

Oracle® Communications

Cloud Native Core, Certificate Management

User Guide



Release 26.1.200
G54040-01
July 2026

ORACLE®

Copyright © 2023, 2026, Oracle and/or its affiliates.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software, software documentation, data (as defined in the Federal Acquisition Regulation), or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs) and Oracle computer documentation or other Oracle data delivered to or accessed by U.S. Government end users are "commercial computer software," "commercial computer software documentation," or "limited rights data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, reproduction, duplication, release, display, disclosure, modification, preparation of derivative works, and/or adaptation of i) Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs), ii) Oracle computer documentation and/or iii) other Oracle data, is subject to the rights and limitations specified in the license contained in the applicable contract. The terms governing the U.S. Government's use of Oracle cloud services are defined by the applicable contract for such services. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle®, Java, MySQL, and NetSuite are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Inside are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Epyc, and the AMD logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

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>.

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.

Contents

1	Introduction	
1.1	Overview	1
1.2	Reference	2
2	OCCM Architecture	
3	OCCM Supported Features	
3.1	OCCM Feature List with Configuration Details	1
3.2	Integration with Certificate Authority	3
3.2.1	Establishing Initial Trust Between OCCM and CA	4
3.2.1.1	MAC Based Trust Establishment	4
3.2.1.2	Certificate Based Trust Establishment	5
3.3	Support for HTTPs Encryption	6
3.4	Accessing OCCM from CNC Console	6
3.5	Managing Issuers	8
3.5.1	Pre-configuration for OCCM Bootstrapping	8
3.5.2	Creating Issuer	10
3.5.3	Updating Issuer	15
3.5.4	Deleting Issuers	16
3.6	Managing Certificates	17
3.6.1	Issuer (CA) HealthCheck	17
3.6.2	Creating CMP (OCCM) Identity Certificate	21
3.6.2.1	OCCM Certificate Configuration Modes	26
3.6.3	Creating End Entity (NF) Certificates	30
3.6.4	Updating Certificate Configurations	36
3.6.5	Recreating Certificates	40
3.6.6	Cloning Certificates	41
3.6.7	Monitoring Certificate Expiry and Renewing Certificates	44
3.6.7.1	Renewing CMP Identity (OCCM) Certificate	45
3.6.7.2	Renewing End Entity (NF) Certificate	45
3.6.8	Polling for Certificates	46
3.6.9	Deleting the Certificate Configuration	46

3.6.10	Bulk Migration of Certificates	48
3.6.10.1	Initiating Bulk Certificate Migration	49
3.6.10.2	Deleting Bulk Certificate Migration	50
3.7	OCCM Retry on Failure	51
3.8	Network Policies	51
3.9	Traffic Segregation	52

4 Introducing OCCM in an Existing NF Deployment

5 Accessing OCCM Resources Through Curl and Postman

5.1	Generate Access Tokens	1
5.2	Refresh Access Tokens	2
5.3	Issuer Configuration API Access	2
5.4	Certificate Configuration API Access	5
5.5	Bulk Certificate Migration API Access	7
5.6	Logging API Access	7

6 OCCM Metrics

6.1	occm_config_http_requests_total	2
6.2	occm_config_http_response_total	3
6.3	occm_cmp_requests_total	3
6.4	occm_cmp_responses_total	4
6.5	occm_cmp_identity_cert_expiration_seconds	4
6.6	occm_end_entity_cert_expiration_seconds	5
6.7	occm_cmp_identity_cert_status	5
6.8	occm_end_entity_cert_status	6
6.9	occm_cmp_cli_durations_seconds	6
6.10	occm_cert_request_status_total	7
6.11	occm_secret_event_total	7
6.12	occm_cert_start_date_seconds	8
6.13	occm_cert_renew_time_seconds	9
6.14	occm_cert_first_fail_time_seconds	9

7 OCCM Alerts

7.1	OccmCmplIdentityCertExpirationMinor	1
7.2	OccmCmplIdentityCertExpirationMajor	2
7.3	OccmCmplIdentityCertExpirationCritical	3
7.4	OccmCmplIdentityCertExpired	4

7.5	OccmEndEntityCertExpirationMinor	5
7.6	OccmEndEntityCertExpirationMajor	6
7.7	OccmEndEntityCertExpirationCritical	7
7.8	OccmEndEntityCertExpired	8
7.9	OccmServiceDown	9
7.10	OccmMemoryUsageMinorThreshold	10
7.11	OccmMemoryUsageMajorThreshold	11
7.12	OccmMemoryUsageCriticalThreshold	12
7.13	OccmCPUUsageMinorThreshold	13
7.14	OccmCMPFailureMinor	14
7.15	OccmCMPFailureMajor	15
7.16	OccmCMPFailureCritical	16
7.17	OccmFailureMinor	17
7.18	OccmFailureMajor	18
7.19	OccmFailureCritical	19
7.20	OccmInputSecretModifyMajor	20
7.21	OccmOutputSecretModifyMinor	21
7.22	OccmK8sResourceDeleteMajor	22

8 OCCM KPIs

8.1	CMP Identity (OCCM) Certificate Expiry Time	1
8.2	End Entity (NF) Certificate Expiry Time	2
8.3	CMP Identity (OCCM) Certificate Readiness Status	3
8.4	End Entity (NF) Certificate Readiness Status	4
8.5	CMP Request	5
8.6	CMP Responses	5
8.7	Configuration Requests	5
8.8	Configuration Responses	6
8.9	CPU Usage	6
8.10	Memory Usage	7
8.11	OpenSSL CLI Duration (occm_cmp_cli_durations_seconds)	7
8.12	Number of requests sent to the CA	7
8.13	Number of responses received from CA	7
8.14	Number of responses based on response code from CA	7
8.15	Type of request sent to CA	8
8.16	Number of certificates issued by CA	8
8.17	Number of CSRs denied by CA or TLS handshake failures or HTTPs connection failures during CA connection	8
8.18	Error while writing the key, certificate, or chain in the Kubernetes secrets	9
8.19	Unable to access or read from Kubernetes secrets	9
8.20	Check Renewed Certificate	9

8.21	Certificate Error and Warnings	9
8.22	Bulk Certificate Migration Error	10
8.23	Issuer Health Check Certificate Readiness Status	11
8.24	Issuer Health Check Certificate Expiry Time	11

Preface

- [Documentation Accessibility](#)
- [Diversity and Inclusion](#)

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.
<i>italic</i>	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 26.1.2.xx.

Release 26.1.200 - G54040-01, July 2026

- **New Features:**
 - **Support for Creating Health Check Certificates:**
 - * Added the [Issuer \(CA\) HealthCheck](#) section to describe the creation of Health Check Certificates.
 - * Updated the [Creating End Entity \(NF\) Certificate Using OCCM - Sample Configuration](#) section to reflect the procedure to create Health Check certificates.
- **New metrics:**

Added the following metrics in the [OCCM Metrics](#) section:

 - occm_cert_start_date_seconds
 - occm_cert_renew_time_seconds
 - occm_cert_first_fail_time_seconds
- **New KPIs:**

Added the following KPIs in the [OCCM KPIs](#) section:

 - Issuer Health Check Certificate Readiness Status
 - Issuer Health Check Certificate Expiry Time

1

Introduction

Oracle Communications Cloud Native core, Certificate Management (OCCM) is an automated solution for managing the certificates needed for Oracle 5G Network Functions (NFs). OCCM constantly monitors and renews the certificates based on their validity or expiry period.

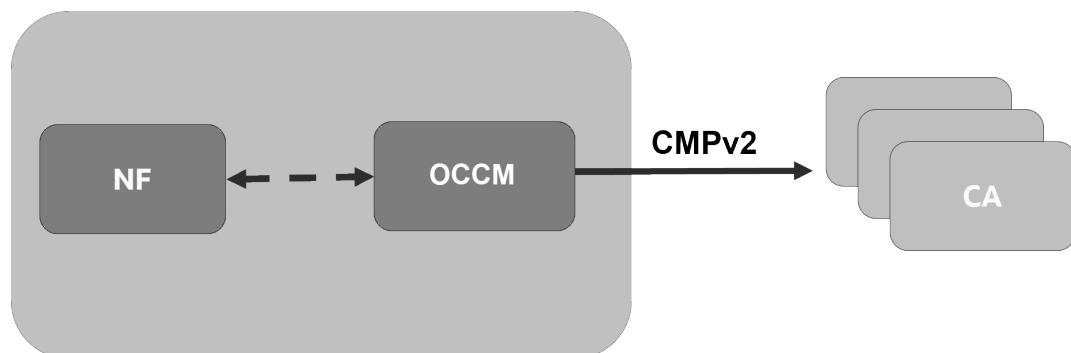
As 3GPP recommends using separate certificates based on the client or server mode and the type of workflow, it leads to many certificates in the network. Automated certificate management eliminates any possibilities of network disruption due to expired certificates. In SBA network deployments, the Network Functions (NFs) are required to support multiple operator certificates for different purposes and interfaces. This amounts to hundreds of certificates in the network with varying validity periods and unwieldy to monitor and renew the certificates manually. Hence, automation of certificate management becomes important to avoid network disruptions due to expired certificates.

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:

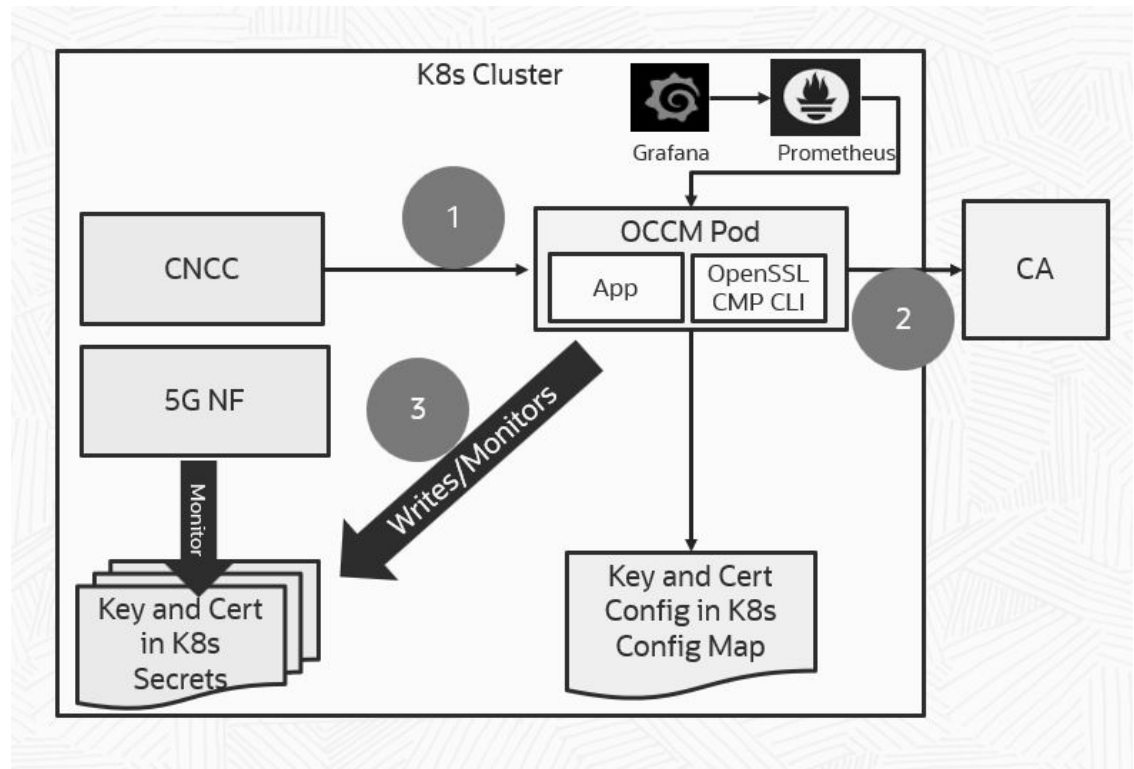
- *Oracle Communications Cloud Native Core, Cloud Native Environment User Guide*
- *Oracle Communications Cloud Native Core, cnDBTier User Guide*
- *Oracle Communications Cloud Native Configuration Console User Guide*
- *Oracle Communications Cloud Native Core Automated Test Suite User Guide*
- *Oracle Communications Cloud Native Core, OCI Deployment Guide*
- *Oracle Communications Cloud Native Core, OCI Adaptor User Guide*
- *Oracle Communications Cloud Native Core Release Notes*
- *Oracle Communications Cloud Native Core Licensing Information User Guide*
- *Oracle Communications Cloud Native Core Solution Upgrade Guide*
- *Oracle Communications Cloud Native Core Security Guide*
- *Oracle Communications Cloud Native Core Licensing Information User Manual*
- *Oracle Communications Cloud Native Core Network Function Data Collector User Guide*
- *Oracle Communications Cloud Native Core, Binding Support Function User Guide*
- *Oracle Communications CD Control Server User Guide*
- *Oracle Communications Cloud Native Core, Operations Services Overlay User Guide*
- *Oracle Communications Cloud Native Core, Converged Policy User Guide*
- *Oracle Communications Cloud Native Core, Service Communication Proxy User Guide*
- *Oracle Communications Cloud Native Core, Security Edge Protection Proxy User Guide*
- *Oracle Communications Cloud Native Core, Network Exposure Function User Guide*
- *Oracle Communications Cloud Native Core, Network Repository Function User Guide*
- *Oracle Communications Cloud Native Core, Network Slice Selection Function User Guide*
- *Oracle Communications Cloud Native Core, Unified Data Repository User Guide*
- *Oracle Communications Cloud Native Core Network Analytics Data Director User Guide*

2

OCCM Architecture

OCCM is a Cloud Native application consisting of a single microservice. The OCCM lifecycle is managed by a Helm chart and delivered as a CSAR package that includes the application container images and the Helm chart.

Figure 2-1 OCCM Architecture



Architecture Description

OCCM is deployed as a single Kubernetes Pod and has a small resource footprint. The OCCM application uses a set of OpenSSL Certificate Management Protocol (CMP) CLI commands based on the provided configuration and the certificate management procedure that needs to be carried out at a point in time. The Output – Key and Certificate – is stored in configuration defined Kubernetes secret.

Operator provides the desired key and certificate configuration through Console. OCCM contacts the CA for certificate signing. After successful Certificate creation, OCCM writes the key and certificate in Kubernetes secrets.

In the diagram above:

1. Operator provides the desired Key and Certificate configuration.
2. OCCM contacts the CA for certificate signing.

3. OCCM writes the key and certificate in Kubernetes Secrets. Starts monitoring of the secret for modification or deletion.

OCCM provides the following deployment models to support certificate management for the integrated NF(s) instantiated within the same cluster:

- Dedicated deployment model - OCCM resides in the same Kubernetes namespace as the NF or Components.
- Shared deployment model - OCCM is deployed in a separate Kubernetes namespace and can manage certificates of multiple NFs or components deployed in other Kubernetes namespaces.

Appropriate permissions must be assigned to OCCM using Kubernetes Service Account, Role and Role Binding, based on the selected deployment model.

OCCM provides secret monitoring capabilities, which help the operator to monitor and manage previously created certificates. OCCM identifies and takes necessary action if certificates are modified or deleted manually, without experiencing loss of service.

Certificate monitoring is useful in the following scenarios:

- The certificate or the Kubernetes secret holding the certificate is deleted.
- The certificate is manually updated.

For more information, see "Monitoring Secrets for Manual Update or Delete" in the *Oracle Communications Cloud Native Configuration Console User Guide*.

3

OCCM Supported Features

This section describes the features supported by Oracle Communications Cloud Native Core, Certificate Management (OCCM).

3.1 OCCM Feature List with Configuration Details

Feature Name	Enabled or Disabled	Configuration Type	Licensing SKU
Integration with Certificate Authority	Enabled	REST API and Console	Oracle Communications Cloud Native Core, Certificate Management - 25K Active Subscribers
Establishing Initial Trust Between OCCM and CA	Enabled	REST API and Console	Oracle Communications Cloud Native Core, Certificate Management - 25K Active Subscribers
MAC Based Trust Establishment	Enabled	REST API and Console	Oracle Communications Cloud Native Core, Certificate Management - 25K Active Subscribers
Certificate Based Trust Establishment	Enabled	REST API and Console	Oracle Communications Cloud Native Core, Certificate Management - 25K Active Subscribers
Support for HTTPS Encryption	Enabled	Helm and Console	Oracle Communications Cloud Native Core, Certificate Management - 25K Active Subscribers
Accessing OCCM from CNC Console	Enabled	Console	Oracle Communications Cloud Native Core, Certificate Management - 25K Active Subscribers
Managing Issuers	Enabled	REST API and Console	Oracle Communications Cloud Native Core, Certificate Management - 25K Active Subscribers
Pre-configuration for OCCM Bootstrapping	Enabled	Helm, REST API, and Console	Oracle Communications Cloud Native Core, Certificate Management - 25K Active Subscribers
Creating Issuer	Enabled	REST API and Console	Oracle Communications Cloud Native Core, Certificate Management - 25K Active Subscribers
Updating Issuer	Enabled	REST API and Console	Oracle Communications Cloud Native Core, Certificate Management - 25K Active Subscribers

Feature Name	Enabled or Disabled	Configuration Type	Licensing SKU
Deleting Issuers	Enabled	REST API and Console	Oracle Communications Cloud Native Core, Certificate Management - 25K Active Subscribers
Managing Certificates	Enabled	REST API and Console	Oracle Communications Cloud Native Core, Certificate Management - 25K Active Subscribers
Issuer (CA) HealthCheck	Enabled	REST API and Console	Oracle Communications Cloud Native Core, Certificate Management - 25K Active Subscribers
Creating CMP (OCCM) Identity Certificate	Enabled	REST API and Console	Oracle Communications Cloud Native Core, Certificate Management - 25K Active Subscribers
OCCM Certificate Configuration Modes	Enabled	REST API and Console	Oracle Communications Cloud Native Core, Certificate Management - 25K Active Subscribers
Creating End Entity (NF) Certificates	Enabled	REST API and Console	Oracle Communications Cloud Native Core, Certificate Management - 25K Active Subscribers
Updating Certificate Configurations	Enabled	REST API and Console	Oracle Communications Cloud Native Core, Certificate Management - 25K Active Subscribers
Recreating Certificates	Enabled	REST API and Console	Oracle Communications Cloud Native Core, Certificate Management - 25K Active Subscribers
Cloning Certificates	Enabled	Console	Oracle Communications Cloud Native Core, Certificate Management - 25K Active Subscribers
Monitoring Certificate Expiry and Renewing Certificates	Enabled	Helm, REST API, and Console	Oracle Communications Cloud Native Core, Certificate Management - 25K Active Subscribers
Renewing CMP Identity (OCCM) Certificate	Enabled	Helm, REST API, and Console	Oracle Communications Cloud Native Core, Certificate Management - 25K Active Subscribers
Renewing End Entity (NF) Certificate	Enabled	Helm, REST API, and Console	Oracle Communications Cloud Native Core, Certificate Management - 25K Active Subscribers
Polling for Certificates	Enabled	REST API and Console	Oracle Communications Cloud Native Core, Certificate Management - 25K Active Subscribers

Feature Name	Enabled or Disabled	Configuration Type	Licensing SKU
Deleting the Certificate Configuration	Enabled	REST API and Console	Oracle Communications Cloud Native Core, Certificate Management - 25K Active Subscribers
Bulk Migration of Certificates	Enabled	REST API and Console	Oracle Communications Cloud Native Core, Certificate Management - 25K Active Subscribers
OCCM Retry on Failure	Enabled	None	Oracle Communications Cloud Native Core, Certificate Management - 25K Active Subscribers
Network Policies	Enabled	Helm	Oracle Communications Cloud Native Core, Certificate Management - 25K Active Subscribers
Traffic Segregation	Disabled	Helm	Oracle Communications Cloud Native Core, Certificate Management - 25K Active Subscribers

3.2 Integration with Certificate Authority

OCCM integrates with one or more Certificate Authorities (CAs) using the Certificate Management Protocol version 2 (CMPv2), as proposed by the 3GPP TS33.310. Operators have the flexibility to configure OCCM to integrate with a single CA or multiple CAs, depending on the layout of CA hierarchy deployed in the network. However, it is recommended that each intermediate CA manage multiple certificates of the same type.

The two CMPv2 procedures for different types of requests towards the CA used by OCCM are:

- Initialization procedure: CMPv2 Initialization Request (IR) is used for certificate create request.
- Key update procedure: CMPv2 Key Update Request (KUR) is used for certificate renewal request.

OCCM employs two modes to establish initial trust between the certificate management and CA:

- Using a pre-shared key
- Using a key and certificate

These options are available when the first request is made towards the CA. For all subsequent requests, OCCM uses the certificate based mechanism to sign the CMPv2 requests in compliance with 3GPP standards.

Note

OCCM supports HTTP 1.0 and HTTP 1.1 versions. OCCM initiates the request using HTTP 1.0. If the CA supports HTTP 1.1 only, then OCCM shifts to using HTTP 1.1 version.

3.2.1 Establishing Initial Trust Between OCCM and CA

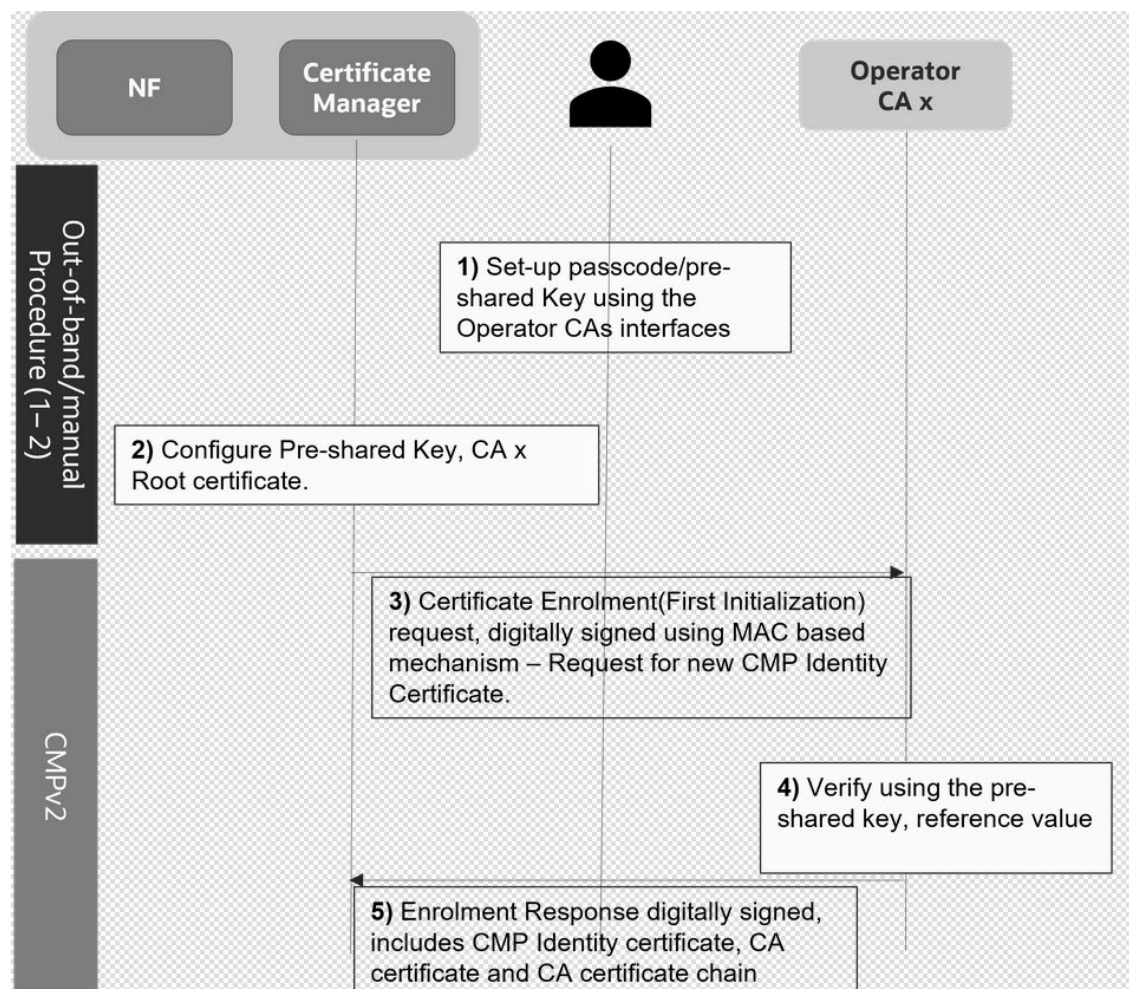
OCCM can be configured to establish trust between Oracle Communication Certificate (OCCM) and Certificate Authorities (CAs) by enabling PKI message protection in the following ways:

- MAC based trust establishment
- Certificate based trust establishment

3.2.1.1 MAC Based Trust Establishment

OCCM supports initial trust establishment with each of the configured CAs using the preconfigured pre-shared (MAC) key.

Figure 3-1 MAC Based Trust Establishment



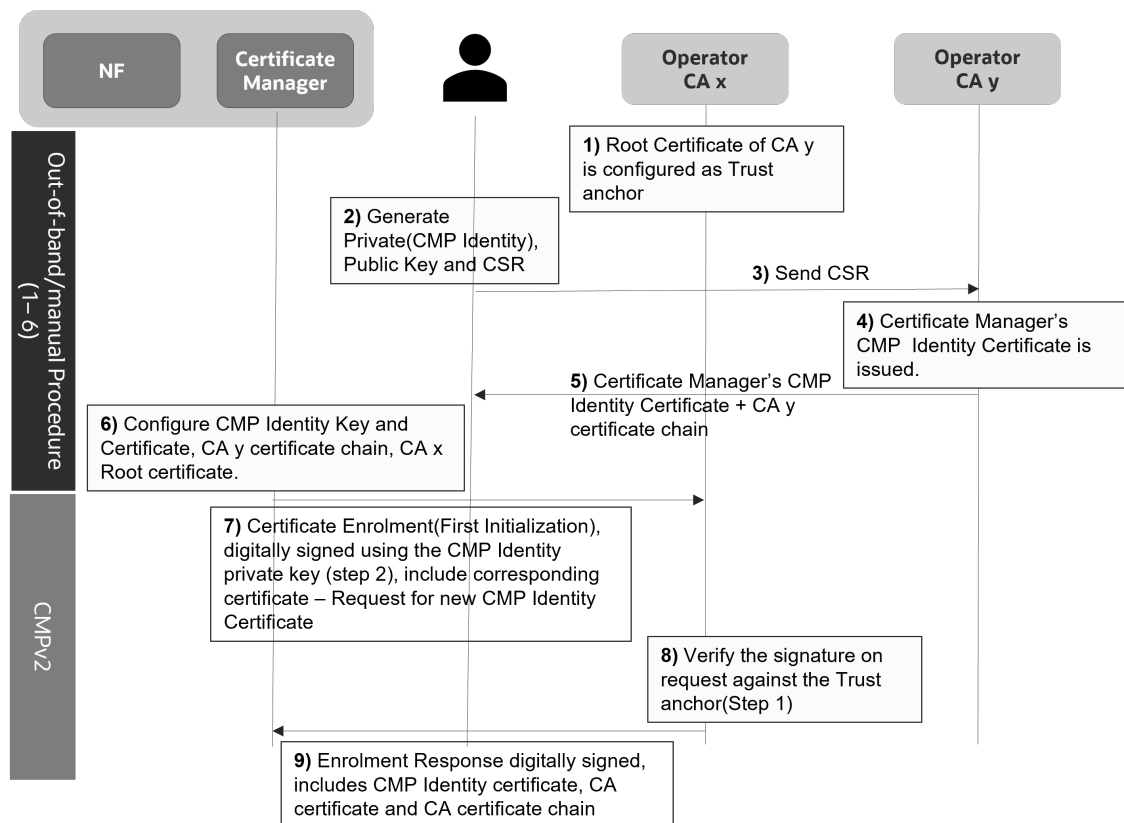
OCCM generates the key pair and requests for the CMP identity certificate for each of the configured CAs using the first Initialization Request. The first Initialization Request towards each of the CAs is signed using the preshared key. The CA authenticates the initialization request and signs the CMP identity certificate. OCCM can be configured to authenticate the

responses of the first initializing procedure using the preshared (MAC) key. All subsequent requests are always signed using the CMP identity key and certificate.

3.2.1.2 Certificate Based Trust Establishment

OCCM supports initial trust establishment with CA using the preconfigured private key and x.509 certificate.

Figure 3-2 Certificate Based Trust Establishment



OCCM signs the first initialization request towards a CA using preconfigured key or certificate.

OCCM can be configured to:

- continue using the same key and certificate to sign the subsequent CMPv2 requests OR
- generate a new certificate using the first Initialization Request

In case OCCM uses the same key and certificate to sign the subsequent CMPv2 requests, OCCM requests for generation of the NF certificate in the first Initialization Request.

In case OCCM generates a new certificate using the first Initialization Request (IR), OCCM requests for generation of CMP identity certificate (OCCM certificate) in the first initialization request. End entity certificate (NF certificate) certificate generation is requested from next Initialization Request onwards.

The End Entity certificate initialization requests and the certificate confirms are digitally signed using the CMP Identity Key. OCCM supports Proof of Possession (PoP) in the initialization request. The PoP of the signing key contains the algorithm identifier and signature. This signature is based on the certificates template structure.

3.3 Support for HTTPs Encryption

Managing HTTPs Encryption

This feature enables you to encrypt the traffic between OCCM and CAs using HTTPs. HTTPs encryption at the transport layer adds an additional layer of security.

OCCM, as a HTTP Client, supports HTTPs connections with CAs using One-Way TLS when authenticating the identity of the CAs. OCCM manages a TrustStore (CA Bundle) to validate the certificates presented by the CAs in the certificate message of the TLS handshake procedure. You can either use the same CA Bundle configuration for all the configured CAs, or different CA Bundles as per your requirements.

OCCM validates the CA certificate as per the RFC 5280 standards, and the TLS handshake can get rejected if the certificate is invalid, or expired:

- Certificate Path validation
- Certificate expiry
- Certificate Strict checking

OCCM supports the following TLS configurations:

- Version TLSv1.2 and TLSv1.3 including support for version rollback to TLSv1.2 in case the CA does not support TLSv1.3
- OCCM acts as the HTTP(s) client while communicating with CA and all the relevant requirements apply.

Configuring HTTPs Encryption

The HTTPs functionality can be manually configured by the operator. The operator can:

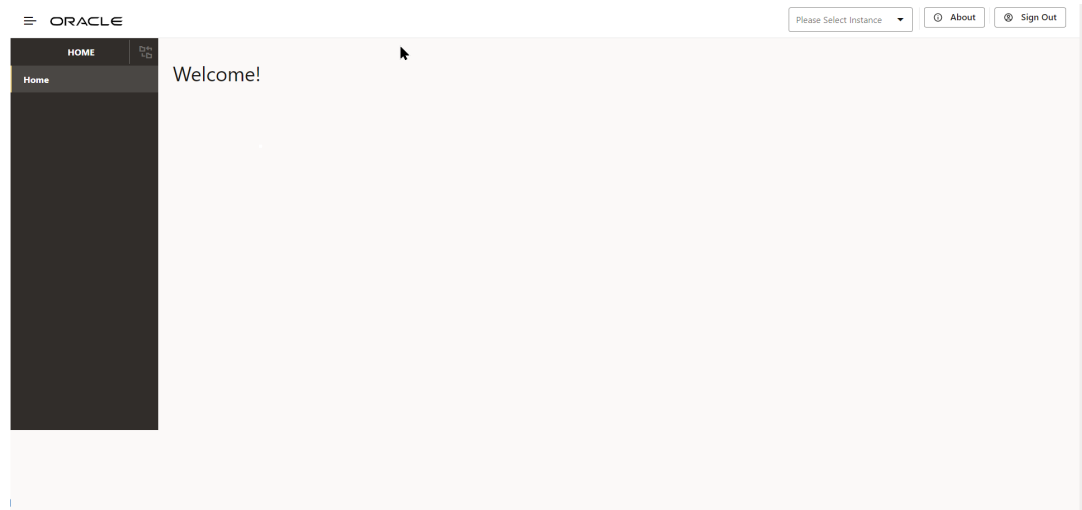
- configure and manually update the CA Bundle used to validate the TLS handshake.
- enable and disable the strict checking of the X.509 certificates presented for HTTPs. This verifies if the certificates are RFC 5280 compliant.
- enable or disable the checking of X.509 certificate critical extensions.

3.4 Accessing OCCM from CNC Console

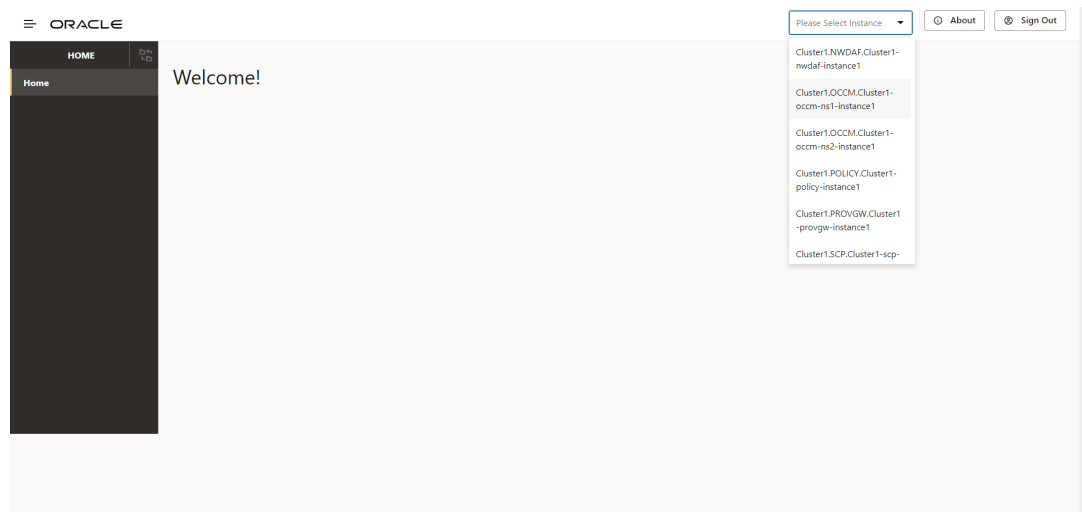
This section describes the procedure to access the OCCM cluster from the CNC Console GUI.

