Oracle® Communications
Cloud Native Core, Certificate Management User Guide

Release 25.1.100
G18470-02
April 2025

Oracle Communications Cloud Native Core, Certificate Management User Guide, Release 25.1.100

G18470-02

Contents

Preface

Documentation Accessibility

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

Access to Oracle Support

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

Diversity and Inclusion

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

Conventions

The following text conventions are used in this document:

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

My Oracle Support

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

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

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

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

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

Acronyms

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

Table Acronyms

Acronym	Description
3GPP	3rd Generation Partnership Project
API	Application Programming Interface
CCA	Client Credentials Assertions
CMP	Certificate Management Protocol
CNC	Cloud Native Core
CNC Console	Cloud Native Configuration Console
OCCM	Oracle Communications Certificate Management
CA	Certification Authority is a trusted entity that issues Secure Sockets Layer (SSL) certificates. CAs are also called issuer in this document.
DNS	Domain Name Server
EE	End Entity
ECC	Elliptic Curve Cryptography
HSM	Hardware Security Module
IDP	Identity Provider
IR	Initialization Requests
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
CMP Identity Key	Private Key used by Certificate Management to sign the CMPv2 requests and establish trust between Certificate Management and CA.
CMP Identity Certificate	Certificate that corresponds to and certifies the CMP Identity Key. It is included in the CMPv2 requests for authentication by CA.

What's New in This Guide

This section introduces the documentation updates for release 25.1.1xx.

Release 25.1.100 - G18470-02, May 2025

Relocated the Preface section for improved document structure and readability.

Release 25.1.100 - G18470-01, April 2025

- **Feature Updates:**
  - **New Features:**
    - **Bulk Migration of Certificates**
      - Added the Bulk Migration of Certificates section to describe the feature.
      - Updated the Updating Issuer section to add information to use bulk certificate migration to update the end point of the CA and Issuer.
      - Added the Bulk Certificate Migration API Access section to access bulk certificate migration APIs.
      - Added the Bulk Certificate Migration Error section for the list of certificates that are failed during bulk migration of certificate.
      - Updated the following KPIs to add bulk certificate migration expression:
        - Configuration Requests
        - Configuration Responses
- **General Updates:**
  - Updated the images in Deleting Issuers section.
  - Updated Integration with Certificate Authority section to add CMPv2 Initialization Request (IR) and CMPv2 Key Update Request (KUR).
  - Updated the image in MAC Based Trust Establishment section.
  - Updated the Certificate Based Trust Establishment section to add information about the End Entity certificate initialization requests.
  - Added information about renewing CMP Identity and End Entity certificates in the Monitoring Certificate Expiry and Renewing Certificates section.
  - Added the Renewing CMP Identity (OCCM) Certificate and Renewing End Entity (NF) Certificate sections to renew CMP identity and End Entity certificates.
  - Updated the images in Deleting the Certificate Configuration section.
  - Updated the Table 4-1 in Introducing OCCM in an Existing NF Deployment section.
  - Added the following metrics for CMP Identity (OCCM) and End Entity (NF):
    - occm_cmp_identity_cert_expiration_seconds
    - occm_end_entity_cert_expiration_seconds
    - occm_cmp_identity_cert_status
    - occm_end_entity_cert_status
  - The following metrics are removed:

    * `occm_cert_expiry`
    * `occm_cert_status`
    * `occm_secret_event_total`
- Added the following alerts for CMP Identity (OCCM) and End Entity (NF):
    * OccmCmpIdentityCertExpirationMinor
    * OccmCmpIdentityCertExpirationMajor
    * OccmCmpIdentityCertExpirationCritical
    * OccmCmpIdentityCertExpired
    * OccmEndEntityCertExpirationMinor
    * OccmEndEntityCertExpirationMajor
    * OccmEndEntityCertExpirationCritical
    * OccmEndEntityCertExpired
- The following alerts are removed:
    * `OccmCertExpiryWithinMinorThreshold`
    * `OccmCertExpiryWithinMajorThreshold`
    * `OccmCertExpiryWithinCriticalThreshold`
    * `OccmCertExpired`
    * `OccmRenewBeforValidityCritical`
    * `OccmCertConfigDeletion`
- Added the following KPIs for CMP Identity (OCCM) and End Entity (NF):
    * CMP Identity (OCCM) Certificate Expiry Time
    * End Entity (NF) Certificate Expiry Time
    * CMP Identity (OCCM) Certificate Readiness Status
    * End Entity (NF) Certificate Readiness Status
- The following KPIs are removed:
    * `Certificate Expiry Time`
    * `Certificate Readiness Status`
- Updated the `occm_cmp_cli_durations_seconds` metric name to add seconds.
- Updated the images and payload in Creating End Entity (NF) Certificate Using OCCM - Sample Configuration section.

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.

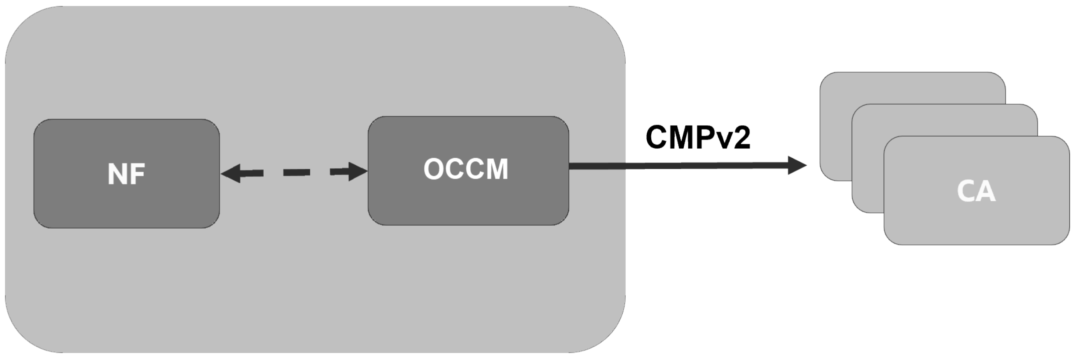
As 3GPP recommends using separate certificates based on the client or server mode and the type of workflow, it leads to many certificates in the network. Automated certificate management eliminates any possibilities of network disruption due to expired certificates. In SBA network deployments, the Network 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:

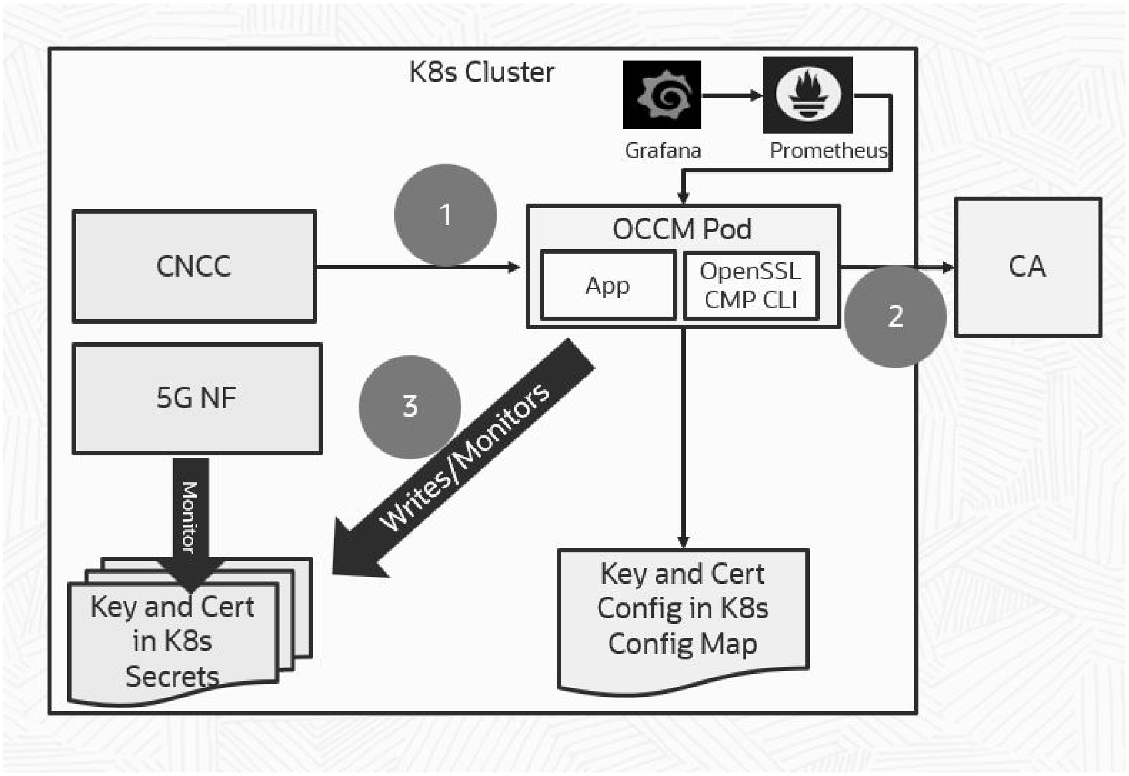
- *Oracle Communications Cloud Native Core, Certificate Management Installation, Upgrade, and Fault Recovery Guide*
- *Oracle Communications Cloud Native Core, Certificate Management Troubleshooting Guide*
- *Oracle Communications Cloud Native Core, Certificate Management REST Specification Guide*
- *Oracle Communications Cloud Native Core, Security Guide*
- *Oracle Communications Cloud Native Core, Solution Upgrade Guide*

2

OCCM Architecture

OCCM is a Cloud Native application consisting of a single microservice. OCCM is packaged and delivered as a CSAR or 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 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.

3.1.1 Establishing Initial Trust Between OCCM and CA

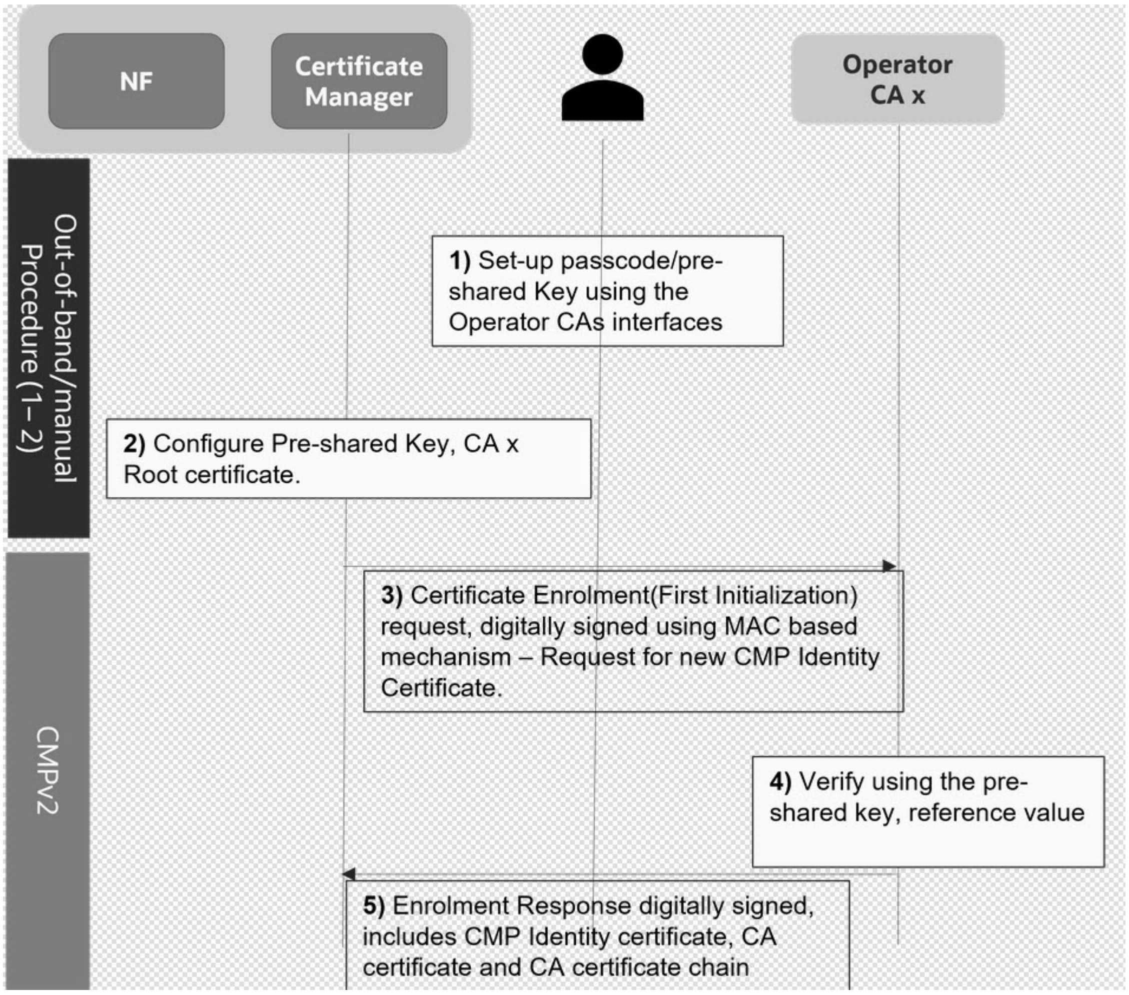
OCCM can be configured to establish trust between Oracle Communication Certificate (OCCM) and Certificate Authorities (CAs) by enabling PKI message protection in the following ways:

- MAC based trust establishment
- Certificate based trust establishment

3.1.1.1 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

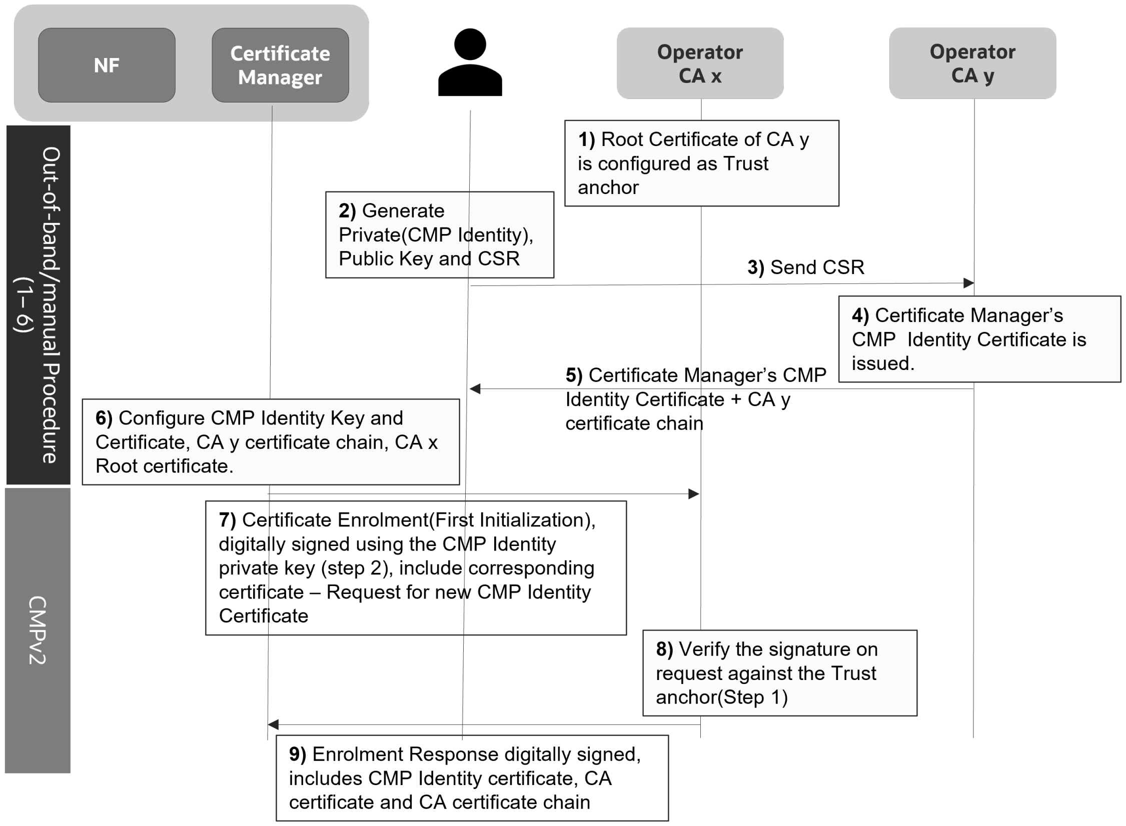

NF
Certificate Manager
Operator CA x
Out-of-band/manual Procedure (1– 2)
CMPv2
1) Set-up passcode/pre-shared Key using the Operator CAs interfaces
2) Configure Pre-shared Key, CA x Root certificate.
3) Certificate Enrolment(First Initialization) request, digitally signed using MAC based mechanism – Request for new CMP Identity Certificate.
4) Verify using the pre-shared key, reference value
5) Enrolment Response digitally signed, includes CMP Identity certificate, CA certificate and CA certificate chain

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.

