },
         "subject": {
             "country": "IN",
             "state": "KA",
             "location": "BLR",
             "organization": "Oracle",
             "organizationUnit": "OracleBU",
             "commonName": "some.example.com"
         },
         "days": "365",
         "subjectAltName": {
             "critical" : null,
             "ipAddress": [
                 "10.10.10.13",
                 "10.10.10.14"
             ],
```

```
                "dns": [
                    "centos8-2.example.com",
                    "centos8-3.example.com"
                ],
                "uriIdUrn": [
                    "urn:uuid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6"
                ],
                "uriIdApiRoot": [
                ]
            },
         "certK8sSecretOut": {
            "namespace": "occm",
            "name": "nrf-tls-secret-54",
            "key": "nrf.cer"
        },
        "certChainK8sSecretOut": {
            "namespace": "occm",
            "name": "nrf-tls-secret-54",
            "key": "nrfcertchain.cer"
        } ,
            "mergeCertAndChain" : false
        },

        "caBundleK8sSecretIn": {
            "namespace": "occm",
            "name": "ca-bundle-secret",
            "key": "cabundle.cer"
        },

        "nf": "NRF",
        "overrideSecret": false
    }'
```

2.2.8 OCCM Certificate Bulk Migrate

Certificate Bulk Migrate Data Model

Table 2-23 OCCM Certificates Bulk Migrate Request Parameters

Field Name	Data Type	Description
uuid	String	This is an optional parameter. Unique identifier for each Bulk Certificate Migration. This must be kept empty in the request.
sourceIssuerName	String	This is a mandatory parameter. Name of the issuer whose linked certificates are migrated.
destinationIssuerName	String	This is a mandatory parameter. Name of the issuer to which the certificates are migrated.

Table 2-23 (Cont.) OCCM Certificates Bulk Migrate Request Parameters

Field Name	Data Type	Description
bulkMigrateInitiatedOnCerts	List<Object>	This is an optional parameter. List of certificates which will be migrated from source to destination issuer. This must be kept empty in the request.
bulkMigrateSkippedOnCerts	List<Object>	This is an optional parameter. List of certificates on which the migration is skipped because the certificate status is other than ready, expired, or another recreate is in process. This must be kept empty in the request.

Table 2-24 OCCM Certificates Bulk Migrate Response Codes

Response codes	Data Type	Cardinality	Description
200 OK	Object (Certificates Bulk Migrate) Or List (Certificates Bulk Migrate)	1	This is a mandatory parameter. Object (Certificates Bulk Migrate) Or List(Certificates Bulk Migrate) matching criteria.
202 Accepted	Object (Certificates Bulk Migrate)	1	This is a mandatory parameter. Object (Certificates Bulk Migrate)
400 Bad request	Problem Details	1	This is a mandatory parameter. Input does not match with the processed request
404 Not found	Problem Details	1	This is a mandatory parameter Bulk cert migration configuration not found
409 Conflict	Problem Details	1	This is a mandatory parameter. Bulk certificate migration already in process.
500 Internal Server Error	Problem Details	1	This is a mandatory parameter. This is displayed when something goes wrong

2.2.8.1 Fetch All the Certificate Bulk Migrate Configurations

Fetch All the Certs Bulk Migrate

OCCM uses the GET operation to fetch all the Certs Bulk Migrate.

Resource URI: /occm-config/v1/certs/bulk-migrate

Table 2-25 Path Parameter

Name	Data type	Mandatory (M) or Optional (O)	Cardinality	Description
uuid	String	M	1	Certificate Bulk Migrate uuid

Table 2-26 Data structures supported by the GET Response Body on this resource

Response codes	Data type	Cardinality	Description
200 OK	List(Certs Bulk Migrate)	1	This is a mandatory parameter. Indicates all certs bulk migrate configurations are successfully fetched.
500 Internal Server Error	Problem Details	1	This is a mandatory parameter. Indicates that something went wrong while getting all the Certs Bulk Migrate configurations.

Sample GET Request:

```
$ curl --location --request GET 'http://{host}:{port}/occm-config/v1/certs/
bulk-migrate' \
--header 'Oc-Cncc-Id: Cluster1' \
--header 'Oc-Cncc-Instance-Id: Cluster1-OCCM-instance1' \
--header 'Authorization: Bearer eyJhbGciOiJSUzI1Ni'
```

Sample GET Response:

```
200 OK Response Body: '[{
        "uuid": "51e40d19-c195-4be1-aedd-ced93ded6b62",
        "sourceIssuerName": "CA1",
        "destinationIssuerName": "CA2",
        "bulkMigrateInitiatedOnCerts": [
            {
                "certUUID": "b7390498-dc44-4f2b-9f0c-86b67fb81c70",
                "certName": "NRF1"
            },
            {
                "certUUID": "134801cd-c7ef-4d0c-af52-49419128b981",
                "certName": "NRF2"
            },
            {
                "certUUID": "d57d19be-e2cd-4b65-a4f1-5bf26cd2be7d",
                "certName": "NRF3"
            }
        ],
        "bulkMigrateSkippedOnCerts": []
```