To access OCCM from CNC Console:

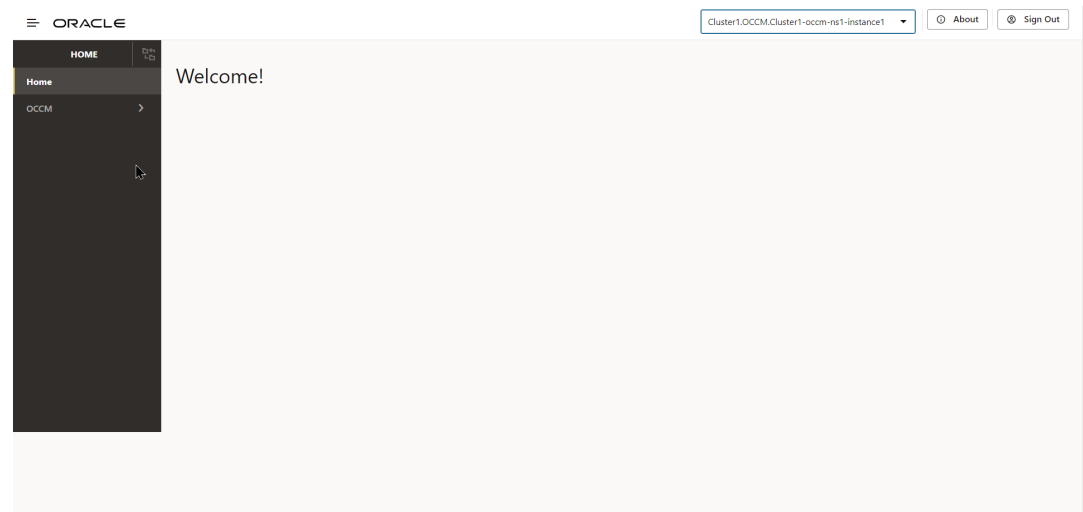
1. Log in to CNC Console using your login credentials.

Figure 3-3 CNC Console Landing Page

2. From the **Select Instance** drop-down, select the **OCCM** Instance.

Figure 3-4 OCCM Instance

The OCCM menu appears on the left pane.

Figure 3-5 OCCM Configuration Options

3.5 Managing Issuers

Issuers, also called Certificate Authorities (CAs), are a trusted entity that issues X.509 certificates. OCCM supports the following aspects of issuer management:

- Pre-configuration for OCCM Bootstrapping
- Creating Issuers
- Updating Issuers
- Deleting Issuers

3.5.1 Pre-configuration for OCCM Bootstrapping

The following secrets can be pre-configured for OCCM bootstrapping:

- **MAC Secret:** The MAC secret is a manually configured pre-shared key or password based MAC secret and reference. This is used by OCCM to sign the first initialization request. CA then validates the request and issues a signed OCCM certificate. For more information, see the 'Using the pre-shared key' section in [OCCM Certificate Configuration Modes](#). To create the MAC Secret, run the following command:

```
kubectl create secret generic <k8s secret name> --from-literal=<mac secret key>=<mac secret value> --from-literal=<reference key>=<reference value> -n <namespace>
```

For example:

```
kubectl create secret generic ca1-mac-secret --from-literal=pwd='pass:****' --from-literal=ref='abcd' -n ns1
```

- **CMP Identity Secret:** The CMP Identity secret is a manually configured private key and certificate, using which OCCM certificate is requested from CA. This is used by OCCM to sign the first initialization request. CA then validates the request and issues a signed OCCM certificate. You can also use the same private key and certificate as OCCM

certificate. For more information, see the 'Using the pre-configured private key and certificate' section in [OCCM Certificate Configuration Modes](#).

To create the CMP Identity Key, run the following command:

```
kubectl create secret generic <k8s secret name> --from-file=<cmp key file location> --from-file=<cmp cert file location> -n <namespace>
```

For example:

```
kubectl create secret generic ca1-cmp-identity-secret --from-file=cmpkey.pem --from-file=cmpcert.pem -n ns1
```

- **OCCM Trust Store Secret:** The OCCM Trust Store secret holds OCCM trust store information (CA certificates), and is used as a trust anchor when validating the digital signature included in the CMP responses.

To create the OCCM Trust Store secret, run the following command:

```
kubectl create secret generic <k8s secret name> --from-file=<CA root cert file location> --from-file=<Intermediate CA cert file location> --from-file=<CMP server cert file location> -n <namespace>
```

For example:

```
kubectl create secret generic ca1-occm-trust-store-secret --from-file=caroot.pem --from-file=intcacert.pem --from-file=servercert.pem -n ns1
```

- **TLS Trust Store Secret:** If TLS is enabled for issuer, TLS Trust Store secret should be provided, else it should be skipped. It holds the CA certificates to be used as trust anchors when authenticating the TLS server certificate. To create the TLS Trust Store secret, run the following command:

```
kubectl create secret generic <k8s secret name> --from-file=<CA cert file location> -n <namespace>
```

For example:

```
kubectl create secret generic ca1-tls-trust-store-secret --from-file=caroot1.pem -n ns1
```

HTTPS communication between OCCM and CA

OCCM supports HTTPS connections with CA using one-way TLS. To enable the same, the operator has to set enableTLS option in the issuer configuration to true and configure the HTTPS scheme server URL. TLS trust store has to be configured with trust anchors in order to authenticate the TLS server.

In order to enable or disable strict checking of the X.509 certificates presented for HTTPS, the following deployment time (helm) parameters can be configured.

- **occmConfig.cmp.config.tls.enableX509StrictCheck:** This field when set to true enables strict checking of the X.509 certificates presented for HTTPS. Errors are thrown for the certificates which are not compliant with RFC 5280.
- **occmConfig.cmp.config.tls.ignoreCriticalExtensionsCheck:**

This field when set to true ignores checking of the critical extensions in X.509 certificates presented for HTTPs.

Normally, if an unhandled critical extension is present that is not supported by OpenSSL, the certificate is rejected in compliance with RFC 5280.

Note

This configuration will be applied only when TLS is enabled for an issuer.

3.5.2 Creating Issuer

Issuers are resources that represent CAs and are able to generate signed certificates. You can configure issuers through REST API or using the CNC Console GUI. The maximum number of issuers that can be supported at a time is 30.

Configuring Issuer Using CNC Console GUI

To manually configure issuer using CNC Console GUI:

1. Log in to CNC Console using your login credentials and select the OCCM Instance.
2. Click **OCCM** from the left pane and then click **Issuer**.
3. Click **Add**. The **Create Issuer** page appears.

Figure 3-6 Create Issuer

Create Issuer

UUID:	UUID
Name:	Name
Server URL:	Server URL
Recipient Distinguished Name:	Recipient Distinguished Name
Issuer Distinguished Name:	Issuer Distinguished Name
Total Timeout (Seconds):	Total Timeout (Seconds) 720
Message Timeout (Seconds):	Message Timeout (Seconds) 120

4. Enter the following information on the Create Issuer page:

Table 3-1 Create Issuer

Field Name	Description
Name	Name of the Issuer
Server URL	Domain URL of CA

Table 3-1 (Cont.) Create Issuer

Field Name	Description
Recipient Distinguished Name	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. For example: /DC=org/DC=OpenSSL/DC=users/UID=123456+CN=John Doe
Issuer Distinguished Name	X509 issuer Distinguished Name of the CA server to place in the requested certificate template in IR or 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. For example: /DC=org/DC=OpenSSL/DC=users/UID=123456+CN=John Doe
Total Timeout (Seconds)	The total time in seconds allowed for the CMP transaction to complete.
Message Timeout (Seconds)	The total time (in seconds) a CMP request-response message round trip is allowed to take.

5. Under **Initial CMP Client(OCCM) Authentication Options**, enter the following information:

Table 3-2 Initial Authentication Options

Field Name	Possible Values
Type	MAC, SIGNATURE For more information, see OCCM Certificate Configuration Modes .
Digest Algorithm	SHA256, SHA384, SHA512
MAC Algorithm	HMACSHA256, HMACSHA384, HMACSHA512

Figure 3-7 Initial CMP Client(OCCM) Authentication Options

Initial CMP Client(OCCM) Authentication Options

Type:

Type

Digest Algorithm:

Digest Algorithm

MAC Algorithm:

MAC Algorithm

6. If you are using the password based MAC authentication mechanism, then under **MAC Authentication Input**, enter the following information:

Table 3-3 MAC Authentication Input

Field Name	Description
Namespace	Select the name of the Kubernetes namespace from the dropdown.
Secret Name	Kubernetes secret name.
Password Key	Kubernetes secret data key against which MAC secret is provided.
Reference Key	Kubernetes secret data key against which reference string is provided.

Figure 3-8 MAC Authentication Input

MAC Authentication Input

Namespace:

Namespace

Secret Name:

Secret Name

Password Key:

Password Key

Reference Key:

Reference Key

7. Under **Signature Authentication Input**, enter the following information:

Table 3-4 Signature Authentication Input

Field Name	Description
Namespace	Select the name of the Kubernetes namespace from the dropdown.
Secret Name	A unique secret name.
Key	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.
Cert	Kubernetes secret data key against which the pre-configured certificate (client's current CMP signer certificate) is provided.
Extra Certs	Extra Certificates, if any, for client authentication.

Figure 3-9 Signature Authentication Input

Signature Authentication Input

Namespace:

Namespace

Secret Name:

Secret Name

Key:

Key

Cert:

Cert

Extra Certs:

Extra Certs

8. Under **CMP Client Authentication Options For Other certificate**, enter the following information:

Table 3-5 CMP Client Authentication Options For Other certificate

Field Name	Possible Values
Type	SIGNATURE
Digest Algorithm	SHA256, SHA384, SHA512

Figure 3-10 CMP Client Authentication Options For Other certificate

CMP Client Authentication Options For Other certificate

Type:

Type

Digest Algorithm:

Digest Algorithm

9. Under **Signature Authentication Input**, enter the following information:

Table 3-6 Signature Authentication Input

Field Name	Description
Namespace	Select the name of the Kubernetes namespace from the dropdown.
Secret Name	A unique secret name
Key	Kubernetes secret data key against which OCCM key is provided or created based on whether OCCM certificate is created in manual or automatic mode.
Cert	Kubernetes secret data key against which OCCM certificate is provided or created based on whether OCCM cert is created in manual or automatic mode.
Extra Certs	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 certificate, based on whether OCCM cert is created in manual or automatic mode.

Figure 3-11 Signature Authentication Input

Signature Authentication Input

Namespace:

Namespace

Secret Name:

Secret Name

Key:

Key

Cert:

Cert

Extra Certs:

Extra Certs

10. Under **Occm Trust-Store Secret Input**, enter the following information:

Table 3-7 Occm Trust-Store Secret Input

Field	Description
Namespace	Select the name of the Kubernetes namespace from the dropdown.
Name	Kubernetes secret which holds OCCM trust store information (CA certificates).
Root CA Certs	The certificate(s), typically of root CAs, the client uses as trust anchors when validating the certificate issued by CA. Note: If server certificate is present, this is ignored.
Intermediate CA Certs	Any untrusted intermediate CA certificate(s) to use when validating newly enrolled certificates.
Server Cert	CMP server or CA server's certificate to expect and directly trust when validating the certificate issued by CA. Note: If this is present, root CA certificates will be ignored.

Figure 3-12 Occm Trust-Store Secret Input

Occm Trust-Store Secret Input

Namespace:

Namespace

Name:

Name

Root CA Certs:

Root CA Certs

Intermediate CA Certs:

Intermediate CA Certs

Server Cert:

Server Cert

11. Enter either the root CA certificates and intermediate CA certificate, or the server certificate in the respective fields.
12. Under TLS Configuration, enter the following information:

Table 3-8 TLS Configuration

Field	Description
Enable TLS	When set to true, HTTPS connection to CA is made. Ensure that you select scheme as HTTPS in server URL if this is set to true.

Table 3-9 TLS Trust-Store secret Input

Field	Description
Namespace	Kubernets namespace where TLS trust store secret is present. Select the name of the namespace from the dropdown.
Name	Kubernetes secret which holds TLS trust store information (CA certificates).
TLS trusted Certs	Trusted certificate(s) to use for validating the TLS server certificate.

Figure 3-13 Enable TLS

The screenshot shows a configuration panel for TLS. At the top, 'Enable TLS' is a toggle switch that is currently turned on. Below this, there is a section titled 'TLS Trust-Store Secret Input'. It contains three fields: 'Namespace' is a dropdown menu showing 'ns1'; 'Name' is a text input field containing 'ca1-tls-trust-store-secret'; and 'TLS Trusted Certs' is a text input field containing a file named 'ManagementCA.pem' with a small 'x' icon to its right.

13. Click **Save**.

3.5.3 Updating Issuer

The bulk certificate migration can be used to update the end point of the CA and Issuer by updating the field, such as server URL, recipient DN, and Issuer DN. As part of the bulk certificate migration, OCCM updates the link of all the certificate configurations previously linked to the Old CA to the new CA. The generated key pair and certificate overwrites the existing certificate. For more information, see [Bulk Migration of Certificates](#).

If no certificate configuration is attached to the issuer, you can update all fields in **Edit Issuer**. If a certificate configuration is attached, you can update only the scheme, from HTTP to HTTPS or from HTTPS to HTTP. Existing certificates are not affected by the scheme update. OCCM uses the updated scheme and server endpoint for CMPv2 procedures that run after the update.

To update the issuer:

1. Log in to CNC Console using your login credentials and select the OCCM Instance.
2. Click **OCCM** from the left pane and then click **Issuer**.

Figure 3-14 Issuer Page

Issuer			
Type to Filter			
UUID	Name	Server URL	Actions
529c448b-1294-423f-89b6-1bea91d9a447	ca21	http://ca-21-openssl-mock.occncc-thrust5-01.svc.thrust5.8080	
9a601658-cab9-4a4a-881e-d6f6c39791223	CA1	http://ca-0211-openssl-mock.occncc-thrust5-01.svc.thrust5.8080	
0f19d193-f8b6-461d-a910-a787a68dacc9	Test-CA	http://test-ca-openssl-mock.occncc-thrust5-01.svc.thrust5.8080	

- Click the edit icon next to the issuer that you want to update. The **Edit Issuer** page appears.

Figure 3-15 Edit Issuer

Edit Issuer	
UUID:	UUID 9a9cc934-c457-487f-bee8-a8d690f7a208
Name:	Name EJBCA-HTTPS-RA
Server URL:	Server URL https://thrust5:8445/ejbcas/publicweb/cmp/occmaliasra
Recipient Distinguished Name:	Recipient Distinguished Name /C=FR/OU=ORACLE/CN=Oracle RA LTE OFR LAB
Issuer Distinguished Name:	Issuer Distinguished Name /CN=
Total Timeout (Seconds):	Total Timeout (Seconds) 720
Message Timeout (Seconds):	Message Timeout (Seconds) 120
▼ Initial CMP Client(OCCM) Authentication Options	
Type:	Type MAC
MAC Algorithm:	MAC Algorithm HMACSHA256

- Edit the fields that you need to update and then click **Save**.

Note

If an issuer is used by a certificate configuration, only the HTTP or HTTPS scheme can be edited. To change other issuer fields, use bulk certificate migration.

3.5.4 Deleting Issuers

To delete issuers from CNC Console GUI:

- Log in to CNC Console using the login credentials, and select the OCCM Instance.
- Click **OCCM** from the left pane and then click **Issuer**. All the available issuers are listed.

Figure 3-16 Issuer

UUID	Name	Server URL	Actions
a852d6da-4e7b-43e9-96b3-59c47c2ca6c7	CA1	https://ca1-ns1.svc.thrust5:30443/...	[Edit] [Delete] [Share]
f2fca868-f8eb-4b8f-ae78-f02acbf278b8	CA2	http://ca2-ns1.svc.thrust5:30080/...	[Edit] [Delete] [Share]
190dd7c6-6040-43ab-a812-2eb530a7cc52	CA3	https://ca3-ns1.svc.thrust5:30443/...	[Edit] [Delete] [Share]

3. Select the issuer to be deleted and click **Delete** and then click **OK** on the confirmation prompt to delete the issuer.

Figure 3-17 Delete Issuer

UUID	Name	Server URL	Actions
a852d6da-4e7b-43e9-96b3-59c47c2ca6c7	CA1	https://ca1-ns1.svc.thrust5:30443/...	[Edit] [Delete] [Share]
f2fca868-f8eb-4b8f-ae78-f02acbf278b8	CA2	http://ca2-ns1.svc.thrust5:30080/...	[Edit] [Delete] [Share]
190dd7c6-6040-43ab-a812-2eb530a7cc52	CA3	https://ca3-ns1.svc.thrust5:30443/...	[Edit] [Delete] [Share]

Note

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

3.6 Managing Certificates

Once an issuer has been configured, an X.509 certificate can be requested. Each certificate configuration in OCCM is a certificate request. It specifies input fields that are used to generate a private key pair and a certificate signing request to obtain a signed certificate from the referenced issuer.

OCCM supports the following key aspects of certificate management:

- Creating CMP identity certificates (OCCM certificate)
- Creating end entity certificates (NF certificates)
- Updating Certificates
- Monitoring certificate expiry and renewing OCCM and NF certificates
- Polling for certificates
- Deleting the certificate configuration

3.6.1 Issuer (CA) HealthCheck

This feature is a lightweight mechanism to regularly monitor the health of OCCM's connectivity with configured issuers (CA). It does so by configuring a special-purpose certificate termed a HealthCheck Certificate. This certificate is manually created per issuer and configured to

renew on configurable interval (preferably daily). This feature enables OCCM to actively verify communication with the Certificate Authority (CA) by initiating certificate lifecycle operations using a dedicated health check certificate. Renewal success or failure events are displayed via a Grafana dashboard, providing real-time visibility into CA integration health. This acts as a proactive monitoring mechanism, helping detect and resolve issues before they impact NF certificates. Instead of performing manual CA health validation, the system:

- Issues a controlled dummy certificate
- Tracks issuance and renewal behavior
- Publishes Prometheus metrics
- Displays real-time status on Grafana dashboard

This enhancement improves system observability without impacting existing certificate workflows.

Here is a procedure to create a health check certificate using CNC Console:

1. Log into CNC Console using your log in credentials and select the OCCM Instance.
2. From the left pane, click on OCCM. Proceed to click certificate.
3. To make the Create Certificate page appear, click on Add. You will be able to create a Health Check Certificate.

Figure 3-18 Health Check Certificate

4. Enter the following information to create a Health Check Certificate:

a. Table 3-10 Information to Create Health Check Certificate

Field Name	Description and Possible Values
Name	Name of the certificate
Cert Type	Select ISSUER_HEALTHCHECK from drop down.
Network Function	HealthCheck-<NF name>
Purpose	Name of the issuer for the certificate <For which operator wants to create health check>
Creation Mode	Possible values is AUTOMATIC. (For health check certificate only Automatic would be preferred)

5. Ensure the following information under Private Key Options:

a. **Table 3-11 Information for Private Key Options**

Field Name	Possible Values
Key Algorithm	RSA, EC
Key Encoding	DER, PEM
Key Size	KEYSIZE_2048, KEYSIZE_4096
Elliptic Curve	SECP384r1, SECP256r1

b. **Figure 3-19 Private Key Options**

Private Key Options

Key Algorithm:

Key Encoding:

Key Size:

Elliptic Curve:

6. The Private Key Output section gets populated automatically from the corresponding issuer when the certificate is saved.

a. **Table 3-12 Private Key Output**

Field Name	Description
Namespace	Name of the namespace
Secret Name	Kubernetes Secret Name
Key	Kubernetes secret key against which the key-pair will be stored.

b. **Figure 3-20 Private Key Output**

Private Key Output

Namespace:

Secret Name:

Key:

7. Enter the following information under the Public Key Certificate Option:

a. **Table 3-13 Public Key Certificate Options**

Field Name	Description
Key Usage	Value(s): DIGITAL_SIGNATURE
Extended Key Usage	Value(s): CLIENT_AUTH and SERVER_AUTH
Basic Constraints	Value(s): END_ENTITY

Table 3-13 (Cont.) Public Key Certificate Options

Field Name	Description
Subject	- Country: Enter country code - State: Enter state code - Location: City or town where company is legally located - Organization: Name of your organization - Organization Unit: Name of business unit - Common Name: The Common Name (CN) represents the server name to be protected by the certificate - Requested Validity (Days): Number of days requested for which the certificate will be valid.

b. Figure 3-21 Public Key Certificate

Public Key Certificate Options

Key Usage

Critical: ☒

Value(s):

Extended Key Usage

Critical: ☐

Value(s):

Basic Constraints

Critical: ☐

Value:

Subject

Country:

State:

Location:

8. Enter the following information under Subject Alternate Names:**a. Table 3-14 Subject Alternate Names**

Field Name	Description
IP Address	The IPs you want to protect under this certificate
DNS Names	List of DNS domain names
URI ID API Roots	List of URI ID
URI ID URNs	List of URI ID

b. **Figure 3-22 Subject Alternate Names**

Subject Alternate Names

Critical: ☐

IP Addresses:

DNS Names:

URI ID API Roots:

URI ID URNs:

9. The Certificate Output section gets populated automatically from the corresponding issuer when the certificate is saved.

a. **Table 3-15 Certificate Output**

Field Name	Description
Namespace	Name of the namespace
Secret Name	Name of the secret
Key	The key against which the certificate will be populated.

b. **Figure 3-23 Certificate Output**

Certificate Output

Namespace:

Secret Name:

Key:

10. Click save.

3.6.2 Creating CMP (OCCM) Identity Certificate

CMP identity certificate (OCCM certificate) is used by OCCM to identify itself to the external CA in CMPv2 messages.

To create a CMP identity certificate using CNC console:

1. Log in to CNC Console using your login credentials and select the OCCM Instance.
2. Click **OCCM** from the left pane and then click **Certificate**.
3. Click **Add**. The **Create Certificate** page appears.

Figure 3-24 Create CMP identity Certificate

The screenshot shows the 'Create Certificate' form in the OCCM console. The form is divided into two columns. The left column contains a sidebar with 'Certificate' selected, and the right column contains the form fields. The fields are: UUID, Name, Cert Type (dropdown), Network Function, Purpose, Issuer (dropdown), Creation Mode (dropdown), Overwrite Secret (toggle), Renew Before Expiration (Days) (input), and Namespace. The 'Cert Type' is set to 'CMP (OCCM) Identity certificate' and 'Issuer' is set to 'OCCM'.

4. Enter the following information:

Table 3-16 Create CMP identity Certificate

Field Name	Description and Possible Values
Name	Name of the certificate.
Cert Type	CMP (OCCM) Identity certificate
Network Function	OCCM
Purpose	Purpose of the OCCM certificate.
Issuer	Select the name of the issuer for the certificate.
Creation Mode	Possible values are MANUAL and AUTOMATIC. For more information, see OCCM Certificate Configuration Modes .
Overwrite Secret	This option is used to overwrite the existing Kubernetes secret with a new certificate.
Renew Before Expiration (Days)	The number of days before the certificate expiration date when renewal should be triggered.

5. Under **Private Key Options**, enter the following information:

Table 3-17 Private Key Options

Field Name	Possible Values
Key Algorithm	RSA, EC
Key Encoding	DER, PEM
Key Size	KEYSIZE_2048, KEYSIZE_4096
Elliptic Curve	SECP256r1, SECP384r1

Figure 3-25 Private Key Options

The screenshot shows the 'Private Key Options' form in the OCCM console. The form is a single column with a header 'Private Key Options' and four dropdown menus: Key Algorithm, Key Encoding, Key Size, and Elliptic Curve.

6. The **Private Key Output** section is auto populated from corresponding issuer after the certificate is saved. You can skip this section.

Table 3-18 Private Key Output

Field Name	Description
Namespace	Select the name of the namespace.
Secret Name	Kubernetes Secret Name.
Key	Kubernetes secret key against which the key-pair will be stored.

Figure 3-26 Private Key Output

Private Key Output

Namespace:

Secret Name:

Key:

7. Under **Public Key Certificate Options**, enter the following:

Table 3-19 Public Key Certificate Options

Field Name	Description
Key Usage	Value(s): DIGITAL_SIGNATURE
Extended Key Usage	Value(s): CLIENT_AUTH and SERVER_AUTH
Basic Constraints	Value(s): END_ENTITY
Subject	Country: Enter country code: State: Enter state code. Location: City or town where company is legally located. Organization: Name of your organization. Organisation Unit: Name of business unit. Common Name: The Common Name (CN) represents the server name to be protected by the certificate. Requested Validity (Days): Number of days requested for which the certificate will be valid.

Figure 3-27 Public Key Certificate Options

Public Key Certificate Options

Key Usage

Critical:☒

Value(s):

Value(s)

DIGITAL_SIGNATURE XKEY_ENCIPHERMENT X

Extended Key Usage

Critical:☐

Value(s):

Value(s)

CLIENT_AUTH XSERVER_AUTH X

Basic Constraints

Critical:☐

Value:

ValueEND_ENTITY

Subject

Country:

Country

State:

State

Location:

Location

Organization:

Organization

Organization Unit:

Organization Unit

Common Name:

Common Name

Requested Validity (Days):

Requested Validity (Days)365

8. Under **Subject Alternate Names**, enter the following:

Table 3-20 Subject Alternate Names

Field Name	Description
IP Address	The IPs you want to protect under this certificate.
DNS Names	List of DNS domain names.
URI ID API Roots	List of URI IDs.
URI ID URNs	List of URI IDs.

Figure 3-28 Subject Alternate Names

Subject Alternate Names

Critical:☒

IP Addresses:

IP Addresses

DNS Names:

DNS Names

URI ID API Roots:

URI ID API Roots

URI ID URNs:

URI ID URNs

9. The **Certificate Output** section is auto populated from corresponding issuer after the certificate is saved. You can skip this section.

Table 3-21 Certificate Output

Field Name	Description
Namespace	Select the name of the namespace.
Secret Name	Name of the secret.
Key	The key against which the certificate will be populated.

Figure 3-29 Certificate Output

Figure 3-29 shows the 'Certificate Output' configuration form. It includes a dropdown for 'Namespace', a text input for 'Secret Name', and a text input for 'Key'.

10. (Optional) Under **Certificate Chain Output**, enter the following:

Table 3-22 Certificate Chain Output

Field Name	Description
Namespace	Select the name of the namespace.
Secret Name	Name of the secret.
Key	Kubernetes secret key against which the certificate chain will be stored.

Figure 3-30 Certificate Chain Output

Figure 3-30 shows the 'Certificate Chain Output' configuration form, which is identical in structure to Figure 3-29, with fields for Namespace, Secret Name, and Key.

If the **Certificate Chain Output** section is filled, then the certificate chain can either be obtained from the CA or can be configured manually. This is based on the `extractCertChainFromCmpResponse` helm parameter. For more information, see *Oracle Communications Cloud Native Core, Certificate Management Installation, Upgrade, and Fault Recovery Guide*.

Note

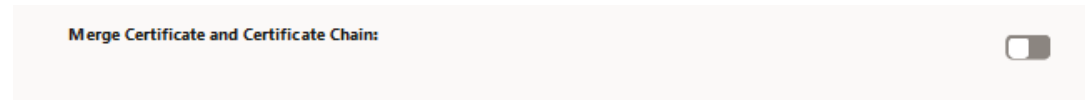
`extractCertChainFromCmpResponse`: This field, when set to true, specifies that certificate chain will be extracted from CA's CMP response message. When false, the operator can configure the chain manually. This certificate chain is used in the TLS handshake along with the certificate.

11. **Merge Certificate and Certificate Chain:**

To get the complete certificate chain including the leaf certificate and the intermediate CA certificate(s), enable the **Merge Certificate and Certificate Chain** option and provide the

same output secret for both **Certificate Output** and **Certificate chain output** fields. The **Certificate Output** secret can be taken from the issuer's **CMP client options for Other Certificate** field.

Figure 3-31 Merge Certificate and Certificate Chain



Merge Certificate and Certificate Chain: ☐

Note

This is an optional field and is set to false by default. In case the issuer CA doesn't respond with the chain (intermediate CA certificates), only the leaf certificate will be populated against the specified Kubernetes secret key.

12. Click **Save**.

3.6.2.1 OCCM Certificate Configuration Modes

The following section highlights the configuration applicable to these modes and control how the OCCM certificates are generated. The purpose of the following issuer configuration and certificate configuration sections is to highlight the difference in the fields for different modes.

OCCM can be configured with one of the following modes available to establish the initial Trust with the CA(s):

- Using the pre-shared key
- Using the pre-configured private key and certificate

Using the Pre-shared Key

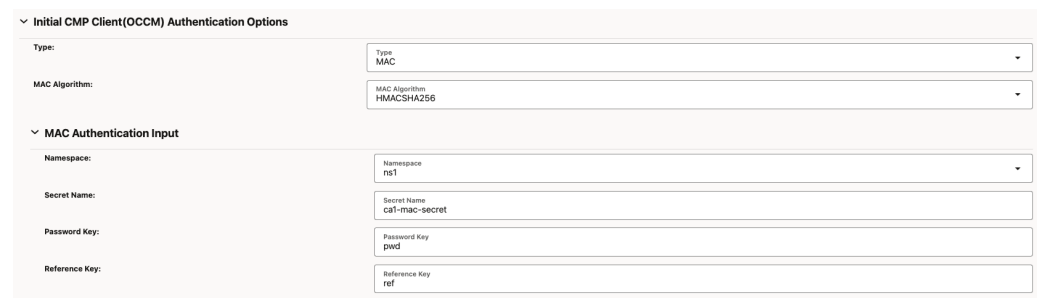
With this mode of configuration, OCCM signs the first initialization request using the pre-shared key. CA validates the request and issues a signed OCCM certificate.

1. **Issuer configuration**

To configure the issuer using the pre-shared key,

- a. The MAC authentication input must be provided under **CMP Client Authentication Options for OCCM Certificate**.

Figure 3-32 Initial CMP Client(OCCM) Authentication Options



Initial CMP Client(OCCM) Authentication Options

Type:

MAC Algorithm:

MAC Authentication Input

Namespace:

Secret Name:

Password Key:

Reference Key:

- b. OCCM key and certificate output location must be specified under **CMP Client Authentication Options for Other Certificate**. OCCM certificate received from CA will be written here.

Figure 3-33 CMP Client Authentication for Other Certificates

▼ **CMP Client Authentication Options For Other certificate**

Type:

Digest Algorithm:

▼ **Signature Authentication Input**

Namespace:

Secret Name:

Key:

Cert:

Extra Certs:

2. Certificate configuration

To configure the OCCM Certificate using the pre-shared key, select OCCM from the **Cert Type** drop down, and select AUTOMATIC from the **Life Cycle Management** on the **Create Certificate** page.

Figure 3-34 CMP Identity (OCCM) Certificate Configuration using Pre-shared Key

< OCCM  

Create Certificate

UUID:

Name:

Cert Type:

Network Function:

Purpose:

Issuer:

Creation Mode:

Certificate lifecycle management

Overwrite Secret: ☐

Option is used to overwrite the existing K8s secret with new certificate

Renew Before Expiration (Days):

Namespace:

3. Select **Issuer** from the **Issuer** dropdown in the **Create Certificate** GUI screen.

Using the pre-configured private key and certificate

The pre-configured private key and certificate mode can be used in the following two ways:

1. The pre-configured private key and certificate (generated out of band) can itself be used as the OCCM certificate.
 - a. **Issuer Configuration**
 - i. Here, you must skip the **CMP Client Authentication Options for OCCM Certificate**.

Figure 3-35 Issuer Configuration

Initial CMP Client(OCCM) Authentication Options

Type:

Digest Algorithm:

MAC Algorithm:

MAC Authentication Input

Namespace:

Secret Name:

Password Key:

Reference Key:

Signature Authentication Input

Namespace:

Secret Name:

Key:

Cert:

Extra Certs:

- ii. OCCM key and certificate output location must be specified under **CMP Client Authentication Options for Other Certificate**. Manually created OCCM key and certificate location should be specified here.

Figure 3-36 CMP Client Authentication Options for Other Certificate

CMP Client Authentication Options For Other certificate

Type:

Digest Algorithm:

Signature Authentication Input

Namespace:

Secret Name:

Key:

Cert:

Extra Certs:

b. **OCCM Certificate Configuration**

To configure the OCCM certificate, select OCCM from the **Cert Type** drop down, and select MANUAL from the **Life Cycle Management** on the **Create Certificate** page.

Figure 3-37 OCCM Certificate Configuration

Create Certificate

UUID:

UUID

Name:

Name
OCCM

Cert Type:

Cert Type
OCCM

Network Function:

Network Function
OCCM

Purpose:

Purpose

Issuer:

Issuer
CA1

Creation Mode:

Creation Mode
MANUAL

Renew Before Expiration (Days):

Renew Before Expiration (Days)
14

Namespace:

Namespace

- 2. OCCM signs the first initialization request using the pre-configured private key and certificate. CA validates the request and issues a signed OCCM certificate.
 - a. **Issuer Configuration**
Here, to configure the issuer,
 - i. Provide the Signature Authentication Input under **CMP Client Authentication Options for OCCM Certificate**.

Figure 3-38 Initial CMP Client(OCCM) Authentication Options

Initial CMP Client(OCCM) Authentication Options

Type:

Type
SIGNATURE

Digest Algorithm:

Digest Algorithm
SHA256

Signature Authentication Input

Namespace:

Namespace
ns1

Secret Name:

Secret Name
ca1-cmp-identity-secret

Key:

Key
cmpkey.pem

Cert:

Cert
cmcert.pem

Extra Certs:

Extra Certs

- ii. OCCM key and certificate output location must be specified under **CMP Client Authentication Options for Other Certificate**. OCCM certificate received from CA will be written here.