3.1.1.2 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


NF
Certificate Manager
Operator CA x
Operator CA y
Out-of-band/manual Procedure (1 – 6)
CMPv2
1) Root Certificate of CA y is configured as Trust anchor
2) Generate Private(CMP Identity), Public Key and CSR
3) Send CSR
4) Certificate Manager's CMP Identity Certificate is issued.
5) Certificate Manager's CMP Identity Certificate + CA y certificate chain
6) Configure CMP Identity Key and Certificate, CA y certificate chain, CA x Root certificate.
7) Certificate Enrolment(First Initialization), digitally signed using the CMP Identity private key (step 2), include corresponding certificate – Request for new CMP Identity Certificate
8) Verify the signature on request against the Trust anchor(Step 1)
9) Enrolment Response digitally signed, includes CMP Identity certificate, CA certificate and CA certificate chain

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.

3.2 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 if 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.

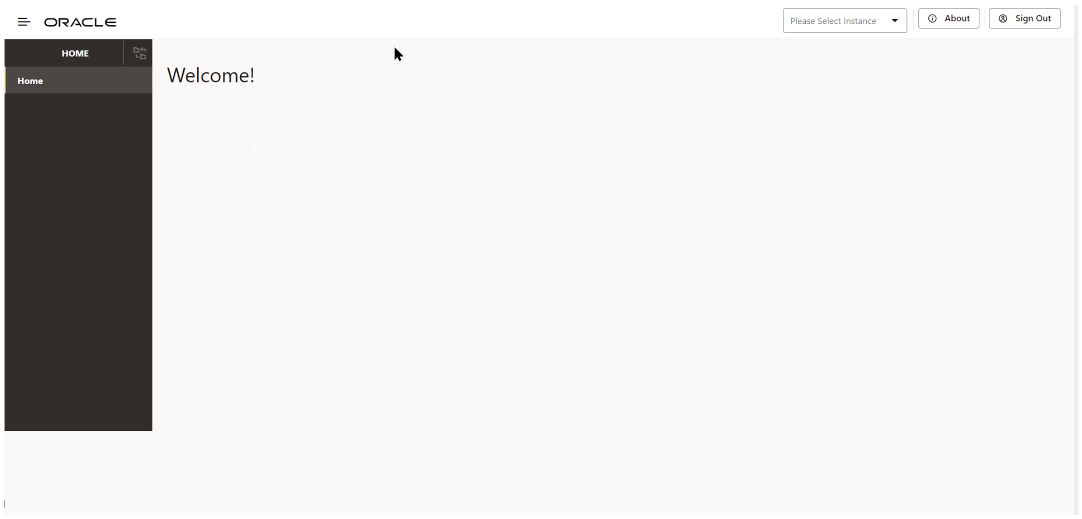
3.3 Accessing OCCM from CNC Console

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.

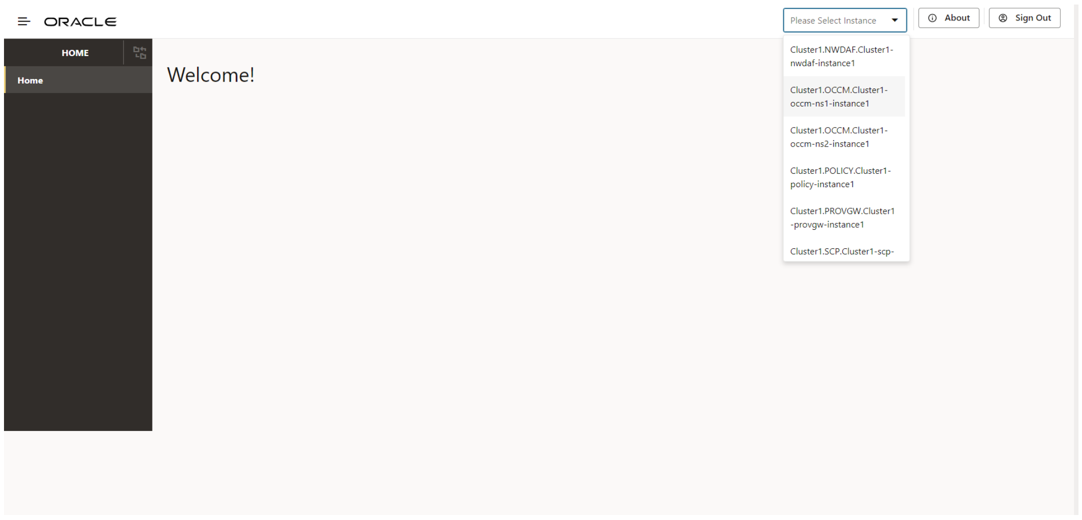
Figure 3-3 CNC Console Landing Page



Welcome!

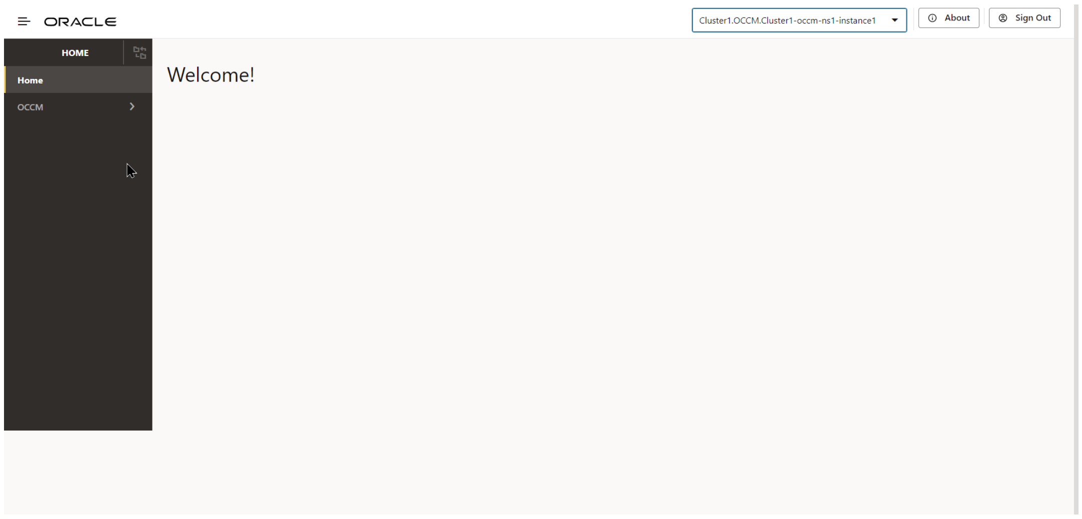
2. From the **Select Instance** drop-down, select the **OCCM** Instance.

Figure 3-4   OCCM Instance


ORACLE
HOME
Home
Welcome!
Please Select Instance
About
Sign Out
Cluster1.NWDAF.Cluster1-nwdaf-instance1
Cluster1.OCCM.Cluster1-occm-ns1-instance1
Cluster1.OCCM.Cluster1-occm-ns2-instance1
Cluster1.POLICY.Cluster1-policy-instance1
Cluster1.PROVGW.Cluster1-provgw-instance1
Cluster1.SCP.Cluster1-scp-

The OCCM menu appears on the left pane.

Figure 3-5   OCCM Configuration Options



3.4 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

3.4.1 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 CMP Identity (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 CMP Identity (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 certiifcate. 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 schemed 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.

3.4.2 Creating Issuer

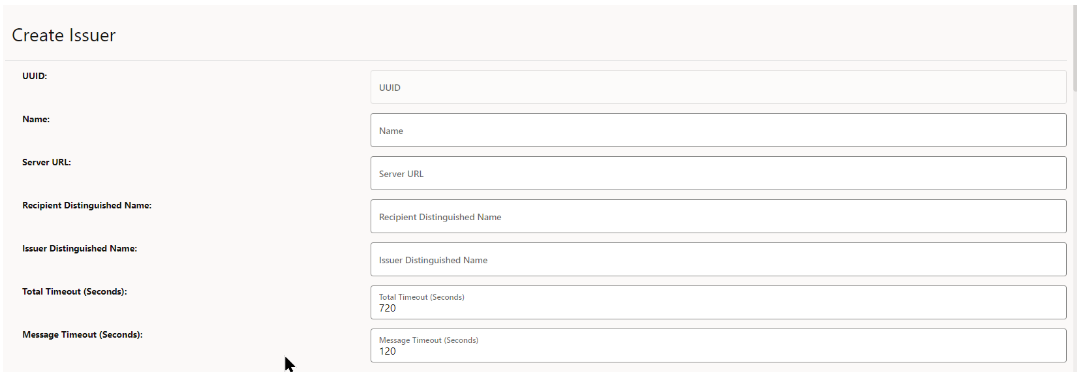
Issuers are resources that represent CAs and are able to generate signed certificates. You can configure issuers through REST API or using the CNC Console GUI. The maximum number of issuers that can be supported at a time is 30.

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
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`
Server URL	Domain URL of CA
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.

Table 3-1 (Cont.) Create Issuer

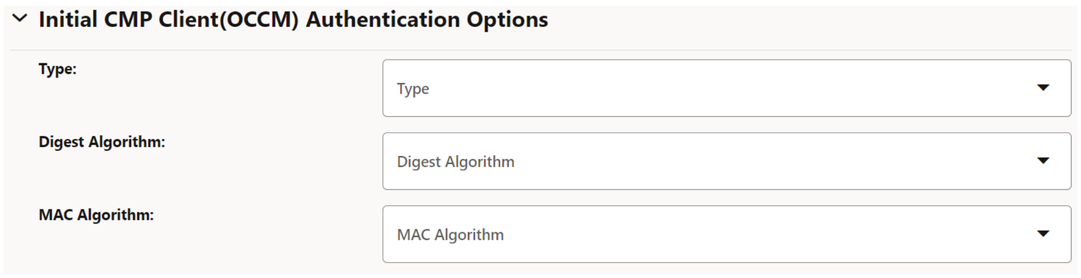
Field Name	Description
Message Timeout (Seconds)	The total time (in seconds) a CMP request-response message round trip is allowed to take.

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 CMP Identity (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

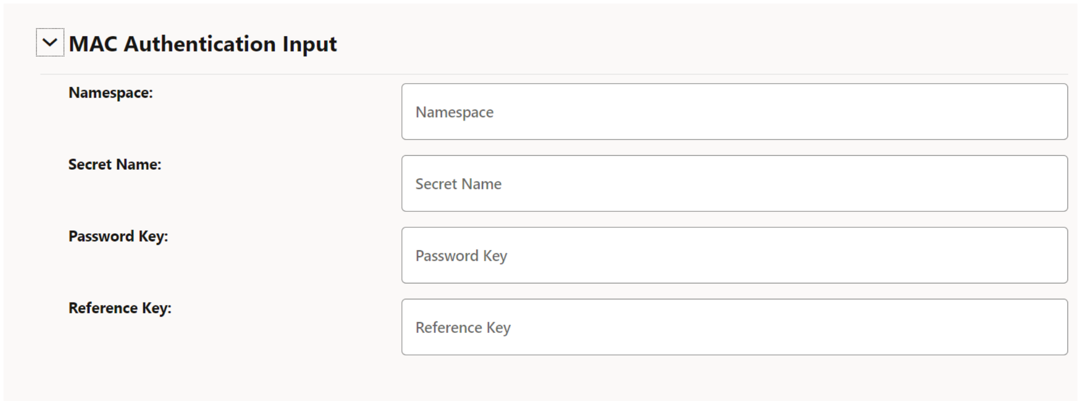
Type:	Type
Digest Algorithm:	Digest Algorithm
MAC Algorithm:	MAC Algorithm

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	Name of the Kubernetes namespace.
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

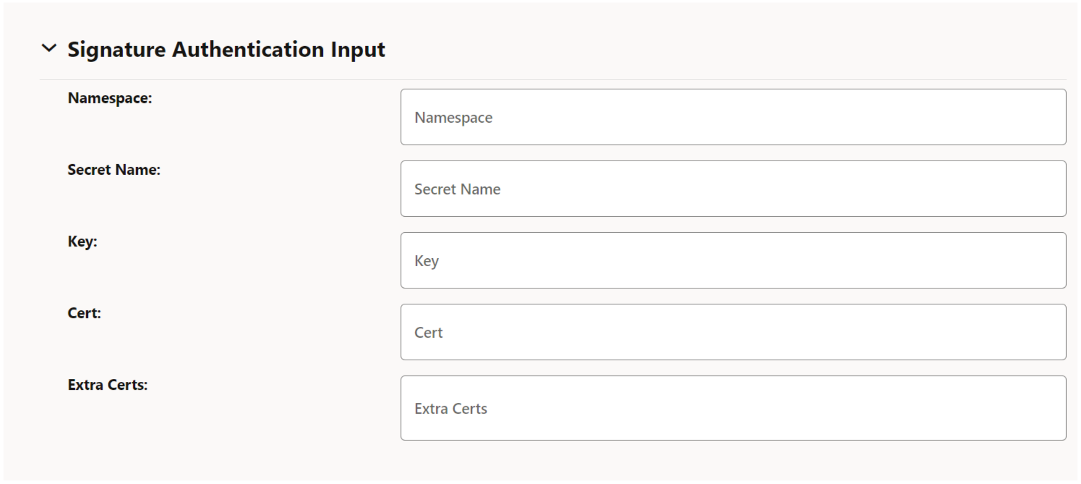


7. Under **Signature Authentication Input,** enter the following information:

Table 3-4 Signature Authentication Input

Field Name	Description
Namespace	Name of the Kubernetes namespace.
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