```
 },

{
     "uuid": "89e40d19-c195-4be1-aedd-ced93ded6b62",
    "sourceIssuerName": "CA3",
    "destinationIssuerName": "CA4",
    "bulkMigrateInitiatedOnCerts": [
        {
            "certUUID": "a9390498-dc44-5e2b-9f0c-86b67fb81c70",
            "certName": "NRF4"
        },
        {
            "certUUID": "154801cd-c7ef-5e0c-af52-49419128b981",
            "certName": "NRF5"
        },
        {
            "certUUID": "d97d19be-e2cd-4b65-pqf1-5bf26cd2be7d",
            "certName": "NRF6"
        }
    ],
    "bulkMigrateSkippedOnCerts": []
}
]'
```

2.2.8.2 Fetch the Certificate Bulk Migrate Configurations by uuid

Fetch the Certificates Bulk Migrate by uuid

OCCM uses the GET operation to fetch all Certificates Bulk Migrate by uuid.

Resource URI: /occm-config/v1/certs/bulk-migrate/{uuid}

URI path parameters supported by the GET method on this resource.

Table 2-27 Path Variable

Field Name	Data Type	Description
uuid	String	Indicates matching certs bulk migrate configuration for the given UUID.

Table 2-28 Data structures supported by the GET Response Body on this resource

Response codes	Data Type	Cardinality	Description
200 OK	Certs Bulk Migrate	1	This is a mandatory parameter. Indicates matching certs bulk migrate configuration for the given UUID.

Table 2-28 (Cont.) Data structures supported by the GET Response Body on this resource

Response codes	Data Type	Cardinality	Description
400 BAD REQUEST	Problem Details	1	This is a mandatory parameter. Indicates invalid request data.
404 NOT FOUND	Problem Details	1	This is a mandatory parameter. Indicates there is no matching certs bulk migrate for the given UUID.
500 Internal Server Error	Problem Details	1	This is a mandatory parameter. Indicates that something went wrong while getting the Certs Bulk Migrate configuration.

Sample GET Request:

```
$ curl --location --request GET 'http://{host}:{port}/occm-config/v1/certs/
bulk-migrate/51e40d19-c195-4be1-aedd-ced93ded6b62' \
--header 'Oc-Cncc-Id: Cluster1' \
--header 'Oc-Cncc-Instance-Id: Cluster1-OCCM-instance1' \
--header 'Authorization: Bearer eyJhbGciOiJSUzI1NiIs'
```

Sample GET Response:

```
200 OK Response Body: '{
        "uuid": "51e40d19-c195-4be1-aedd-ced93ded6b62",
        "sourceIssuerName": "CA1",
        "destinationIssuerName": "CA2",
        "bulkMigrateInitiatedOnCerts": [
            {
                "certUUID": "b7390498-dc44-4f2b-9f0c-86b67fb81c70",
                "certName": "NRF1"
            },
            {
                "certUUID": "134801cd-c7ef-4d0c-af52-49419128b981",
                "certName": "NRF2"
            },
            {
                "certUUID": "d57d19be-e2cd-4b65-a4f1-5bf26cd2be7d",
                "certName": "NRF3"
            }
        ],
        "bulkMigrateSkippedOnCerts": []
}'
```

2.2.8.3 Create the Certificate Bulk Migrate Configurations Using Request Body

Create Certs Bulk Migrate Using Request Body

OCCM uses the POST operation to create Certs Bulk Migrate using request body.

Resource URI: /occm-config/v1/certs/bulk-migrate

Table 2-29 Data structures supported by the POST Response Body on this resource

Response codes	Data type	Cardinality	Description
202 ACCEPTED	Object (Certs Bulk Migrate)	1	This is a mandatory parameter. Indicates there is no matching certs bulk migrate for the given UUID.
400 BAD REQUEST	Problem Details	1	This is a mandatory parameter. Indicates invalid request data.
500 INTERNAL SERVER ERROR	Problem Details	1	This is a mandatory parameter. Indicates that something went wrong while initiating the bulk migration.

Sample POST Request:

```
$ curl --location --request POST 'http://{host}:{port}/occm-config/v1/certs/
bulk-migrate' \
--header 'Oc-Cncc-Id: Cluster1' \
--header 'Oc-Cncc-Instance-Id: Cluster1-OCCM-instance1' \
--header 'Authorization: Bearer LxuLeX9di....hXDUcoFwDw' \
--header 'Content-Type: application/json' \
--data-raw '{
    "sourceIssuerName": "CA1",
    "destinationIssuerName": "CA2"
}'
```

Sample POST Response:

```
202 Success Response Body: '{
        "uuid": "51e40d19-c195-4be1-aedd-ced93ded6b62",
        "sourceIssuerName": "CA1",
        "destinationIssuerName": "CA2",
        "bulkMigrateInitiatedOnCerts": [
            {
                "certUUID": "b7390498-dc44-4f2b-9f0c-86b67fb81c70",
                "certName": "NRF1"
            },
            {
```