Figure 3-39 CMP Client Authentication Options for Other Certificate

CMP Client Authentication Options For Other certificate

Type:

Type
SIGNATURE

Digest Algorithm:

Digest Algorithm
SHA256

Signature Authentication Input

Namespace:

Namespace
ns1

Secret Name:

Secret Name
ca1-occm-key-cert-secret

Key:

Key
occmkey.pem

Cert:

Cert
occmcert.pem

Extra Certs:

Extra Certs

b. OCCM Certificate Configuration

To configure the OCCM certificate, select OCCM from the **Cert Type** drop down, and select AUTOMATIC from the **Life Cycle Management** on the **Create Certificate** page.

Figure 3-40 Certificate Configuration

The screenshot shows the 'Create Certificate' page in the OCCM console. The left sidebar has a dark theme with 'Certificate' selected under the 'OCCM' tab. The main form is light gray and contains the following fields:

- UUID:** A text input field.
- Name:** A text input field with the value 'OCCM CA1'.
- Cert Type:** A dropdown menu with 'OCCM' selected.
- Network Function:** A text input field with the value 'OCCM'.
- Purpose:** A text input field with the value 'CMP Client Authentication'.
- Issuer:** A dropdown menu with 'CA1' selected.
- Creation Mode:** A dropdown menu with 'AUTOMATIC' selected.
- Overwrite Secret:** A toggle switch that is disabled.
- Renew Before Expiration (Days):** A text input field with the value '14'.
- Namespace:** A text input field.

Below the 'Creation Mode' dropdown, there is a note: 'Certificate lifecycle management' and 'Option is used to overwrite the existing K8s secret with new certificate'.

Note

This configuration is available for each of the issuers, therefore the modes for the CAs can be controlled individually.

3.6.3 Creating End Entity (NF) Certificates

End entity certificate (NF Certificate) are used by NF and applications for various purposes, such as TLS.

To create End Entity certificate using CNC Console GUI:

1. Log in to CNC Console using your login credentials and select the OCCM Instance.
2. Click **OCCM** from the left pane and then click **Certificate**.
3. Click **Add**. The **Create Certificate** page appears.

Figure 3-41 Create NF Certificate

This screenshot is identical to Figure 3-40, showing the 'Create Certificate' page in the OCCM console. The left sidebar has a dark theme with 'Certificate' selected under the 'OCCM' tab. The main form is light gray and contains the following fields:

- UUID:** A text input field.
- Name:** A text input field.
- Cert Type:** A dropdown menu.
- Network Function:** A text input field.
- Purpose:** A text input field.
- Issuer:** A dropdown menu.
- Creation Mode:** A dropdown menu.
- Overwrite Secret:** A toggle switch.
- Renew Before Expiration (Days):** A text input field with the value '14'.
- Namespace:** A text input field.

Below the 'Creation Mode' dropdown, there is a note: 'Option is used to overwrite the existing K8s secret with new certificate'.

4. Enter the following information:

Table 3-23 Create NF Certificate

Field Name	Description and Possible Values
Name	Name of the certificate.
Cert Type	Select OTHER for End Entity (NF) certificates.
Network Function	Name of the End Entity (NF).
Purpose	Purpose of the End Entity (NF) certificate.
Issuer	Select the name of the issuer for the certificate.
Creation Mode	The mode of certificate creation. Possible values are MANUAL and AUTOMATIC.
Overwrite Secret	This option is used to overwrite the existing Kubernetes secret with a new certificate.
Renew Before Expiration (Days)	The number of days before the certificate expiration date when renewal should be triggered. The maximum number of days for renewal is certificate validity minus 1 day and the minimum is 1 day.

5. Under **Private Key Options**, enter the following information:

Table 3-24 Private Key Options

Field Name	Description and Possible Values
Key Algorithm	The cryptographic algorithm used to generate the key pair. Possible values are RSA, EC
Key Encoding	The file format used to store or transmit the key material. Possible values are DER, PEM
Key Size	The strength/length of the RSA modulus(not applicable to EC). Possible values are KEYSIZE_2048, KEYSIZE_4096
Elliptic Curve	The specific curve used when Key Algorithm is EC (ignored for RSA). Possible values are SECP256r1, SECP384r1

Figure 3-42 Private Key Options

Private Key Options

Key Algorithm:

Key Encoding:

Key Size:

Elliptic Curve:

6. Under **Private Key Output**, enter the following information:

Table 3-25 Private Key Output

Field Name	Description
Namespace	Select the name of the namespace.

Table 3-25 (Cont.) Private Key Output

Field Name	Description
Secret Name	Kubernetes Secret Name.
Key	Kubernetes secret key against which the key-pair will be stored.

Figure 3-43 Private Key Output

Private Key Output

Namespace:

Secret Name:

Key:

7. Under **Public Key Certificate Options**, enter the following:

Table 3-26 Public Key Certificate Options

Field Name	Description and Possible Values
Key Usage	X.509 extension that defines the permitted cryptographic operations for the certificate's public key. Possible Values are DIGITAL_SIGNATURE
Extended Key Usage	X.509 extension that narrows acceptable application contexts for the certificate. Possible Values are CLIENT_AUTH and SERVER_AUTH
Basic Constraints	X.509 extension indicating whether the certificate is a End Entity and, if so, its path length constraints. Possible values are END_ENTITY
Subject	Country: Enter country code. State: Enter state code. Location: City or town where company is legally located. Organization: Name of your organization. Organisation Unit: Name of business unit. Common Name: The Common Name (CN) represents the server name to be protected by the certificate. Requested Validity (Days): Number of days requested for which the certificate will be valid.

Figure 3-44 Public Key Certificate Options

Public Key Certificate Options

Key Usage

Critical: ☒

Value(s):

Value(s)
DIGITAL_SIGNATURE x KEY_ENCIIPHERMENT x

Extended Key Usage

Critical: ☐

Value(s):

Value(s)
CLIENT_AUTH x SERVER_AUTH x

Basic Constraints

Critical: ☐

Value:

Value
END_ENTITY

Subject

Country:

Country

State:

State

Location:

Location

Organization:

Organization

Organization Unit:

Organization Unit

Common Name:

Common Name

Requested Validity (Days):

Requested Validity (Days)
365

Note

The **Critical** toggle on an X.509 extension tells certificate consumers they must understand and enforce that extension or reject the certificate. If an extension is non-critical, a consumer that doesn't understand it may ignore it. Mark only those extensions critical that clients must understand to safely use the certificate (e.g., Basic Constraints and Key Usage).

- Under **Subject Alternate Names**, enter the following:

Table 3-27 Subject Alternate Names

Field Name	Description
IP Address	The IPs you want to protect under this certificate.
DNS Names	List of DNS domain names.
URI ID API Roots	List of URI ID (API root of the NF Instance).
URI ID URNs	List of URI ID (URN of the NFInstanceId).

Figure 3-45 Subject Alternate Names

▼ Subject Alternate Names

Critical: ☒

IP Addresses:

DNS Names:

URI ID API Roots:

URI ID URNs:

9. Under **Certificate Output**, enter the following for the NF certificate:

Table 3-28 Certificate Output

Field Name	Description
Namespace	Select the name of the namespace. The list of certificates on the certificate page can be filtered by the namespace name selected in this field.
Secret Name	Name of the secret.
Key	The key against which the certificate will be populated.

Figure 3-46 Certificate Output

▼ Certificate Output

Namespace:

Secret Name:

Key:

10. (Optional) Under **Certificate Chain Output**, enter the following:

Table 3-29 Certificate Chain Output

Field Name	Description
Namespace	Select the name of the namespace.
Secret Name	Name of the secret.
Key	Kubernetes secret key against which the certificate chain will be stored.

Figure 3-47 Certificate Chain Output

▼ Certificate Chain Output

Namespace:

Secret Name:

Key:

If the **Certificate Chain Output** section is filled, then the certificate chain (intermediate CA certificates) can either be obtained from the CA or can be configured manually. This is

based on the `extractCertChainFromCmpResponse` helm parameter. For more information, see *Oracle Communications Cloud Native Core, Certificate Management Installation, Upgrade, and Fault Recovery Guide*.

Note

`extractCertChainFromCmpResponse`: This field when set to true specifies that certificate chain will be extracted from CA's CMP response message. When false, the operator can configure the chain manually. This certificate chain can be used in the TLS handshake along with the certificate.

11. Merge Certificate and Certificate Chain.

To get the complete chain including the leaf certificate and the intermediate CA certificate(s), enable the **Merge Certificate and Certificate Chain** option and provide the same output secret for both **Certificate Output** and **Certificate chain output** fields. The **Certificate Output** secret can be taken from the Issuer's **CMP client options for Other Certificate** field.

Figure 3-48 Merge Certificate and Certificate Chain

The image shows a horizontal toggle switch labeled "Merge Certificate and Certificate Chain:". The switch is currently in the "off" position, indicated by a grey slider.

Note

This is an optional field and is set to false by default. In case the issuer CA doesn't respond with the chain (intermediate CA certificates), only the leaf certificate will be populated against the specified Kubernetes secret key.

For example, The certificate chain (leaf certificate and intermediate CA certificate(s)) will be populated against the key `nrfcertchain.pem` of the Kubernetes secret `nrf-tls-secret` present in namespace `ns1`.

Figure 3-49 Sample Certificate Output and Certificate Chain Output

The image shows a configuration form with two sections: "Certificate Output" and "Certificate Chain Output". Each section has three input fields: "Namespace:" (dropdown menu), "Secret Name:" (text input), and "Key:" (text input). Below these sections is a toggle switch labeled "Merge Certificate and Certificate Chain:".

Section	Field	Value
Certificate Output	Namespace:	ns1
	Secret Name:	nrf-tls-secret
	Key:	nrfcertchain.pem
Certificate Chain Output	Namespace:	ns1
	Secret Name:	nrf-tls-secret
	Key:	nrfcertchain.pem
Merge Certificate and Certificate Chain:		☐

12. (Optional) Under **CA Bundle Input**, enter the following information:

Table 3-30 CA Bundle Input

Field Name	Description
Namespace	Select the name of the namespace.
Secret Name	Name of the secret.
Key	Kubernetes secret key against which CA bundle certificate(s) will be stored.

Figure 3-50 CA Bundle Input

13. Click **Save**.

For sample NF configuration, see [Creating End Entity \(NF\) Certificate Using OCCM - Sample Configuration](#).

3.6.4 Updating Certificate Configurations

This feature enables users to edit specific fields within the certificate configuration. Subsequently, OCCM performs actions based on the updated fields, thereby modifying the end-entity certificate. The specific action undertaken hinges on the chosen field and the presence or absence of the corresponding certificate (secret) within the system. The edit scenarios can be categorized as follows:

- [When the corresponding certificate secret is not created](#)
- [When the corresponding certificate secret is already created or already exists](#)

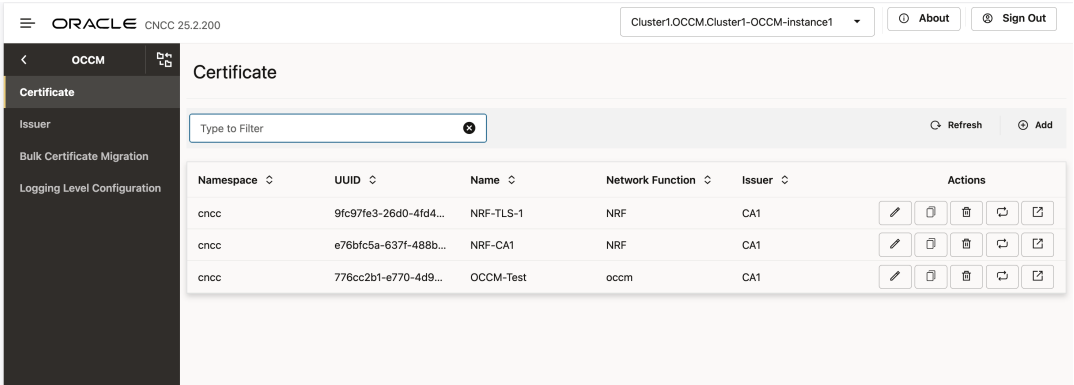
Note

Only end entity certificates can be edited. Editing CMP identity certificate is not supported.

To update the certificate using CNC console:

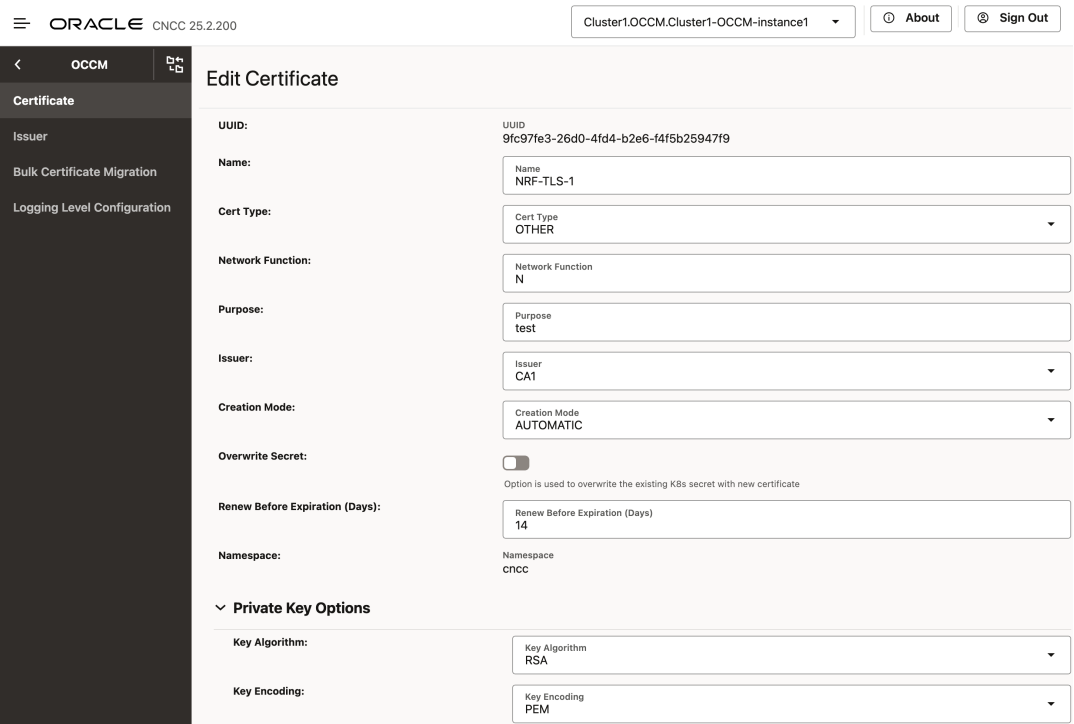
1. Log in to CNC Console using your login credentials and select the OCCM Instance.
2. Click **OCCM** from the left pane and then click **Certificate**.
This will list all the certificate configurations and can be filtered by the following:
 - Namespace name
 - UUID
 - Name of the certificate
 - Network Function
 - Issuer Name

Figure 3-51 Certificate



3. Click the edit icon next to the certificate that you want to update. The Edit Certificate page appears.

Figure 3-52 Edit Certificate



4. Edit the fields that you need to update and then click **Save**.

Figure 3-53 Edit Certificate: Private Key Options

Private Key Options

Key Algorithm: RSA

Key Encoding: PEM

Key Size: KEYSIZE_2048

Private Key Output

Namespace: ns1

Secret Name: nrf-secret-tls

Key: nrkey.pem

Public Key Certificate Options

Key Usage

Critical: ☒

Value(s): DIGITAL_SIGNATURE X KEY_ENIPHERMENT X

Extended Key Usage

Critical: ☐

Value(s): CLIENT_AUTH X SERVER_AUTH X

Figure 3-54 Edit Certificate: Certificate Output

Certificate Output

Namespace: ns1

Secret Name: nrf-secret-tls

Key: nrcert.pem

Certificate Chain Output

Namespace: ns1

Secret Name: nrf-secret-tls

Key: nrcertchain.pem

Merge Certificate and Certificate Chain: ☐

CA Bundle Input

Namespace:

Secret Name:

Key:

Save Cancel

Modification of the End Entity Certificate Configuration when the Corresponding Certificate Secret is not Created

OCCM supports the update of all the end entity certificate configuration fields when the certificate secret is not created. The updated values are considered for the subsequent retries. Click **View** next to certificate configuration to verify the configuration. The updated configuration must be reflected.

Modification of the End Entity Certificate Configuration when the Corresponding Certificate Secret is Already Created or Exists

Non-editable Fields

The following fields cannot be edited when the certificate secret is already created:

- Name
- Cert Type
- Network Function
- Creation Mode
- Overwrite Secret
- kubernetes secret details such as namespace, name, and key

Editable Fields

The remaining fields are editable and can be further categorized as follows:

1. When the following fields are updated, OCCM uses the CMPv2 IR to recreate the certificate. The generated certificate overwrites the existing certificate secret, if present.
 - Linked CA (Issuer)
 - Private Key options:
 - Key Algorithm
 - Elliptic curve
 - Key Encoding
 - Key Size
 - Subject
 - Values of Certificate Extensions:
 - Key Usage
 - Extended Key Usage
 - SAN

Click the **View** button next to certificate configuration to verify the configuration. The updated configuration must reflect. You can also verify the status of recreate certificate action triggered from the Grafana dashboard. For more information, see [Recreating Certificates](#).

2. When the following fields are updated, no immediate action is taken. The updated values are considered for subsequent operations, such as renewal or recreate.
 - Purpose
 - Requested Validity
 - "Critical" toggles for the following certificate extensions:
 - Key Usage
 - Extended Key Usage
 - SAN
 - Basic Constraints

Click the **View** button next to certificate configuration to verify the configuration. The updated configuration must be reflected.

3. When you update the **Renew Before** field, the updated configuration is considered for renewal with immediate effect. The renewal is rescheduled based on the updated configuration. Click **View** next to certificate configuration to verify the configuration. The updated configuration must be reflected.

For more information, see [Updating Certificates - Sample Configuration](#).

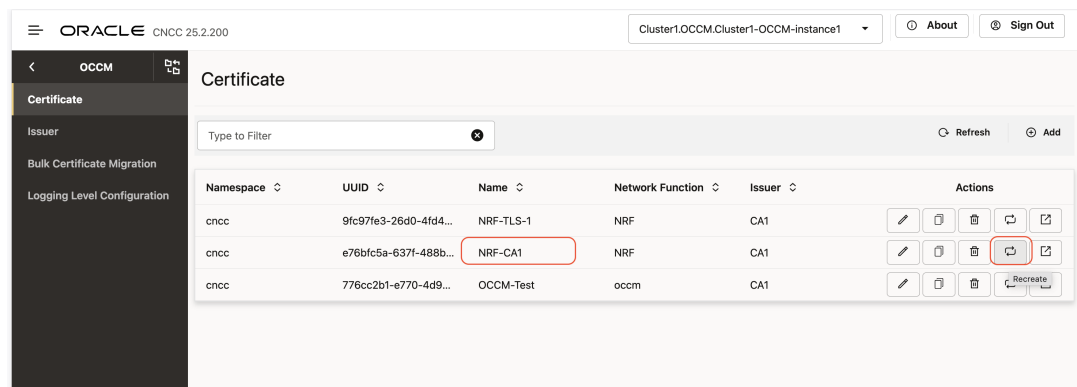
3.6.5 Recreating Certificates

OCCM supports on-demand reissuance of certificate with the same configuration by using the certificate recreate functionality. The CMPv2 IR procedure is used to recreate the certificate. Based on the CMPv2 response, the corresponding key and certificates are overwritten in the Kubernetes secrets. Both the end entity and the CMP identity certificates can be recreated.

To recreate the certificate using CNC console:

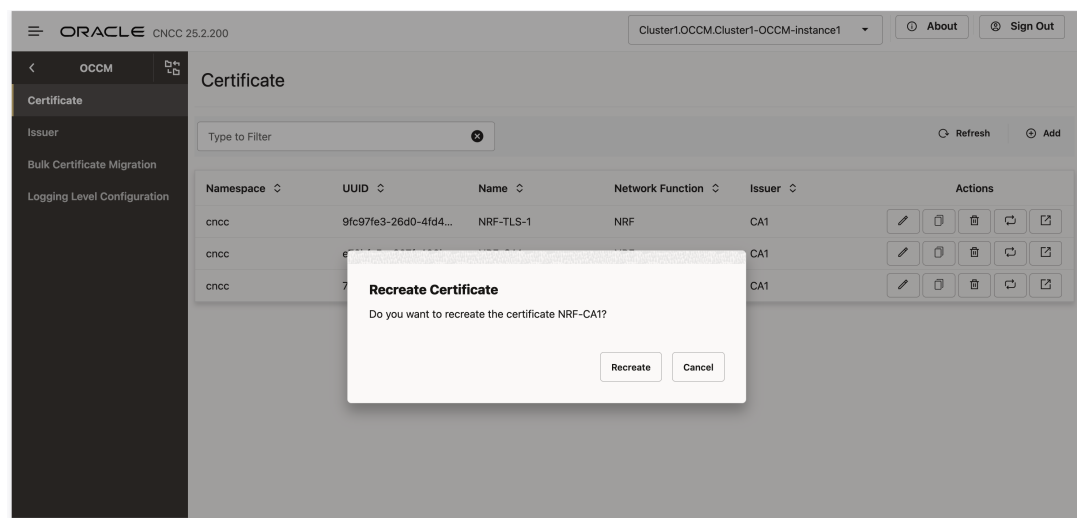
1. Log in to CNC Console using your login credentials and select the OCCM Instance.
2. Click **OCCM** from the left pane and then click **Certificate**. This will list all the certificate configurations.

Figure 3-55 Recreate Certificate



3. Click the **recreate** icon next to the certificate that you want to recreate. The Recreate Certificate dialog box appears.

Figure 3-56 Recreate Certificate Dialog Box



Note

- The Certificate Edit button (pencil icon) was earlier being used for triggering certificate recreate prior to 25.1.200. From 25.1.200, a separate button for recreate has been added.



- If certificate recreation fails, OCCM retries certificate recreation repeatedly until it is successful or the configuration is deleted.
- Certificates that were created successfully and are in status READY, EXPIRED, or FAILED can be recreated.

4. Click **Recreate** to proceed. Click **Cancel** to discard your progress and close the dialog box.
5. Check the Grafana dashboard to view the recreated certificate.

For more information, see [Recreating Certificates- Sample Configuration](#).

3.6.6 Cloning Certificates

Overview

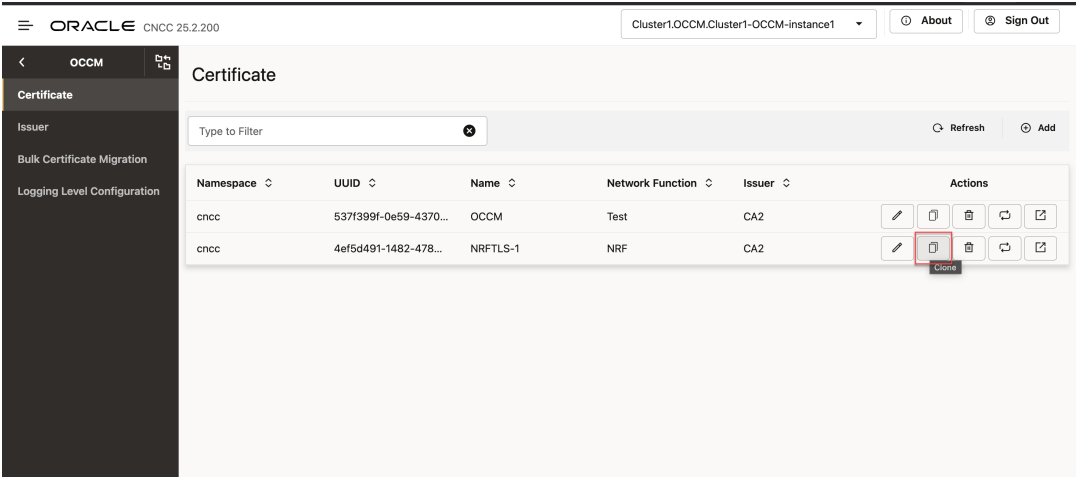
The Certificate Cloning feature streamlines the creation of new digital certificates by allowing users to clone or duplicate an existing certificate's configuration. This significantly reduces manual entry, minimizes errors, and ensures consistency across certificate deployments.

When using the CNC Console user interface, you can select an existing certificate and click the **Clone** button. This opens a screen where most configuration fields are automatically populated based on the original certificate. You can review and edit any details before saving the new certificate.

Perform the following procedure to clone certificates:

1. Log in to CNC Console using your login credentials and select the OCCM instance.
2. Click **OCCM** in the left pane and click **Certificate**.
3. Click the **Clone** icon next to the certificate you want to clone.

Figure 3-57 Clone Certificate



4. The **Clone Certificate** page appears. The fields on the **Clone Certificate** page are already populated but can be edited if required. Edit the required fields, and click **Save**.

Figure 3-58 Clone Certificate Page

ORACLE

CNCC 25.2.200

Cluster1.OCCM.Cluster1-OCCM-instance1

AboutSign Out

<OCCM

Certificate

IssuerBulk Certificate MigrationLogging Level Configuration

Clone Certificate

UUID:

Name:

NRFTLS-1

Cert Type:

OTHER

Network Function:

Network Function
NRF

Purpose:

Purpose
Test

Issuer:

CA2

Creation Mode:

Creation Mode
AUTOMATIC

Overwrite Secret:

Option is used to overwrite the existing X8s secret with new certificate

Renew Before Expiration (Days):

Renew Before Expiration (Days)
14

Namespace:

Namespace
CNCC

Private Key Options

Key Algorithm:

Key Algorithm
RSA

Key Encoding:

Key Encoding
PEM

Key Size:

Key Size
KEYSIZE_2048

Private Key Output

Namespace:

Namespace
CNCC

Secret Name:

Secret Name
nrftls-secret-1

Key:

Key
nrf-private-key.pem

Public Key Certificate Options

Key Usage

Critical:

Value(s):

Value(s)
DIGITAL_SIGNATURE × KEY_ENIPHERMENT ×

Extended Key Usage

Critical:

Value(s):

Value(s)
CLIENT_AUTH × SERVER_AUTH ×

Basic Constraints

Critical:

Value:

Value
END_ENTITY

Subject

Country:

Country

State:

State

Location:

Location

Organization:

Organization

Organization Unit:

Organization Unit

Common Name:

Common Name
OCCM/a

Requested Validity (Days):

Requested Validity (Days)
365

Subject Alternate Names

Critical:

IP Addresses:

IP Addresses

DNS Names:

DNS Names

URI ID API Roots:

URI ID API Roots

URI ID URNs:

URI ID URNs

Certificate Output

Namespace:

Namespace
CNCC

Secret Name:

Secret Name
nrftls-secret-1

Key:

Key
nrf-cert.cer

Certificate Chain Output

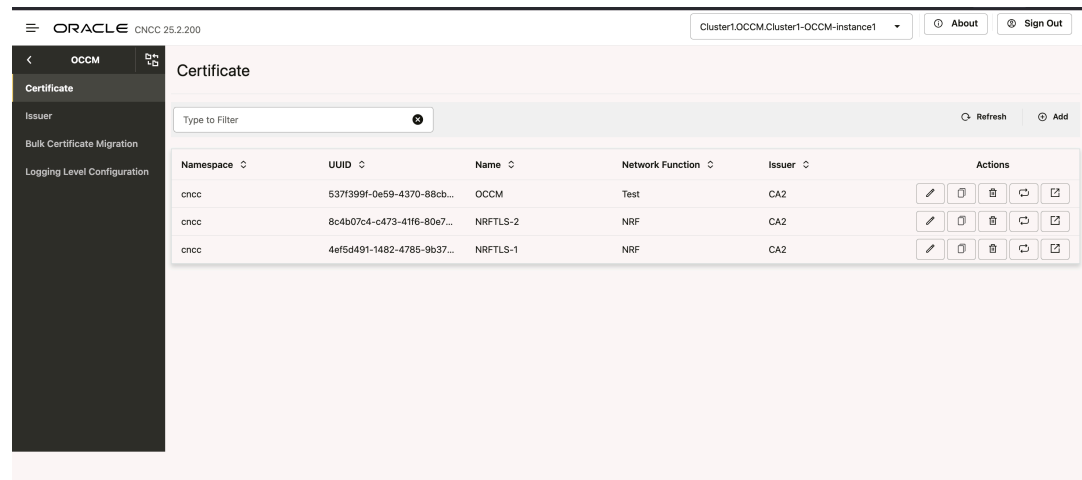
Certificate Management User Guide

Oracle and/or its affiliates.

July 30, 2026Page 43 of 53

5. A new certificate is created. Click the certificate name to verify the certificate configurations.

Figure 3-59 New Certificate



6. Check the Grafana dashboard to verify the status of the new certificate.

Managing Cloning Certificates

Enable

Cloning certificates is a core functionality of OCCM. You do not need to enable or disable this feature.

Configure

There are no Helm Configurations associated with this feature. You can clone certificates using the OCCM UI.

Observe

There are no specific metrics, or KPIs dedicated to certificate cloning. Existing OCCM metrics, and KPIs are used for this functionality. For more information, see [OCCM Metrics](#) and [OCCM KPIs](#).

Maintain

If you encounter alerts at system or application levels, see [OCCM Alerts](#).

In case the alerts still persists, perform the following:

1. Collect the logs: See *Oracle Communications Cloud Native Core, Certificate Management Troubleshooting Guide* for more information on how to collect logs.
2. Raise a service request: See [My Oracle Support](#) for more information on how to raise a service request.

3.6.7 Monitoring Certificate Expiry and Renewing Certificates

OCCM monitors the certificate validity and initiates automatic certificate renewal based on the renew before period configuration. You can update the **Renew Before Expiration (Days)** field on the **Create Certificate** page at the time of certificate creation. This field specifies the number of days before the certificate expiry date when the certificate must be renewed.

Figure 3-60 Renew Before Expiration (Days)

The screenshot shows the 'Create Certificate' interface. On the left is a sidebar with a dark background and white text, containing a back arrow, the 'OCCM' logo, and a 'Create' button. Below these are four menu items: 'Certificate' (highlighted), 'Issuer', 'Bulk Certificate Migration', and 'Logging Level Configuration'. The main area is a light gray form titled 'Create Certificate'. It contains several input fields and a toggle switch. The fields are: 'UUID' (text input), 'Name' (text input), 'Cert Type' (dropdown menu), 'Network Function' (text input), 'Purpose' (text input), 'Issuer' (dropdown menu), 'Creation Mode' (dropdown menu), 'Overwrite Secret' (toggle switch, currently off), 'Renew Before Expiration (Days)' (text input with '14' entered), and 'Namespace' (text input). Below the 'Overwrite Secret' toggle, there is a small note: 'Option is used to overwrite the existing K8s secret with new certificate'.

A CMPv2 Key Update Request (KUR) is sent to the issuing CA for certificate renewal. Once the renewal is successful, OCCM overwrites the configured kubernetes secret holding the old key and certificate with the new one and continues to monitor.

Note

Grafana dashboards can be used to visualize certificate status, such as, expiry time the cert readiness status and the error reason in case the certificate creation fails etc.

3.6.7.1 Renewing CMP Identity (OCCM) Certificate

The Key Update Request (KUR) for CMP identity certificate is signed by the CMP identity key and certificate that is being renewed. The corresponding certificate is included in the extraCerts of the outgoing CMP message.

3.6.7.2 Renewing End Entity (NF) Certificate

The Key Update Request (KUR) for an end entity certificate is signed in the following two ways:

- Using CMP Identity key and certificate.
- Using the old end entity key and certificate that is being renewed.

The corresponding certificate is included in the extraCerts of the outgoing CMP message. To use the CMP Identity key and certificate as the signer, you must set the Key Update Request mode as follows:

- Set the `occmConfig.cmp.config.useOccmCertSignForKur` parameter to true at the time of OCCM deployment, then CMP Identity key and certificate is used to sign the CMP KUR message.
- If the `occmConfig.cmp.config.useOccmCertSignForKur` parameter is set to false, the certificate that is being renewed will be used as the signer.

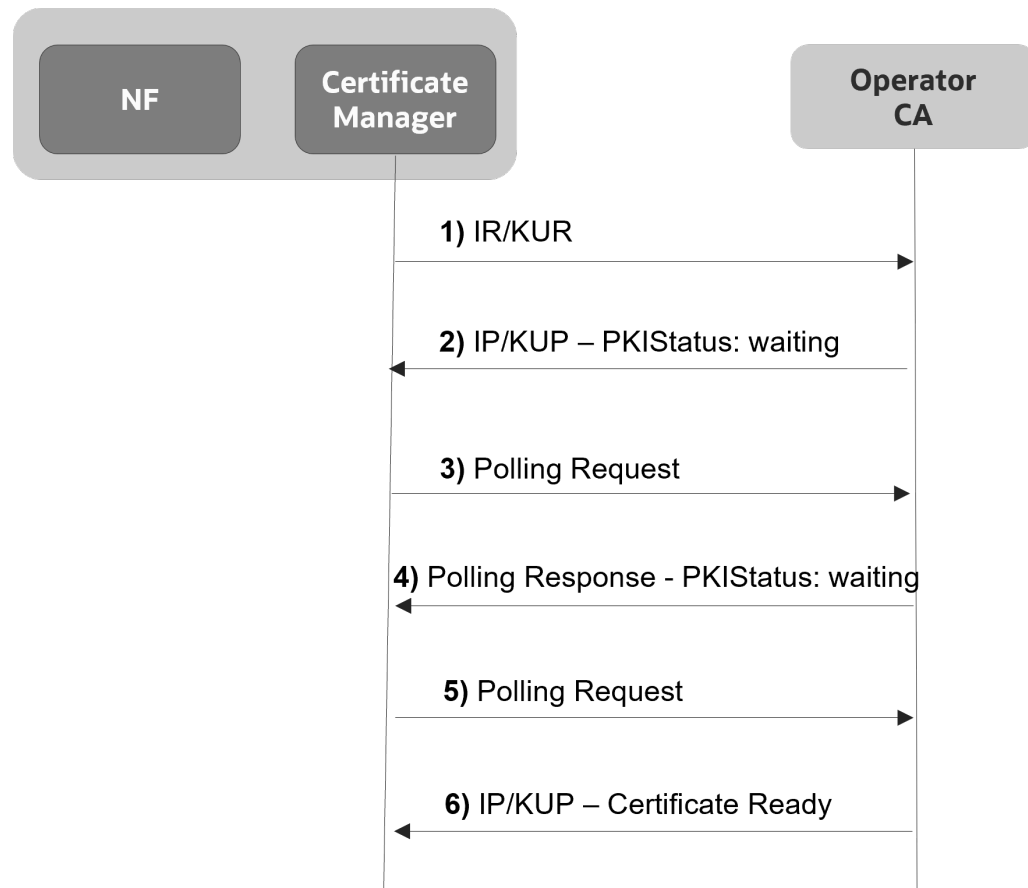
Note

Grafana dashboard can be used to visualize the status of the certificate, such as, the remaining expiry of the certificate, the cert readiness status, and the error reason in case the certificate creation fails etc.