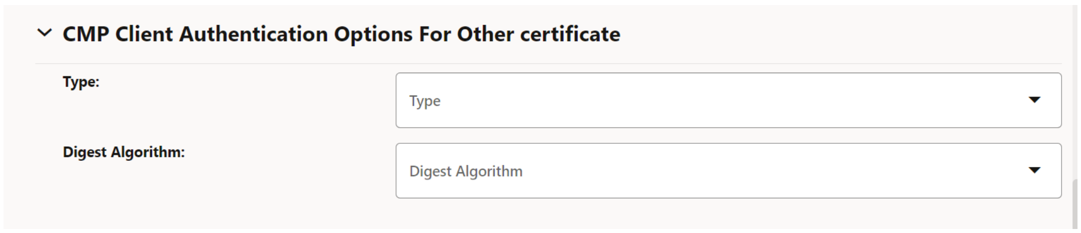


8. Under **CMP Client Authentication Options For Other certificate**, enter the following information:

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

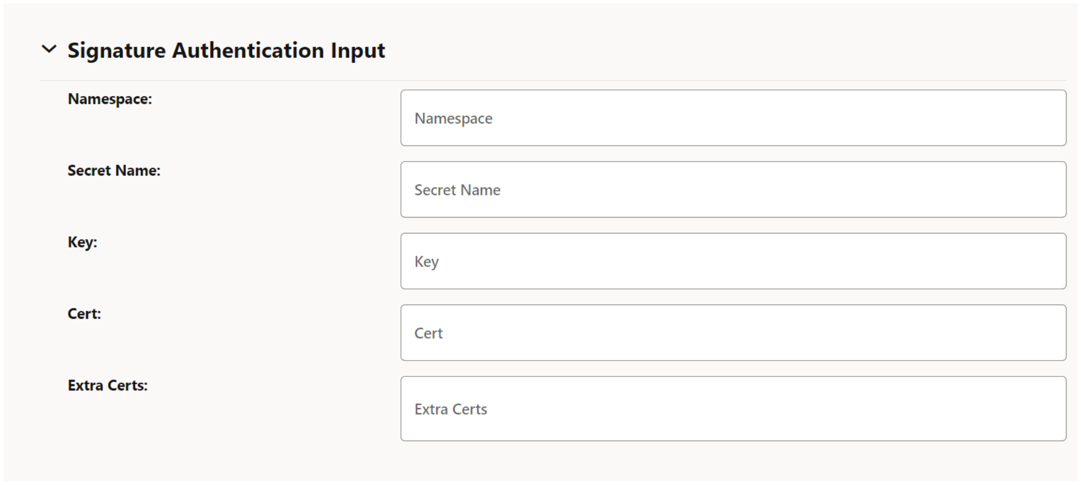
Type:	Type
Digest Algorithm:	Digest Algorithm

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

Table 3-6 Signature Authentication Input

Field Name	Description
Namespace	Name of the Kubernetes namespace
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



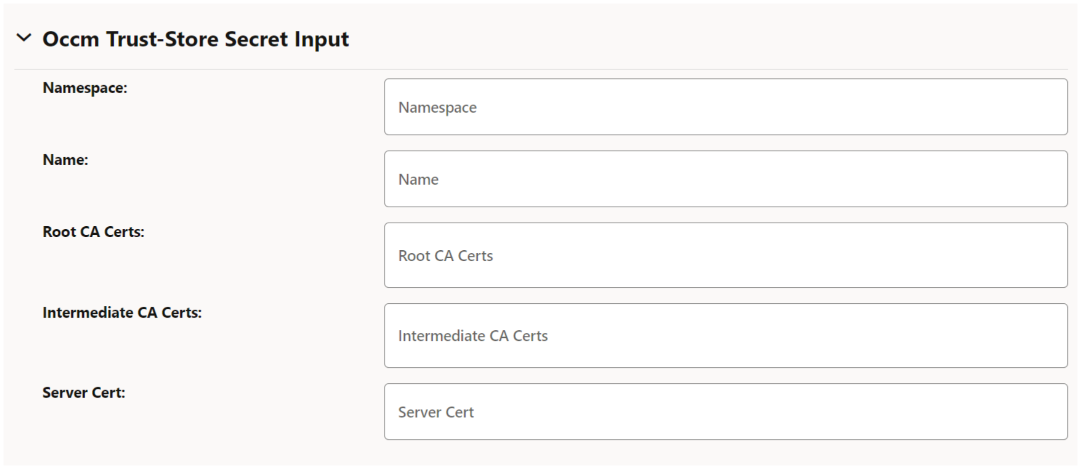
Namespace:	Namespace
Secret Name:	Secret Name
Key:	Key
Cert:	Cert
Extra Certs:	Extra Certs

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

Table 3-7   Occm Trust-Store Secret Input

Field	Description
Namespace	Name of the Kubernetes namepace.
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 certtificate 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.

Figure 3-13 TLS Configuration

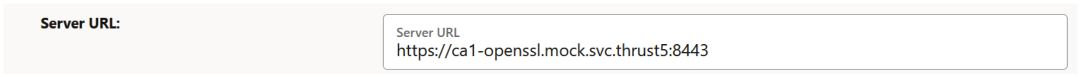
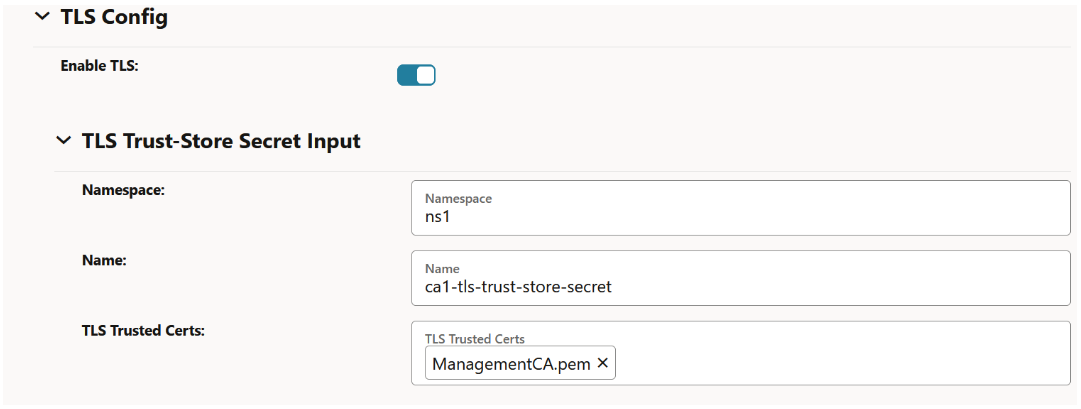

Server URL: Server URL https://ca1-openssl.mock.svc.thrust5:8443

Table 3-9 TLS Trust-Store secret Input

Field	Description
Namespace	Kubernetets namespace where TLS trust store secret is present.
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 3-14 Enable TLS



Enable TLS: on

Namespace: ns1

Name: ca1-tls-trust-store-secret

TLS Trusted Certs: ManagementCA.pem

13. Click **Save**.

3.4.3 Updating Issuer

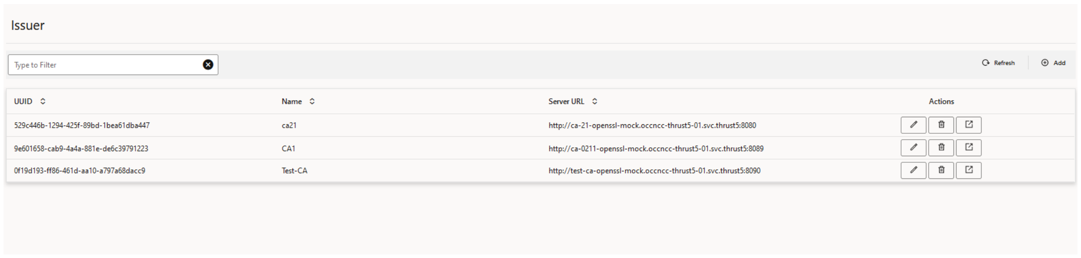
The bulk certificate migration can be used to update the end point of the CA and Issuer by updating the field, such as server URL, recipient DN, and Issuer DN. As part of the bulk certificate migration, OCCM updates the link of all the certificate configurations previously linked to the Old CA to the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

You can update all the fields in Edit issuer if no certificate configuration is attached to it. However, if any certification configuration is mapped to the given issuer, only the HTTP scheme is updated (HTTP to HTTPS and vice versa). When the HTTP scheme is updated, the certificates that are created will not be impacted. The updated server endpoint is contacted for any further CMPv2 procedures performed after the update. The scheme (HTTP or HTTPS) will be effective.

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**.

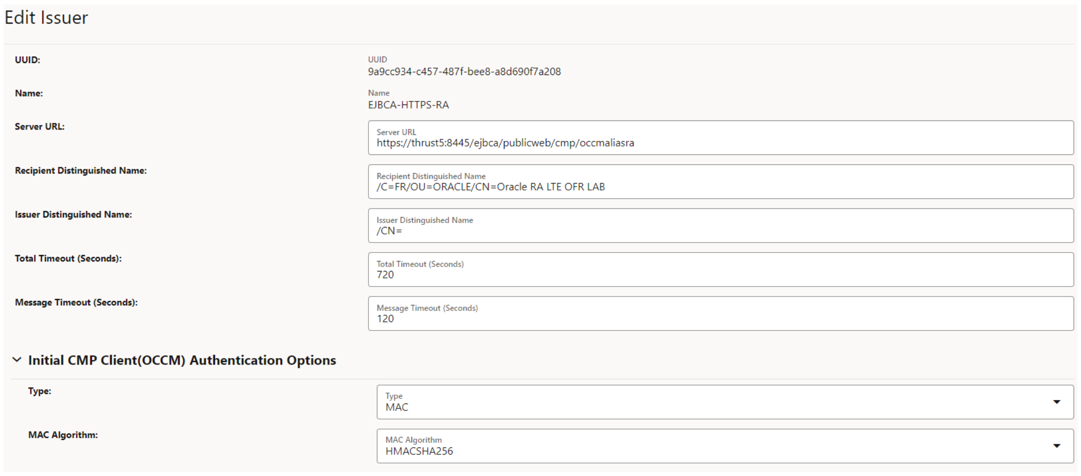
Figure 3-15 Issuer Page



UUID	Name	Server URL	Actions
529c446b-1294-425f-89bd-1bea61dba447	ca21	http://ca-21-openssl-mock.occncc-thrust5-01.svc.thrust5:8080	
9e601658-cab9-4a4a-881e-de6c39791223	CA1	http://ca-0211-openssl-mock.occncc-thrust5-01.svc.thrust5:8089	
0f19d193-ff86-461d-aa10-a797a68dacc9	Test-CA	http://test-ca-openssl-mock.occncc-thrust5-01.svc.thrust5:8090	

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

Figure 3-16 Edit Issuer



Edit Issuer

UUID:	UUID 9a9cc934-c457-487f-bee8-a8d690f7a208
Name:	Name EJBCA-HTTPS-RA
Server URL:	Server URL https://thrust5:8445/ejbca/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

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

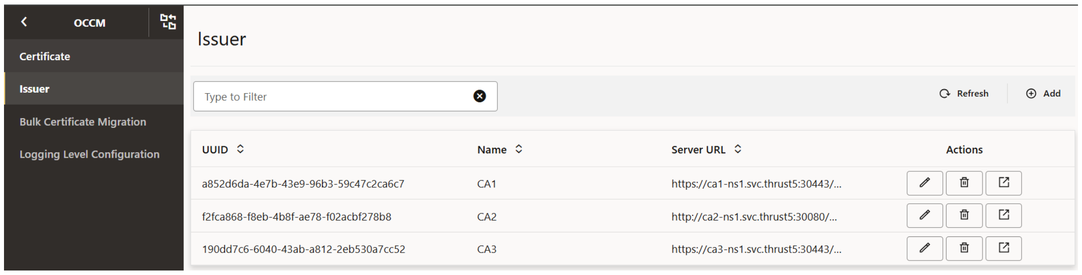
> **Note**
>
> Issuer can't be edited if it is in use by any certificate.

3.4.4 Deleting Issuers

To delete issuers from CNC Console GUI:

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

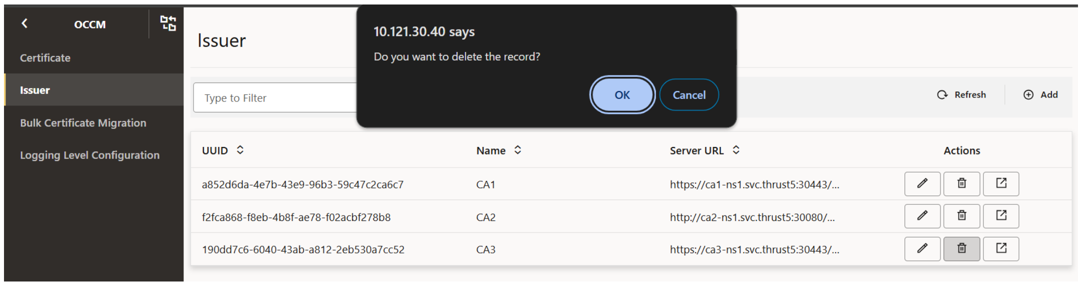
Figure 3-17 Issuer



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

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

Figure 3-18 Delete Issuer



10.121.30.40 says

Do you want to delete the record?

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

Note

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

3.5 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)
- Monitoring certificate expiry and renewing OCCM and NF certificates
- Polling for certificates
- Deleting the certificate configuration

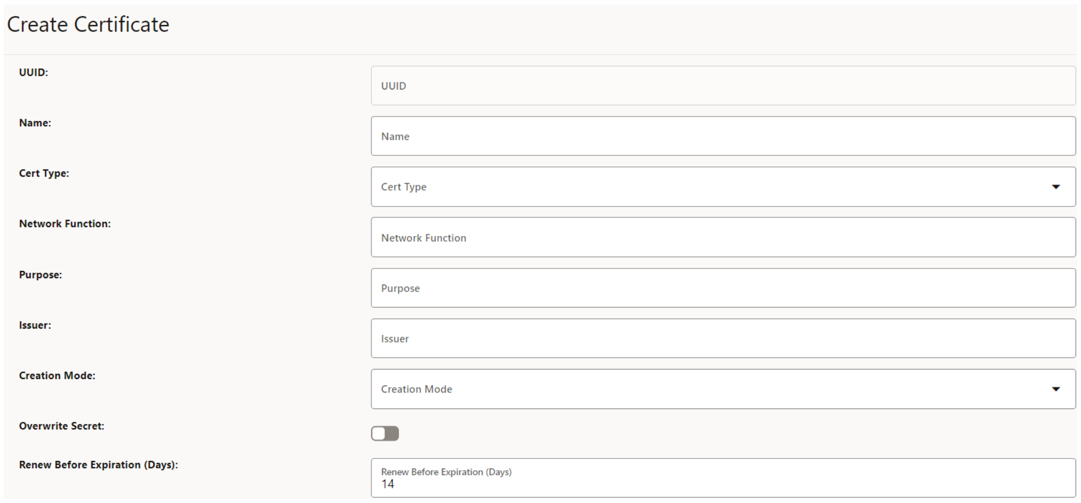
3.5.1 Creating CMP (OCCM) Identity Certificate

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 3-19 Create CMP identity Certificate



Create Certificate

UUID:	UUID
Name:	Name
Cert Type:	Cert Type
Network Function:	Network Function
Purpose:	Purpose
Issuer:	Issuer
Creation Mode:	Creation Mode
Overwrite Secret:	
Renew Before Expiration (Days):	Renew Before Expiration (Days) 14

4. Enter the following information:

Table 3-10 Create CMP identity Certificate

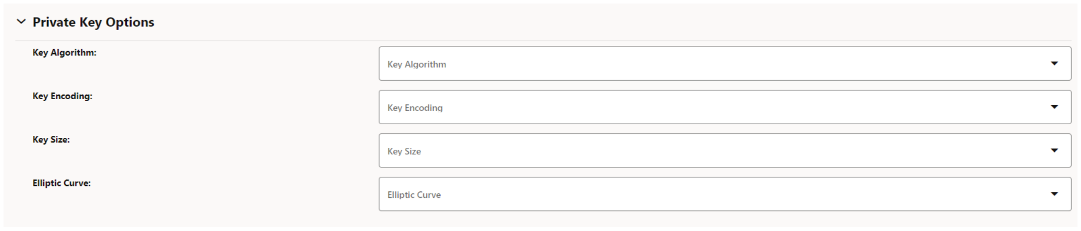
Field Name	Description and Possible Values
Name	Name of the certificate.
Cert Type	CMP (OCCM) Identity certificate
Network Function	OCCM
Purpose	Purpose of the OCCM certificate.
Issuer	Name of the issuer for the certificate.
Creation Mode	Possible values are MANUAL and AUTOMATIC. For more information, see CMP Identity (OCCM) Certificate Configuration Modes.