```
                    "certUUID": "134801cd-c7ef-4d0c-af52-49419128b981",
                    "certName": "NRF2"
                },
                {
                    "certUUID": "d57d19be-e2cd-4b65-a4f1-5bf26cd2be7d",
                    "certName": "NRF3"
                }
            ],
            "bulkMigrateSkippedOnCerts": []
}'
```

2.2.8.4 Delete the Certificate Bulk Migrate Configurations by uuid

Delete the Certificates Bulk Migrate by uuid

OCCM uses the DELETE operation to delete the Certificates Bulk Migrate by uuid.

Resource URI: /occm-config/v1/certs/bulk-migrate/{uuid}

URI query parameters supported by the DELETE method on this resource.

Table 2-30 Query Parameters

Field Name	Mandatory (M) or Optional (O)	Cardinality	Data Type	Description
uuid	O	1	String	uuid of the Certificates Bulk Migrate configuration.

Table 2-31 Data structures supported by the DELETE Response Body on this resource

Response codes	Data type	Cardinality	Description
200 OK	Certificates Bulk Migrate	1	This is a mandatory parameter. Indicates that Certs Bulk Migrate deletion for the given UUID is successful.
400 BAD REQUEST	Problem Details	1	This is a mandatory parameter. Indicates invalid request data.
404 NOT FOUND	Problem Details	1	This is a mandatory parameter. Indicates that there is no matching Certs Bulk Migrate for the given UUID.

Table 2-31 (Cont.) Data structures supported by the DELETE Response Body on this resource

Response codes	Data type	Cardinality	Description
500 Internal Server Error	Problem Details	1	This is a mandatory parameter. Indicates that something went wrong while deleting the Certs Bulk Migrate configuration.

Sample DELETE Request:

```
$ curl --location --request DELETE 'http://{host}:{port}/occm-config/v1/certs/
bulk-migrate/51e40d19-c195-4be1-aedd-ced93ded6b62' \
--header 'Oc-Cncc-Id: Cluster1' \
--header 'Oc-Cncc-Instance-Id: Cluster1-OCCM-instance1' \
--header 'Authorization: Bearer eyJhbGciOi.......g-atjhQ'
```

Sample DELETE Response:

```
200 OK Response Body: '{
        "uuid": "51e40d19-c195-4be1-aedd-ced93ded6b62",
        "sourceIssuerName": "CA1",
        "destinationIssuerName": "CA2",
        "bulkMigrateInitiatedOnCerts": [
            {
                "certUUID": "b7390498-dc44-4f2b-9f0c-86b67fb81c70",
                "certName": "NRF1"
            },
            {
                "certUUID": "134801cd-c7ef-4d0c-af52-49419128b981",
                "certName": "NRF2"
            },
            {
                "certUUID": "d57d19be-e2cd-4b65-a4f1-5bf26cd2be7d",
                "certName": "NRF3"
            }
        ],
        "bulkMigrateSkippedOnCerts": []
}'
```

2.3 OCCM Logging Resource

2.3.1 Fetch Logging Configuration for a Service

OCCM uses the GET operation to fetch the logging configuration for a service

Resource URI:/occm-config/v1/occm/logging

Sample GET request:

```
curl --location --request GET 'http://{host}:{port}/occm-config/v1/occm/
logging' \
--header 'oc-cncc-id: Cluster1' \
--header 'oc-cncc-instance-id: Cluster1-occm-instance1' \
--header 'Authorization: Bearer eyJhbGciOiJSUzI1NiI...' \
--data-raw ''
```

Sample Response:

```
{"appLogLevel":"INFO","packageLogLevel":
[{"packageName":"root","logLevelForPackage":"ERROR"}]}
```

2.3.2 Fetch Logging Configurations for All Services

OCCM uses the GET operation to fetch logging configurations for all services.

Resource URI:/occm-config/v1/all/logging

Sample GET Request:

```
curl --location --request GET 'http://{host}:{port}/occm-config/v1/all/
logging' \
--header 'oc-cncc-id: Cluster1' \
--header 'oc-cncc-instance-id: Cluster1-occm-instance1' \
--header 'Authorization: Bearer eyJhbGciOiJSUzI1NiIsInR5cCIgOiAi...'
```

Sample Response:

```
[{"occm":"{\"appLogLevel\":\"INFO\",\"packageLogLevel\":
[{\"packageName\":\"root\",\"logLevelForPackage\":\"ERROR\"}]}"}]
```

2.3.3 Update Logging Configurations for a Service

OCCM uses the PUT operation to update logging configurations for a service.

Resource URI:/occm-config/v1/occm/logging

Sample PUT Request:

```
curl --location --request PUT 'http://{host}:{port}/occm-config/v1/occm/
logging' \
--header 'oc-cncc-id: Cluster1' \
--header 'oc-cncc-instance-id: Cluster1-occm-instance1' \
--header 'Authorization: Bearer eyJhbGc...h8bJI_Owc_nb_hA' \
--header 'Content-Type: application/json' \
--data-raw '{
    "appLogLevel":"INFO",
    "packageLogLevel":[
    {
        "packageName":"root",
        "logLevelForPackage":"ERROR"
        }
```

```
    ]
    }'
```

Sample Response:

```
201 Created
```