3.6.8 Polling for Certificates

After the IR or KUR, if the certificate is not available yet, the CA responds with PKI status 'Waiting'. The application keeps polling until the CA is ready with the certificate. Openssl implicitly handles polling. No additional configuration is required at the application level in this regard. However, the Total Timeout field can be set in the issuer configuration, which can restrict this polling time. It is the maximum number of seconds a transaction may take, including polling etc. If the time specified by total timeout has elapsed, the polling will stop.

Figure 3-61 Polling for Certificates



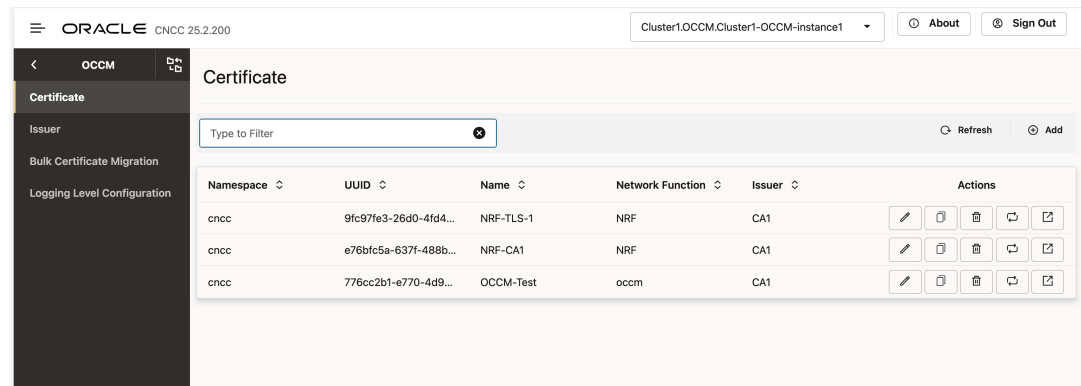
3.6.9 Deleting the Certificate Configuration

To delete the certificate configuration and secret:

1. Log in to CNC Console using your login credentials and select the **OCCM** Instance.

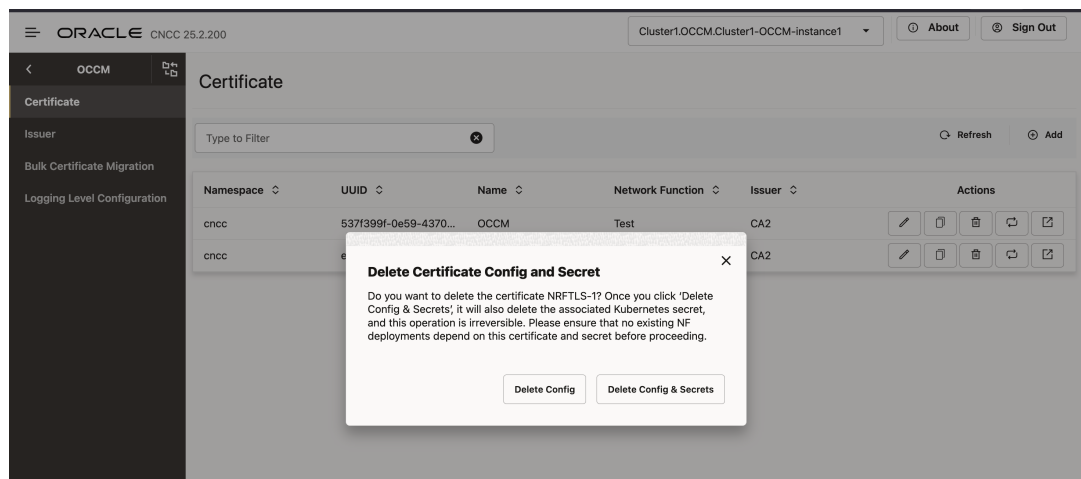
- Click **OCCM** from the left pane and click **Certificate**. All the available certificates will be listed.

Figure 3-62 Certificate

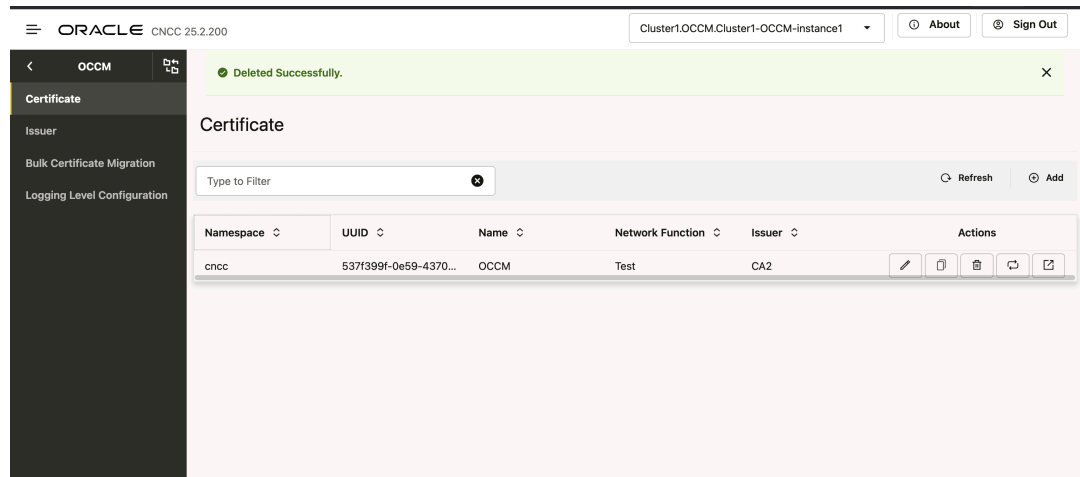


- Click the **Delete** icon and select one of the following from the pop up confirmation dialog:
 - Delete Config** to delete certificate configuration only.
 - Delete certificate config and secret** to delete certificate and secret both.

Figure 3-63 Certificate Delete



- Click **Confirm Delete** on the confirmation prompt to delete the certificate.
- On successful deletion, the *"Deleted successfully"* confirmation message appears on the screen

Figure 3-64 Deleted successfully**Note**

The **Delete Config** button only deletes the configuration even if a corresponding secret exists.

Before deletion, run `kubectl get secret <secret-name> -n <namespace> -o yaml` and save the output in the approved backup location. Confirm no active NF workload still depends on the secret. Success: `kubectl delete secret <secret-name> -n <namespace>` returns `secret "<secret-name>" deleted`, and a follow-up `kubectl get secret` returns `NotFound`. Failure: `Forbidden`, unexpected `NotFound`, or dependent workload errors mean stop and restore from the saved YAML if service impact occurs.

```
kubectl delete secrets <secret name> -n <namespace>
```

For example:

```
kubectl delete secrets nrf-tls-secret -n ns1
```

3.6.10 Bulk Migration of Certificates

The bulk migration of certificates is utilized when the issuer configuration needs to be updated, specifically to modify the issuer endpoint by updating fields, such as, the server URL, recipient DN, and issuer DN. This update is carried out by migrating the certificates in bulk from the current issuer to a newly created issuer with the necessary configuration.

To start the bulk migration, the operator must specify both the source and destination issuers in the request. During the process, all the certificates linked to the source issuer are retrieved and an automatic recreate (CMPv2 Initialization Request) is triggered for each certificate to the destination issuer. Based on the CMPv2 response, the relevant keys and certificates are updated in the existing Kubernetes secrets.

3.6.10.1 Initiating Bulk Certificate Migration

Perform the following steps to initiate the bulk certificate migration:

1. Configure the destination issuer in the issuer configuration. For more information, see [Managing Issuers](#).
2. Create CMP (OCCM) Identity certificate corresponding to the destination issuer. For more information, see [Managing Certificates](#).
3. Initiate the bulk certificate migration as follows:
 - a. Log in to CNC Console using your login credentials and select the OCCM Instance.
 - b. Click **OCCM** from the left pane and then click **Bulk Certificate Migration**. The Bulk Certificate Migration page appears.

Figure 3-65 Bulk Certificate Migration

- c. Click **Add**. The Initiate Bulk Certificate Migration page appears.

Figure 3-66 Initiate Bulk Certificate Migration

- d. Choose the required information from the drop-down list on the **Initiate Bulk Certificate Migration** page:

Table 3-31 Initiate Bulk Certificate Migration

Field Name	Description
Source Issuer	The issuer whose linked certificates is migrated.
Destination Issuer	The issuer to which the certificates are migrated.

The following fields must be kept empty in the request.

Table 3-32 Initiate Bulk Certificate Migration

Field Name	Description
ID	Unique identifier for each bulk certificate migration.

Table 3-32 (Cont.) Initiate Bulk Certificate Migration

Field Name	Description
Bulk Certificate Migration Initiated Certificates	List of certificates that will be migrated from source issuer to destination issuer.
Bulk Certificate Migration Skipped Certificates	List of certificates on which the migration is skipped because of the certificate status is other than ready, expired, or another recreate is already in process.

4. View the configuration to check the certificates on which migration is initiated.
5. Check Grafana dashboard to verify the status of the migration of each certificate. For sample configuration, see [Initiating Bulk Certificate Migration - Sample Configuration](#).

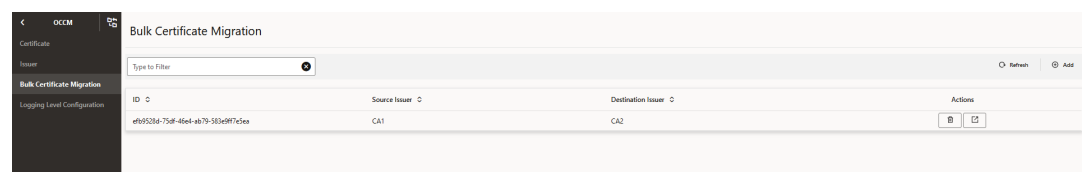
Note

- Bulk certificate migration is supported for only linked End Entity (NF) certificates. It does not apply to CMP (OCCM) identity certificates.
- Certificates that have been created successfully and are in READY, EXPIRED or FAILED status (say in renew retries) are considered for bulk migration.
- Only one bulk certificate migration can be triggered at a time.
- Issuer edit remains the same and allows the edits of all the fields if there are no certificate configuration attached. If the certificate is attached, then you can edit only the HTTP scheme (HTTP to HTTPS and vice-versa). If the CA end point is changed (with the change in parameters, such as server URL, Issuer DN, and recipient DN) then bulk migration is used to recreate all the certificates by linking the certificates to the configured destination issuer.

3.6.10.2 Deleting Bulk Certificate Migration

Perform the following steps to delete the bulk certificate migration:

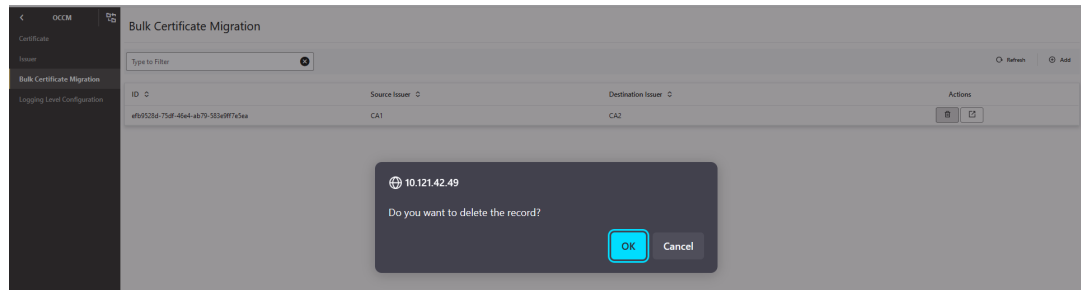
1. Log in to CNC Console using your login credentials and select the OCCM Instance.
2. Click **OCCM** from the left pane and then click **Bulk Certificate Migration**.

Figure 3-67 Bulk Certificate Migration

3. Click **Delete** icon available under the Actions column for the specific certificate. Click **OK** on the confirmation prompt to delete the configuration.

Note

- The bulk certificate migration configuration cannot be deleted if the migration is in progress.
- The **Delete** option only removes the corresponding configuration.
- A maximum of 15 bulk certificate migration configuration are supported.

Figure 3-68 Delete Bulk Certificate Migration

3.7 OCCM Retry on Failure

OCCM supports retry on encountering failures during the certificate creation, certificate re-creation, certificate renewal, and manipulation of Kubernetes secrets.

- The procedure is retried until successful or interrupted by an action executed by the operator.
- Retry is not controlled through any maximum limit.
- The retry interval is an pre-defined value and set to 30s.

Some of the failure scenarios for which retries will be attempted:

- CA is unavailable, not reachable, or busy.
- Any errors returned by CA.

OCCM also provides a retry mechanism for errors encountered during Kubernetes secret update with the generated key and certificate. Based on the error encountered (insufficient permissions, Kubernetes internal errors etc), once the User fixes the issue, the Kubernetes secrets are automatically updated due to the ongoing retries.

Note

In this case, there is no attempt to recreate the key and certificate. The retry is restricted to updating the Kubernetes secrets with the key and certificate that are already generated.

3.8 Network Policies

Network Policies are an application-centric construct that allow you to specify how a pod communicates with various network entities. It creates pod-level rules to control

communication between the cluster pods and services, and to determine which pods and services can access one another inside a cluster.

Previously, the pods under deployment could be contacted by any other pods in the Kubernetes cluster without any restrictions. Now, Network Policy provides namespace-level isolation, which allows secured communications to and from OCCM with rules defined in the respective Network Policies. The network policies enforce access restrictions for all the applicable data flows except communication from Kubernetes node to pod for invoking container probe. For example, OCCM internal microservices can't be contacted directly by any other pods.

Managing Support for Network Policies

Enable

To use this feature, network policies need to be applied to the namespace where OCCM is applied.

Configure

You can configure this feature using Helm. For information about Configuring Network Policy for OCCM deployment, see *Oracle Communications Certificate Management Installation, Upgrade, and Fault Recovery Guide*.

Observe

There are no specific metrics and alerts required for the Support of Network Policy functionality.

3.9 Traffic Segregation

This feature provides end-to-end traffic segregation to OCCM based on traffic types. Within a Kubernetes cluster, traffic segregation can divide applications or workloads into distinct sections such as OAM, SBI, Kubernetes control traffic, etc. The Multus CNI container network interface (CNI) plugin for Kubernetes enables attaching multiple network interfaces to pods to help segregate traffic from OCCM microservice.

This feature addresses the challenge of logically separating IP traffic of different profiles, which are typically handled through a single network (Kubernetes overlay). The new functionality ensures that critical networks are not cross-connected or sharing the same routes, thereby preventing network congestion.

With traffic segregation, operators can segregate traffic to external feeds and applications more effectively. Previously, all external traffic was routed through the same external network, but now, egress traffic from the OCCM pods can be directed through non-default networks to third-party applications. This separation is achieved by leveraging cloud-native infrastructure and the load balancing algorithms in OCCNE.

The feature supports the configuration of separate networks, Network Attachment Definitions (NADs), and the Cloud Native Load Balancer (CNLB). These configurations are crucial for enabling cloud native load balancing, facilitating ingress-egress traffic separation, and optimizing load distribution within OCCM.

Prerequisites

The CNLB feature is only available in OCCM if OCCNE is installed with CNLB and Multus.

Cloud Native Load Balancer (CNLB)

CNE provides Cloud Native Load Balancer (CNLB) for managing the ingress and egress network as an alternate to the existing LBVM, lb-controller, and egress-controller solutions. You can enable or disable this feature only during a fresh CNE installation. When this feature is enabled, CNE automatically uses CNLB to control ingress traffic. To manage the egress traffic, you must preconfigure the egress network details in the `cnlb.ini` file before installing CNE.

For more information about enabling and configuring CNLB, see *Oracle Communications Cloud Native Core, Cloud Native Environment User Guide*, and *Oracle Communications Cloud Native Core, Cloud Native Environment Installation, Upgrade, and Fault Recovery Guide*.

Network Attachment Definitions for CNLB

A Network Attachment Definition (NAD) is a resource used to set up a network attachment, in this case, a secondary network interface to a pod. OCCM supports following types of CNLB NADs:

Egress Only Network Attachment Definitions

Egress Only NADs enable outbound traffic only. An NF pod can initiate traffic and route it through a CNLB application, translating the source IP address to an external egress IP address. An egress NAD contains network information to create interfaces for NF pods and routes to external subnets.

- Requirements:
 - Ingress NADs are already created for the desired internal networks.
 - Destination (egress) subnet addresses are known beforehand and defined under the `cnlb.ini` file's `egress_dest` variable to generate NADs.
 - The use of an Egress NAD on a deployment can be combined with Ingress NADs to route traffic through specific CNLB apps.
- Naming Convention `nf-<service_network_name>-egr`

Traffic Segregation

The traffic segregation feature enables OCCM users to manage egress traffic, that is, all outgoing data and communication from OCCM to CAs. It ensures that the traffic directed towards CAs is segregated and managed to maintain security and improve efficiency.

Note: Incoming traffic like REST API requests are managed separately using CNC Console. CNC Console is responsible for managing and processing these incoming requests, ensuring that they are appropriately routed and secured.

Enable and Configure

This feature is disabled by default. To enable this feature, you must configure the network attachment annotations in the custom values file. For more information, see the "Installing OCCM Package" section in the *Oracle Communications Certificate Management Installation, Upgrade, and Fault Recovery Guide*.

Observe

There are no metrics, KPIs, or alerts required for this feature.

4

Introducing OCCM in an Existing NF Deployment

This section describes the procedure to introduce OCCM in an existing NF deployment where certificates are managed manually. OCCM helps in automating certificate management.

You can move from manual management to automated manages in one of 2 ways:

- Using existing key and certificate.
- Using a new key and certificate.

Moving NFs from Manual Certificate Management to Automated Certificate Management with Existing Key and Certificate

To move NFs from manual certificate management to automated certificate management with existing key and certificate:

1. Configure a key and certificate on OCCM. You must reuse the same Kubernetes secret and the content as used by NF with manually generated key and certificate. The NF configuration must not be updated.
2. OCCM monitors the existing key and certificate in the configured Kubernetes secret and renews it. The metrics attached to the key and certificate are generated.

Note

The existing key and certificate are not validated against the configuration. However, the renewed certificate will be aligned with the configuration.

Moving NFs from Manual Certificate Management to Automated Certificate Management With new Key and Certificate

To move NFs from manual certificate management to automated certificate management with new key and certificate:

1. Configure a key and certificate on OCCM making sure to reuse the same Kubernetes secret as used by NF with manually generated key and certificate. Reusing the Kubernetes secret make sure that the NF configuration is not updated.
2. OCCM creates a new key and certificate in the configured Kubernetes secret and deletes the old key and certificate. The old key and certificate is deleted to generate OCCM metrics attached to the certificate creation.

Procedure

The operator can select the following values for the Creation Mode field:

- Manual (With existing key and certificate)
- Automatic (With new key and certificate)

Manual

- In Manual mode, OCCM is configured to manage the lifecycle of existing the certificates. For example, the certificates that are already being used by NFs can be monitored by OCCM and further renewed by OCCM. In this case, the same Kubernetes secret and the content as used by NF with manually generated key and certificate is reused by OCCM.

Note

The existing key and certificate are not validated against the configuration. Renewed certificate will be aligned with the configuration though.

Automatic

- In Automatic mode, OCCM can create fresh certificates, or overwrite the existing certificate with a new one. For example, if NFs want to create a new key and certificate to overwrite old one through OCCM, and monitor them, then a key and certificate can be created on OCCM using the same Kubernetes secret as used by NF with manually generated key and certificate
- OCCM creates a new key and certificate in the configured Kubernetes secret and deletes the old key and certificate

Table 4-1 Dependency of Creation Mode on Kubernetes Secret

Creation Mode	Description
Manual	Manual mode must be selected to continue using the existing key and certificate which are used by NF in the manual management. Operator needs to configure the same Kubernetes secret as used by the NF configuration.
Automatic	Automatic mode must be selected to generate a new key and certificate. OCCM generates a fresh key pair and certificate and creates a new Kubernetes secret. You must configure the Kubernetes secret details. Operator can also choose to overwrite an existing secret with the generated key pair and certificate by setting the <code>Overwrite Secret</code> flag to true and configuring the existing secret details.

Table 4-2 Behaviour of different Creation Modes

Creation Mode	Preexisting Kubernetes Secret	overwrite Secret Flag	Behaviour
Automatic	No	No Impact	Certificate is created irrespective of the overwrite flag.

Table 4-2 (Cont.) Behaviour of different Creation Modes

Creation Mode	Preexisting Kubernetes Secret	overwrite Secret Flag	Behaviour
Automatic	Yes	True or False	True: The Kubernetes secret is overridden. False: An error is thrown because you must either use a new secret or set the overwrite flag to true. This error is thrown upfront on the user interface or in the response if APIs are used.
Manual	No	NA	An error is thrown because OCCM expects a preconfigured Kubernetes secret. This error is thrown upfront on the user interface or in the response if APIs are used.
Manual	Yes	NA	Certificate configuration is created at OCCM for further certificate renewal and monitoring.

Moving Back to Manual Certificate Management

- If the operator wants to move back to manual certificate monitoring, then they can delete the entry from the OCCM configuration. OCCM doesn't delete the secret when the entry is deleted and the certificate can be monitored manually (if operator used same secret location).
- If user creates a separate secret during certificate management from OCCM, and the operator doesn't want to use the secret further, then operator can delete the entry from OCCM and must also delete the Kubernetes secret.

5

Accessing OCCM Resources Through Curl and Postman

CNC Console provides a secure option for accessing OCCM resources through curl and postman using the CNC Console IAM access token. This section describes how to generate access tokens and access OCCM APIs.

5.1 Generate Access Tokens

CNC Console IAM provides a REST API for generating and refreshing access tokens.

To generate access tokens:

1. Send a POST request to the following URL to get an access token from CNC Console IAM:
`http://${cncc-iam-ingress-extrenal-ip}:${cncc-iam-ingress-service-port}/cncc/auth/realms/${realm}/protocol/openid-connect/token`

For example: `https://{host}:{port}/cncc/auth/realms/cncc/protocol/openid-connect/token`

2. The body of the request must be *x-www-form-urlencoded* encoded as follows:

```
'client_id': 'your_client_id',  
'username': 'your_username',  
'password': 'your_password',  
'grant_type': 'password'
```

For example:

```
'client_id': 'cncc-api-access',  
'username': 'user1',  
'password': '*****',  
'grant_type': 'password'
```

3. Run the following curl command to generate access tokens:

```
curl --location --request POST 'http://{host}:{port}/cncc/auth/realms/cncc/protocol/openid-connect/token' \  
--header 'Content-Type: application/x-www-form-urlencoded' \  
--data-urlencode 'grant_type=password' \  
--data-urlencode 'username=user1' \  
--data-urlencode 'password=*****' \  
--data-urlencode 'client_id=cncc-api-access'
```

4. In response, you will get an **access_token** and a **refresh_token**:

```
{  
  "access_token": "<access-token>",
```

```

    "expires_in": 300,
    "refresh_expires_in": 1800,
    "refresh_token": "<refresh-token>",
    "token_type": "bearer",
    "not-before-policy": 0,
    "session_state": "6c42d978-14ac-4793-a1e3-789cfbdb2b74",
    "scope": "email profile"
  }

```

5.2 Refresh Access Tokens

If the access token has expired, you can refresh it by sending a POST request to the same URL, but containing the refresh token instead of username and password:

Perform the following procedure to refresh the access tokens:

If the `access_token` has expired, it can be refreshed by sending a POST request to the same URL as above; but the POST method must have the refresh token instead of username and password. The format is as follows:

```

'client_id': 'your_client_id',
'refresh_token': refresh_token_from_previous_request,
'grant_type': 'refresh_token'

```

For Example:

```

'client_id': 'cncc-api-access',
'refresh_token': 'eyJhbGciOiJIU...dKnMfb5vKPF-ZIg',
'grant_type': 'refresh_token'

```

In response, you will receive a new **access_token** and **refresh_token**.

5.3 Issuer Configuration API Access

You need the CNC Console IAM access tokens to access the OCCM Issuer APIs through CNC Console.

You must include the following headers when you send an API request:

- **Authorization:** The access token must be used in every request to a NF resource by placing it in the *Authorization* header.
- **oc-cncc-id:** M-CNCC uses the `oc-cncc-id` header to find the agent or manager owning the instance.
- **oc-cncc-instance-id:** A-CNCC Core (or M-CNCC Core) uses the `oc-cncc-instance-id` header to find the NF instance for routing.

Following headers must be passed in the curl or postman request while accessing the OCCM Issuers resource:

HTTP Request:

```

curl --request POST 'http://${occm-external-ip}:${occm-service-port}/occm-
config/v1/issuers/'
--header 'Content-Type: application/json'

```

```

--header 'oc-cncc-id: Cluster1'
--header 'oc-cncc-instance-id: Cluster1-occm-instance1'
--header 'Authorization: Bearer <Token>'
--data-raw '{
  "name": "CA1",
  "server": "http://ca1-openssl-mock.ns1.svc.thrust5:8090",
  "recipientDN": "/CN=x.company.com",
  "issuerDN": "/CN=x.company.com",
  "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.pem"
    ],
    "intCACerts": [
      "intcacert.pem"
    ],
    "serverCert": "servercert.pem"
  },
  "tlsConfig": {
    "enableTLS": false,
    "tlsTrustStoreK8sSecretItem": {
      "namespace": "",
      "name": "",

```

```

        "tlsTrustedCerts": []
    }
}
}'

```

HTTPS Request

```

curl --request POST 'http://${occm-external-ip}:${occm-service-port}/occm-
config/v1/issuers/'
--header 'Content-Type: application/json'
--header 'oc-cncc-id: Cluster1'
--header 'oc-cncc-instance-id: Cluster1-occm-instance1'
--header 'Authorization: Bearer <Token>'
--data-raw '{
    "name": "CA1",
    "server": "https://ca1-openssl-mock.ns1.svc.thrust5:8443",
    "recipientDN": "/CN=x.company.com",
    "issuerDN": "/CN=x.company.com",
    "totalTimeout": "720",
    "messageTimeout": "120",
    "cmpProtectionOcmCert": {
        "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.pem"
        ]
    }
},

```

```

        "intCACerts": [
            "intcacert.pem"
        ],
        "serverCert": "servercert.pem"
    },
    "tlsConfig": {
        "enableTLS": true,
        "tlsTrustStoreK8sSecretItem": {
            "namespace": "ns1",
            "name": "ca1-tls-trust-store-secret",
            "tlsTrustedCerts": ["caroot.cer"]
        }
    }
}
}'

```

5.4 Certificate Configuration API Access

You need the CNC Console IAM access token that you generated to access OCCM Certificates Configuration API:

```

curl --request POST 'http://${occm-external-ip}:${occm-service-port}/occm-
config/v1/certs/'
--header 'Content-Type: application/json'
--header 'oc-cncc-id: Cluster1'
--header 'oc-cncc-instance-id: Cluster1-occm-instance1'
--header 'Authorization: Bearer <Token>'
--data-raw ' {

    "name": "NRF TLS Cert",
    "lcmType": "AUTOMATIC",
    "certType": "OTHER",
    "renewBefore": "14",
    "certPurpose": "NRF SBI",
    "issuer": "CA1",
    "privateKey": {
        "keyAlgo": "RSA",
        "keySize": "KEYSIZE_2048",
        "keyEncoding": "PEM",
        "ecCurve": null,
        "privateKeyK8sSecretOut": {
            "namespace": "ns1",
            "name": "nrf-tls-secret",
            "key": "nrfkey.pem"
        }
    },
    "csr": {
        "extendedKeyUsage": {
            "critical": false,
            "extendedKeyUsageValues": [
                "CLIENT_AUTH",
                "SERVER_AUTH"
            ]
        },
        "keyUsage": {

```

```

        "critical": true,
        "keyUsageValues": [
            "DIGITAL_SIGNATURE"
        ]
    },
    "basicConstraints": {
        "critical": false,
        "basicConstraintsValue": "END_ENTITY"
    },
    "subject": {
        "country": "IN",
        "state": "KA",
        "location": "BLR",
        "organization": "Oracle",
        "organizationUnit": "CGBU",
        "commonName": "a.company.com"
    },
    "days": "365",
    "subjectAltName": {
        "critical": false,
        "ipAddress": [
            "10.10.10.20",
            "10.10.10.21"
        ],
        "dns": [
            "y.company.com",
            "z.company.com"
        ],
        "uriIdApiRoot": null,
        "uriIdUrn": [
            "urn:uuid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6"
        ]
    },
    "certK8sSecretOut": {
        "namespace": "ns1",
        "name": "nrf-tls-secret",
        "key": "nrfcert.pem"
    },
    "certChainK8sSecretOut": {
        "namespace": "ns1",
        "name": "nrf-tls-secret",
        "key": "nrfcertchain.pem"
    },
    "mergeCertAndChain" : false
},
"caBundleK8sSecretIn": {
    "namespace": "ns1",
    "name": "ca-bundle-secret",
    "key": "ca-bundle.pem"
},
"nf": "NRF",
"overrideSecret": false
}'

```

5.5 Bulk Certificate Migration API Access

You need the CNC Console IAM access token that you generated to access bulk certificate migration APIs.

```
curl --request POST 'http://${occm-external-ip}:${occm-service-port}/occm-
config/v1/certs/bulk-migrate'
--header 'Content-Type: application/json'
--header 'oc-cncc-id: Cluster1'
--header 'oc-cncc-instance-id: Cluster1-occm-instance1'
--header 'Authorization: Bearer <Token>'
--data-raw '{
  "sourceIssuerName": "CA1",
  "destinationIssuerName": "CA2"
}'
```

5.6 Logging API Access

You need the CNC Console IAM access token that you generated to access OCCM Logging APIs.

```
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 eyJhbGciOiJSUzI1NiIs...' \
--header 'Content-Type: application/json' \
--data-raw '{
  "appLogLevel": "DEBUG",
  "packageLogLevel": [
    {
      "packageName": "root",
      "logLevelForPackage": "ERROR"
    }
  ]
}'
```

6

OCCM Metrics

This chapter provides information about metrics for OCCM.

Table 6-1 Metric Type

Metric Type	Description
Counter	Represents the total number of occurrences of an event or traffic, such as measuring the total amount of traffic received and transmitted by OCCM, and so on.
Gauge	Represents a single numerical value that changes randomly. This metric type is used to measure various parameters, such as OCCM load values, memory usage, and so on.
Histogram	Represents samples of observations (such as request durations or response sizes) and counts them in configurable buckets. It also provides a sum of all observed values.

Dimension Description

The following table describes different types of metric dimensions:

Table 6-2 OCCM Dimension Description

Dimension	Description	Possible Values
method	Http method	GET, PUT, POST, DELETE
httpVersion	Http protocol version	HTTP/1.1
scheme	Http protocol scheme	HTTP, UNKNOWN
uri	URL of requested API	/occm-config/v1/certs
nfType	API called by NF	eg: SCP, NRF, OCCM, NA
statusCode	Http status code	200, 202
certUuid	Unique ID for the purpose of logging and tracking	eg: 7523a545-089b-49e9-a05c-ae5141db544b
requestType	Type of request	IR, KUR
certName	Name of the certificate	NRFTLS-1, SCPTLS-1
certPurpose	Purpose of the certificate creation	NRF SBI
issuerName	Name of the Issuer	CA
errorReason	Reason of the error	eg:ERR_K8S_SECRET_CREATION_ERROR
operationType	Type of operation	CREATE, RENEW, DELETE, RECREATE
host	Application hosted on cluster	eg: occm.occncc-thrust5-01.svc.thrust5
application	Name of the application	OCCM
caServer	URL of the Certificate Authority (Issuer)	eg: http://ca1-openssl-mock.occncc-thrust5-01.svc.thrust5:8089, https://ca2-openssl-mock.occncc-thrust5-01.svc.thrust5:8443

Table 6-2 (Cont.) OCCM Dimension Description

Dimension	Description	Possible Values
status	To know the status of openssl CMP cmd	SUCCESS, FAILED
belongs	To determine the secret belongs to which entity	certificate-other, certificate-occm, issuer
type	To determine the type of the secret	input-secret, output-secret
secret	Name of the secret	nrf-tls-secret
uuid	Unique id of the entity	eg: 7523a545-089b-49e9-a05c-ae5141db544b
event	Name of the event	eg: modify, delete
secretNamespace	Name of the secret's namespace	eg: occncc-thrust5-01
cerType	Cert type of the certificate	eg: OCCM, END_ENTITY, ISSUER_HEALTHCHECK

Note

After OCCM application starts, the default metrics are visible for the following metrics without pegging any metric to list down the available metrics. The default metrics are visible all the time and the values for the labels will be empty. It is possible that in prometheus the labels of the metrics are not visible and only Kubernetes default labels are visible.