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

Table 3-11 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 3-20 Private Key Options



Private Key Options

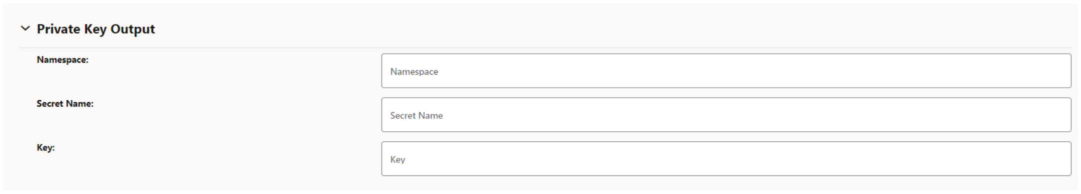
Key Algorithm:	Key Algorithm
Key Encoding:	Key Encoding
Key Size:	Key Size
Elliptic Curve:	Elliptic Curve

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

Table 3-12 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.

Figure 3-21 Private Key Output


Private Key Output

Namespace: Namespace

Secret Name: Secret Name

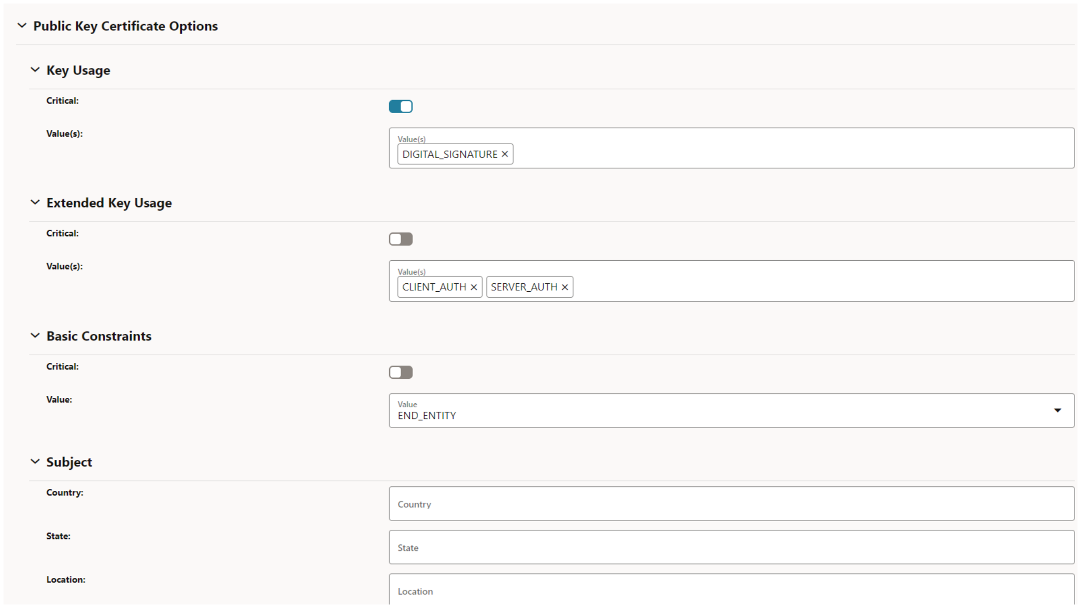
Key: Key

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

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
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 3-22 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

Subject

Country: Country

State: State

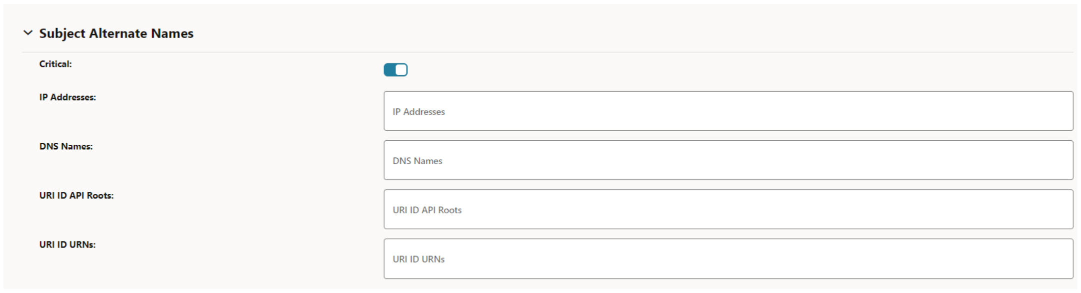
Location: Location

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

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 IDs.
URI ID URNs	List of URI IDs.

Figure 3-23 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.

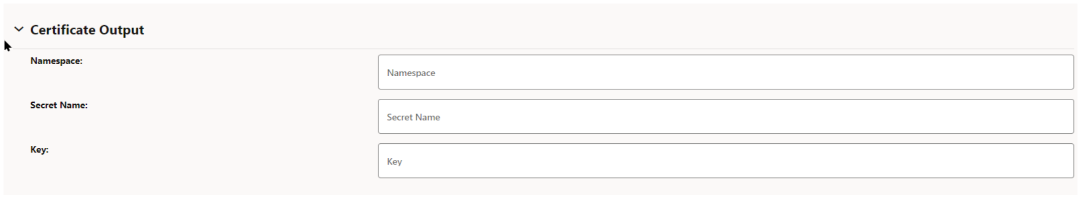
Table 3-15 Certificate Output

Field Name	Description
Namespace	Name of the namespace.
Secret Name	Name of the secret.

Table 3-15 (Cont.) Certificate Output

Field Name	Description
Key	The key against which the certificate will be populated.

Figure 3-24 Certificate Output



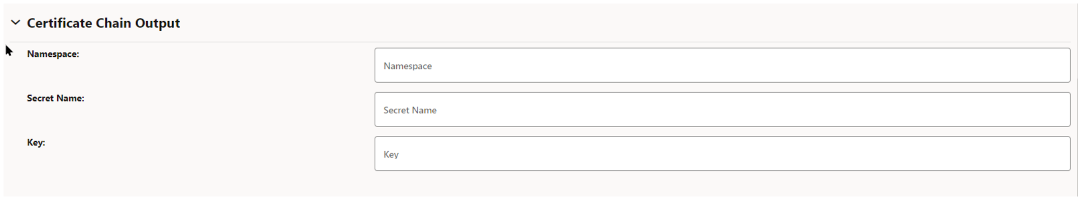
Namespace: Namespace
Secret Name: Secret Name
Key: Key

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

Table 3-16 Certificate Chain Output

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

Figure 3-25 Certificate Chain Output



Namespace: Namespace
Secret Name: Secret Name
Key: Key

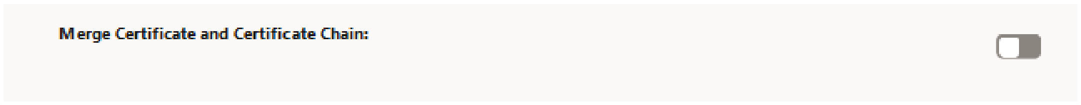
If the **Certificate Chain Output** section is filled, then the certificate chain can either be obtained from the CA or can be configured manually. This is based on the `extractCertChainFromCmpResponse` helm parameter. For more information, see *Oracle Communications Cloud Native Core, Certificate Management 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 3-26 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**.

3.5.1.1 CMP Identity (OCCM) Certificate Configuration Modes

The following section highlights the configuration applicable to these modes and control how the CMP identity 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):

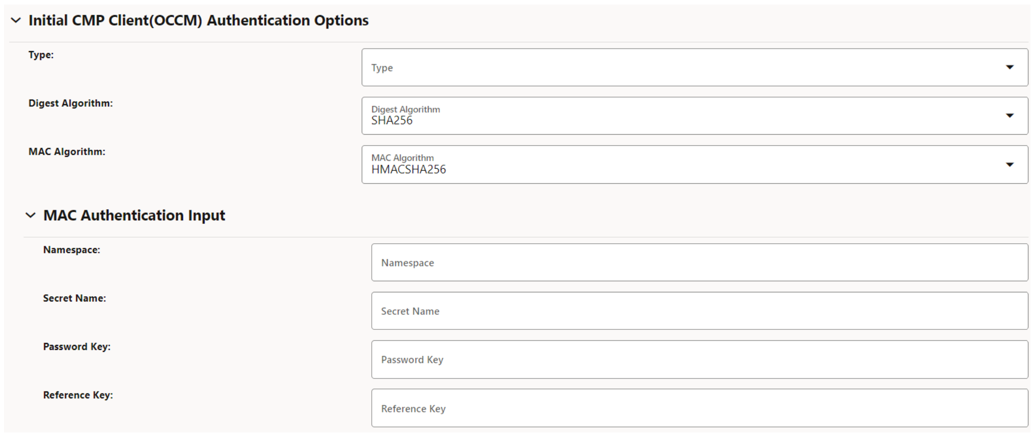
- Using the pre-shared key
- Using the pre-configured private key and certificate

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 **Initial CMP Client (OCCM) Authentication Options**.

Figure 3-27 Initial CMP Client(OCCM) Authentication Options



Initial CMP Client(OCCM) Authentication Options

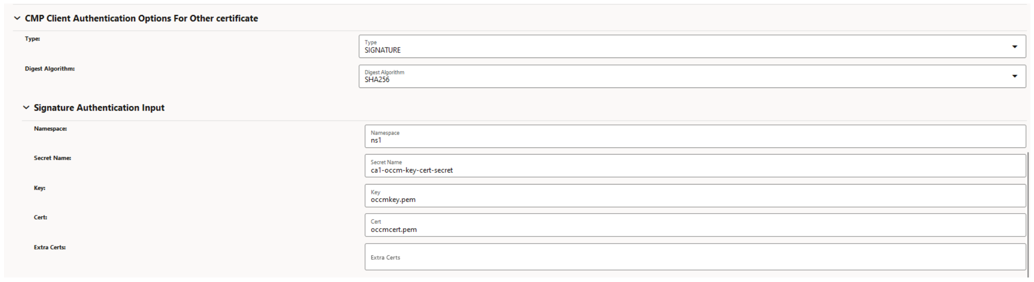
Type:	Type
Digest Algorithm:	Digest Algorithm SHA256
MAC Algorithm:	MAC Algorithm HMACSHA256

MAC Authentication Input

Namespace:	Namespace
Secret Name:	Secret Name
Password Key:	Password Key
Reference 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 3-28 CMP Client Authentication Options for Other Certificate



CMP Client Authentication Options For Other certificate

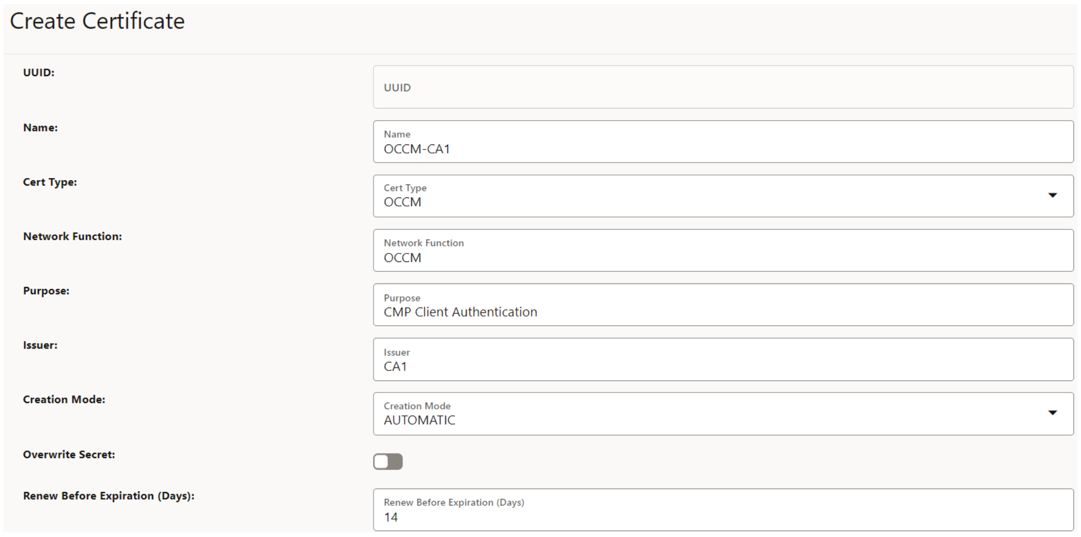
Type:	Type SIGNATURE
Digest Algorithm:	Digest Algorithm SHA256

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

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 **Creation Mode** on the **Create Certificate** page.

Figure 3-29 OCCM Certificate Configuration using Pre-shared Key



Create Certificate

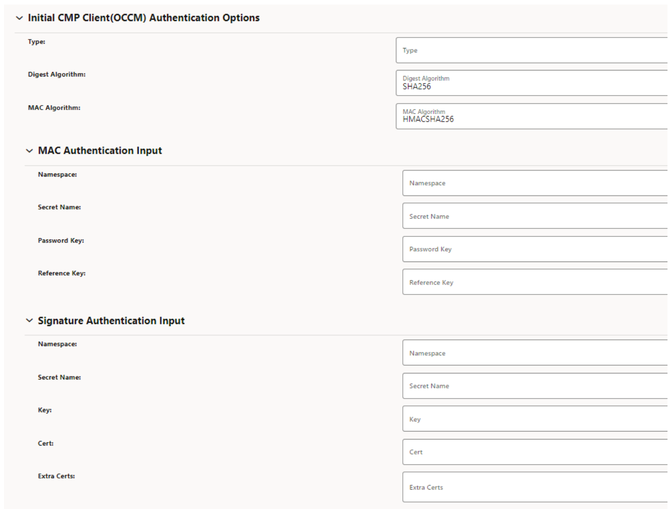
UUID:	UUID
Name:	Name OCCM-CA1
Cert Type:	Cert Type OCCM
Network Function:	Network Function OCCM
Purpose:	Purpose CMP Client Authentication
Issuer:	Issuer CA1
Creation Mode:	Creation Mode AUTOMATIC
Overwrite Secret:	
Renew Before Expiration (Days):	Renew Before Expiration (Days) 14

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. 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 **Initial CMP Client(OCCM) Authentication Options.**

Figure 3-30 Initial CMP Client(OCCM) Authentication Options



Initial CMP Client(OCCM) Authentication Options

Type:	Type
Digest Algorithm:	Digest Algorithm SHA256
MAC Algorithm:	MAC Algorithm HMACSHA256

MAC Authentication Input

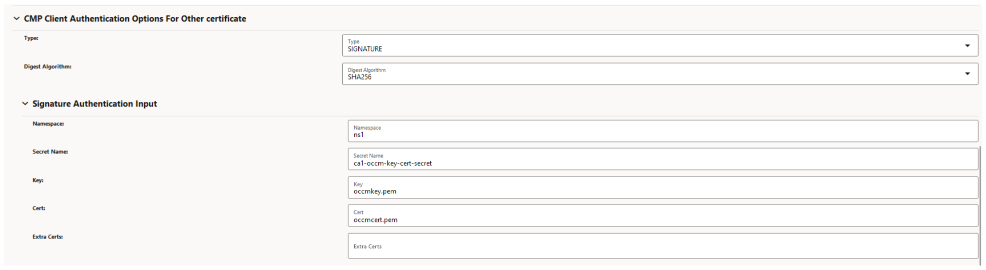
Namespace:	Namespace
Secret Name:	Secret Name
Password Key:	Password Key
Reference Key:	Reference Key

Signature Authentication Input

Namespace:	Namespace
Secret Name:	Secret Name
Key:	Key
Cert:	Cert
Extra Certs:	Extra Certs

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

Figure 3-31 CMP Client Authentication Options for Other Certificate



CMP Client Authentication Options For Other certificate

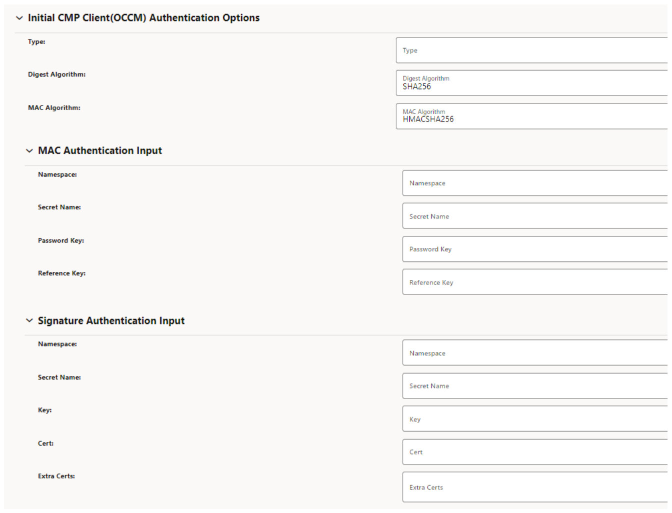
Type:	Type SIGNATURE
Digest Algorithm:	Digest Algorithm SHA256

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 **Creation Mode** on the **Create Certificate** page.
2. The pre-configured private key and certificate (generated out of band) can be used as the OCCM certificate.
   a. **Issuer Configuration**
      i. Here, you must skip the **Initial CMP Client(OCCM) Authentication Options.**

Figure 3-32 Issuer Configuration



Initial CMP Client(OCCM) Authentication Options

Type:	Type
Digest Algorithm:	Digest Algorithm SHA256
MAC Algorithm:	MAC Algorithm HMACSHA256

MAC Authentication Input

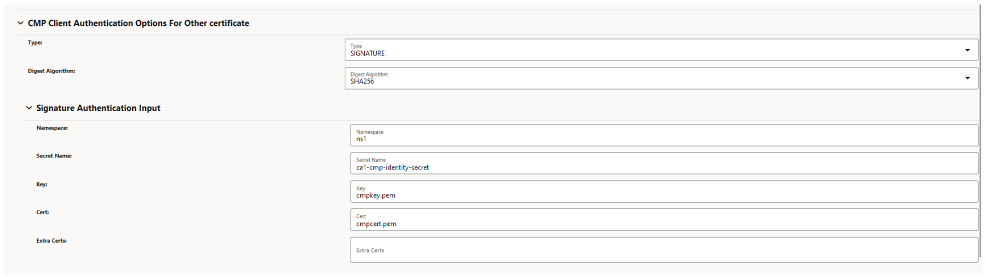
Namespace:	Namespace
Secret Name:	Secret Name
Password Key:	Password Key
Reference Key:	Reference Key

Signature Authentication Input

Namespace:	Namespace
Secret Name:	Secret Name
Key:	Key
Cert:	Cert
Extra Certs:	Extra Certs

ii. OCCM key and certificate output location need to be specified under **CMP Client Authentication Options for Other Certificate.** Specify the manually created OCCM key and certificate location here.

Figure 3-33 CMP Client Authentication Options for Other Certificate



CMP Client Authentication Options For Other certificate

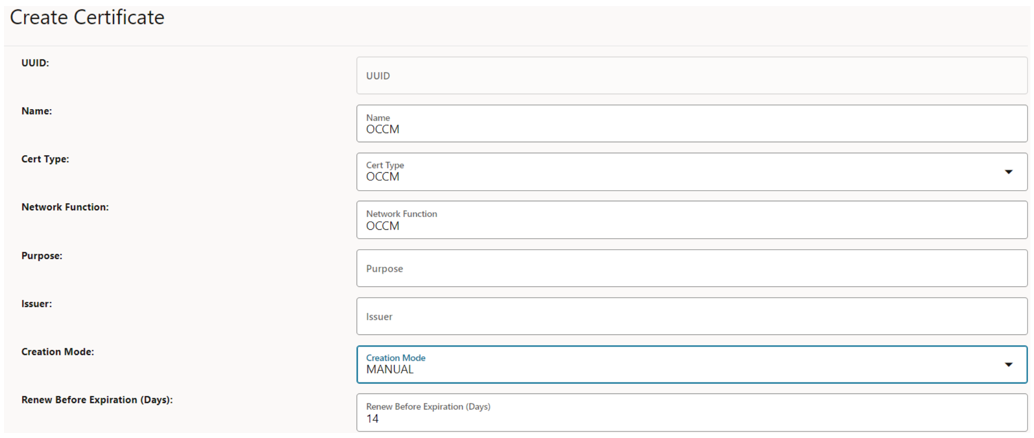
Type:	Type SIGNATURE
Digest Algorithm:	Digest Algorithm SHA256

Signature Authentication Input

Namespace:	Namespace ns1
Secret Name:	Secret Name ca1-cmp-identity-secret
Key:	Key cmpkey.pem
Cert:	Cert cmpcert.pem
Extra Certs:	Extra Certs

b. **OCCM Certificate Configuration**
To configure the OCCM Certificate, select OCCM from the **Cert Type** drop-down and select MANUAL from the **Creation Mode** on the **Create Certificate** page.

Figure 3-34 OCCM Certificate Configuration



Create Certificate

UUID:	UUID
Name:	Name OCCM
Cert Type:	Cert Type OCCM
Network Function:	Network Function OCCM
Purpose:	Purpose
Issuer:	Issuer
Creation Mode:	Creation Mode MANUAL
Renew Before Expiration (Days):	Renew Before Expiration (Days) 14

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

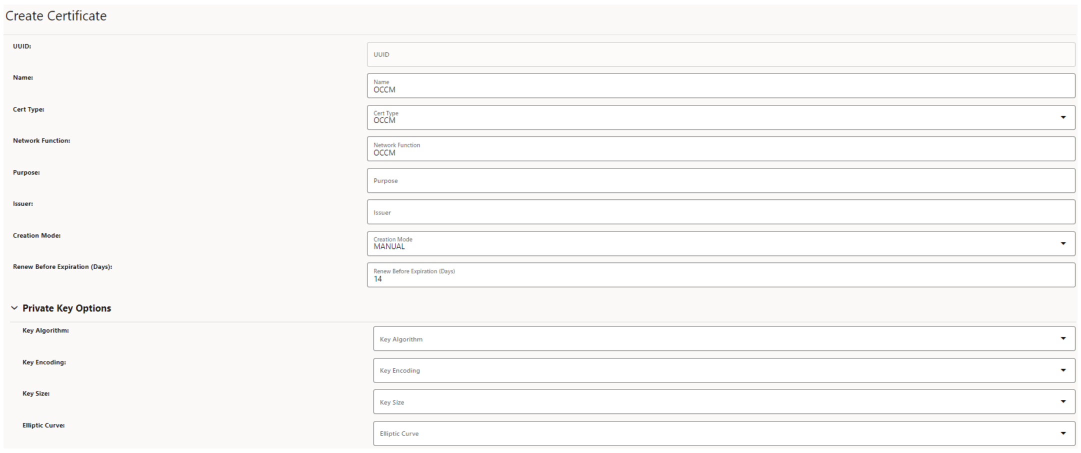
3.5.2 Creating End Entity (NF) Certificates

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 3-35 Create NF Certificate



Create Certificate

UUID:	UUID
Name:	Name: OCCM
Cert Type:	Cert Type: OCCM
Network Function:	Network Function: OCCM
Purpose:	Purpose
Issuer:	Issuer
Creation Mode:	Creation Mode: MANUAL
Renew Before Expiration (Days):	Renew Before Expiration (Days): 14

Private Key Options

Key Algorithm:	Key Algorithm
Key Encoding:	Key Encoding
Key Size:	Key Size
Elliptic Curve:	Elliptic Curve

4. Enter the following information:

Table 3-17 Create NF Certificate

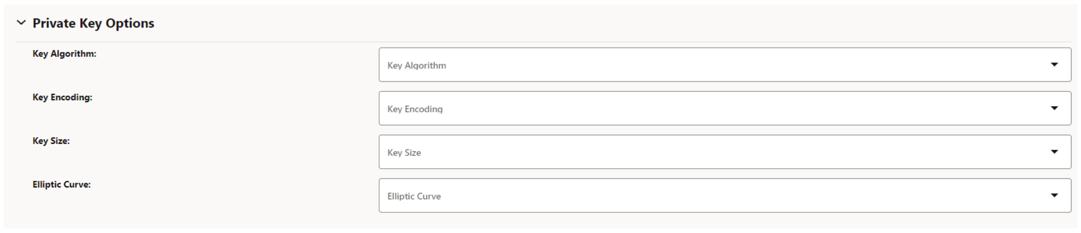
Field Name	Description and Possible Values
Name	Name of the certificate.
Cert Type	Select OTHER for End Entity (NF) certificates.
Network Function	Name of the End Entity (NF).
Purpose	Purpose of the End Entity (NF) certificate.
Issuer	Name of the issuer for the certificate.
Creation Mode	Possible values are MANUAL and AUTOMATIC.

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

Table 3-18 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 3-36 Private Key Options

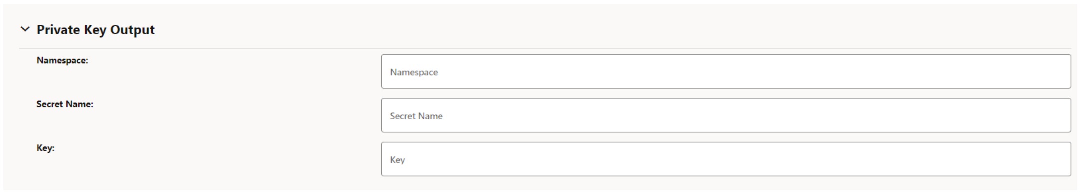


6. **Under Private Key Output,** enter the following information:

Table 3-19 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.

Figure 3-37 Private Key Output



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

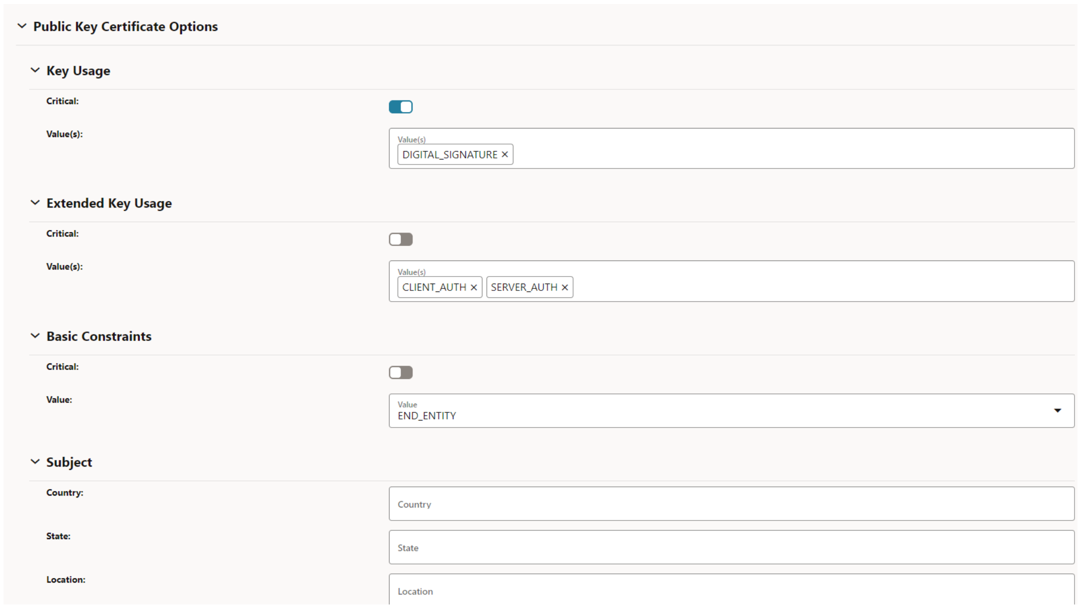
Table 3-20 Public Key Certificate Options

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 3-20 (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. 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 3-38 Public Key Certificate Options



Public Key Certificate Options

Key Usage

Critical: on

Value(s): DIGITAL_SIGNATURE

Extended Key Usage

Critical: off

Value(s): CLIENT_AUTH, SERVER_AUTH

Basic Constraints

Critical: off

Value: END_ENTITY

Subject

Country: Country

State: State

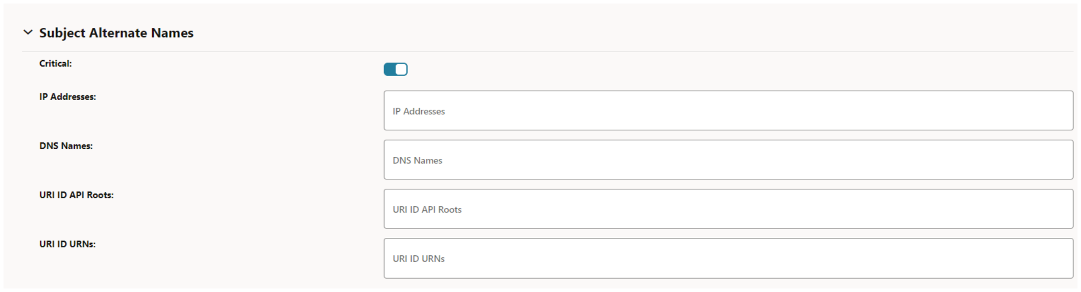
Location: Location

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

Table 3-21 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 3-39 Subject Alternate Names

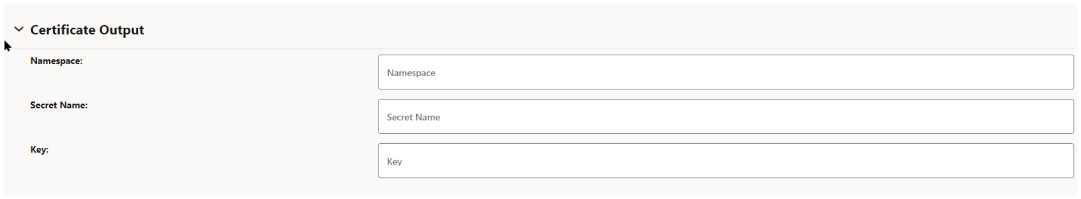

Subject Alternate Names
Critical:
IP Addresses: IP Addresses
DNS Names: DNS Names
URI ID API Roots: URI ID API Roots
URI ID URNs: URI ID URNs

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

Table 3-22 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.