6.1 occm_config_http_requests_total

Table 6-3 occm_config_http_requests_total

Field	Details
Description	The total number of requests to the OCCM configuration service.
Type	Counter
Dimensions	• host • application • httpVersion • scheme • method • nfType • uri
Example	<pre>occm_config_http_requests_total{ application="occm", container="occm", endpoint="cnc-metrics", helm_sh_chart="occm-26.1.200", host="occm.occncc-thrust5-01.svc.thrust5", httpVersion="HTTP/1.1", instance="10.233.121.228:9000", job="occne-infra/occne-nf-cnc-podmonitor", method="POST", namespace="occncc-thrust5-01", nfType="NRF", pod="occm-occm-67764765f8-7rpm8", pod_template_hash="67764765f8", scheme="http", uri="/occm-config/v1/certs" }</pre>

6.2 occm_config_http_response_total

Table 6-4 occm_config_http_response_total

Field	Details
Description	The total number of responses from the OCCM configuration service
Type	Counter
Dimensions	• host • httpVersion • scheme • method • nfType • statusCode • uri
Example	<pre>occm_config_http_responses_total{ application="occm", container="occm", endpoint="cnc-metrics", helm_sh_chart="occm-26.1.200", host="occm.occncc-thrust5-01.svc.thrust5", httpVersion="HTTP/1.1", instance="10.233.121.228:9000", job="occne-infra/occne-nf-cnc-podmonitor", method="POST", namespace="occncc-thrust5-01", nfType="NRF", pod="occm-occm-67764765f8-7rpm8", pod_template_hash="67764765f8", scheme="http", statusCode="202", uri="/occm-config/v1/certs"}</pre>

6.3 occm_cmp_requests_total

Table 6-5 occm_cmp_requests_total

Field	Details
Description	The number of CMP requests to the CA.
Type	Counter
Dimensions	• certUid • certName • nfType • requestType • issuerName • caServer
Example	<pre>occm_cmp_requests_total{ application="occm", caServer="http://ca90-openssl-mock.occncc-thrust5-01.svc.thrust5:8083", certName="NRFTLS-47", certUid="c0578b02-caab-454a-bd97-422b0e1c575b", container="occm", endpoint="cnc-metrics", helm_sh_chart="occm-26.1.200", instance="10.233.121.228:9000", issuerName="CA90", job="occne-infra/occne-nf-cnc-podmonitor", namespace="occncc-thrust5-01", nfType="NRF", pod="occm-occm-67764765f8-7rpm8", pod_template_hash="67764765f8", requestType="ir"}</pre> <pre>occm_cmp_requests_total{ application="occm", caServer="http://ca90-openssl-mock.occncc-thrust5-01.svc.thrust5:8083", certName="NRFTLS-47", certUid="c0578b02-caab-454a-bd97-422b0e1c575b", container="occm", endpoint="cnc-metrics", helm_sh_chart="occm-26.1.200", instance="10.233.121.228:9000", issuerName="CA90", job="occne-infra/occne-nf-cnc-podmonitor", namespace="occncc-thrust5-01", nfType="NRF", pod="occm-occm-67764765f8-7rpm8", pod_template_hash="67764765f8", requestType="kur"}</pre>

6.4 occm_cmp_responses_total

Table 6-6 occm_cmp_responses_total

Field	Details
Description	The number of CMP responses from the CA.
Type	Counter
Service Operation	
Dimensions	- certUid - certName - nfType - requestType - status - statusCode - issuerName - caServer
Example	<pre>occm_cmp_responses_total{ application="occm", caServer="http://ca90-openssl-mock.occncc-thrust5-01.svc.thrust5:8083", certName="NRFTLS-47", certUid="c0578b02-caab-454a-bd97-422b0e1c575b", container="occm", endpoint="cnc-metrics", helm_sh_chart="occm-26.1.200", instance="10.233.121.228:9000", issuerName="CA90", job="occne-infra/occne-nf-cnc-podmonitor", namespace="occncc-thrust5-01", nfType="NRF", pod="occm-occm-67764765f8-7rpm8", pod_template_hash="67764765f8", requestType="ir", status="SUCCESS", statusCode="OK"}</pre> <pre>occm_cmp_responses_total{ application="occm", caServer="http://ca90-openssl-mock.occncc-thrust5-01.svc.thrust5:8083", certName="NRFTLS-47", certUid="c0578b02-caab-454a-bd97-422b0e1c575b", container="occm", endpoint="cnc-metrics", helm_sh_chart="occm-26.1.200", instance="10.233.121.228:9000", issuerName="CA90", job="occne-infra/occne-nf-cnc-podmonitor", namespace="occncc-thrust5-01", nfType="NRF", pod="occm-occm-67764765f8-7rpm8", pod_template_hash="67764765f8", requestType="kur", status="SUCCESS", statusCode="OK"}</pre>

6.5 occm_cmp_identity_cert_expiration_seconds

Table 6-7 occm_cmp_identity_cert_expiration_seconds

Field	Details
Description	It is the CMP identity (OCCM) Certificate expiry gauge metrics. It indicates the CMP identity (OCCM) certificate expiry timestamp.
Type	Gauge
Dimensions	- certUid - certName - nfType - issuerName - certPurpose
Example	<pre>occm_cmp_identity_cert_expiration_seconds{ application="occm", certName="OCCM-HTTPS-1", certPurpose="HTTPS-RA", certUid="164dea8d-a54e-4556-81d0-f0a923030416", container="occm", endpoint="cnc-metrics", helm_sh_chart="occm-26.1.200-COCCM-2520-metrics-improve-241128052710-9f49f6ef", instance="10.233.80.166:9000", issuerName="CA-1", job="occne-infra/occne-nf-cnc-podmonitor", namespace="occm-ns", nfType="OCCM", pod="occm-occm-59898cf755-vr2hq", pod_template_hash="59898cf755"} 1734674763</pre>

6.6 occm_end_entity_cert_expiration_seconds

Table 6-8 occm_end_entity_cert_expiration_seconds

Field	Details
Description	It is the End Entity (NF) Certificate expiry gauge metrics. It indicates the End Entity (NF) Certificate expiry timestamp.
Type	Gauge
Dimensions	- certUuid - certName - nfType - issuerName - certPurpose
Example	<pre>occm_end_entity_cert_expiration_seconds{ app_kubernetes_io_vendor="Oracle", app_kubernetes_io_version="25.1.0.0.0", application="occm", certName="nrfcert- nov2711-1", certPurpose="NRF SBI", certUuid="591e3743-70d0-4425-9300-10ebb989d02c", container="occm", endpoint="cnc- metrics", helm_sh_chart="occm-26.1.200-COCCM-2520-metrics- improve-241128052710-9f49f6ef", instance="10.233.80.166:9000", issuerName="CA-1", job="occne-infra/occne-nf-cnc-podmonitor", namespace="occm-ns", nfType="NRF", pod="occm-occm-59898cf755-vr2hq", pod_template_hash="59898cf755"} 1733306455</pre>

6.7 occm_cmp_identity_cert_status

Table 6-9 occm_cmp_identity_cert_status

Field	Details
Description	It is the CMP Identity (OCCM) certificate status gauge metric. The Gauge values indicate the CMP Identity (OCCM) Certificate status. For example, CREATING(1), READY(2), FAILED(3), DELETED(6), EXPIRED(7), and WAITING(8).
Type	Gauge
Dimensions	- certUuid - nfType - certName - certPurpose - issuerName
Example	<pre>occm_cmp_identity_cert_status{ application="occm", certName="OCCM-HTTPS-1", certPurpose="HTTPS-RA", certUuid="164dea8d-a54e-4556-81d0-f0a923030416", container="occm", endpoint="cnc-metrics",helm_sh_chart="occm-26.1.200-COCCM-2520- metrics-improve-241128052710-9f49f6ef", instance="10.233.80.166:9000", issuerName="CA-1", job="occne-infra/occne-nf-cnc-podmonitor", namespace="occm-ns", nfType="OCCM", pod="occm-occm-59898cf755-vr2hq", pod_template_hash="59898cf755"} 2</pre>

6.8 occm_end_entity_cert_status

Table 6-10 occm_end_entity_cert_status

Field	Details
Description	It is the End Entity (NF) Cert status gauge metric. The Gauge values indicate the End Entity (NF) Certificate status. For example, CREATING(1), READY(2), FAILED(3), DELETED(6), EXPIRED(7), and WAITING(8).
Type	Gauge
Dimensions	- certUuid - nfType - certName - certPurpose - issuerName
Example	occm_end_entity_cert_status{ application="occm", certName="nrfcert-nov2711-1", certPurpose="NRF SBI", certUuid="591e3743-70d0-4425-9300-10ebb989d02c", container="occm", endpoint="cnc-metrics", helm_sh_chart="occm-26.1.200-COCCM-2520-metrics-improve-241128052710-9f49f6ef", instance="10.233.80.166:9000", issuerName="CA-1", job="occm-ne-infra/occm-ne-nf-cnc-podmonitor", namespace="occm-ns", nfType="NRF", pod="occm-occm-59898cf755-vr2hq", pod_template_hash="59898cf755"} 2

6.9 occm_cmp_cli_durations_seconds

Table 6-11 occm_cmp_cli_durations_seconds

Field	Details
Description	OCCM CMP CLI duration histogram metrics. It indicates the time taken for CMP CLI between request and response from CA.
Type	Histogram
Dimensions	- certUuid - nfType - certName - requestType - caServer
Example	occm_cmp_cli_durations_seconds{ application="occm", caServer="http://ejbca-ejbca-community-helm.occm-ns.svc.thrust5:30080/ejbca/publicweb/cmp/occmaliasra", certName="OCCM-test1", certUuid="c7343daa-9eb0-4ba5-9e02-67a945f2bd9b", container="occm", endpoint="cnc-metrics", helm_sh_chart="occm-26.1.200", namespace="occm-ns", nfType="OCCM", pod="occm-occm-ccfcbbbf-bnl22k", quantile="0.5", requestType="ir"}

6.10 occm_cert_request_status_total

Table 6-12 occm_cert_request_status

Field	Details
Description	OCCM certificate request status counter metric. This metrics includes both CMP Identity (OCCM) and End Entity (NF). It indicates CMP Identity (OCCM) and End Entity (NF) certificate status, error reason, operation type whether Create, Renew, or Recreate etc.
Type	Counter
Dimensions	- certName - certUuid - errorReason - issuerName - metadata - nfType - operationType
Example	occm_cert_request_status_total{ application="occm", certName="NRFTLS-47", certUuid="c0578b02-caab-454a-bd97-422b0e1c575b", container="occm", endpoint="cnc-metrics", errorReason="OK", helm_sh_chart="occm-26.1.200", instance="10.233.121.228:9000", issuerName="CA90", job="occne-infra/occne-nf-cnc-podmonitor", metadata="NA", namespace="occncc-thrust5-01", nfType="NRF", operationType="RENEW", pod="occm-occm-67764765f8-7rpm8", pod_template_hash="67764765f8"} For bulk certificate migration, operationType will be RECREATE and the metadata will have the bulk migration UUID. The errorReason is OK_BULK_CERT_MIGRATION or ERR_BULK_CERT_MIGRATION to depict success and failure cases respectively. occm_cert_request_status_total{ application="occm", certName="NRFTLS-48", certUuid="c1278b02-caab-454a-bd97-422b0e1c805b", container="occm", endpoint="cnc-metrics", errorReason="OK_BULK_CERT_MIGRATION ", helm_sh_chart="occm-26.1.200", instance="10.233.121.228:9000", issuerName="CA90", job="occne-infra/occne-nf-cnc-podmonitor", metadata="7787cac2-5250-4aee-9f7b-fe3e469d5f0e", namespace="occncc-thrust5-01", nfType="NRF", operationType="RECREATE", pod="occm-occm-67764765f8-7rpm8", pod_template_hash="67764765f8"}

6.11 occm_secret_event_total

Table 6-13 occm_secret_event_total

Field	Details
Description	Kubernetes secret event count. It will indicate the number of operations that have been performed to delete or modify a secret linked to OCCM.
Type	Counter

Table 6-13 (Cont.) occm_secret_event_total

Field	Details
Dimensions	- name - uuid - type - belongs - secret - secretNamespace - event
Example	<pre>occm_secret_event_total{app_kubernetes_io_application="occm", app_kubernetes_io_component="occm", app_kubernetes_io_engVersion="26.1.200- COCCM-1332-240722084346-939119a5", app_kubernetes_io_instance="occm", app_kubernetes_io_managed_by="Helm", app_kubernetes_io_microservice="occm", app_kubernetes_io_mktgVersion="26.1.200.0.0", app_kubernetes_io_name="occm", app_kubernetes_io_part_of="occm", app_kubernetes_io_vendor="Oracle", app_kubernetes_io_version="26.1.200.0.0", application="occm", belongs="certificate-other", container="occm", endpoint="cnc-metrics", event="deleted", exported_namespace="occm- ns", helm_sh_chart="occm-26.1.200-COCCM-1332-240722084346-939119a5", instance="10.233.80.225:8989", job="occne-infra/occne-nf-cnc-podmonitor", name="Nrf- tls8", namespace="occm-ns", pod="occm-occm-7b6fd5dcf7-7n4ld", pod_template_hash="7b6fd5dcf7", secret="nrf-tls-secret8", type="output-secret", uuid="2111e512-10d7-4ffd-a0db-a995d606bc60"}</pre>

6.12 occm_cert_start_date_seconds

Table 6-14 occm_cert_start_date_seconds

Field	Details
Description	The metric will indicate the cert creation/start time of the healthcheck certificate when cert is successfully created (IR/ KUR)
Type	gauge
Dimensions	- certName - certUuid - certType - issuerName - metadata - nfType
Example	<pre>occm_cert_start_date_seconds{application="occm", certName="Healthcheck-Cert-CA1", certPurpose="CA- Healthcheck", certType="ISSUER_HEALTHCHECK", certUuid=Uuid, instance="localhost:8989", issuerName="CA1", job="occm"}</pre>

6.13 occm_cert_renew_time_seconds

Table 6-15 occm_cert_renew_time_seconds

Field	Details
Description	The metric will indicate the next renewal time of certificate to check if it is created or failed. The scope of this metric is limited to healthcheck cert.
Type	gauge
Dimensions	• certName • certUuid • certType • issuerName • metadata • nfType
Example	occm_cert_renew_time_seconds{application="occm", certName="Healthcheck-Cert-CA1", certPurpose="CA-Healthcheck", certType="ISSUER_HEALTHCHECK", certUuid=Uuid, instance="localhost:8989", issuerName="CA1", job="occm"}

6.14 occm_cert_first_fail_time_seconds

Table 6-16 occm_cert_first_fail_time_seconds

Field	Details
Description	The metric will indicate the first fail when the certificate was created successfully and failed later in renew or in recreation. The scope for this metric is limited to the healthcheck feature.  Note When OCCM is restarted or upgraded, this metric will reset.
Type	gauge
Dimensions	• certName • certUuid • certType • issuerName • metadata • nfType
Example	occm_cert_first_fail_time_seconds{application="occm", certName="Healthcheck-Cert-CA1", certPurpose="CA-Healthcheck", certType="ISSUER_HEALTHCHECK", certUuid=Uuid, instance="localhost:8989", issuerName="CA1", job="occm"}

7

OCCM Alerts

This section describes the alerts available for OCCM.

Note

Alert file is packaged with OCCM CSAR package.

- Review the `occm_alerting_rules_promha_<version>.yaml` file and edit the value of the parameters in the `occm_alerting_rules_promha_<version>.yaml` file (if needed to be changed from default values) before configuring the alerts. See above table for details.
- `kubernetes_namespace` is configured as `kubernetes` namespace in which OCCM is deployed. Default value is `occm`. Please update the `occm_alerting_rules_promha_<version>.yaml` file to reflect the correct OCCM `kubernetes` namespace.

Table 7-1 Alerts Levels or Severity Types

Alerts Levels / Severity Types	Definition
Critical	Indicates a severe issue that poses a significant risk to safety, security, or operational integrity. It requires immediate response to address the situation and prevent serious consequences. Raised for conditions may affect the service of OCCM.
Major	Indicates a more significant issue that has an impact on operations or poses a moderate risk. It requires prompt attention and action to mitigate potential escalation. Raised for conditions may affect the service of OCCM.
Minor	Indicates a situation that is low in severity and does not pose an immediate risk to safety, security, or operations. It requires attention but does not demand urgent action. Raised for conditions may affect the service of OCCM.
Info or Warn (Informational)	Provides general information or updates that are not related to immediate risks or actions. These alerts are for awareness and do not typically require any specific response. WARN and INFO alerts may not impact the service of OCCM.

7.1 OccmCmplIdentityCertExpirationMinor

Table 7-2 OccmCmplIdentityCertExpirationMinor

Field	Details
Description	CMP Identity (OCCM) Certificate Expiry Alert. The certificate <code>{{labels.certName}}</code> used by <code>{{labels.nfType}}</code> for <code>{{labels.certPurpose}}</code> will expire within 90 days.

Table 7-2 (Cont.) OccmCmpIdentityCertExpirationMinor

Field	Details
Summary	namespace: {{labels.namespace}}, podname: {{labels.pod}}, timestamp: {{ with query "time()" }}{{ . first value humanizeTimestamp }}{{ end }}: The certificate {{labels.certName}} used by {{labels.nfType}} for {{labels.certPurpose}} will expire soon within 90 days'
Severity	Minor
Condition	The CMP Identity (OCCM) certificate will expire within 90 days.
OID	1.3.6.1.4.1.323.5.3.54.1.2.7001
Metric Used	occm_cmp_identity_cert_expiration_seconds
Recommended Actions	Cause: When Certificate is going to expire within 90 days. Diagnostic Information: 1. Check certificate configuration for renew before days. 2. Refer to the application logs on Kibana and filter based on occm service name. Check for ERROR WARNING logs related to thread exceptions. Recovery:The alert is cleared when the certificate is renewed. If alert is not cleared: Check for remaining validity of the cert. If this is unexpected, contact My Oracle Support. Ensure to capture the occm pod logs and relevant metrics to help Support analyze the issue. Note The threshold is configurable in the <code>occm_alertingrules_<version>.yaml</code> file.

7.2 OccmCmpIdentityCertExpirationMajor

Table 7-3 OccmCmpIdentityCertExpirationMajor

Field	Details
Description	CMP Identity (OCCM) Certificate Expiry Alert. The certificate {{labels.certName}} used by {{labels.nfType}} for {{labels.certPurpose}} will expire within 30 days.
Summary	namespace: {{labels.namespace}}, podname: {{labels.pod}}, timestamp: {{ with query "time()" }}{{ . first value humanizeTimestamp }}{{ end }}: The certificate {{labels.certName}} used by {{labels.nfType}} for {{labels.certPurpose}} will expire soon within 30 days'
Severity	Major
Condition	The CMP Identity (OCCM) certificate will expire within 30 days.
OID	1.3.6.1.4.1.323.5.3.54.1.2.7001
Metric Used	occm_cmp_identity_cert_expiration_seconds

Table 7-3 (Cont.) OccmCmplIdentityCertExpirationMajor

Field	Details
Recommended Actions	Cause: When Certificate is going to expire within 30 days. Diagnostic Information: Check certificate configuration for renew before days. Recovery:The alert is cleared when the certificate is renewed or when the Certificate expiry days crosses the Critical threshold. If alert is not cleared: Check for remaining validity of the cert. If this is unexpected, contact My Oracle Support. Ensure to capture the occm pod logs and relevant metrics to help Support analyze the issue. Note The threshold is configurable in the <code>occm_alertingrules_<version>.yaml</code> file.

7.3 OccmCmplIdentityCertExpirationCritical

Table 7-4 OccmCmplIdentityCertExpirationCritical

Field	Details
Description	CMP Identity (OCCM) Certificate Expiry Alert. The certificate <code>{{ \$labels.certName }}</code> used by <code>{{ \$labels.nfType }}</code> for <code>{{ \$labels.certPurpose }}</code> will expire within one week.
Summary	namespace: <code>{{ \$labels.namespace }}</code> , podname: <code>{{ \$labels.pod }}</code> , timestamp: <code>{{ with query "time()" }}{ . first value humanizeTimestamp }}{ end }}</code> : The certificate <code>{{ \$labels.certName }}</code> used by <code>{{ \$labels.nfType }}</code> for <code>{{ \$labels.certPurpose }}</code> will expire soon within 1 week'
Severity	Critical
Condition	The CMP Identity (OCCM) certificate will expire within one week.
OID	1.3.6.1.4.1.323.5.3.54.1.2.7001
Metric Used	occm_cmp_identity_cert_expiration_seconds

Table 7-4 (Cont.) OccmCmplIdentityCertExpirationCritical

Field	Details
Recommended Actions	Cause: The Certificate is going to expire within 1 week. Diagnostic Information: Check certificate configuration for renew before days. Recovery:The alert is cleared when the certificate is renewed or when the Certificate expiry days crosses the Critical threshold. If alert is not cleared: Check for remaining validity of the cert. If this is unexpected, contact My Oracle Support. Ensure to capture the occm pod logs and relevant metrics to help Support analyze the issue. Note The threshold is configurable in the <code>occm_alertingrules_<version>.yaml</code> file.

7.4 OccmCmplIdentityCertExpired

Table 7-5 OccmCmplIdentityCertExpired

Field	Details
Description	Alert is raised when the certificate expires and then recreation will be triggered. If the certificate recreation is successful then alert will be cleared automatically or the operator has to clear the alert manually. The certificate <code>{{labels.certName}}</code> used by <code>{{labels.nfType}}</code> for <code>{{labels.certPurpose}}</code> is expired.
Summary	'namespace: <code>{{labels.namespace}}</code> , podname: <code>{{labels.pod}}</code> , timestamp: <code>{{ with query "time()" }}{{ . first value humanizeTimestamp }}{{ end }}</code> : The certificate <code>{{labels.certName}}</code> used by <code>{{labels.nfType}}</code> for <code>{{labels.certPurpose}}</code> is expired'
Severity	Critical
Condition	The CMP Identity (OCCM) certificate has expired.
OID	1.3.6.1.4.1.323.5.3.54.1.2.7002
Metric Used	occm_cmp_identity_cert_expiration_seconds

Table 7-5 (Cont.) OccmCmplIdentityCertExpired

Field	Details
Recommended Actions	Cause: The Certificate is going to expire. Diagnostic Information: 1. Check certificate configuration for renew before days. 2. Check logs to identify the root cause, one of the possible cause is CA connection failure. in this case operator needs to manually configure the CMP Identity certificate. 3. Manual recreate can be triggered once CA connection is resume. Recovery: The alert is cleared when the certificate is recreated. 1. Check logs to identify the root cause, one of the possible cause is CA connection failure. in this case operator needs to manually configure the CMP Identity certificate. 2. Get the k8s secret name corresponding to OCCM key/certificate location from the mapped issuer. This information will be present under CMP client authentication options for Other Cert section of the issuer. 3. Create CMP Identity (OCCM) Certificate manually and Update the secret. 4. Manual recreate can be triggered once CA connection is resume. 5. For more information, refer <i>Oracle Communications, Cloud Native Core, Certificate Management Troubleshooting Guide</i> for troubleshooting scenarios. 6. If this is unexpected, contact My Oracle Support. Ensure to capture the occm pod logs and relevant metrics to help Support analyze the issue.

7.5 OccmEndEntityCertExpirationMinor

Table 7-6 OccmEndEntityCertExpirationMinor

Field	Details
Description	End Entity (NF) Certificate Expiry Alert. The certificate {{ \$labels.certName }} used by {{ \$labels.nfType }} for {{ \$labels.certPurpose }} will expire within 90 days.
Summary	namespace: {{ \$labels.namespace }}, podname: {{ \$labels.pod }}, timestamp: {{ with query "time()" }}{{ . first value humanizeTimestamp }}{{ end }}: The certificate {{ \$labels.certName }} used by {{ \$labels.nfType }} for {{ \$labels.certPurpose }} will expire soon within 90 days'
Severity	Minor
Condition	The End Entity (NF) certificate will expire within 90 days.
OID	1.3.6.1.4.1.323.5.3.54.1.2.7003
Metric Used	occm_end_entity_cert_expiration_seconds

Table 7-6 (Cont.) OccmEndEntityCertExpirationMinor

Field	Details
Recommended Actions	Cause: The Certificate is going to expire within 90 days. Diagnostic Information: 1. Check certificate configuration for renew before days. Recovery: The alert is cleared when the certificate is renewed so that the Certificate expiry days falls below the Minor threshold or when the Certificate expiry days crosses the Major threshold • If this is unexpected, contact My Oracle Support. Ensure to capture the occm pod logs and relevant metrics to help Support analyze the issue. Note The threshold is configurable in the <code>occm_alertingrules_<version>.yaml</code> file.

7.6 OccmEndEntityCertExpirationMajor

Table 7-7 OccmEndEntityCertExpirationMajor

Field	Details
Description	End Entity (NF) Certificate Expiry Alert. The certificate {{labels.certName}} used by {{labels.nfType}} for {{labels.certPurpose}} will expire within 30 days.
Summary	'namespace: {{labels.namespace}}, podname: {{labels.pod}}, timestamp: {{ with query "time()" }}{{ . first value humanizeTimestamp }}{{ end }}: The certificate {{labels.certName}} used by {{labels.nfType}} for {{labels.certPurpose}} will expire soon within 30 days.
Severity	Major
Condition	End Entity (NF) certificate will expire soon within 30 days.
OID	1.3.6.1.4.1.323.5.3.54.1.2.7003
Metric Used	occm_end_entity_cert_expiration_seconds

Table 7-7 (Cont.) OccmEndEntityCertExpirationMajor

Field	Details
Recommended Actions	Cause: When Certificate is going to expire within 30 days. Diagnostic Information: 1. Check certificate configuration for renew before days. 2. Refer to the application logs on Kibana and filter based on occm service name. Check for ERROR WARNING logs related to thread exceptions. Recovery: The alert is cleared when the certificate is renewed or when the Certificate expiry days crosses the Critical threshold. • If this is unexpected, contact My Oracle Support. Ensure to capture the occm pod logs and relevant metrics to help Support analyze the issue. Note The threshold is configurable in the <code>occm_alertingrules_<version>.yaml</code> file.

7.7 OccmEndEntityCertExpirationCritical

Table 7-8 OccmEndEntityCertExpirationCritical

Field	Details
Description	End Entity (NF) Certificate Expiry Alert. The certificate <code>{{ \$labels.certName }}</code> used by <code>{{ \$labels.nfType }}</code> for <code>{{ \$labels.certPurpose }}</code> will expire within one week.
Summary	'namespace: <code>{{ \$labels.namespace }}</code> , podname: <code>{{ \$labels.pod }}</code> , timestamp: <code>{{ with query "time()" }}{{ . first value humanizeTimestamp }}{{ end }}</code> : The certificate <code>{{ \$labels.certName }}</code> used by <code>{{ \$labels.nfType }}</code> for <code>{{ \$labels.certPurpose }}</code> will expire soon within 1 week'
Severity	Critical
Condition	End Entity (NF) certificate will expire soon within one week.
OID	1.3.6.1.4.1.323.5.3.54.1.2.7003
Metric Used	occm_end_entity_cert_expiration_seconds

Table 7-8 (Cont.) OccmEndEntityCertExpirationCritical

Field	Details
Recommended Actions	Cause: The Certificate is going to expire within 1 week. Diagnostic Information: - Check certificate configuration for renew before days. - Refer to the application logs on Kibana and filter based on occm service name. Check for ERROR WARNING logs related to thread exceptions. Recovery: The alert is cleared when the certificate is renewed or when the Certificate expiry days crosses the Critical threshold. If alert is not cleared: - Check for remaining validity of the cert. - If this is unexpected, contact My Oracle Support. Ensure to capture the occm pod logs and relevant metrics to help Support analyze the issue. Note The threshold is configurable in the <code>occm_alertingrules_<version>.yaml</code> file.

7.8 OccmEndEntityCertExpired

Table 7-9 OccmEndEntityCertExpired

Field	Details
Description	Alert is raised when the certificate expires and then recreation will be triggered. If the certificate recreation is successful then alert will be cleared automatically or the operator has to clear the alert manually. The certificate <code>{{labels.certName}}</code> used by <code>{{labels.nfType}}</code> for <code>{{labels.certPurpose}}</code> is expired'
Summary	'namespace: <code>{{labels.namespace}}</code> , podname: <code>{{labels.pod}}</code> , timestamp: <code>{{ with query "time()" }}{ . first value humanizeTimestamp }}{ end }}</code> : The certificate <code>{{labels.certName}}</code> used by <code>{{labels.nfType}}</code> for <code>{{labels.certPurpose}}</code> is expired'
Severity	Critical
Condition	End Entity (NF) certificate has expired.
OID	1.3.6.1.4.1.323.5.3.54.1.2.7009
Metric Used	occm_end_entity_cert_expiration_seconds

Table 7-9 (Cont.) OccmEndEntityCertExpired

Field	Details
Recommended Actions	Cause: The Certificate is going to expire. Diagnostic Information: 1. Refer to the application logs on Kibana and filter based on occm service name. Check for ERROR WARNING logs related to thread exceptions. 2. Check logs to identify the root cause, one of the possible cause is CA connection failure. Recovery: The alert is cleared when the certificate is recreated. 1. Perform recreate operation when is CA is accessible. Alert will be cleared once recreation is successful. 2. If CA is still down then create End-Entity(NF) certificate manually and update details in secret which will be automatically monitored by OCCM. 3. Manual recreate can be triggered once CA connection is resume. 4. Manual recreate can be triggered once CA connection is resume. 5. For more information, refer <i>Oracle Communications, Cloud Native Core, Certificate Management Troubleshooting Guide</i> for troubleshooting scenarios. 6. If this is unexpected, contact My Oracle Support. Ensure to capture the occm pod logs and relevant metrics to help Support analyze the issue.

7.9 OccmServiceDown

Table 7-10 OccmServiceDown

Field	Details
Description	OCCM Service Down Alert New certificates will not be created, and existing ones can not be renewed until OCCM is back
Summary	namespace: {{\$labels.namespace}}, podname: {{\$labels.pod}}, timestamp: {{ with query "time()" }}{{ . first value humanizeTimestamp }}{{ end }}: OCCM service is down
Severity	Critical
Condition	The pods of the occm service is unavailable.
OID	1.3.6.1.4.1.323.5.3.54.1.2.7004
Metric Used	up Note: This is a prometheus metric used for instance availability monitoring. If this metric is not available, use the similar metric as exposed by the monitoring system.

Table 7-10 (Cont.) OccmServiceDown

Field	Details
Recommended Actions	Cause: The Certificate is going to expire within 1 week. Diagnostic Information: - Check the orchestration logs of occm service and check for readiness probe failures. - Refer to the application logs on Kibana and filter based on occm service names. Check for ERROR WARNING logs related to thread exceptions. Recovery:The alert is cleared when the occm service is available. If alert is not cleared: - Depending on the failure reason, take the resolution steps from <i>Oracle Communications, Cloud Native Core, Certificate Management Troubleshooting Guide</i>. - If this is unexpected, contact My Oracle Support. Ensure to capture the occm pod logs and relevant metrics to help Support analyze the issue.

7.10 OccmMemoryUsageMinorThreshold

Table 7-11 OccmMemoryUsageMinorThreshold

Field	Details
Description	OCCM Memory Usage Alert OCCM Memory Usage for pod {{ \$labels.pod }} has crossed the configured minor threshold (70%) (value={{ \$value }}) of its limit.
Summary	namespace: {{ \$labels.namespace }}, podname: {{ \$labels.pod }}, timestamp: {{ with query "time()" }}{{ . first value humanizeTimestamp }}{{ end }}: Memory Usage of pod exceeded 70% of its limit.
Severity	Minor
Condition	A pod has reached the configured minor threshold(70%) of its memory resource limits.
OID	1.3.6.1.4.1.323.5.3.54.1.2.7005
Metric Used	container_memory_usage_bytes, Note : This is a kubernetes metric used for instance availability monitoring. If the metric is not available, use the similar metric as exposed by the monitoring system.

Table 7-11 (Cont.) OccmMemoryUsageMinorThreshold

Field	Details
Recommended Actions	Cause: The Certificate is going to expire within 1 week. Diagnostic Information: - Refer to the application logs on Kibana and filter based on occm service name. Check for ERROR WARNING logs related to thread exceptions. Recovery:The alert gets cleared when the memory utilization falls below the Minor Threshold or crosses the major threshold, in which case OccmMemoryUsageMajorThreshold alert shall be raised. If alert is not cleared: - Depending on the failure reason, take the resolution steps from <i>Oracle Communications, Cloud Native Core, Certificate Management Troubleshooting Guide</i>. - If this is unexpected, contact My Oracle Support. Ensure to capture the occm pod logs and relevant metrics to help Support analyze the issue. Note Note: The threshold is configurable in the <code>occm_alertingrules_<version>.yaml</code> file.

7.11 OccmMemoryUsageMajorThreshold

Table 7-12 OccmMemoryUsageMajorThreshold

Field	Details
Description	OCCM Memory Usage Alert OCCM Memory Usage for pod {{ \$labels.pod }} has crossed the configured major threshold (80%) (value={{ \$value }}) of its limit.
Summary	namespace: {{ \$labels.namespace }}, podname: {{ \$labels.pod }}, timestamp: {{ with query "time()" }}{{ . first value humanizeTimestamp }}{{ end }}: Memory Usage of pod exceeded 80% of its limit.
Severity	Major
Condition	A pod has reached the configured major threshold(80%) of its memory resource limits.
OID	1.3.6.1.4.1.323.5.3.54.1.2.7005
Metric Used	container_memory_usage_bytes, Note : This is a kubernetes metric used for instance availability monitoring.If the metric is not available, use the similar metric as exposed by the monitoring system.

Table 7-12 (Cont.) OccmMemoryUsageMajorThreshold

Field	Details
Recommended Actions	Cause: The Certificate is going to expire within 1 week. Diagnostic Information: - Refer to the application logs on Kibana and filter based on occm service name. Check for ERROR WARNING logs related to thread exceptions. Recovery:The alert gets cleared when the memory utilization falls below the Major Threshold or crosses the critical threshold, in which case OccmMemoryUsageMajorThreshold alert shall be raised If alert is not cleared: - Depending on the failure reason, take the resolution steps from <i>Oracle Communications, Cloud Native Core, Certificate Management Troubleshooting Guide</i>. - If this is unexpected, contact My Oracle Support. Ensure to capture the occm pod logs and relevant metrics to help Support analyze the issue. Note Note: The threshold is configurable in the <code>occm_alertingrules_<version>.yaml</code> file.