Figure 3-40 Certificate Output

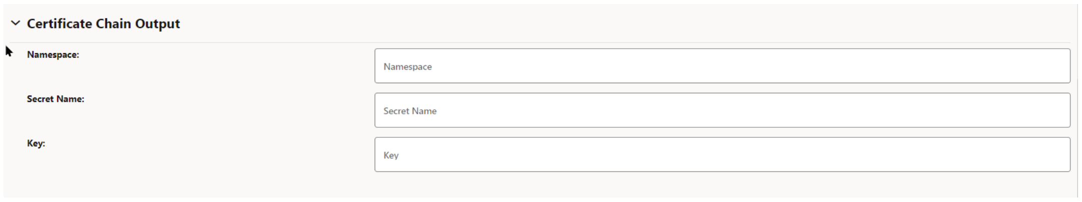

Certificate Output
Namespace: Namespace
Secret Name: Secret Name
Key: Key

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

Table 3-23 Certificate Chain Output

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

Figure 3-41 Certificate Chain Output

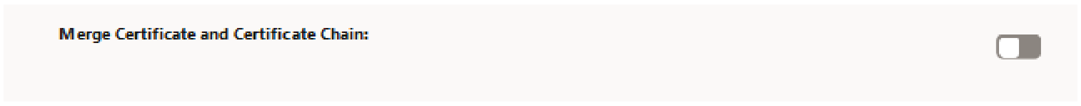

Certificate Chain Output
Namespace: Namespace
Secret Name: Secret Name
Key: Key

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 3-42 Merge Certificate and Certificate Chain

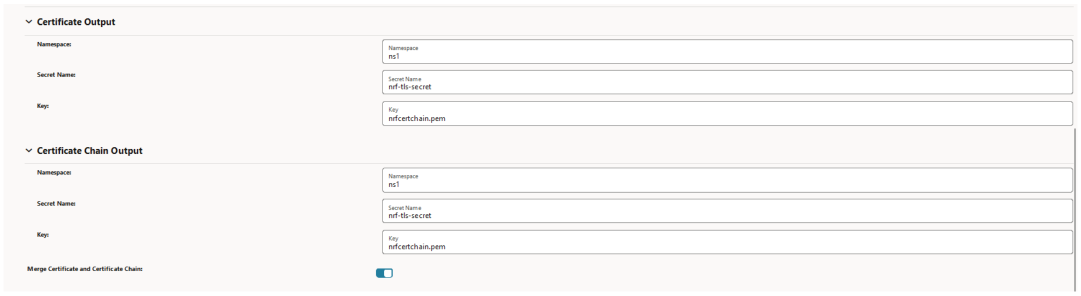


Merge Certificate and Certificate Chain:

> **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.

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 3-43 Sample Certificate Output and Certificare Chain Output



Certificate Output

Namespace:	Namespace ns1
Secret Name:	Secret Name nrf-tls-secret
Key:	Key nrfcertchain.pem

Certificate Chain Output

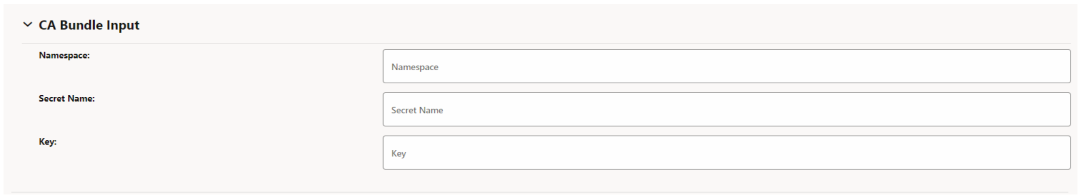
Namespace:	Namespace ns1
Secret Name:	Secret Name nrf-tls-secret
Key:	Key nrfcertchain.pem
Merge Certificate and Certificate Chain:	

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

Table 3-24 CA Bundle Input

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

Figure 3-44 CA Bundle Input



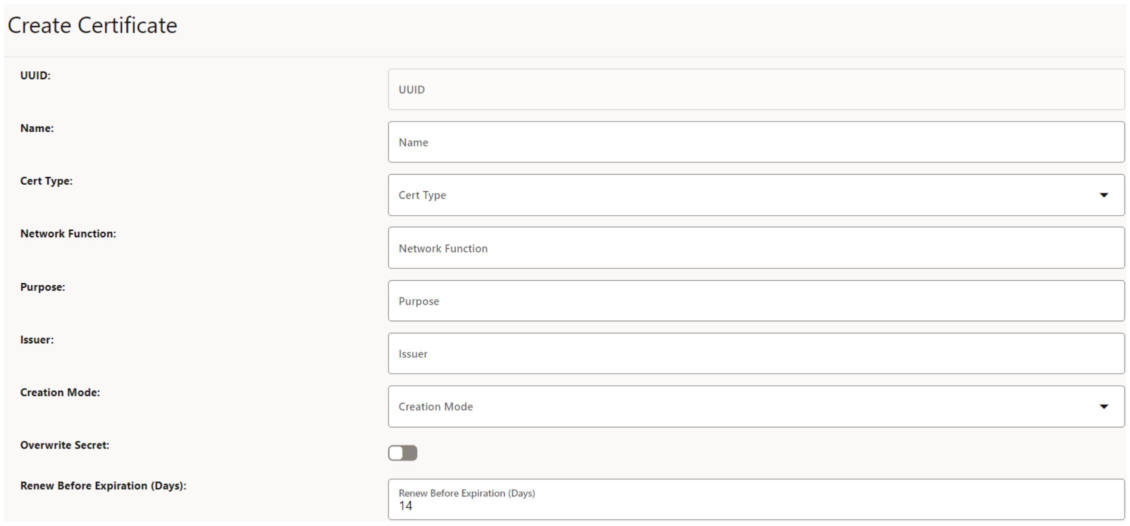
13. Click **Save**.

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

3.5.3 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 3-45 Renew Before Expiration (Days)



Renew Before Expiration (Days): 14

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.

3.5.3.1 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.

3.5.3.2 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.

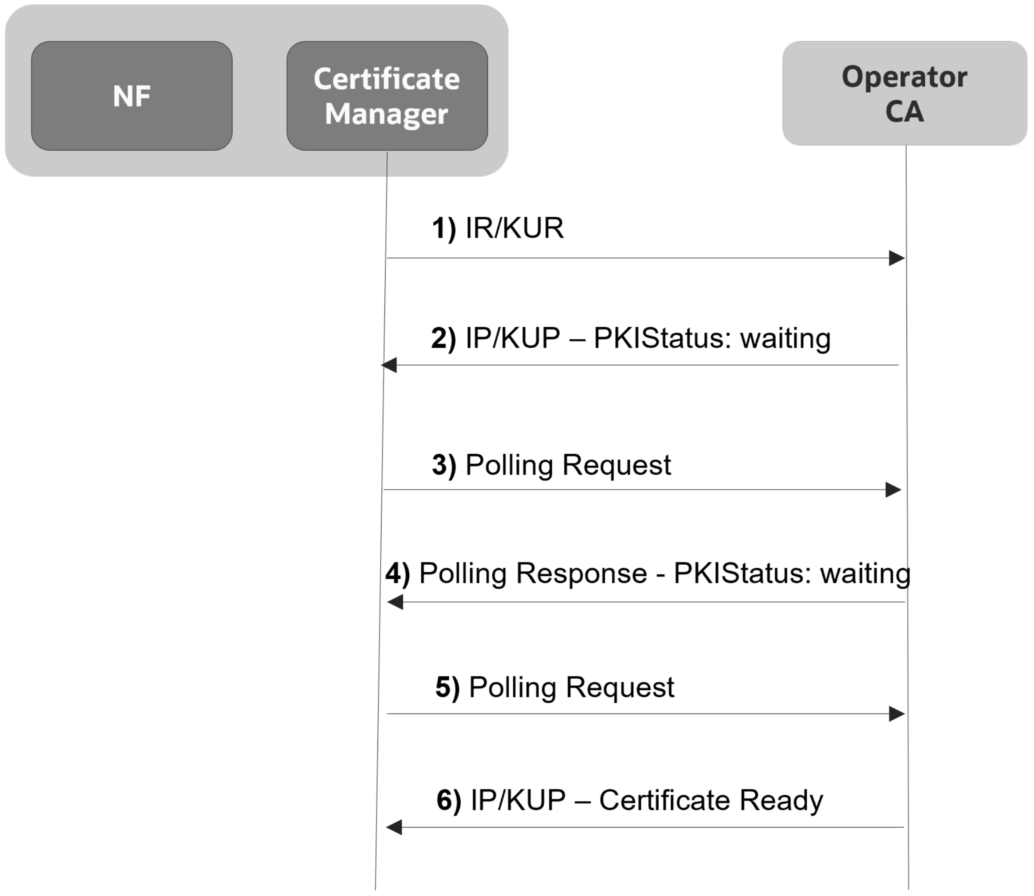
Note

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.

3.5.4 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 3-46 Polling for Certificates

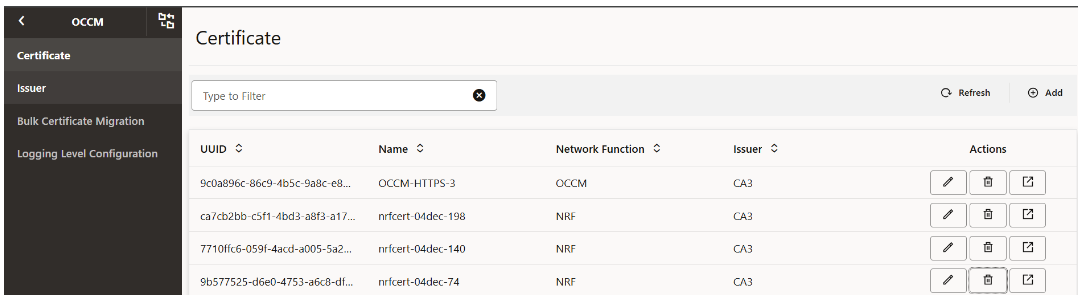

NF
Certificate Manager
Operator CA
1) IR/KUR
2) IP/KUP – PKIStatus: waiting
3) Polling Request
4) Polling Response - PKIStatus: waiting
5) Polling Request
6) IP/KUP – Certificate Ready

3.5.5 Deleting the Certificate Configuration

To delete the certificate configuration:

1. Login to CNC Console using your login credentials and select the **OCCM** Instance.
2. Click **OCCM** from the left pane and then click **Certificate**. All the available certificates will be listed

Figure 3-47 Certificate

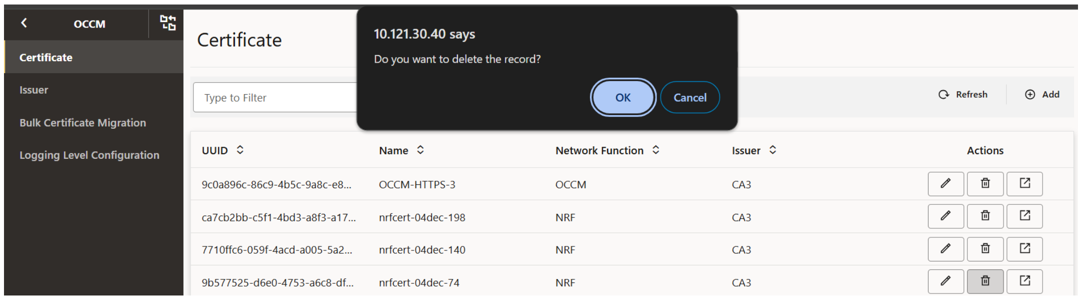


Certificate

UUID	Name	Network Function	Issuer	Actions
9c0a896c-86c9-4b5c-9a8c-e8...	OCCM-HTTPS-3	OCCM	CA3	
ca7cb2bb-c5f1-4bd3-a8f3-a17...	nrfcert-04dec-198	NRF	CA3	
7710ffc6-059f-4acd-a005-5a2...	nrfcert-04dec-140	NRF	CA3	
9b577525-d6e0-4753-a6c8-df...	nrfcert-04dec-74	NRF	CA3	

3. Click **Delete** and click **OK** on the confirmation prompt to delete the certificate.

Figure 3-48 Certificate Delete



Certificate

10.121.30.40 says
Do you want to delete the record?

UUID	Name	Network Function	Issuer	Actions
9c0a896c-86c9-4b5c-9a8c-e8...	OCCM-HTTPS-3	OCCM	CA3	
ca7cb2bb-c5f1-4bd3-a8f3-a17...	nrfcert-04dec-198	NRF	CA3	
7710ffc6-059f-4acd-a005-5a2...	nrfcert-04dec-140	NRF	CA3	
9b577525-d6e0-4753-a6c8-df...	nrfcert-04dec-74	NRF	CA3	

Note

This procedure only deletes the certificate configuration from OCCM.
Run the following command to delete the Kubernetes secret holding the certificates:

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

For example:

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

3.5.6 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.

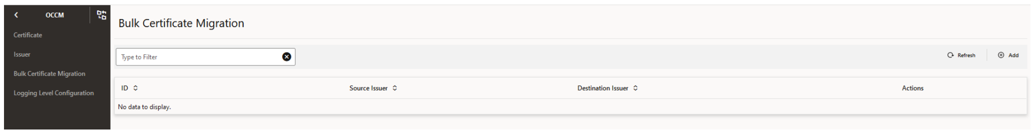
To start the bulk migration, the operator must specify both the source and destination issuers in the request. During the process, all the certificates linked to the source issuer are retrieved and an automatic recreate (CMPv2 Initialization Request) is triggered for each certificate to the destination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

3.5.6.1 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.

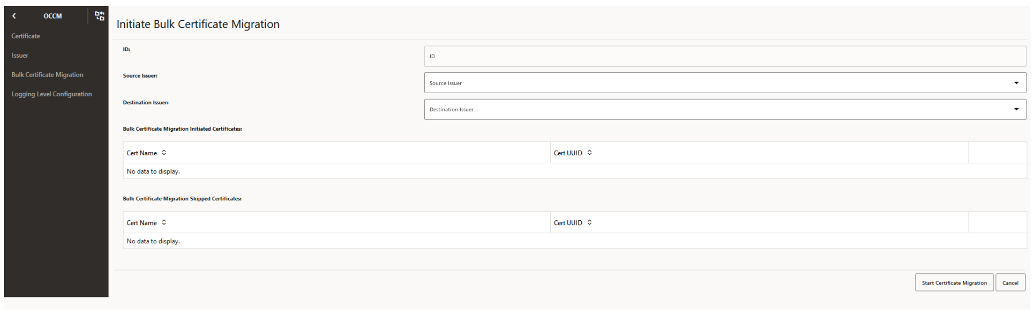
Figure 3-49 Bulk Certificate Migration


OCCM
Certificate
Issuer
Bulk Certificate Migration
Logging Level Configuration
Bulk Certificate Migration
Type to Filter
Refresh
Add

ID	Source Issuer	Destination Issuer	Actions
No data to display.			

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

Figure 3-50 Initiate Bulk Certificate Migration


OCCM
Certificate
Issuer
Bulk Certificate Migration
Logging Level Configuration
Initiate Bulk Certificate Migration
Start Certificate Migration
Cancel

ID: ID
Source Issuer: Source Issuer
Destination Issuer: Destination Issuer

Bulk Certificate Migration Initiated Certificates:

Cert Name	Cert UUID
No data to display.	

Bulk Certificate Migration Skipped Certificates:

Cert Name	Cert UUID
No data to display.	

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

Table 3-25 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 3-26 Initiate Bulk Certificate Migration

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

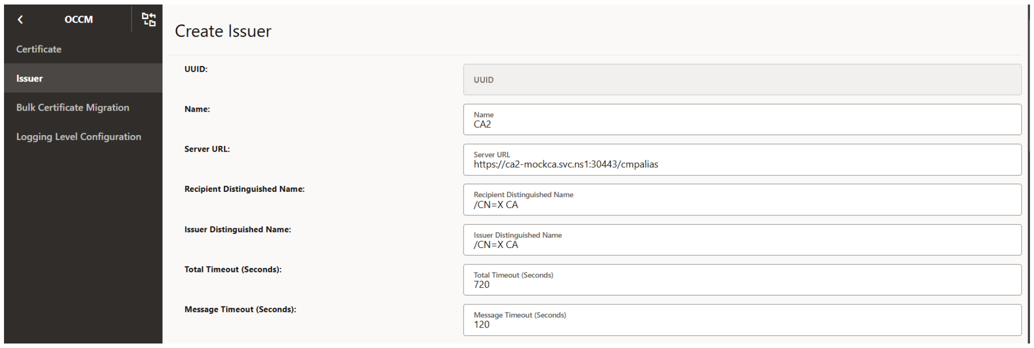
Note

- Bulk certificate migration is supported for only linked End Entity (NF) certificates. It does not apply to CMP (OCCM) identity certificates.
- Bulk migration is triggered only on READY or EXPIRED certificates.
- 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.

3.5.6.1.1 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:

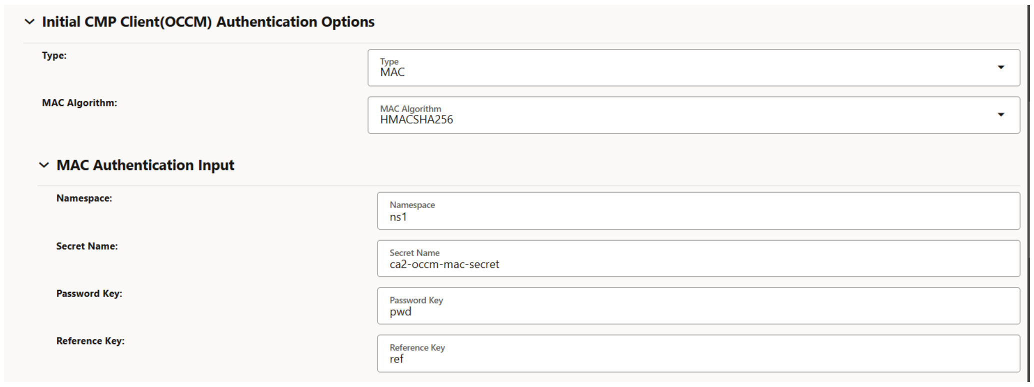
1. Create the destination issuer CA2 with the requisite configuration. For more information about the steps, see Managing Issuers.
   a. **Figure 3-51 Create Issuer**



Create Issuer

UUID:	UUID
Name:	Name: CA2
Server URL:	Server URL: https://ca2-mockca.svc.ns1:30443/cmpalias
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. **Figure 3-52 Initial CMP Client (OCCM) Authentication Options**



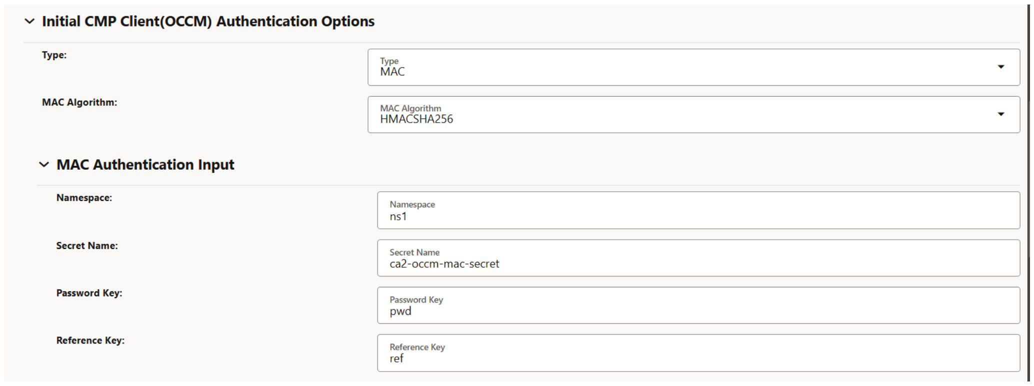
Initial CMP Client(OCCM) Authentication Options

Type:	Type: MAC
MAC Algorithm:	MAC Algorithm: HMACSHA256

MAC Authentication Input

Namespace:	Namespace: ns1
Secret Name:	Secret Name: ca2-occm-mac-secret
Password Key:	Password Key: pwd
Reference Key:	Reference Key: ref

c. **Figure 3-53 MAC Authentication Input**



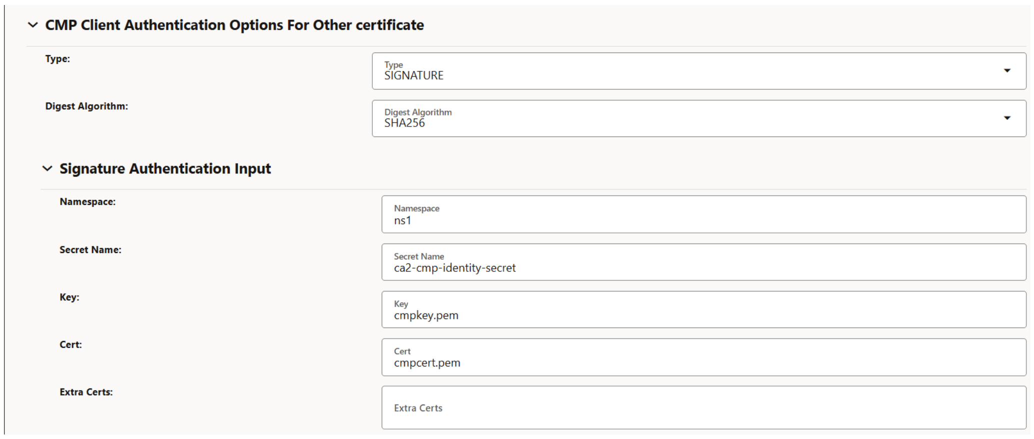
Initial CMP Client(OCCM) Authentication Options

Type:	Type: MAC
MAC Algorithm:	MAC Algorithm: HMACSHA256

MAC Authentication Input

Namespace:	Namespace: ns1
Secret Name:	Secret Name: ca2-occm-mac-secret
Password Key:	Password Key: pwd
Reference Key:	Reference Key: ref

d. **Figure 3-54 CMP Client Authentication Options For Other certificate**



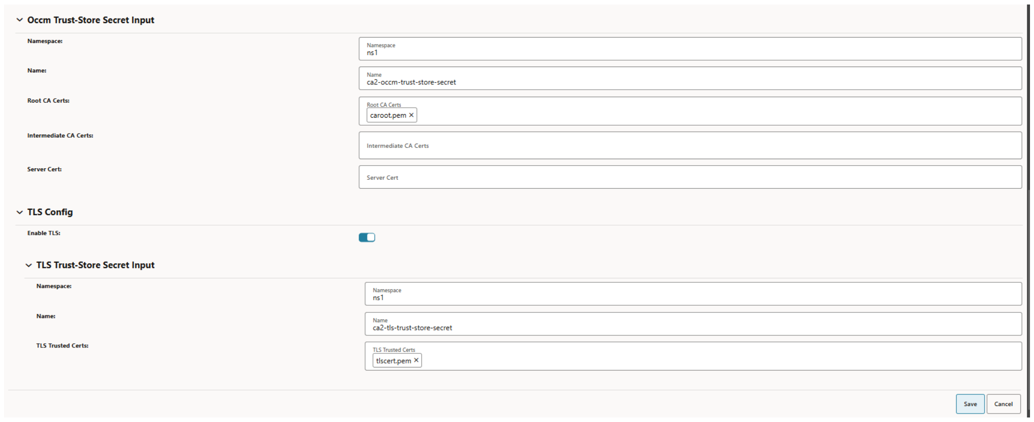
CMP Client Authentication Options For Other certificate

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. **Figure 3-55 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 ×
Intermediate CA Certs:	Intermediate CA Certs
Server Cert:	Server Cert

TLS Config

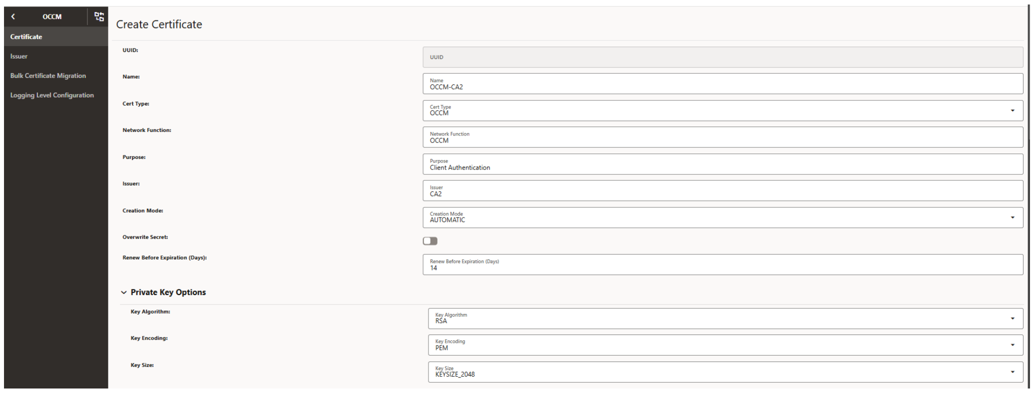
Enable TLS:	●

TLS Trust-Store Secret Input

Namespace:	Namespace: ns1
Name:	Name: ca2-tls-trust-store-secret
TLS Trusted Certs:	TLS Trusted Certs: tlscert.pem ×

2. Create CMP (OCCM) Identity certificate OCCM-CA2 corresponding to the destination issuer CA2. For more information about the steps, see Managing Certificates.

a. **Figure 3-56 Create Certificate**



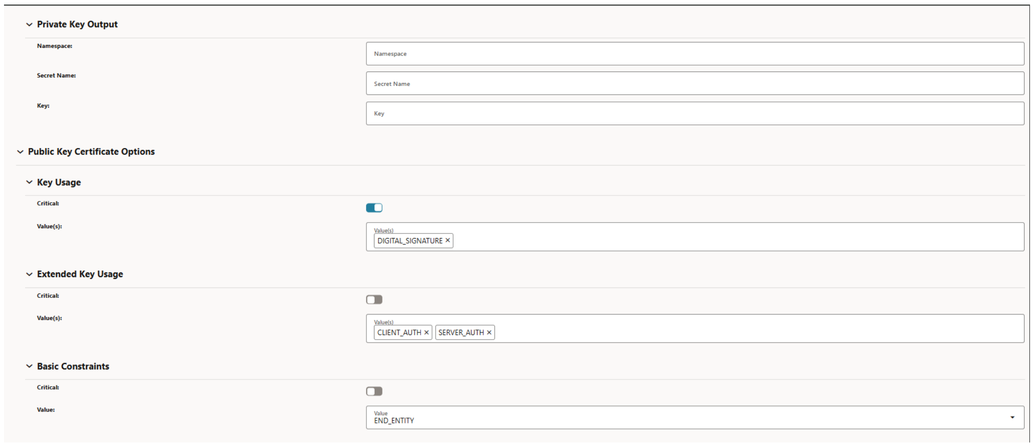
Create Certificate

UUID:	UUID
Name:	OCCM-CA2
Cert Type:	OCCM
Network Function:	OCCM
Purpose:	Client Authentication
Issuer:	CA2
Creation Mode:	AUTOMATIC
Overwrite Secret:	
Renew Before Expiration (Days):	14