7.12 OccmMemoryUsageCriticalThreshold

Table 7-13 OccmMemoryUsageCriticalThreshold

Field	Details
Description	OCCM Memory Usage Alert OCCM Memory Usage for pod {{ \$labels.pod }} has crossed the configured critical threshold (90%) (value={{ \$value }}) of its limit..
Summary	namespace: {{ \$labels.namespace }}, podname: {{ \$labels.pod }}, timestamp: {{ with query "time()" }}{{ . first value humanizeTimestamp }}{{ end }}: Memory Usage of pod exceeded 90% of its limit.
Severity	Critical
Condition	A pod has reached the configured critical threshold (90%) of its memory resource limits
OID	1.3.6.1.4.1.323.5.3.54.1.2.7005
Metric Used	container_memory_usage_bytes, Note : This is a kubernetes metric used for instance availability monitoring.If the metric is not available, use the similar metric as exposed by the monitoring system.

Table 7-13 (Cont.) OccmMemoryUsageCriticalThreshold

Field	Details
Recommended Actions	Cause: The Certificate is going to expire within 1 week. Diagnostic Information: - Refer to the application logs on Kibana and filter based on occm service name. Check for ERROR WARNING logs related to thread exceptions. Recovery:The alert gets cleared when the memory utilization falls below the Critical Threshold. If alert is not cleared: - Depending on the failure reason, take the resolution steps from <i>Oracle Communications, Cloud Native Core, Certificate Management Troubleshooting Guide</i>. - If this is unexpected, contact My Oracle Support. Ensure to capture the occm pod logs and relevant metrics to help Support analyze the issue. Note Note: The threshold is configurable in the <code>occm_alertingrules_<version>.yaml</code> file.

7.13 OccmCPUUsageMinorThreshold

Table 7-14 OccmCPUUsageMinorThreshold

Field	Details
Description	OCCM CPU Usage Alert OCCM Pod {{ \$labels.pod }} has high CPU usage detected.
Summary	namespace: {{ \$labels.namespace }}, podname: {{ \$labels.pod }}, timestamp: {{ with query "time()" }}{{ . first value humanizeTimestamp }}{{ end }}: CPU usage is {{ \$value printf "%.2f" }} which is usage is above 70% (current value is: {{ \$value }})
Severity	Minor
Condition	CPU usage is above 70%
OID	1.3.6.1.4.1.323.5.3.54.1.2.7006
Metric Used	container_cpu_usage_seconds_total

Table 7-14 (Cont.) OccmCPUUsageMinorThreshold

Field	Details
Recommended Actions	Cause: The CPU usage is above 70%. Diagnostic Information: - Refer to the application logs on Kibana and filter based on occm service name. Check for ERROR WARNING logs related to thread exceptions. Recovery:The alert gets cleared when the CPU usage falls below the Minor Threshold. If alert is not cleared: - Depending on the failure reason, take the resolution steps from <i>Oracle Communications, Cloud Native Core, Certificate Management Troubleshooting Guide</i>. - If this is unexpected, contact My Oracle Support. Ensure to capture the occm pod logs and relevant metrics to help Support analyze the issue. Note The threshold is configurable in the <code>occm_alertingrules_<version>.yaml</code> file.

7.14 OccmCMPFailureMinor

Table 7-15 OccmCMPFailureMinor

Field	Details
Description	OCCM CMP Command Execution Failure Alert The certificate {{labels.certName}} used by {{labels.nfType}} has failed while executing CMP cmd with {{labels.statusCode}}.
Summary	namespace: {{labels.namespace}}, podname: {{labels.pod}}, timestamp: {{ with query "time()" }}{{ . first value humanizeTimestamp }}{{ end }}: The certificate {{labels.certName}} used by {{labels.nfType}} has failed while executing CMP cmd with {{labels.statusCode}}.
Severity	Minor
Condition	Certificate has failed while executing CMP cmds.
OID	1.3.6.1.4.1.323.5.3.54.1.2.7007
Metric Used	occm_cmp_responses_total

Table 7-15 (Cont.) OccmCMPFailureMinor

Field	Details
Recommended Actions	Cause: When the rate of certificate failure due to CMP command execution error has crossed the threshold. Diagnostic Information: - Refer to the application logs on Kibana and filter based on occm service name. Check for ERROR WARNING logs related to thread exceptions. Recovery: The alert is cleared when the rate of certificate failure due to CMP command execution error falls below the Minor threshold or when the error rate crosses the Major threshold, in which case the OccmCMPFailureMajor alert is raised. If alert is not cleared: - Depending on the failure reason, take the resolution steps from <i>Oracle Communications, Cloud Native Core, Certificate Management Troubleshooting Guide</i>. - If this is unexpected, contact My Oracle Support. Ensure to capture the occm pod logs and relevant metrics to help Support analyze the issue. Note The threshold is configurable in the <code>occ_m_alertingrules_<version>.yaml</code> file.

7.15 OccmCMPFailureMajor

Table 7-16 OccmCMPFailureMajor

Field	Details
Description	OCCM CMP Command Execution Failure Alert The certificate <code>{{ \$labels.certName }}</code> used by <code>{{ \$labels.nfType }}</code> has failed while executing CMP cmd with <code>{{ \$labels.statusCode }}</code>.
Summary	namespace: <code>{{ \$labels.namespace }}</code>, podname: <code>{{ \$labels.pod }}</code>, timestamp: <code>{{ with query "time()" }}{ . first value humanizeTimestamp }}{ end }}</code>: The certificate <code>{{ \$labels.certName }}</code> used by <code>{{ \$labels.nfType }}</code> has failed while executing CMP cmd with <code>{{ \$labels.statusCode }}</code>.
Severity	Major
Condition	Certificate has failed while executing CMP cmds
OID	1.3.6.1.4.1.323.5.3.54.1.2.7007
Metric Used	occ_m_cmp_responses_total

Table 7-16 (Cont.) OccmCMPFailureMajor

Field	Details
Recommended Actions	Cause: When the rate of certificate failure due to CMP command execution error has crossed the threshold. Diagnostic Information: - Refer to the application logs on Kibana and filter based on occm service name. Check for ERROR WARNING logs related to thread exceptions. Recovery:The alert is cleared when the rate of certificate failure due to CMP command execution error falls below the Minor threshold or when the error rate crosses the Major threshold, in which case the OccmCMPFailureMajor alert is raised. If alert is not cleared: - Depending on the failure reason, take the resolution steps from <i>Oracle Communications, Cloud Native Core, Certificate Management Troubleshooting Guide</i>. - If this is unexpected, contact My Oracle Support. Ensure to capture the occm pod logs and relevant metrics to help Support analyze the issue. Note The threshold is configurable in the <code>occm_alertingrules_<version>.yaml</code> file.

7.16 OccmCMPFailureCritical

Table 7-17 OccmCMPFailureCritical

Field	Details
Description	OCCM CMP Command Execution Failure Alert The certificate <code>{{labels.certName}}</code> used by <code>{{labels.nfType}}</code> has failed while executing CMP cmd with <code>{{labels.statusCode}}</code>.
Summary	namespace: <code>{{labels.namespace}}</code>, podname: <code>{{labels.pod}}</code>, timestamp: <code>{{ with query "time()" }}{ . first value humanizeTimestamp }}{ end }}</code>. The certificate <code>{{labels.certName}}</code> used by <code>{{labels.nfType}}</code> has failed while executing CMP cmd with <code>{{labels.statusCode}}</code>.
Severity	Critical
Condition	Certificate has failed while executing CMP cmds
OID	1.3.6.1.4.1.323.5.3.54.1.2.7007
Metric Used	occm_cmp_responses_total

Table 7-17 (Cont.) OccmCMPFailureCritical

Field	Details
Recommended Actions	Cause: When the rate of certificate failure due to CMP command execution error has crossed the threshold. Diagnostic Information: - Refer to the application logs on Kibana and filter based on the OCCM service name. Check for ERROR WARNING logs related to thread exceptions. Recovery: The alert is cleared when the rate of certificate failure due to CMP command execution error falls below the Critical threshold. If the alert is not cleared: - Depending on the failure reason, take the resolution steps from <i>Oracle Communications, Cloud Native Core, Certificate Management Troubleshooting Guide</i>. - If this is unexpected, contact My Oracle Support. Ensure to capture the occm pod logs and relevant metrics to help Support analyze the issue. Note The threshold is configurable in the <code>occm_alertingrules_<version>.yaml</code> file.

7.17 OccmFailureMinor

Table 7-18 OccmFailureMinor

Field	Details
Description	OCCM Internal Failure Alert The certificate <code>{{ \$labels.certName }}</code> used by <code>{{ \$labels.nfType }}</code> has failed while creating cert with <code>{{ \$labels.errorReason }}</code> .
Summary	namespace: <code>{{ \$labels.namespace }}</code> , podname: <code>{{ \$labels.pod }}</code> , timestamp: <code>{{ with query "time()" }}{{ . first value humanizeTimestamp }}{{ end }}</code> : The certificate <code>{{ \$labels.certName }}</code> used by <code>{{ \$labels.nfType }}</code> has failed while creating cert with <code>{{ \$labels.errorReason }}</code> .
Severity	Minor
Condition	Certificate has failed while creating
OID	1.3.6.1.4.1.323.5.3.54.1.2.7008
Metric Used	occm_cert_request_status_total

Table 7-18 (Cont.) OccmFailureMinor

Field	Details
Recommended Actions	Cause: The rate of OCCM errors has crossed the threshold. Diagnostic Information: - Refer to the application logs on Kibana and filter based on occm service name. Check for ERROR WARNING logs related to thread exceptions. Recovery:The alert is cleared when the rate OCCM error falls below the Minor threshold or when the error rate crosses the Major threshold, in which case the OccmFailureMajor alert is raised. If alert is not cleared: - Depending on the failure reason, take the resolution steps from <i>Oracle Communications, Cloud Native Core, Certificate Management Troubleshooting Guide</i>. - If this is unexpected, contact My Oracle Support. Ensure to capture the occm pod logs and relevant metrics to help Support analyze the issue. Note Note: The threshold is configurable in the <code>occm_alertingrules_<version>.yaml</code> file.

7.18 OccmFailureMajor

Table 7-19 OccmFailureMajor

Field	Details
Description	OCCM Internal Failure Alert The certificate {{ \$labels.certName }} used by {{ \$labels.nfType }} has failed while creating cert with {{ \$labels.errorReason }}.
Summary	namespace: {{ \$labels.namespace }}, podname: {{ \$labels.pod }}, timestamp: {{ with query "time()" }}{{ . first value humanizeTimestamp }}{{ end }}: The certificate {{ \$labels.certName }} used by {{ \$labels.nfType }} has failed while creating cert with {{ \$labels.errorReason }}.
Severity	Major
Condition	Certificate has failed while creating
OID	1.3.6.1.4.1.323.5.3.54.1.2.7008
Metric Used	occm_cert_request_status_total

Table 7-19 (Cont.) OccmFailureMajor

Field	Details
Recommended Actions	Cause: The rate of OCCM errors has crossed the threshold. Diagnostic Information: - Refer to the application logs on Kibana and filter based on occm service name. Check for ERROR WARNING logs related to thread exceptions. Recovery:The alert is cleared when the rate OCCM error falls below the Major threshold or when the error rate crosses the Critical threshold, in which case the OccmFailureCritical alert is raised. If alert is not cleared: - Depending on the failure reason, take the resolution steps from <i>Oracle Communications, Cloud Native Core, Certificate Management Troubleshooting Guide</i>. - If this is unexpected, contact My Oracle Support. Ensure to capture the occm pod logs and relevant metrics to help Support analyze the issue. Note Note: The threshold is configurable in the <code>occm_alertingrules_<version>.yaml</code> file.

7.19 OccmFailureCritical

Table 7-20 OccmFailureCritical

Field	Details
Description	OCCM CMP Command Execution Failure Alert The certificate {{ \$labels.certName }} used by {{ \$labels.nfType }} has failed while creating cert with {{ \$labels.errorReason }}.
Summary	namespace: {{ \$labels.namespace }}, podname: {{ \$labels.pod }}, timestamp: {{ with query "time()" }}{{ . first value humanizeTimestamp }}{{ end }}: The certificate {{ \$labels.certName }} used by {{ \$labels.nfType }} has failed while creating cert with {{ \$labels.errorReason }}.
Severity	critical
Condition	Certificate has failed while creating
OID	1.3.6.1.4.1.323.5.3.54.1.2.7008
Metric Used	occm_cert_request_status_total

Table 7-20 (Cont.) OccmFailureCritical

Field	Details
Recommended Actions	Cause: The rate of OCCM errors has crossed the threshold. Diagnostic Information: - Refer to the application logs on Kibana and filter based on occm service name. Check for ERROR WARNING logs related to thread exceptions. Recovery:The alert is cleared when the rate OCCM error falls below the Major threshold or when the error rate crosses the Critical threshold, in which case the OccmFailureCritical alert is raised. If alert is not cleared: - Depending on the failure reason, take the resolution steps from <i>Oracle Communications, Cloud Native Core, Certificate Management Troubleshooting Guide</i>. - If this is unexpected, contact My Oracle Support. Ensure to capture the occm pod logs and relevant metrics to help Support analyze the issue. Note Note: The threshold is configurable in the <code>occm_alertingrules_<version>.yaml</code> file.

7.20 OccmInputSecretModifyMajor

Table 7-21 OccmInputSecretModifyMajor

Field	Details
Description	Input secret is modified by non-OCCM user The Secret <code>{{ \$labels.secret }}</code> in <code>{{ \$labels.secretNamespace }}</code> is modified by non-occm user, which is used by <code>{{ \$labels.name }}</code>.'
Summary	'namespace: <code>{{ \$labels.namespace }}</code>, podname: <code>{{ \$labels.pod }}</code>, timestamp: <code>{{ with query "time()" }}{{ . first value humanizeTimestamp }}{{ end }}</code>: The Secret <code>{{ \$labels.secret }}</code> in <code>{{ \$labels.secretNamespace }}</code> is modified by non-occm user, which is used by <code>{{ \$labels.name }}</code> and <code>{{ \$labels.type }}</code>.'
Severity	Major
Condition	Input secrets are modified by non-OCCM users or by the operator manually.
OID	1.3.6.1.4.1.323.5.3.54.1.2.7010
Metric Used	occm_secret_event_total

Table 7-21 (Cont.) OccmInputSecretModifyMajor

Field	Details
Recommended Actions	Cause: The input secret is modified by non-occm user. Diagnostic Information: - Check input secrets for any modifications. Recovery:The alert is cleared when the new input secret is created manually. If alert is not cleared: - Refer to the alert's label for which namespace and for which secret alert is triggered. - Update input secrets with correct data, if require. - If this is unexpected, contact My Oracle Support. Ensure to capture the occm pod logs and relevant metrics to help Support analyze the issue.

7.21 OccmOutputSecretModifyMinor

Table 7-22 OccmOutputSecretModifyMinor

Field	Details
Description	Output secret is modified by non-OCCM user The Secret <code>{{labels.secret}}</code> in <code>{{labels.secretNamespace}}</code> is modified by non-occm user, which is used by <code>{{labels.name}}</code>.'
Summary	'namespace: <code>{{labels.namespace}}</code>, podname: <code>{{labels.pod}}</code>, timestamp: <code>{{ with query "time()" }}{{ . first value humanizeTimestamp }}{{ end }}</code>: The Secret <code>{{labels.secret}}</code> in <code>{{labels.secretNamespace}}</code> is modified by non-occm user, which is used by <code>{{labels.name}}</code> and <code>{{labels.type}}</code>.'
Severity	Minor
Condition	Output secrets are modified by non-OCCM user or by operator manually
OID	1.3.6.1.4.1.323.5.3.54.1.2.7011
Metric Used	occm_secret_event_total

Table 7-22 (Cont.) OccmOutputSecretModifyMinor

Field	Details
Recommended Actions	Cause: The output secret is modified by non-occm user. Diagnostic Information: - Check output secrets for any modifications. - Automatic recreation will be triggered if certificate which is modified doesn't match with cert config. - The update for validity will be done, if modified certificate validation is successful with cert configuration. No recreation will be triggered in this case. Recovery:The alert is cleared when the new output secret is created by occm. If alert is not cleared: - Depending on the failure reason, take the resolution steps from <i>Oracle Communications, Cloud Native Core, Certificate Management Troubleshooting Guide</i>. - If this is unexpected, contact My Oracle Support. Ensure to capture the occm pod logs and relevant metrics to help Support analyze the issue.

7.22 OccmK8sResourceDeleteMajor

Table 7-23 OccmK8sResourceDeleteMajor

Field	Details
Description	Kubernetes resource (secret or namespace) is deleted by non-OCCM user The Kubernetes resource is deleted, which is used in {{labels.name}} of type {{labels.type}}. K8s resources, secretNamespace: {{labels.secretNamespace}} and secret: {{labels.secret}}'
Summary	{{labels.namespace}}, podname: {{labels.pod}}, timestamp: {{ with query "time()" }}{{ . first value humanizeTimestamp }}{{ end }}: The k8s resource is deleted, which is used in {{labels.name}} of type {{labels.type}}. K8s resources, namespace: {{labels.secretNamespace}} and secret: {{labels.secret}}.'
Severity	Major
Condition	Kubernetes resources (secret or namespace) are deleted by non-OCCM user or by operator manually.
OID	1.3.6.1.4.1.323.5.3.54.1.2.7012
Metric Used	occm_secret_event_total

Table 7-23 (Cont.) OccmK8sResourceDeleteMajor

Field	Details
Recommended Actions	Cause: The k8s resource (secret/namespace) is deleted by non-occm user. Diagnostic Information: • Check secret or namespace for any deletion. • If secret is deleted, Automatic recreation of cert will be triggered. • If namespace is deleted, no auto-recreation will happen. Recovery:The alert is cleared when the new secret/namespace is created again. If alert is not cleared: • Try to create the secret/namespace again and check for alert. • If namespace is deleted then there might be issue even after recreation of namespace. So if namespaces is deleted, delete cert config and create again. • Depending on the failure reason, take the resolution steps from <i>Oracle Communications, Cloud Native Core, Certificate Management Troubleshooting Guide</i>. • If this is unexpected, contact My Oracle Support. Ensure to capture the occm pod logs and relevant metrics to help Support analyze the issue.

8

OCCM KPIs

This section describes the KPIs available for OCCM.

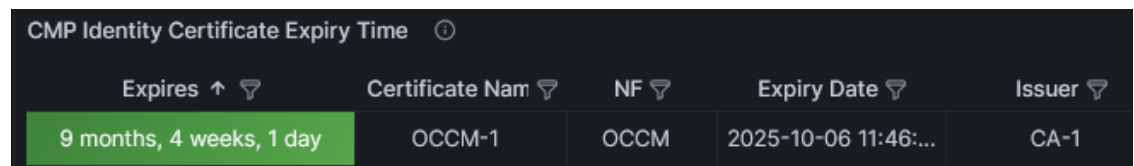
8.1 CMP Identity (OCCM) Certificate Expiry Time

Table 8-1 CMP Identity (OCCM) Certificate Expiry Time

Field	Details
Description	CMP Identity (OCCM) certificate expiry time to list Certificate Name and Expiry Date.
Expression	OCCM dashboard in grafana will show CMP Identity (OCCM) Certificate Expiry Time panel with columns. Table visualization listing Expires, NF, Certificate Name, Expiry Date. Expires column uses color coding to indicate near expiry status. all:occm_cmp_identity_cert_expiration_seconds{namespace="\$namespace"} * 1000 != 0 Expires column: ((occm_cmp_identity_cert_expiration_seconds{namespace="\$namespace"} != 0) - time())*1000

OCCM KPI Dashboard

Figure 8-1 CMP Identity (OCCM) Certificate Expiry Time



CMP Identity Certificate Expiry Time ⓘ				
Expires ↑ ⌵	Certificate Name ⌵	NF ⌵	Expiry Date ⌵	Issuer ⌵
9 months, 4 weeks, 1 day	OCCM-1	OCCM	2025-10-06 11:46:...	CA-1

Color coding description:-

Red (Critical):- Certificate expiring within $0 \leq 7$ days Or Certificate expired ≤ 0 days

Light Red(Major):- Certificate expiring within $> 7 \leq 30$ days

Orange (Minor):- Certificate expiring within $> 30 \leq 90$

Yellow :- Certificate expiring within $> 90 \leq 180$

Green :- Certificates not expiring sooner

8.2 End Entity (NF) Certificate Expiry Time

Table 8-2 End Entity (NF) Certificate Expiry Time

Field	Details
Description	End Entity (NF) certificate expiry time to list Certificate Name and Expiry Date.
Expression	OCCM dashboard in grafana will show End Entity (NF) Certificate Expiry Time panel with columns. Table visualization listing Expires, NF, Certificate Name, Expiry Date. Expires column uses color coding to indicate near expiry status. all:occm_end_entity_cert_expiration_seconds{certType!="ISSUER_HEALTHCHECK", namespace="\$namespace"} * 1000 != 0 Expires column: ((occm_end_entity_cert_expiration_seconds{certType!="ISSUER_HEALTHCHECK", namespace="\$namespace"} != 0)-time())*1000

OCCM KPI Dashboard

Figure 8-2 End Entity (NF) Certificate Expiry Time

Expires ↑	Certificate Name	NF	Expiry Date	Issuer
6 days, 23 hours, 51 minutes	nrfcert-10dec2...	NRF	2024-12-17 11:49:...	CA-1
4 weeks, 0 days, 23 hours	nrfcert-10dec2...	NRF	2025-01-08 11:51:15	CA-1
1 month, 4 weeks, 1 day	nrfcert-10dec2...	NRF	2025-02-08 11:51:...	CA-1
3 months, 4 weeks, 1 day	nrfcert-10dec2...	NRF	2025-04-09 11:55...	CA-1
4 months, 4 weeks, 1 day	nrfcert-10dec2...	NRF	2025-05-09 11:53...	CA-1
6 months, 2 weeks, 5 days	nrfcert-10dec2...	NRF	2025-06-28 11:54...	CA-1

Color coding description:-

Red (Critical):- Certificate expiring within 0 <= 7 days Or Certificate expired <= 0 days

Light Red(Major):- Certificate expiring within > 7 <= 30 days

Orange (Minor):- Certificate expiring within > 30 <= 90

Yellow :- Certificate expiring within > 90 <= 180

Green :- Certificates not expiring sooner

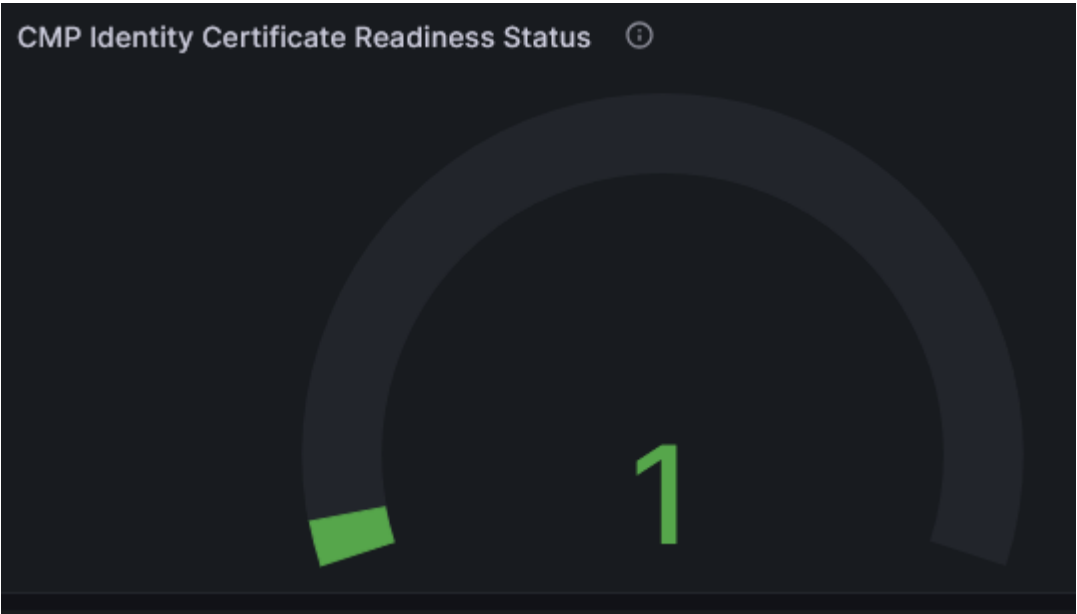
8.3 CMP Identity (OCCM) Certificate Readiness Status

Table 8-3 CMP Identity (OCCM) Certificate Readiness Status

Field	Details
Description	CMP Identity (OCCM) Certificate Readiness Status to indicate if number of Ready and Failed Certificates.
Expression	OCCM dashboard in grafana shows the CMP Identity (OCCM) Certificate Readiness Status panel panel gauge visualization to indicate if number of Ready and Failed Certificates Creating:count(occm_cmp_identity_cert_status{namespace="\$namespace"} == 1) (Color:Orange) Ready:count(occm_cmp_identity_cert_status{namespace="\$namespace"} == 2) (Color:Green) Failed:count(occm_cmp_identity_cert_status{namespace="\$namespace"} == 3) (Color:Red) Waiting:count(occm_cmp_identity_cert_status{namespace="\$namespace"} == 8) (Color:Light Orange) Expired:count(occm_cmp_identity_cert_status{namespace="\$namespace"} == 7) (Color:Red) During bulk certificate migration, the intermediate status displayed on the gauge includes duplicate certificate and might show an increased number. However, once the process is completed the eventual state will be consistent and will show the correct count of all the certificates.

OCCM KPI Dashboard

Figure 8-3 CMP Identity (OCCM) Certificate Readiness Status



Creating: Orange

Ready: Green

Failed: Red
Waiting: Light Orange
Expired : Red

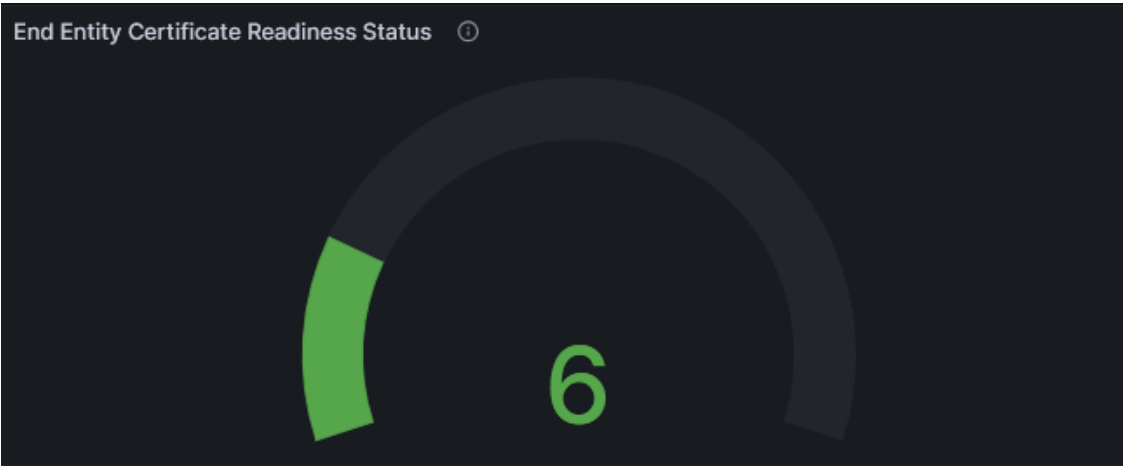
8.4 End Entity (NF) Certificate Readiness Status

Table 8-4 End Entity (NF) Certificate Readiness Status

Field	Details
Description	End Entity (NF) Certificate Readiness Status to indicate if number of Ready and Failed Certificates.
Expression	OCCM dashboard in grafana shows the End Entity (NF) Certificate Readiness Status panel gauge visualization to indicate if number of Ready and Failed Certificates Creating:count(occm_end_entity_cert_status{certType!="ISSUER_HEALTHCHECK",namespace="\$namespace"} == 1) (Color:Orange) Ready:count(occm_end_entity_cert_status{certType!="ISSUER_HEALTHCHECK",namespace="\$namespace"} == 2) (Color:Green) Failed:count(occm_end_entity_cert_status{certType!="ISSUER_HEALTHCHECK",namespace="\$namespace"} == 3) (Color:Red) Waiting:count(occm_end_entity_cert_status{certType!="ISSUER_HEALTHCHECK",namespace="\$namespace"} == 8) (Color:Light Orange) Expired:count(occm_end_entity_cert_status{namespace="\$namespace"} == 7) (Color:Red) During bulk certificate migration, the intermediate status displayed on the gauge includes duplicate certificate and might show an increased number. However, once the process is completed the eventual state will be consistent and will show the correct count of all the certificates.

OCCM KPI Dashboard

Figure 8-4 End Entity (NF) Certificate Readiness Status



Creating: Orange
Ready: Green

Failed: Red

Waiting: Light Orange

Expired : Red

8.5 CMP Request

Table 8-5 CMP Request

Field	Details
Description	Total CMP requests initiated from OCCM towards CA per NF
Expression	OCCM dashboard in grafana will show CMP Request panel which is total CMP requests per NF. all:sum(rate(occm_cmp_requests_total{namespace="\$namespace"}[2m])) SCP:sum(rate(occm_cmp_requests_total{namespace="\$namespace", nfType=~"SCP scp"}[2m])) NRF:sum(rate(occm_cmp_requests_total{namespace="\$namespace", nfType=~"NRF nrf"}[2m]))

8.6 CMP Responses

Table 8-6 CMP Responses

Field	Details
Description	Total CMP responses received from CA per NF by OCCM
Expression	OCCM dashboard in grafana will show CMP Response panel which is total CMP responses per NF. all:sum(rate(occm_cmp_responses_total{namespace="\$namespace"}[2m])) SCP:sum(rate(occm_cmp_responses_total{namespace="\$namespace", nfType=~"SCP scp"}[2m])) NRF:sum(rate(occm_cmp_responses_total{namespace="\$namespace", nfType=~"NRF nrf"}[2m]))

8.7 Configuration Requests

Table 8-7 Configuration Requests

Field	Details
Description	Total Issuer, Certificate Configuration, and Bulk Certificate Migration requests.

Table 8-7 (Cont.) Configuration Requests

Field	Details
Expression	OCCM dashboard in grafana will show Config Requests panel. Total Issuer, Certificate configuration, and Bulk Certificate Migration requests. <pre>all:sum(rate(occm_config_http_requests_total{namespace="\$namespace"}[2m])) SCP certs:sum(rate(occm_config_http_requests_total{namespace="\$namespace", uri=~".*/certs.*", nfType=~"SCP scp"}[2m])) NRF certs:sum(rate(occm_config_http_requests_total{namespace="\$namespace", uri=~".*/certs.*", nfType=~"NRF nrf"}[2m])) issuers:sum(rate(occm_config_http_requests_total{namespace="\$namespace", uri=~".*/issuers.*"}[2m])) Bulk cert migrations:sum(rate(occm_config_http_requests_total{namespace="\$namespace", uri=~".*/certs/bulk-migrate.*"}[2m]))</pre>

8.8 Configuration Responses

Table 8-8 Configuration Responses

Field	Details
Description	Total Issuer, Certificate Configuration, and Bulk Certificate Migration responses.
Expression	OCCM dashboard in grafana will show Config Responses panel. Total Issuer, Certificate configuration, and Bulk Certificate Migration responses. <pre>all:sum(rate(occm_config_http_responses_total{namespace="\$namespace"}[2m])) SCP certs:sum(rate(occm_config_http_responses_total{namespace="\$namespace", uri=~".*/certs.*", nfType=~"SCP scp"}[2m])) NRF certs:sum(rate(occm_config_http_responses_total{namespace="\$namespace", uri=~".*/certs.*", nfType=~"NRF nrf"}[2m])) issuers:sum(rate(occm_config_http_responses_total{namespace="\$namespace", uri=~".*/issuers.*"}[2m])) Bulk cert migrations:sum(rate(occm_config_http_responses_total{{namespace="\$namespace", uri=~".*/certs/bulk-migrate.*"}[2m]))</pre>

8.9 CPU Usage

Table 8-9 CPU Usage

Field	Details
Description	CPU usage of OCCM pod
Expression	Time series indicates CPU usage of OCCM pod. <pre>sum(rate(container_cpu_usage_seconds_total{image!="", namespace="\$namespace", pod=~"occm-.*"}[2m])) by(pod)</pre>

8.10 Memory Usage

Table 8-10 Memory Usage

Field	Details
Description	Memory usage of OCCM pod
Expression	Time series indicates Memory usage of OCCM pod. (avg_over_time(container_memory_usage_bytes{container=~"occm", namespace="\$namespace"}[2m]))

8.11 OpenSSL CLI Duration (occm_cmp_cli_durations_seconds)

Table 8-11 OpenSSL CLI Duration (occm_cmp_cli_durations_seconds)

Field	Details
Description	The time taken by CMP CLI between request and response from CA.
Expression	Used to show the duration of openssl cmp calls occm_cmp_cli_durations_seconds{namespace="occm-ns", uuid="fdsfds-9880-fsd99"}

8.12 Number of requests sent to the CA

Table 8-12 Number of requests sent to the CA

Field	Details
Description	Metric will peg when request cmd prepared and send to CA for generate certificate.
Expression	count(occm_cmp_requests_total{namespace="\$namespace"})

8.13 Number of responses received from CA

Table 8-13 Number of responses received from CA

Field	Details
Description	Metric will peg when response received from CA for generate certificate.
Expression	count(occm_cmp_responses_total{namespace="occm-ns"})

8.14 Number of responses based on response code from CA

Table 8-14 Number of responses based on response code from CA

Field	Details
Description	Metric will peg when response received from CA for generate certificate.

Table 8-14 (Cont.) Number of responses based on response code from CA

Field	Details
Expression	count(occm_cmp_responses_total{namespace="occm-ns", statusCode="OK", status = "SUCCESS"}) or count(occm_cmp_responses_total{namespace="occm-ns", statusCode="ERR_CMP_COMMAND_FAILED", status = "FAILED"})

8.15 Type of request sent to CA

Table 8-15 Type of request sent to CA

Field	Details
Description	Metric will peg when request cmd prepared and send to CA for generate certificate.
Expression	count(occm_cmp_requests_total{namespace="occm-ns", requestType="ir"}) or count(occm_cmp_requests_total{namespace="occm-ns", requestType="kur"})

8.16 Number of certificates issued by CA

Table 8-16 Number of certificates issued by CA

Field	Details
Description	Metric will peg when response received from CA for generate certificate.
Expression	count(occm_cmp_responses_total{namespace="occm-ns", status = "SUCCESS", statusCode = "OK"})

8.17 Number of CSRs denied by CA or TLS handshake failures or HTTPs connection failures during CA connection

Table 8-17 Number of CSRs denied by CA or TLS handshake failures or HTTPs connection failures during CA connection

Field	Details
Description	Metric will peg when response received from CA for generate certificate.
Expression	count(occm_cmp_responses_total{namespace="occm-ns", status = "FAILED"}) or count(occm_cmp_responses_total{namespace="occm-ns", statusCode="ERR_CMP_COMMAND_FAILED", status="FAILED"})

8.18 Error while writing the key, certificate, or chain in the Kubernetes secrets

Table 8-18 Error while writing the key, certificate, or chain in the Kubernetes secrets

Field	Details
Description	Metric will peg when cert renew or create worker complete its process
Expression	<code>occm_cert_request_status_total{namespace="occm-ns", errorReason="ERR_SECRET_FAILED"}</code>

8.19 Unable to access or read from Kubernetes secrets

Table 8-19 Unable to access or read from Kubernetes secrets

Field	Details
Description	Metric will peg when cert renew or create worker complete its process
Expression	<code>occm_cert_request_status_total{namespace="occm-ns", errorReason="ERR_SECRET_EXIST"}</code>

8.20 Check Renewed Certificate

Table 8-20 Check Renewed Certificate

Field	Details
Description	Metric will peg when cert renew or create worker complete its process
Expression	<code>occm_cert_request_status_total{namespace="occm-ns", operationType="RENEW"}</code>

8.21 Certificate Error and Warnings

Table 8-21 Certificate Error and Warnings

Field	Details
Description	List of certificates having Error and Warnings for duration of 5 mins.
Expression	<code>rate(occm_cert_request_status_total{namespace="occm-ns", errorReason!="OK.*"}[5m]) > 0</code>

OCCM KPI Dashboard

Figure 8-5 Certificate Error and Warnings

Certificate Error					
Cert Name	UUID	Operation	Reason	Issuer	nType
OCCM-TLS-Wrong	60761a60-5fe8-4579-a06c-02c549b...	CREATE	ERR_CMP_COMMAND_FAILED	CA-TLS-Wrong	OCCM

Displayed Columns

- 1. Cert Name - Certificate Name
- 2. UUID - Certificate UUID
- 3. Operation - Certificate Operation Type (CREATE or RENEW)
- 4. Reason - Error code indicating Certificate Error or Warning Reason
- 5. Issuer - Issuer Name linked to the Certificate

8.22 Bulk Certificate Migration Error

Table 8-22 Bulk Certificate Migration Error

Field	Details
Description	List of certificates that are failed during bulk migration of certificate.
Expression	occm_cert_request_status_total{namespace="\$namespace", errorReason="ERR_BULK_CERT_MIGRATION"}

OCCM KPI Dashboard

Figure 8-6 Bulk Certificate Migration Error

Bulk Cert Migrations Error			
Cert Name	Cert UUID	Issuer	Bulk Migration UUID
NRF-Beta-1	ee0c8126-e7cb-4c9e-af11-bfbaaf8c60b3	CA1	233e69ef-073c-417a-8d88-d498a95688ef
NRF-Beta-2	b8626c2f-f42e-44ae-834f-7918ec94add4	CA1	233e69ef-073c-417a-8d88-d498a95688ef
NRF-Beta-3	2e388ba9-283e-4ebc-b000-de530f9e2294	CA1	233e69ef-073c-417a-8d88-d498a95688ef

Displayed Columns

- 1. Cert Name - Name of the certificate whose migration failed.
- 2. UUID - UUID of the certificate whose migration failed.
- 3. Issuer - Name of the destination issuer to which the certificate is to be migrated.
- 4. Bulk Migration UUID - The UUID of the bulk certificate migration.

Note

Filtering and sorting are available on all the fields. To get the certificates that failed the migration, you must filter the rows based on the bulk migration UUID.

8.23 Issuer Health Check Certificate Readiness Status

Table 8-23 Issuer Health Check Certificate Readiness Status

Field	Details
Description	Issuer Health Check Certificate Readiness Status to display certificate lifecycle state counts for Creating, Ready, Failed, Waiting, and Expired certificates.
Expression	OCCM dashboard in Grafana will show Issuer Health Check Certificate Readiness Status gauge visualization displaying certificate readiness states for Issuer Health Check certificates. CREATING: count(occm_end_entity_cert_status{namespace="\$namespace", certType="ISSUER_HEALTHCHECK"} == 1) OR vector(0) READY: count(occm_end_entity_cert_status{namespace="\$namespace", certType="ISSUER_HEALTHCHECK"} == 2) OR vector(0) FAILED: count(occm_end_entity_cert_status{namespace="\$namespace", certType="ISSUER_HEALTHCHECK"} == 3) OR vector(0) WAITING: count(occm_end_entity_cert_status{namespace="\$namespace", certType="ISSUER_HEALTHCHECK"} == 8) EXPIRED: count(occm_end_entity_cert_status{namespace="\$namespace", certType="ISSUER_HEALTHCHECK"} == 7) OR vector(0)

Color coding description:

- Orange :- Creating certificates
- Green :- Ready certificates
- Red :- Failed / Expired certificates
- Light Orange :- Waiting certificates

8.24 Issuer Health Check Certificate Expiry Time

Table 8-24 Issuer Health Check Certificate Expiry Time

Field	Details
Description	Issuer Health Check Certificate Expiry Time to list Issuer Name, Certificate Name, Last Successful Update, Next Renewal Time, Down Time, and Status.

Table 8-24 (Cont.) Issuer Health Check Certificate Expiry Time

Field	Details
Expression	Last Successful Update: <pre>(occm_cert_start_date_seconds{namespace="\$namespace"} * 1000)>=0 and on(issuerName, certName, certType) (occm_end_entity_cert_status{namespace="\$namespace" certType="ISSUER_HEALTHCHECK"} >= 1 and occm_end_entity_cert_status{namespace="\$namespace",certType="ISSUER_HEALT HCHECK"} != 6)</pre> Next Renewal Time: <pre>occm_cert_renew_time_seconds{namespace="\$namespace"} - time() and on(issuerName, certName, certType) (occm_end_entity_cert_status{namespace="\$namespace",certType="ISSUER_HEAL THCHECK"} > 1 and occm_end_entity_cert_status{namespace="\$namespace", certType="ISSUER_HEALTHCHECK"} != 6)</pre> Down Time: <pre>(time() - occm_cert_first_fail_time_seconds{namespace="\$namespace"}) unless occm_cert_first_fail_time_seconds{namespace="\$namespace"} == -1</pre> Status: <pre>max by (issuerName, certName, certType) (((occm_end_entity_cert_status{namespace="\$namespace", certType="ISSUER_HEALTHCHECK"}>0 and occm_end_entity_cert_status{namespace="\$namespace", certType="ISSUER_HEALTHCHECK"} !=6) == bool 2) + (2*((occm_end_entity_cert_status{namespace="\$namespace",certType="ISSUER_H EALTHCHECK"}>0 and occm_end_entity_cert_status{namespace="\$namespace", certType="ISSUER_HEALTHCHECK"} !=6) == bool 7)))</pre>

Status mapping description:

- UP = 1
- DOWN = 0
- EXPIRED = 2

A.1 Certificate Configuration Examples

A.1.1 Creating End Entity (NF) Certificate Using OCCM - Sample Configuration

This section describes the sequence of steps to be performed to generate a signed certificate (NF certificate) using OCCM

1. Creating End Entity (NF) Certificate:

The following images provide a sample configuration for creating the issuer using CNC Console GUI

- a. This image provides a sample configuration for the **Create Issuer** screen.

Figure 7 Create Issuer

Field	Value
UUID:	UUID
Name:	CA1
Server URL:	http://ca1-openssl-mock.ns1.svc.ns1:8090
Recipient Distinguished Name:	/CN=X CA
Issuer Distinguished Name:	/CN=X CA
Total Timeout (Seconds):	720
Message Timeout (Seconds):	120

- b. This image shows the **Initial CMP Client (OCCM) Authentication Options** screen configured for issuer creation.

Figure 8 Initial CMP Client (OCCM) Authentication Options

Initial CMP Client(OCCM) Authentication Options	
Type:	Type
Digest Algorithm:	SHA256
MAC Algorithm:	HMACSHA256
MAC Authentication Input	
Namespace:	Namespace
Secret Name:	Secret Name
Password Key:	Password Key
Reference Key:	Reference Key
Signature Authentication Input	
Namespace:	Namespace
Secret Name:	Secret Name
Key:	Key
Cert:	Cert
Extra Certs:	Extra Certs

- c. This image shows the **CMP Client Authentication Options for Other Certificate** screen configured for additional certificate authentication.

Figure 9 CMP Client Authentication Options for Other Certificate

The screenshot shows the 'CMP Client Authentication Options For Other certificate' configuration screen. It is divided into two main sections: 'Signature Authentication Input' and 'Ocm Trust-Store Secret Input'.

Signature Authentication Input:

- Type: SIGNATURE
- Digest Algorithm: SHA256
- Namespace: ns1
- Secret Name: ca1-cmp-identity-secret
- Key: cmpkey.pem
- Cert: cmpcert.pem
- Extra Certs: (empty)

Ocm Trust-Store Secret Input:

- Namespace: ns1
- Name: ca1-occm-trust-store-secret
- Root CA Certs: caroot.pem X
- Intermediate CA Certs: intcacert.pem X
- Server Cert: (empty)

- d. To enable HTTPS communication, provide HTTPS scheme in the server URL field and provide the TLS trust store certificates under TLS config.

Figure 10 HTTPS Scheme

The screenshot shows the 'Server URL' configuration field. The 'Server URL' label is on the left, and the input field contains the value: `https://ca1-openssl-mock.ns1.svc.ns1:8443`.

Figure 11 Enable TLS Config

The screenshot shows the 'TLS Config' configuration screen. It includes a toggle for 'Enable TLS' which is turned on, and a section for 'TLS Trust-Store Secret Input'.

Enable TLS: ☒

TLS Trust-Store Secret Input:

- Namespace: ns1
- Name: ca1-tls-trust-store-secret
- TLS Trusted Certs: tiscert.pem X

2. Issuer (CA) HealthCheck:

The following screenshots provide a sample configuration for Issuer HealthCheck certificate using CNC Console.

- a. This image provides a sample configuration for the **Create Issuer (CA) HealthCheck** screen.

Figure 12 Create Issuer (CA) HealthCheck

The screenshot shows the 'Create Certificate' form in the OCCM console. The left sidebar lists navigation options: Certificate, Issuer, Bulk Certificate Migration, and Logging Level Configuration. The main form fields are as follows:

Field	Value
UUID:	UUID
Name:	HealthCheck-Cert-CA1
Cert Type:	ISSUER-HEALTHCHECK
Network Function:	OCCM
Purpose:	CA-Healthcheck
Issuer:	ca-1
Creation Mode:	AUTOMATIC
Overwrite Secret:	☐ Option is used to overwrite the existing KBs secret with new certificate
Renew Before Expiration (Days):	14
Namespace:	Namespace

- b. This image shows the **Private Key Options** screen configured for certificate creation.

Figure 13 Private Key Options

The screenshot shows the 'Private Key Options' form. The fields are configured as follows:

Field	Value
Key Algorithm:	RSA
Key Encoding:	PEM
Key Size:	KEYSIZE_2048
Private Key Output	
Namespace:	cncc
Secret Name:	healthcheck-ca1-secret
Key:	private-key.pem

- c. This image shows the **Public Key Certificate Options** screen configured for certificate creation.

Figure 14 Public Key Certificate Options

The screenshot shows the 'Public Key Certificate Options' form, which is identical in structure and values to Figure 13. The fields are configured as follows:

Field	Value
Key Algorithm:	RSA
Key Encoding:	PEM
Key Size:	KEYSIZE_2048
Private Key Output	
Namespace:	cncc
Secret Name:	healthcheck-ca1-secret
Key:	private-key.pem

- d. This image shows the **Subject and Subject Alternate Name** screen configured with sample certificate identity values.

Figure 15 Subject and Subject Alternate Name

The screenshot shows a configuration form for a certificate subject. It is divided into two main sections: 'Subject' and 'Subject Alternate Names'. The 'Subject' section includes fields for Country, State, Location, Organization, Organization Unit, Common Name (set to 'occmra'), and Requested Validity (Days) (set to '365'). The 'Subject Alternate Names' section includes a 'Critical' toggle (checked), and fields for IP Addresses, DNS Names, URI ID API Roots, and URI ID URNs.

Subject	
Country:	Country
State:	State
Location:	Location
Organization:	Organization
Organization Unit:	Organization Unit
Common Name:	Common Name occmra
Requested Validity (Days):	Requested Validity (Days) 365

Subject Alternate Names	
Critical:	☒
IP Addresses:	IP Addresses
DNS Names:	DNS Names
URI ID API Roots:	URI ID API Roots
URI ID URNs:	URI ID URNs

- e. This image shows the **Certificate Output and Certificate Chain Output** screen configured with sample output locations.

Figure 16 Certificate Output and Certificate Chain Output

The screenshot shows a configuration form for certificate output. It is divided into two main sections: 'Certificate Output' and 'Certificate Chain Output'. The 'Certificate Output' section includes fields for Namespace (set to 'cncc'), Secret Name (set to 'healthcheck-ca1-secret'), and Key (set to 'healthcheck-cert.pem'). The 'Certificate Chain Output' section includes fields for Namespace, Secret Name, and Key. At the bottom, there is a 'Merge Certificate and Certificate Chain' toggle (unchecked).

Certificate Output	
Namespace:	Namespace cncc
Secret Name:	Secret Name healthcheck-ca1-secret
Key:	Key healthcheck-cert.pem

Certificate Chain Output	
Namespace:	Namespace
Secret Name:	Secret Name
Key:	Key

Merge Certificate and Certificate Chain: ☐

- f. This image shows the **CA Bundle Input** screen configured with sample CA bundle details.

Figure 17 CA bundle input

The screenshot shows a configuration form for CA bundle input. It includes fields for Namespace (set to 'Namespace'), Secret Name (set to 'Secret Name'), and Key (set to 'Key').

CA Bundle Input	
Namespace:	Namespace
Secret Name:	Secret Name
Key:	Key

3. Create CMP Identity (OCCM) Certificate:

The following screenshots provide a sample configuration for creating CMP Identity (OCCM) certificate using CNC Console GUI. Here, OCCM certificate is configured manually.

- a. This image provides a sample configuration for the **Create CMP Identity (OCCM) Certificate** screen.

Figure 18 Create CMP Identity (OCCM) Certificate

The screenshot shows the 'Create Certificate' screen in the OCCM console. The left sidebar has 'Certificate' selected. The main form contains the following fields and values:

- UUID:** (empty)
- Name:** OCCM CA1
- Cert Type:** OCCM
- Network Function:** OCCM
- Purpose:** CMP Client Authentication
- Issuer:** CA1
- Creation Mode:** AUTOMATIC
- Overwrite Secret:** (toggle off)
- Renew Before Expiration (Days):** 14
- Namespace:** (empty)

Note

Select **Issuer** from the **Issuer dropdown** in the **Create Certificate** GUI screen.

- b. This image provides a sample configuration for the **Create End Entity (NF) Certificate** screen.

Figure 19 Private Key Options

The screenshot shows the 'Private Key Options' screen. It contains the following fields and values:

- Key Algorithm:** RSA
- Key Encoding:** PEM
- Key Size:** KEYSIZE_2048
- Namespace:** (empty)
- Secret Name:** (empty)
- Key:** (empty)

- c. This image shows the **Public Key Options** screen configured for the End Entity (NF) certificate.

Figure 20 Public Key Certificate Options

Public Key Certificate Options

Key Usage

Critical: ☒

Value(s):

Extended Key Usage

Critical: ☒

Value(s):

Basic Constraints

Critical: ☒

Value:

- d. This image shows the **Subject and Subject Alternate Names** screen configured for the End Entity (NF) certificate.

Figure 21 Subject and Subject Alternate Name

Subject

Country:

State:

Location:

Organization:

Organization Unit:

Common Name:

Requested Validity (Days):

Subject Alternate Names

Critical: ☒

IP Addresses:

DNS Names:

URI ID API Route:

URI ID URIth:

- e. This image shows the **Certificate Output** screen configured for the End Entity (NF) certificate.

Figure 22 Certificate Output and Certificate Chain Output

Certificate Output

Namespace:

Secret Name:

Key:

Certificate Chain Output

Namespace:

Secret Name:

Key:

Merge Certificate and Certificate Chain: ☐

4. **Create End Entity (NF) Certificate: (PEM encoding):**
The following screenshots provide a sample configuration for creating NF Certificate using CNC Console GUI.
- a. This image provides a sample configuration for the **Create End Entity (NF) Certificate** screen.

Figure 23 Create End Entity (NF) Certificate

Create Certificate

UUID:

Name:

Cert Type:

Network Function:

Purpose:

Issuer:

Creation Mode:

Overwrite Secret: ☐

Option is used to overwrite the existing K8s secret with new certificate

Renew Before Expiration (Days):

Namespace:

- b. This image shows the **Private Key Options** screen configured for the End Entity (NF) certificate.

Figure 24 Private Key Options

Private Key Options

Key Algorithm:

Key Encoding:

Key Size:

Private Key Output

Namespace:

Secret Name:

Key:

- c. This image shows the **Public Key Options** screen configured for the End Entity (NF) certificate.

Figure 25 Public Key Options

Public Key Certificate Options

Key Usage

Critical: ☒

Value(s):

Extended Key Usage

Critical: ☐

Value(s):

Basic Constraints

Critical: ☐

Value:

- d. This image shows the **Subject and Subject Alternate Names** screen configured for the End Entity (NF) certificate.

Figure 26 Subject and Subject Alternate Names

Subject

Country:	Country IN
State:	State KA
Location:	Location BLR
Organizations:	Organization Oracle
Organization Units:	Organization Unit CORU
Common Name:	Common Name *.company.com
Expected Validity (Days):	Expected Validity (Days) 365

Subject Alternate Names

Critical: ☐

IP Address:	IP Addresses: 10.10.10.20 × 10.10.10.21 ×
DNS Names:	DNS Names: *.company.com × *.company.com ×
URI ID API Route:	URI ID API Routes
URI ID URNs:	URI ID URNs: urn:uuid:f3d4fae-7dec-11db-a785-00ac91a6af65 ×

- e. This image shows the **Certificate Output** screen configured for the End Entity (NF) certificate.

Figure 27 Certificate Output

▼ Certificate Output

Namespace:

Namespace
ns1

Secret Name:

Secret Name
nrf-tls-secret

Key:

Key
nrfcert.pem

▼ Certificate Chain Output

Namespace:

Namespace
ns1

Secret Name:

Secret Name
nrf-tls-secret

Key:

Key
nrfcertchain.pem

Merge Certificate and Certificate Chain:

☐

▼ CA Bundle Input

Namespace:

Namespace
ns1

Secret Name:

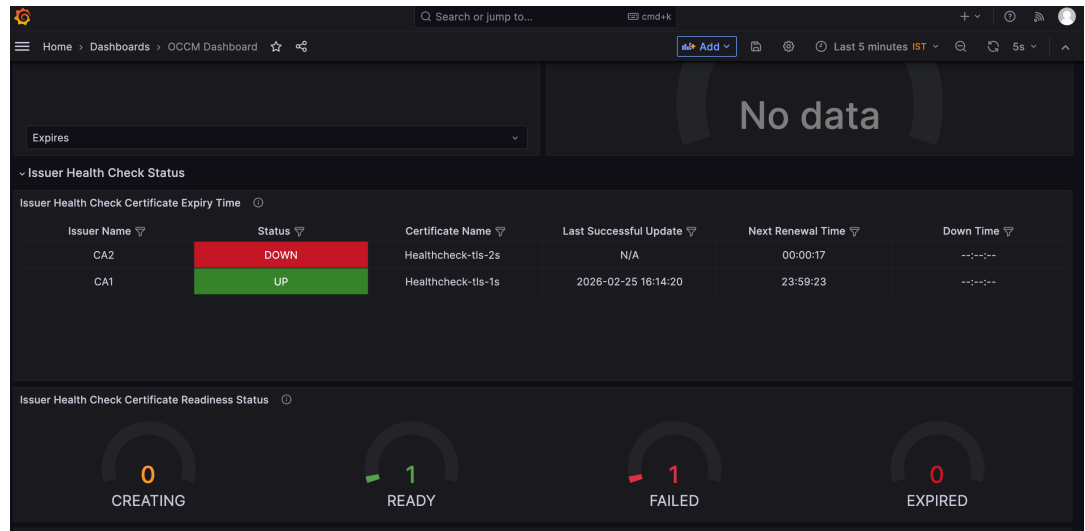
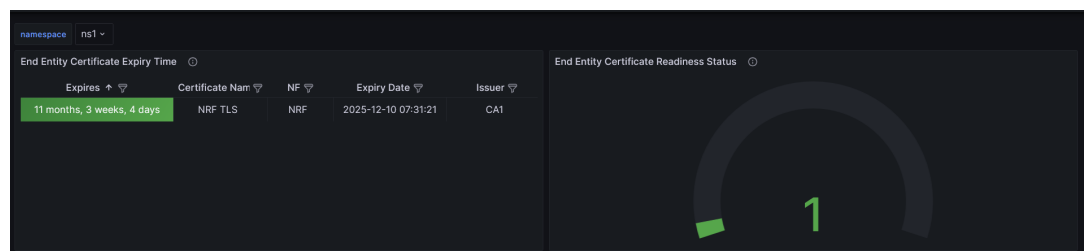
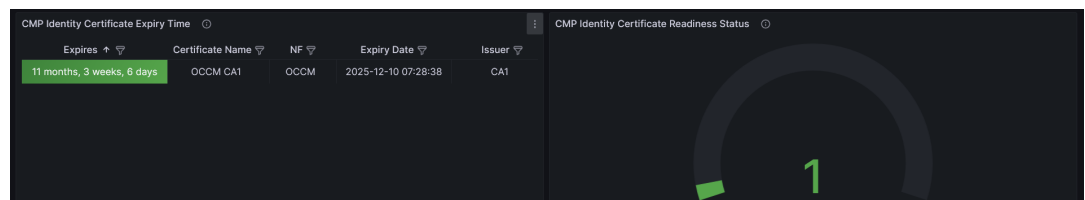
Secret Name
ca-bundle-secret

Key:

Key
ca-bundle.pem

5. Check Grafana Dashboard

Check the Grafana dashboard to view the certificates created.

Figure 28 Health check Sample Grafana Dashboard**Figure 29 End Entity (NF) Sample Grafana Dashboard****Figure 30 CMP Identity (OCCM) Sample Grafana Dashboard**

The End Entity (NF) and CMP Identity (OCCM) samples show that NRF TLS Cert and CA1 certificates are created successfully. The left panel indicates their expiry time and the right panel shows that both are ready to be consumed.

6. Verify Kubernetes Secret

After the certificate request is submitted, verify whether the k8s secret specified under private key output and certificate output location is created or not.

Run the following command to get the content of the Kubernetes secret:

```
kubectl get secret <k8s-secret-name> -n <namespace> -o yaml
```

For example:

```
$ kubectl get secret nrf-tls-secret -n ns1 -o yaml
apiVersion: v1
data:
  nrfcert.pem: LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1XXXXXXX
  nrfcertchain.pem: LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1XXXXXXX
  nrfkey.pem: LS0tLS1CRUdJTiBQUklWQVRFIEtFWS0tLS0tCk1XXXXXXX
kind: Secret
metadata:
  creationTimestamp: "2024-12-10T07:31:21Z"
  name: nrf-tls-secret
  namespace: ns1
  resourceVersion: "563348905"
  uid: f0eb452d-e977-4809-99b0-c541b154dabe
type: Opaque
```

Output of openssl x509 command for the certificate:

```
kubectl get secret <k8s-secret-name> -n <namespace> -o=go-
template='{{index .data "<certificate-output-K8s-secret-key>"}}' | base64 -
d | openssl x509 -text -noout
```

For example:

```
$ kubectl get secret nrf-tls-secret -n ns1 -o=go-template='{{index .data
"nrfcert.pem"}}' | base64 -d | openssl x509 -text -noout
```

Certificate:

```
Data:
  Version: 3 (0x2)
  Serial Number:
    XXXXXXXX
  Signature Algorithm: sha256WithRSAEncryption
  Issuer: CN = x.company.com
  Validity
    Not Before: Dec 10 07:31:22 2024 GMT
    Not After : Dec 10 07:31:21 2025 GMT

  Subject: C = IN, ST = KA, L = BLR, O = Oracle, OU = CGBU, CN =
a.company.com
  Subject Public Key Info:
    Public Key Algorithm: rsaEncryption
    Public-Key: (2048 bit)
    Modulus:
      00:c9:1b:35:bf:21:e6:1f:69:9e:78:25:07:4b:6e:
      XXXXXXXX

    Exponent: 65537 (0x10001)
  X509v3 extensions:
    X509v3 Key Usage:
      Digital Signature
    X509v3 Extended Key Usage:
      TLS Web Client Authentication, TLS Web Server Authentication
    X509v3 Basic Constraints:
```

```

CA:FALSE
X509v3 Subject Alternative Name:
      IP Address:10.10.10.20, IP Address:10.10.10.21,
DNS:y.commpany.com, DNS:z.commpany.com, URI:urn:uuid:f81d4fae-7dec-11d0-
a765-00a0c91e6bf6
X509v3 Subject Key Identifier:
      2B:0D:XXXXXXXXXXXX
X509v3 Authority Key Identifier:
      20:03:XXXXXXXXXXXX
Signature Algorithm: sha256WithRSAEncryption
Signature Value:
XXXXXXXXXXXXXXXXXXXXXXXXXXXX

```

Create NF Certificate (DER encoding):

The following screenshots provide a sample configuration for creating DER encoded NF Certificate using CNC Console GUI.

1. Certificate metadata

Figure 31 Certificate Metadata

The screenshot shows the 'Create Certificate' form in the OCCM console. The left sidebar contains a menu with 'Certificate' selected. The form fields are as follows:

- UUID: (empty field)
- Name: NRF TLS
- Cert Type: OTHER (dropdown menu)
- Network Function: NRF
- Purpose: NRF SBI
- Issuer: CA1 (dropdown menu)
- Creation Mode: AUTOMATIC (dropdown menu)
- Overwrite Secret: ☐ (disabled)
- Renew Before Expiration (Days): 14
- Namespace: (empty field)

2. Private Key Options

Figure 32 Private Key Options

The screenshot shows the 'Private Key Options' form in the OCCM console. The form fields are as follows:

- Key Algorithm: RSA (dropdown menu)
- Key Encoding: DER (dropdown menu)
- Key Size: KEYSIZE_2048 (dropdown menu)
- Namespace: ns1 (dropdown menu)
- Secret Name: nrf-tls-secret
- Key: nrfkey.der

3. Public Key Certificate Options

Figure 33 Public Key Certificate Options

Public Key Certificate Options

Key Usage

Critical:

Value(s):

Value(s)

DIGITAL_SIGNATURE ×

Extended Key Usage

Critical:

Value(s):

Value(s)

CLIENT_AUTH ×

SERVER_AUTH ×

Basic Constraints

Critical:

Value:

Value

END_ENTITY

4. Subject

Figure 34 Subject

Subject

Country:

Country

IN

State:

State

KA

Location:

Location

BLR

Organization:

Organization

Oracle

Organization Unit:

Organization Unit

CGBU

Common Name:

Common Name

a.company.com

Requested Validity (Days):

Requested Validity (Days)

365

5. Subject Alternate names

Figure 35 Subject Alternate names

Subject Alternate Names

Critical:

IP Addresses:

IP Addresses

10.10.10.20 ×

10.10.10.21 ×

DNS Names:

DNS Names

y.company.com ×

z.company.com ×

URI ID API Roots:

URI ID API Roots

URI ID URNs:

URI ID URNs

Certificate Output

Namespace:

Namespace

ns1

Secret Name:

Secret Name

nrf-tls-secret

Key:

Key

nrf.cer

Cloud Native Core, Certificate Management User Guide
G54040-01
Copyright © 2023, 2026, Oracle and/or its affiliates.

July 30, 2026
Page 12 of 34

6. Optional Certificate chain output and CA bundle input fields

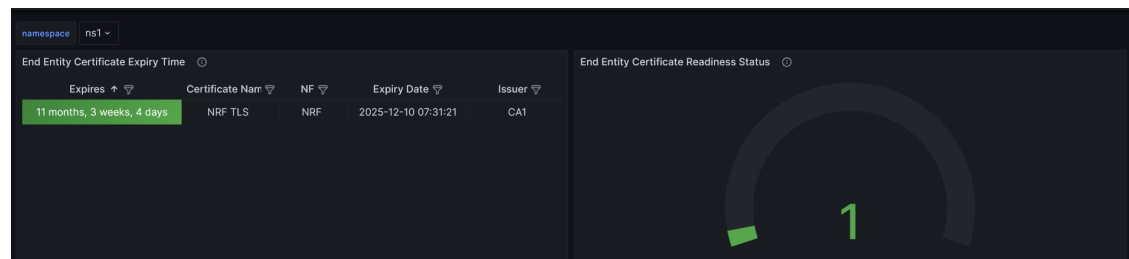
Figure 36 Optional Certificate chain output and CA bundle input fields

The screenshot shows two sections: 'Certificate Chain Output' and 'CA Bundle Input'. Each section has fields for 'Namespace', 'Secret Name', and 'Key'. The 'Certificate Chain Output' section also includes a 'Merge Certificate and Certificate Chain' toggle switch.

Section	Field	Value
Certificate Chain Output	Namespace	Namespace
	Secret Name	Secret Name
	Key	Key
	Merge Certificate and Certificate Chain	☐
CA Bundle Input	Namespace	Namespace
	Secret Name	Secret Name
	Key	Key

Check Grafana dashboard

Figure 37 End Entity (NF) Sample Grafana Dashboard



The screenshot shows that NRF-TLS Certificate is created successfully. The left panel indicates its expiry time and the right panel shows that it is ready to be consumed.

Verify Kubernetes secret

After the certificate request is submitted, verify whether the Kubernetes secret specified under private key output and certificate output location is created or not.

Run the following command to get the content of the Kubernetes secret:

```
kubectl get secret <k8s-secret-name> -n <namespace> -o yaml
```

For example:

```
$ kubectl get secret nrf-tls-secret -n ns1 -o yaml
apiVersion: v1
```

data:

```
  nrf.cer: MIIDrTCCApWgAwIBXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
```

```
  nrfkey.der: MIIEogIBAAKXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
```

```
kind: Secret
```

metadata:

```

creationTimestamp: "2024-12-10T07:31:21Z"

name: nrf-tls-secret

namespace: ns1

resourceVersion: "346496359"

uid: 2bdbb2d7-313d-45d9-a634-642d14f01fa5

type: Opaque

```

Output of openssl x509 command for the certificate:

```

kubectl get secret <k8s-secret-name> -n <namespace> -o=go-
template='{{index .data "<certificate-output-K8s-secret-key>"}}' | base64 -d
| openssl x509 -text -noout -inform DER

```

For example:

```

$ kubectl get secret nrf-tls-secret -n ns1 -o=go-template='{{index .data
"nrf.cer"}}' | base64 -d | openssl x509 -text -noout -inform DER
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            3c:47:05:d7:ee:4c:ce:bb:8f:26:07:c2:a1:9b:92:2c:87:e1:7c:3f
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: CN = x.company.com
        Validity
            Not Before: Dec 10 07:31:22 2024 GMT
            Not After : Dec 10 07:31:21 2025 GMT
        Subject: C = IN, ST = KA, L = BLR, O = Oracle, OU = CGBU, CN =
a.company.com
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            RSA Public-Key: (2048 bit)
            Modulus:
                00:ba:95:23:61:2f:31:55:e3:06:7b:b6:b7:67:cd:
                XXXXXXX
            Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Key Usage: critical
                Digital Signature
            X509v3 Extended Key Usage:
                TLS Web Client Authentication, TLS Web Server Authentication
            X509v3 Basic Constraints:
                CA:FALSE
            X509v3 Subject Alternative Name: critical
                IP Address:10.10.10.20, IP Address:10.10.10.21,
DNS:y.company.com, DNS:z.company.com
            X509v3 Authority Key Identifier:

keyid:FB:4A:01:07:D4:8D:BB:0B:E4:50:72:75:10:8E:81:57:33:66:0D:3E

```

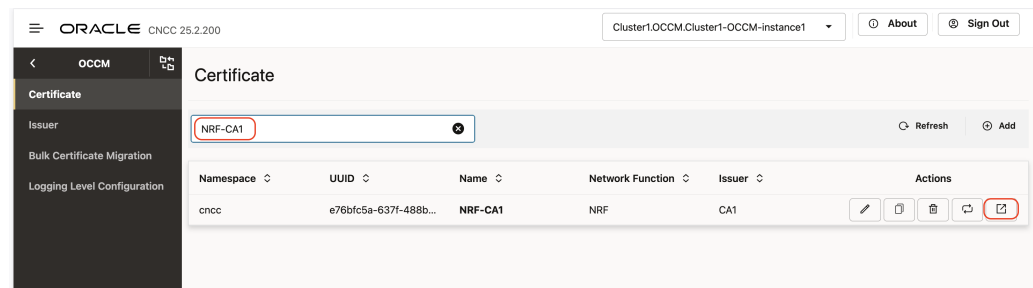
X509v3 Subject Key Identifier:
 A3:82:F6:67:94:35:37:A6:0B:4B:03:9C:0D:B9:A8:72:8D:59:73:85
 Signature Algorithm: sha256WithRSAEncryption
 0a:c2:81:ec:89:91:b4:aa:24:22:33:54:e1:92:db:07:cf:6f:
 XXXXXXXX

A.1.2 Updating Certificates - Sample Configuration

The following procedure provides an example on how to update certificates:

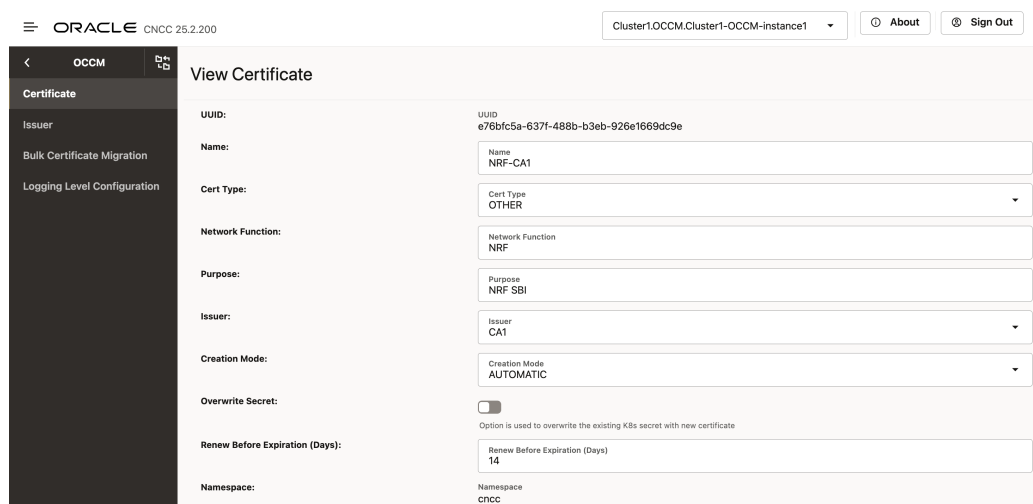
1. Navigate to the **Certificate** page and select the certificate to be edited. View the current configuration before editing.
 - a. This image shows the **View Certificate** screen with the current certificate configuration.

Figure 38 Certificate Page



- b. This image shows the **View Certificate** screen with the current certificate configuration.

Figure 39 View Certificate



- c. This image shows the **View Certificate: Private Key Options** screen with the existing private key configuration.

Figure 40 View Certificate: Private Key Options

Private Key Options

Key Algorithm:	Key Algorithm RSA
Key Encoding:	Key Encoding PEM
Key Size:	Key Size KEYSIZE_2048

Private Key Output

Namespace:	Namespace ns1
Secret Name:	Secret Name nrf-secret-ca-one
Key:	Key nrfkey.pem

- d. This image shows the **View Certificate: Public Key Certificate Options** screen with the existing public key certificate configuration.

Figure 41 View Certificate: Public Key Certificate Options

Public Key Certificate Options

Key Usage

Critical: ☒

Value(s):

Extended Key Usage

Critical: ☐

Value(s):

Basic Constraints

Critical: ☐

Value:

- e. This image shows the **View Certificate: Subject** screen with the existing subject details.

Figure 42 View Certificate: Subject

Subject

Country:	Country IN
State:	State KA
Location:	Location BLR
Organization:	Organization Oracle
Organization Unit:	Organization Unit CGBU
Common Name:	Common Name NRF
Requested Validity (Days):	Requested Validity (Days) 365

- f. This image shows the **View Certificate: Subject Alternate Names (SANs)** screen where the SAN values are missing.

Figure 43 View Certificate: Subject Alternate Names (SANs)

The screenshot shows the 'Subject Alternate Names' configuration page. It includes a 'Critical' checkbox and four input fields: 'IP Addresses', 'DNS Names', 'URI ID API Roots', and 'URI ID URNs'. A red rectangular box is drawn around the 'IP Addresses' and 'DNS Names' fields, indicating they are the focus of the configuration step.

- g. This image shows the **View Certificate: Certificate Output** screen with the existing certificate output configuration.

Figure 44 View Certificate: Certificate Output

The screenshot displays the 'Certificate Output' configuration page. It is divided into three main sections: 'Certificate Output', 'Certificate Chain Output', and 'CA Bundle Input'. Each section contains three fields: 'Namespace' (a dropdown menu), 'Secret Name' (a text input), and 'Key' (a text input). In the 'Certificate Output' section, the values are 'ns1', 'nrf-secret-ca-one', and 'nrfcert.pem' respectively. The 'Certificate Chain Output' section has similar values, with the key being 'nrfcertchain.pem'. A toggle switch for 'Merge Certificate and Certificate Chain' is present and currently turned off.

2. Edit scenarios are applicable based on the field to be updated and whether the certificate (secret) exists in the system. Here, in the example the certificate to be updated is **NRF-CA1** and field to be updated is **SAN (Subject alternate names)**. To check if the certificate exists, see the expiry panel in the Grafana dashboard. If the certificate exists, it will be listed there. The panel displays that the certificate exists in the system.

Figure 45 Grafana Dashboard

Expires	Certificate Name	NF	Expiry Date	Issuer
11 months, 3 weeks, 6 days	NRF-CA1	NRF	2026-05-22 14:59:14	CA1

3. If the certificate (secret) exists in the system, verify its secret content with respect to current configuration before editing. Here in the example, no SAN entry can be seen on the certificate as certificate configuration related to SAN are missing.

Figure 46 Missing SANs

```
$ kubectl get secret nrf-secret-ca-one -o go-template={{index .data "nrfcert.pem"}} | base64 -d | openssl x509 -text -noout

Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            A:B:C:D:XX
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: UID = xxx, CN = My CA, O = yyy
        Validity
            Not Before: May 22 14:59:14 2025 GMT
            Not After : May 22 14:59:14 2026 GMT
        Subject: CN = NRF, OU = CGBU, O = Oracle, L = BLR, ST = KA, C = IN
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            Public-Key: (2048 bit)
            Modulus:
                pqrs:XX
                Fennosee: A5537 (0x10000)
            X509v3 extensions:
                X509v3 Key Usage: critical
                Digital Signature
                X509v3 Extended Key Usage:
                TLS Web Client Authentication, TLS Web Server Authentication
                X509v3 Basic Constraints:
                CA:FALSE
                X509v3 Authority Key Identifier:
                xyz:XX
                X509v3 Subject Key Identifier:
                mno:XX
        Signature Algorithm: sha256WithRSAEncryption
        Signature Value:
            12:33:34:45:XXX
```

4. Edit the certificate configuration field(s) and click **Save**.

Figure 47 Certificate

The screenshot shows the Oracle Cloud Native Core (CNCC) Certificate Management interface. The sidebar on the left has a 'Certificate' section. The main area displays a table of certificates. The first row shows a certificate with Namespace 'cncc', UUID 'e76bfc5a-637f-488b...', Name 'NRF-CA1', Network Function 'NRF', and Issuer 'CA1'. The 'Actions' column for this row contains a red 'Edit' icon (pencil) and other icons for delete, refresh, and add.

Figure 48 Edit Certificate

ORACLE CNCC 25.2.200 Cluster1.OCCM.Cluster1-OCCM-instance1 About Sign Out

< OCCM Edit Certificate

Certificate

Issuer

Bulk Certificate Migration

Logging Level Configuration

UUID: e76bfc5a-637f-488b-b3eb-926e1669dc9e

Name: NRF-CA1

Cert Type: OTHER

Network Function: NRF

Purpose: NRF-SBI

Issuer: CA1

Creation Mode: AUTOMATIC

Overwrite Secret: ☐ Option is used to overwrite the existing KBs secret with new certificate

Renew Before Expiration (Days): 14

Namespace: cncc

Figure 49 Edit Certificate: Private Key Options

Private Key Options

Key Algorithm: RSA

Key Encoding: PEM

Key Size: KEYSIZE_2048

Private Key Output

Namespace: ns1

Secret Name: nrf-secret-ca-one

Key: nrfkey.pem

Figure 50 Edit Certificate: Public Key Certificate Options

Public Key Certificate Options

Key Usage

Critical: ☒

Value(s): DIGITAL_SIGNATURE

Extended Key Usage

Critical: ☐

Value(s): CLIENT_AUTH SERVER_AUTH

Basic Constraints

Critical: ☐

Value: END_ENTITY

Figure 51 Edit Certificate: Subject

The 'Subject' configuration form contains the following fields and values:

Field	Value
Country:	IN
State:	KA
Location:	BLR
Organization:	Oracle
Organization Unit:	Organization Unit CGBU
Common Name:	Common Name NRF
Requested Validity (Days):	Requested Validity (Days) 365

Figure 52 Edit Certificate: Subject ALternate Names (SANs)

The 'Subject Alternate Names' configuration form includes the following fields:

Field	Value
Critical:	☒
IP Addresses:	IP Addresses
DNS Names:	DNS Names
URI ID API Roots:	URI ID API Roots
URI ID URNs:	URI ID URNs

Figure 53 Updated IP address and DNS Names

The 'Subject Alternate Names' configuration form is shown with updated values. A red box highlights the 'IP Addresses' and 'DNS Names' fields, which now contain '127.0.0.1 X' and 'localhost X' respectively.

Field	Value
Critical:	☒
IP Addresses:	IP Addresses 127.0.0.1 X
DNS Names:	DNS Names localhost X
URI ID API Roots:	URI ID API Roots
URI ID URNs:	URI ID URNs

Figure 54 Click Save

▼ Certificate Output

Namespace:

Secret Name:

Key:

▼ Certificate Chain Output

Namespace:

Secret Name:

Key:

Merge Certificate and Certificate Chain: ☐

▼ CA Bundle Input

Namespace:

Secret Name:

Key:

Save **Cancel**

5. The updated configuration appears when you view the edited certificate.

Figure 55 Certificate Page

ORACLE CNCC 25.2.200 Cluster1.OCCM.Cluster1-OCCM-instance1 About Sign Out

Certificate

Issuer:

Refresh Add

Namespace	UUID	Name	Network Function	Issuer	Actions
cncc	e76bfc5a-637f-488b...	NRF-CA1	NRF	CA1	

Figure 56 View Certificate

ORACLE CNCC 25.2.200 Cluster1.OCCM.Cluster1-OCCM-instance1 About Sign Out

View Certificate

UUID: e76bfc5a-637f-488b-b3eb-926e1669dc9e

Name:

Cert Type:

Network Function:

Purpose:

Issuer:

Creation Mode:

Overwrite Secret: ☐

Option is used to overwrite the existing KBs secret with new certificate

Renew Before Expiration (Days):

Namespace:

Figure 57 View Certificate: Private Key Options

Private Key Options

Key Algorithm: RSA

Key Encoding: PEM

Key Size: KEYSIZE_2048

Private Key Output

Namespace: ns1

Secret Name: nrf-secret-ca-one

Key: nrfkey.pem

Figure 58 View Certificate: Public Key Certificate Options

Public Key Certificate Options

Key Usage

Critical: ☒

Value(s): DIGITAL_SIGNATURE ✕

Extended Key Usage

Critical: ☐

Value(s): CLIENT_AUTH ✕ SERVER_AUTH ✕

Basic Constraints

Critical: ☐

Value: END_ENTITY

Figure 59 View Certificate: Subject

Subject

Country: IN

State: KA

Location: BLR

Organization: Oracle

Organization Unit: CGBU

Common Name: NRF

Requested Validity (Days): 365

Figure 60 Updated Subject Alternate Names (SANs)

Subject Alternate Names

Critical: ☒

IP Addresses:

DNS Names:

URI ID API Roots:

URI ID URNs:

Figure 61 View Certificate: Certificate Output

Certificate Output

Namespace:

Secret Name:

Key:

Certificate Chain Output

Namespace:

Secret Name:

Key:

Merge Certificate and Certificate Chain: ☐

CA Bundle Input

Namespace:

Secret Name:

Key:

6. If the field update triggers recreation, verify that certificate recreation with updated configuration is successful. See the Grafana dashboard to verify that the certificate expiry has been extended, the certificate status is ready, and the secret content has been updated as per the new configuration. In the following screenshot, the certificate expiry has been extended as shown in the End Entity Certificate Expiry Time panel.

Figure 62 End Entity Certificate Expiry Time panel

Expires ↑	Certificate Nam	NF	Expiry Date	Issuer
11 months, 3 weeks, 6 days	NRF-CA1	NRF	2026-05-28 07:35:29	CA1

Verify that the secret content is updated as per the edited configuration.

Figure 63 Secret Updated with SAN

```

$ kubect| get secret nci-secret-ca-one -o go-template='[{{index .data "ncicert.pem"}}] |
base64 -d | openssl x509 -text -noout

Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            A:B:C:D:XX
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: UID=xxx, CN = My CA, O = yyy
        Validity
            Not Before: May 28 07:35:29 2025 GMT
            Not After: May 28 07:35:29 2026 GMT
        Subject: CN = NAF, OU = CGDU, O = Oracle, L = BLR, ST = KA, C = IN
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            Public-Key: (2048 bit)
            Module:
                [qr]=XX
            Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Key Usage: critical
                Digital Signature
            X509v3 Extended Key Usage:
                TLS Web Client Authentication, TLS Web Server Authentication
            X509v3 Basic Constraints:
                CA:FALSE
            X509v3 Subject Alternative Name: critical
                IP Address: 177.0.0.1, DNS:localhost
            X509v3 Authority Key Identifier:
                xyz:XX
            X509v3 Subject Key Identifier:
                mcnz:XX
        Signature Algorithm: sha256WithRSAEncryption
        Signature Value:
            12:23:34:45:XX

```

Note

If certificate recreation fails,

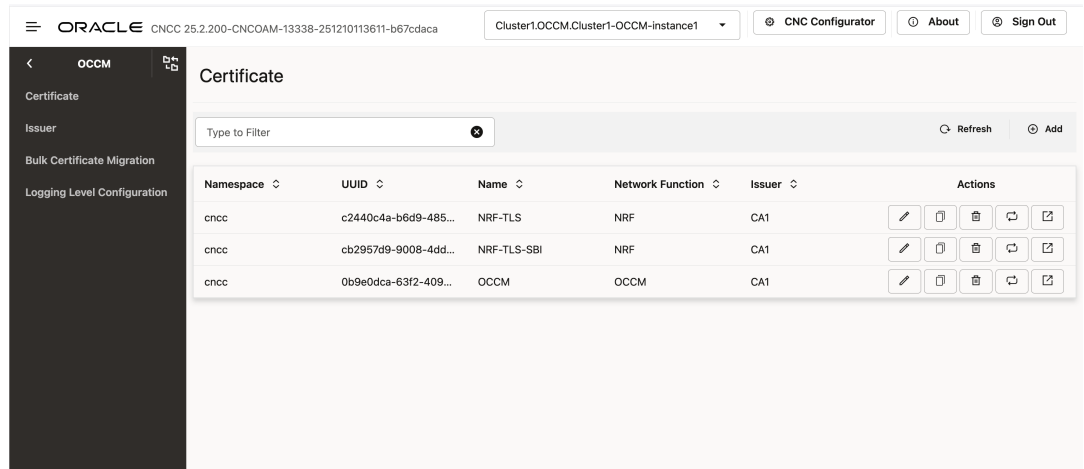
- OCCM will continue to reattempt certificate recreation with the updated configuration.
- Certificate expiry will not be extended.
- Certificate status will be set to FAILED.
- Certificate Errors table in the Grafana dashboard will list the certificate and the cause of failure.

A.1.3 Recreating Certificates- Sample Configuration

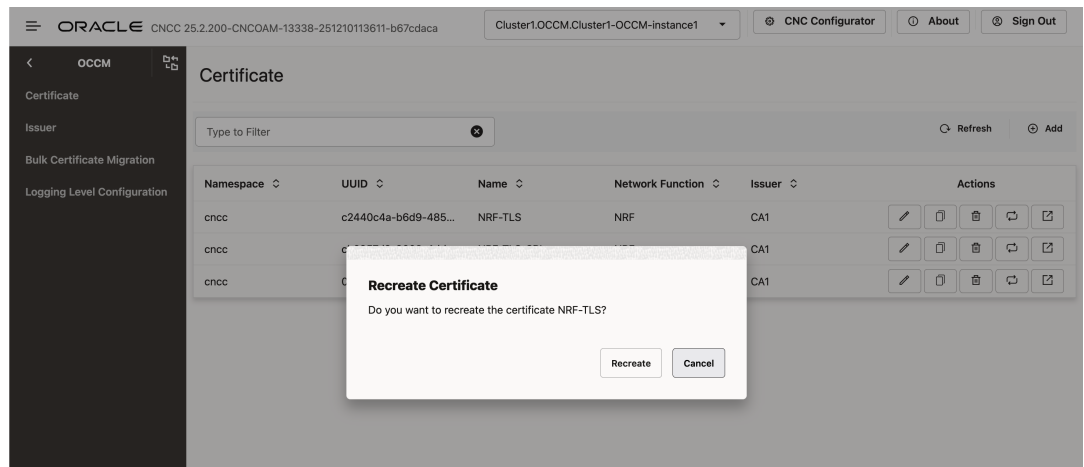
This section describes the procedure to recreate a certificate.

To recreate certificates:

1. Log in to CNC Console using your login credentials and select the OCCM Instance.
2. Click **OCCM** from the left pane and then click **Certificate**.
3. Click the **Recreate** button next to the certificate that you want to recreate.

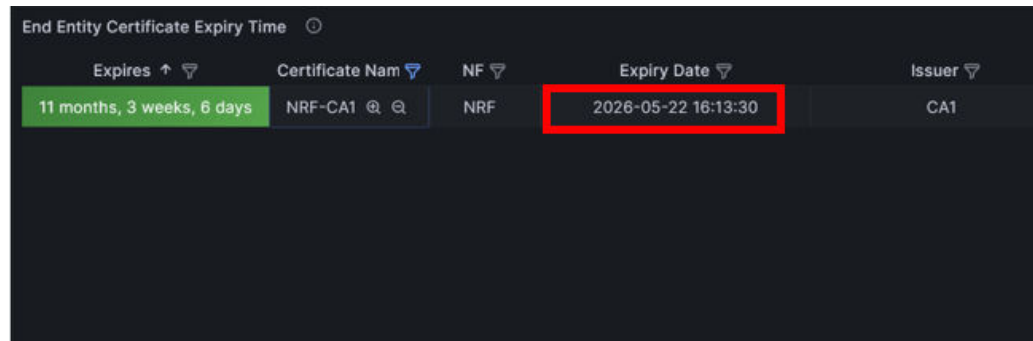
Figure 64 Recreate Icon

The **Recreate Certificate** dialog box appears.

Figure 65 Recreate Certificate

4. Click **Recreate** to proceed. Click **Cancel** to discard your progress and close the dialog box.
5. Check the Grafana dashboard to view the recreated certificate.
 - a. Check the '**End Entity Certificate Expiry Time**' panel to verify that the certificate expiry date has extended after successful recreation. In case the recreation fails, the certificate expiry date will not be extended and will display the remaining validity.

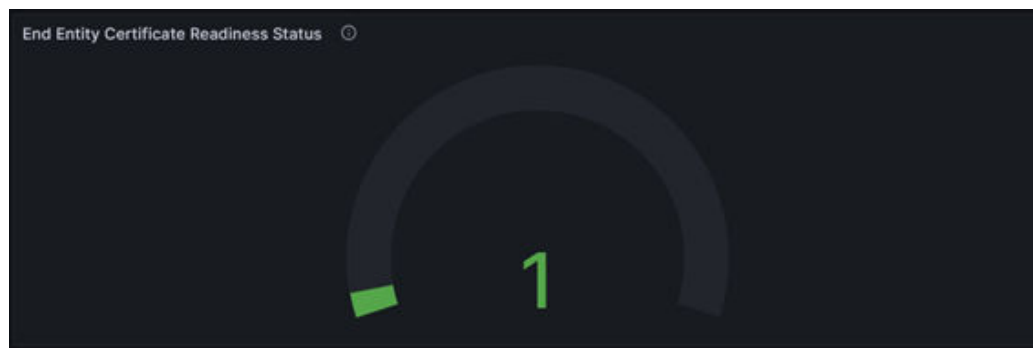
Figure 66 End Entity Certificate Expiry Time



Expires	Certificate Name	NF	Expiry Date	Issuer
11 months, 3 weeks, 6 days	NRF-CA1	NRF	2026-05-22 16:13:30	CA1

- b. Check the **End Entity Certificate Readiness Status** panel to verify the certificate status. If recreation succeeds, the status will be set to **READY**, otherwise the status will be set to **FAILED**.

Figure 67 End Entity Certificate Readiness Status



6. Optionally, verify that the certificate validity has been extended by checking the content of the Kubernetes secret corresponding to the configuration. Run the following command to verify the Kubernetes secret:

```
$ kubectl get secret nrf-secret-ca-one -o=go-template='{{index .data "nrfcert.pem"}}' -n ns1 | base64 -d | openssl x509 -text -noout
```

A sample response is as follows:

Figure 68 Secret with Extended Validity

```

$ kubectl get secret nri-secret-ca-one -o go-template='{{index .data "nricert.pem"}}' -n nri |
base64 -d | openssl x509 -text -noout

Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            A:B:C:D:XX
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: UID = xxx, CN = My CA, O = yyy
        Validity
            Not Before: May 22 16:19:30 2025 GMT
            Not After: May 22 16:19:30 2026 GMT
        Subject: CN = MSP, OU = CGBU, O = Oracle, L = BLR, ST = KA, C = IN
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            Public Key: (2048 bit)
            Module:
                pqrs:XX
            Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Key Usage: critical
            Digital Signature
            X509v3 Extended Key Usage:
            TLS Web Client Authentication, TLS Web Server Authentication
            X509v3 Basic Constraints:
            CA:FALSE
            X509v3 Authority Key Identifier:
            xyz:XX
            X509v3 Subject Key Identifier:
            mno:XX
        Signature Algorithm: sha256WithRSAEncryption
        Signature Value:
            12:34:56:XX

```

A.1.4 Initiating Bulk Certificate Migration - Sample Configuration

The following steps provides an example on how to initiate bulk migration of certificates from source issuer CA1 to destination issuer CA2:

1. Create the destination issuer CA2 with the requisite configuration. For more information about the steps, see [Managing Issuers](#).
 - a. This image provides a sample configuration for the **Create Issuer** screen used to create the destination issuer CA2.

Figure 69 Create Issuer

Create Issuer	
UUID:	UUID
Name:	Name CA2
Server URL:	Server URL https://ca2-mockca.svc.ns1:30443/cmpallias
Recipient Distinguished Name:	Recipient Distinguished Name /CN=X CA
Issuer Distinguished Name:	Issuer Distinguished Name /CN=X CA
Total Timeout (Seconds):	Total Timeout (Seconds) 720
Message Timeout (Seconds):	Message Timeout (Seconds) 120

- b. This image provides a sample configuration for the **Create Issuer** screen used to create the destination issuer CA2.

Figure 70 Initial CMP Client (OCCM) Authentication Options

Initial CMP Client(OCCM) Authentication Options

Type:

Type
MAC

MAC Algorithm:

MAC Algorithm
HMACSHA256

MAC Authentication Input

Namespace:

Namespace
ns1

Secret Name:

Secret Name
ca1-occm-mac-secret

Password Key:

Password Key
pwd

Reference Key:

Reference Key
ref

- c. This image shows the **MAC Authentication Input** screen configured for CMP client authentication.

Figure 71 MAC Authentication Input

Initial CMP Client(OCCM) Authentication Options

Type:

Type
MAC

MAC Algorithm:

MAC Algorithm
HMACSHA256

MAC Authentication Input

Namespace:

Namespace
ns1

Secret Name:

Secret Name
ca1-occm-mac-secret

Password Key:

Password Key
pwd

Reference Key:

Reference Key
ref

- d. This image shows the **CMP Client Authentication Options For Other Certificate** screen configured for additional certificate authentication.

Figure 72 CMP Client Authentication Options For Other certificate

CMP Client Authentication Options For Other certificate

Type:

Type
SIGNATURE

Digest Algorithm:

Digest Algorithm
SHA256

Signature Authentication Input

Namespace:

Namespace
ns1

Secret Name:

Secret Name
ca2-cmp-identity-secret

Key:

Key
cmpkey.pem

Cert:

Cert
cmpcert.pem

Extra Certs:

Extra Certs

- e. This image shows the **OCCM Trust-Store Secret Input** screen configured with the trust-store secret details.

Figure 73 Occm Trust-Store Secret Input

Occm Trust-Store Secret Input

Namespace:

Namespace

ns1

Name:

Name

ca2-occm-trust-store-secret

Root CA Certs:

Root CA Certs

caroot.pem x

Intermediate CA Certs:

Intermediate CA Certs

Server Cert:

Server Cert

TLS Config

Enable TLS:

TLS Trust-Store Secret Input

Namespace:

Namespace

ns1

Name:

Name

ca2-tls-trust-store-secret

TLS Trusted Certs:

TLS Trusted Certs

tlscert.pem x

2. Create CMP (OCCM) Identity certificate OCCM-CA2 corresponding to the destination issuer CA2. For more information about the steps, see [Managing Certificates](#).
 - a. This image provides a sample configuration for the **Create Certificate** screen used to create the CMP (OCCM) Identity certificate OCCM-CA2.

Figure 74 Create Certificate

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

OC

Figure 75 Private Key

Private Key Output

Namespace:

Secret Name:

Key:

Public Key Certificate Options

Key Usage

Critical: ☒

Value(s):

Extended Key Usage

Critical: ☐

Value(s):

Basic Constraints

Critical: ☐

Value:

- c. This image shows the **Subject** screen configured for the OCCM-CA2 identity certificate.

Figure 76 Subject

Subject

Country:

State:

Location:

Organization:

Organization Unit:

Common Name:

Requested Validity (Days):

Subject Alternate Names

Critical: ☒

IP Addresses:

DNS Names:

URI ID API Base:

URI ID URNs:

- d. This image shows the **Certificate Output** screen configured for the OCCM-CA2 identity certificate.

Figure 77 Certificate Output

Certificate Output

Namespace:

Secret Name:

Key:

Certificate Chain Output

Namespace:

Secret Name:

Key:

Merge Certificate and Certificate Chain: ☐

3. Initiate the bulk certificate migration by choosing the source issuer as CA1 and destination issuer as CA2 from the drop downs list. Other fields must be empty.
 - a. Log in to CNC Console using your login credentials and select the OCCM Instance.
 - b. Click **OCCM** from the left pane and then click **Bulk Certificate Migration**.
 - c. The Bulk Certificate Migration page appears. Click **Add**. The Initiate Bulk Certificate Migration page appears.
 - d. Choose the source and destination issuer from the drop-down list and click on **Start Certificate Migration**.

Figure 78 Initiate Bulk Certificate Migration

The screenshot shows the 'Initiate Bulk Certificate Migration' page. On the left is a sidebar with 'OCCM' selected, and sub-items 'Certificate', 'Issuer', 'Bulk Certificate Migration' (highlighted), and 'Logging Level Configuration'. The main area has the title 'Initiate Bulk Certificate Migration'. It contains:

- An 'ID' field.
- A 'Source Issuer' dropdown menu.
- A 'Destination Issuer' dropdown menu that is open, showing 'Please Select', 'CA1', and 'CA2'.
- A section 'Bulk Certificate Migration Initiated Certificates' with a 'Cert Name' dropdown and 'No data to display.' below it.
- A section 'Bulk Certificate Migration Skipped Certificates' with a table header 'Cert Name' and 'Cert UUID', and 'No data to display.' below it.
- At the bottom right, 'Start Certificate Migration' and 'Cancel' buttons.

- e. A configuration is created and displayed as follows. This configuration can be viewed to know on which certificates migration was triggered and if it is skipped on any certificate.

Figure 79 Bulk Certificate Migration

The screenshot shows the 'Bulk Certificate Migration' page. On the left is the same sidebar as Figure 78, with 'Bulk Certificate Migration' highlighted. The main area has the title 'Bulk Certificate Migration'. It contains:

- A search bar 'Type to Filter' with a magnifying glass icon.
- 'Refresh' and 'Add' buttons.
- A table with the following columns: 'ID', 'Source Issuer', 'Destination Issuer', and 'Actions'.
- One row in the table:

ID	Source Issuer	Destination Issuer	Actions
6147ffa7-8914-40da-9dce-15cd8d9bea55	CA1	CA2	[Delete] [Edit]

4. View the configuration to check the certificates on which migration is initiated.

Figure 80 View Bulk Certificate Migration

View Bulk Certificate Migration

ID: 6147ffa7-8914-40da-9dcc-15cd8d9bea55

Source Issuer: CA1

Destination Issuer: CA2

Bulk Certificate Migration Initiated Certificates:

Cert Name	Cert UID
NRF-TLS-2	2d5bad6f-cfc4-4cbf-af1f-c754310f18b0
NRF-TLS-3	960ad503-1f03-45ee-bc55-96d15c105760
NRF-TLS-1	9b26f73b-8374-4871-93c4-afe5bb00e931

Bulk Certificate Migration Skipped Certificates:

Cert Name	Cert UID
No data to display.	

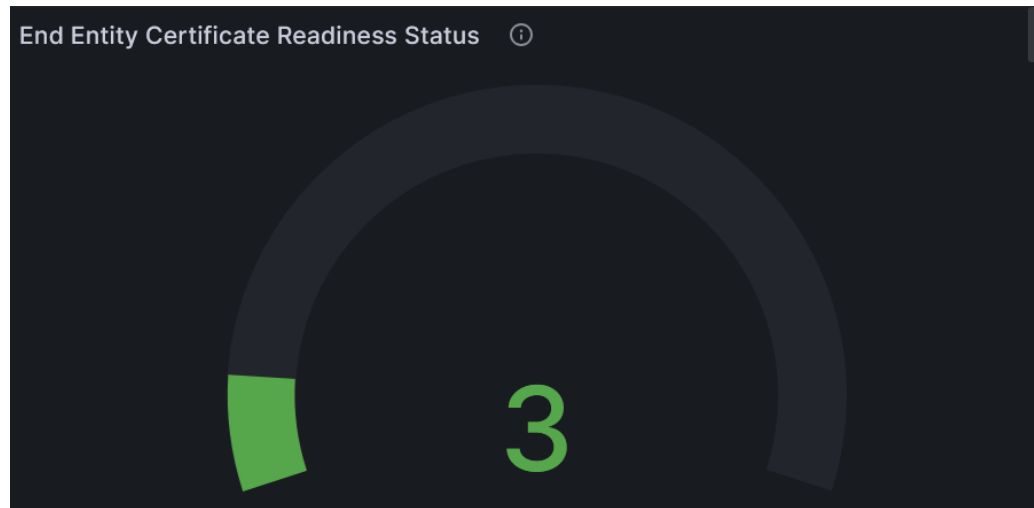
5. Check the Grafana dashboard to verify the migration status of each certificate displayed under the field *Bulk Certificate Migration Initiated certificates*:
 - a. When the migration is successful on a certificate.
 - To verify the migration status for each certificate refer to the *End Entity Certificate Expiry Time* and the *End Entity Certificate Readiness Status* panels .
 - On the *End Entity Certificate Expiry Time* the corresponding issuer of the certificate considered in bulk migration is updated to the destination issuer (CA2) and the expiry date will be extended based on the configuration.
 - On the *End Entity Certificate Readiness Status* the certificate will be in the READY state.

In the following example, the bulk migration is triggered on three certificates. The updated issuer, expiry, and the readiness status are shown as follows:

Figure 81 End Entity Certificate Expiry Time

End Entity Certificate Expiry Time

Expires	Certificate Name	NF	Expiry Date	Issuer
11 months, 3 weeks, 6 days	NRF-TLS-1	NRF	2025-12-19 08:32:...	CA2
11 months, 3 weeks, 6 days	NRF-TLS-3	NRF	2025-12-19 08:32:...	CA2
11 months, 3 weeks, 6 days	NRF-TLS-2	NRF	2025-12-19 08:32:...	CA2

Figure 82 End Entity Certificate Readiness Status

b. When the migration fails on a certificate.

- The corresponding issuer of the certificate will NOT be updated to the destination issuer on the *End Entity Certificate Expiry Time* panel and it will keep pointing to the source issuer (CA1). The expiry will not be updated since recreation has failed. Similarly, the end entity certificate readiness status will also preserve the previous status. The failed migration on all the three certificates are as follows:

Figure 83 End Entity Certificate Expiry Time Failed

End Entity Certificate Expiry Time ⓘ				
Expires ↑ ▾	Certificate Name ▾	NF ▾	Expiry Date ▾	Issuer ▾
11 months, 3 weeks, 6 days	NRF-TLS-1	NRF	2025-12-19 08:49:16	CA1
11 months, 3 weeks, 6 days	NRF-TLS-2	NRF	2025-12-19 08:49:16	CA1
11 months, 3 weeks, 6 days	NRF-TLS-3	NRF	2025-12-19 08:49:16	CA1

- The *Bulk Cert Migrations Error* panel will have an entry of the certificate whose migration has failed. To check for a given migration, you must filter on the Bulk Migration UUID field. The issuer displayed is the configured destination issuer. The failed migration on all the three certificates for the given bulk migration are as follows.

Figure 84 Bulk Cert Migrations Error

Bulk Cert Migrations Error ⓘ			
Cert Name ▾	Cert UUID ▾	Issuer ▾	Bulk Migration UUID ▾
NRF-TLS-3	88580d83-4582-4c6a-948b-5f12b9c1ce56	CA2	f7f97318-2027-4a4c-b39a-d583545b1a1f
NRF-TLS-2	b8fe1262-ff5f-4743-b60b-89f8dabb6376	CA2	f7f97318-2027-4a4c-b39a-d583545b1a1f
NRF-TLS-1	4f324dd7-e08f-4d0a-9e9d-bb365f0e0b6d	CA2	f7f97318-2027-4a4c-b39a-d583545b1a1f

- Logs must be checked to find out the reason of the failure and corrective action must be taken.

- Another migration can be initiated to retry for all the failed certificates from the source to the destination issuer.