Private Key Options

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

b. **Figure 3-57 Private Key**



Private Key Output

Namespace:	Namespace
Secret Name:	Secret Name
Key:	Key

Public Key Certificate Options

Key Usage

Critical:	
Value(s):	DIGITAL_SIGNATURE ×

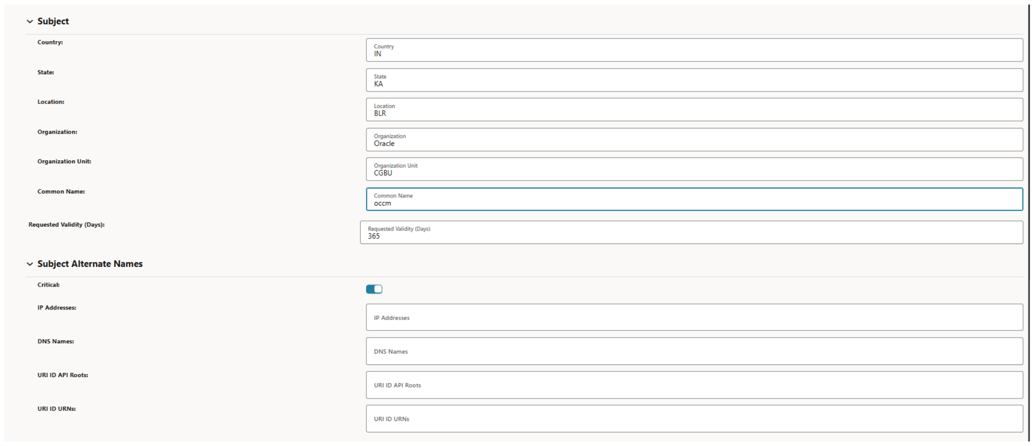
Extended Key Usage

Critical:	
Value(s):	CLIENT_AUTH × SERVER_AUTH ×

Basic Constraints

Critical:	
Value:	END_ENTITY

c. **Figure 3-58 Subject**



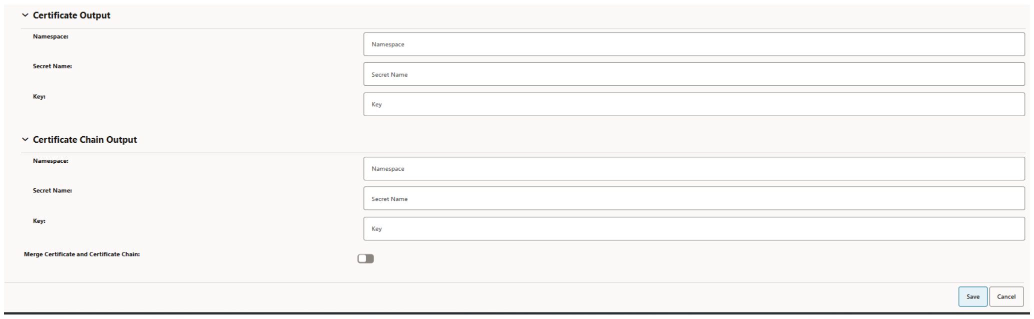
Subject

Country:	IN
State:	KA
Location:	BLR
Organization:	Oracle
Organization Unit:	CGBU
Common Name:	occm
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

d. **Figure 3-59 Certificate Output**



Certificate Output

Namespace:	Namespace
Secret Name:	Secret Name
Key:	Key

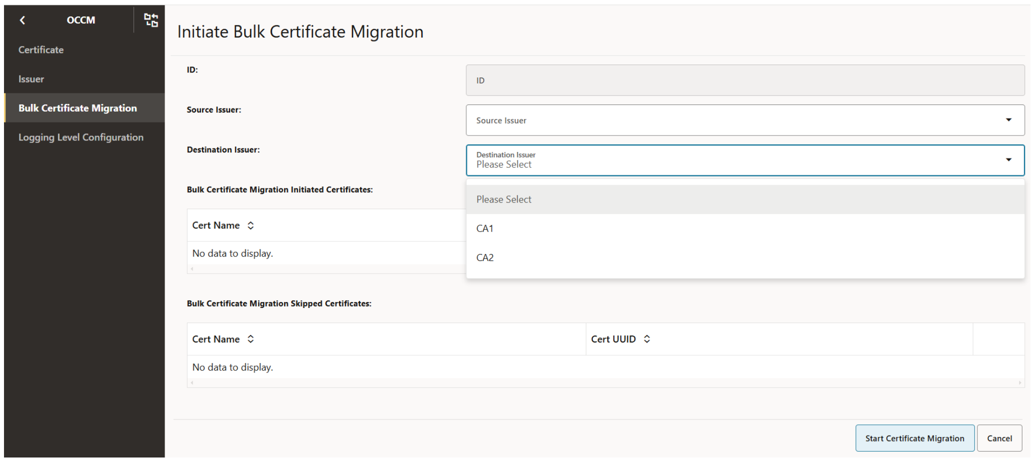
Certificate Chain Output

Namespace:	Namespace
Secret Name:	Secret Name
Key:	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 3-60 Initiate Bulk Certificate Migration



OCCM
Certificate
Issuer
Bulk Certificate Migration
Logging Level Configuration

Initiate Bulk Certificate Migration

ID:	ID
Source Issuer:	Source Issuer
Destination Issuer:	Please Select

Bulk Certificate Migration Initiated Certificates:

Cert Name
No data to display.

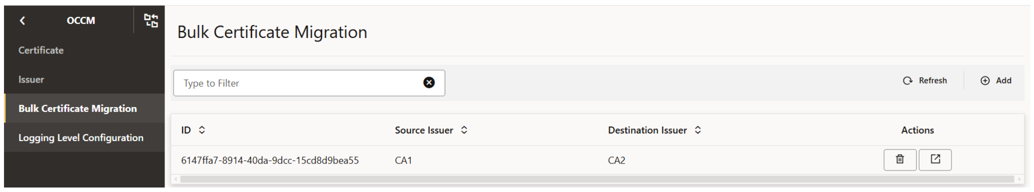
Bulk Certificate Migration Skipped Certificates:

Cert Name	Cert UUID
No data to display.	

Start Certificate Migration
Cancel

   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 3-61 Bulk Certificate Migration

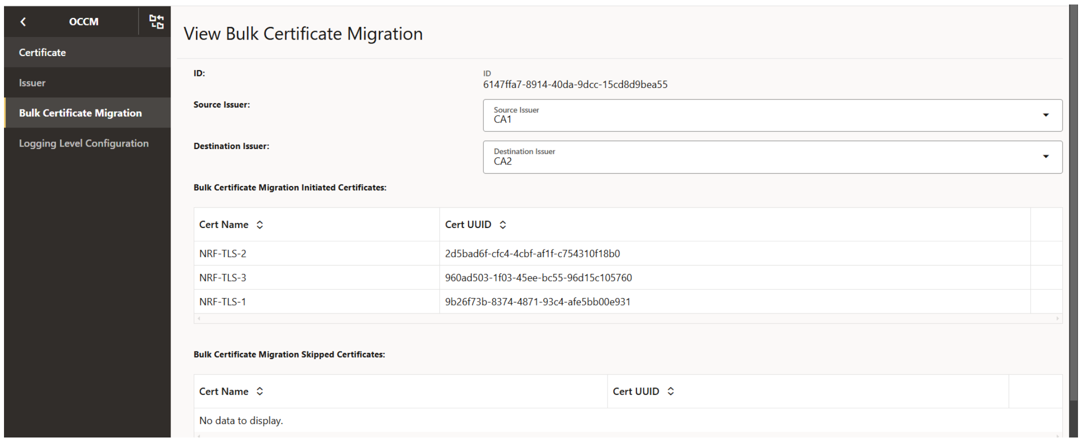


Bulk Certificate Migration

ID	Source Issuer	Destination Issuer	Actions
6147ffa7-8914-40da-9dcc-15cd8d9bea55	CA1	CA2	

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

Figure 3-62   View Bulk Certificate Migration



View Bulk Certificate Migration

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

Source Issuer: Source Issuer CA1

Destination Issuer: Destination Issuer CA2

Bulk Certificate Migration Initiated Certificates:

Cert Name	Cert UUID
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 UUID
No data to display.	

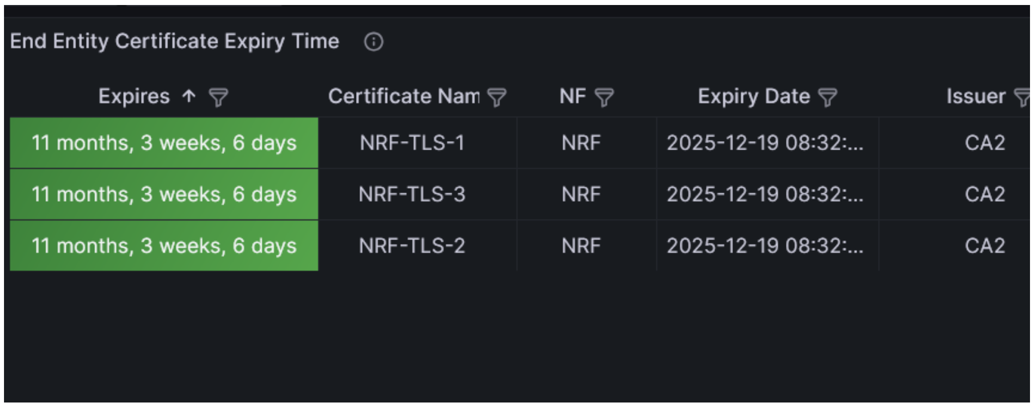
5. Check the Grafana dashboard to verify the migration status of each certificate displayed under the field *Bulk Certificate Migration Initiated certificates*:

   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:

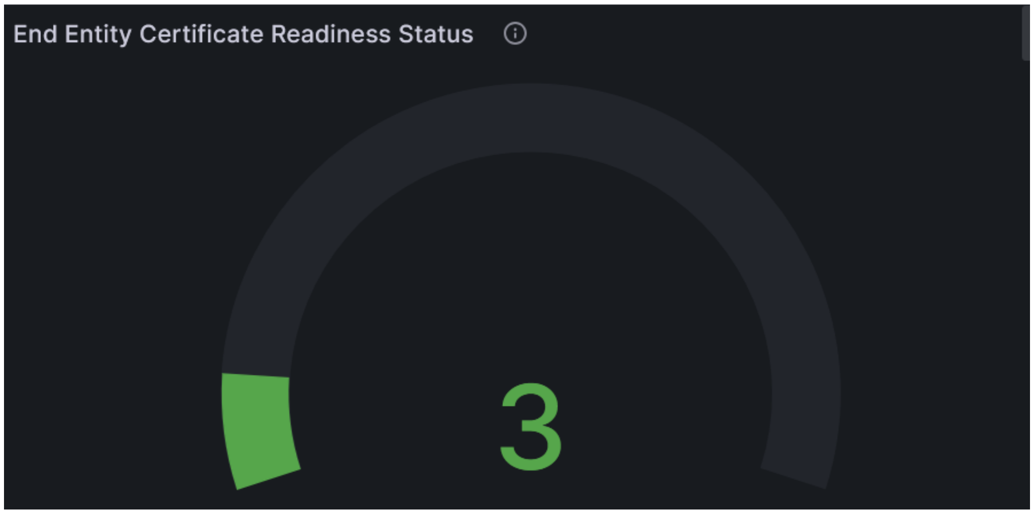
Figure 3-63   End Entity Certificate Expiry Time



End Entity Certificate Expiry Time

Expires	Certificate Nam	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

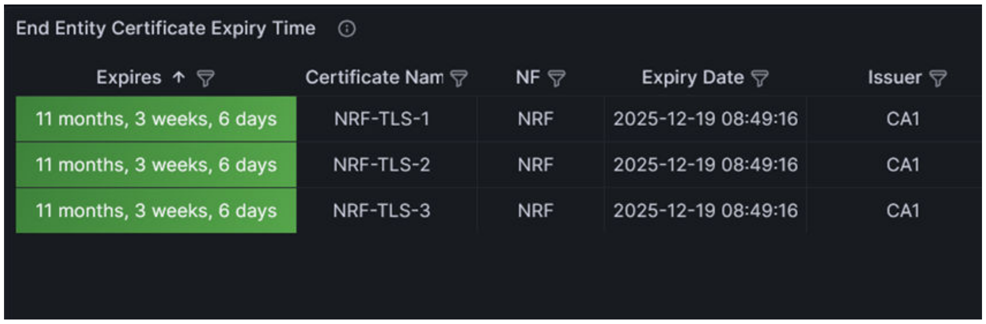
Figure 3-64 End Entity Certificate Readiness Status



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 3-65 End Entity Certificate Expiry Time Failed

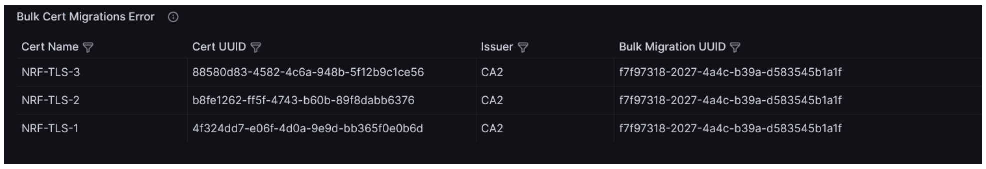


End Entity Certificate Expiry Time

Expires	Certificate Nam	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 3-66 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-e06f-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.

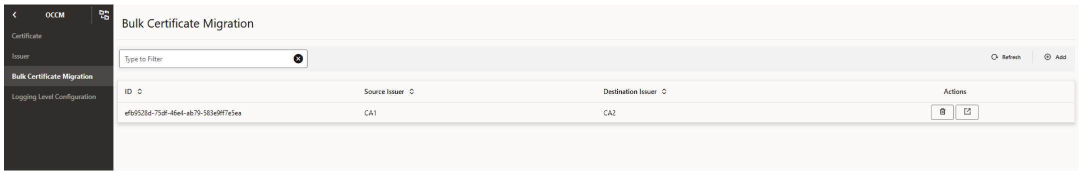
- Another migration can be initiated to retry for all the failed certificates from the source to the destination issuer.

3.5.6.2 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

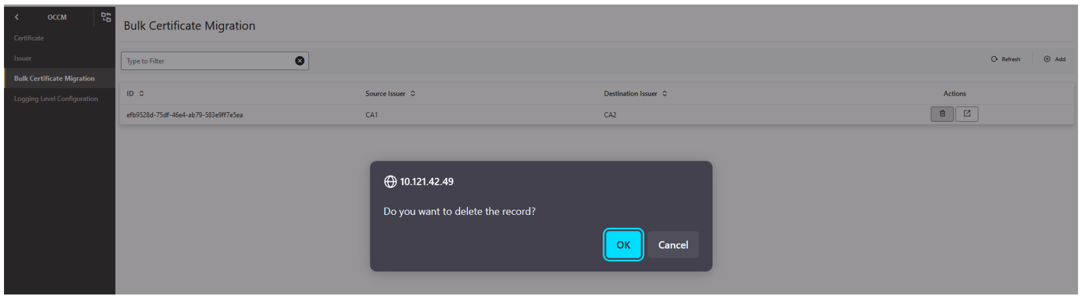

OCCM
Certificate
Issuer
Bulk Certificate Migration
Logging Level Configuration
Bulk Certificate Migration
Type to Filter
Refresh
Add

ID	Source Issuer	Destination Issuer	Actions
efb9528d-75df-46e4-ab79-583e9ff7e5ea	CA1	CA2	

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


OCCM
Certificate
Issuer
Bulk Certificate Migration
Logging Level Configuration
Bulk Certificate Migration
Type to Filter
Refresh
Add
OK
Cancel

ID	Source Issuer	Destination Issuer	Actions
efb9528d-75df-46e4-ab79-583e9ff7e5ea	CA1	CA2	

10.121.42.49

Do you want to delete the record?

3.6 OCCM Retry on Failure

OCCM supports retry on encountering failures during the certificate 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.

3.7 Network Policies

Network Policies are an application-centric construct that allows 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 wherein 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.

3.8 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 `Overwrite Secret` 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": "eyJhbGc...O9l2Q",
   ```

```
        "expires_in": 300,
        "refresh_expires_in": 1800,
        "refresh_token": "eyJhbG...5vKPF-ZIg",
        "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",
    "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": 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

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	occm_config_http_requests_total{ application="occm", container="occm", endpoint="cnc-metrics", helm_sh_chart="occm-25.1.100", 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" }

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	occm_config_http_responses_total{ application="occm", container="occm", endpoint="cnc-metrics", helm_sh_chart="occm-25.1.100", 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"}

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	• certUuid • certName • nfType • requestType • issuerName • caServer

Example	occm_cmp_requests_total{ application="occm", caServer="http://ca90-openssl-mock.occncc-thrust5-01.svc.thrust5:8083", certName="NRFTLS-47", certUuid="c0578b02-caab-454a-bd97-422b0e1c575b", container="occm", endpoint="cnc-metrics", helm_sh_chart="occm-25.1.100", 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"}
	occm_cmp_requests_total{ application="occm", caServer="http://ca90-openssl-mock.occncc-thrust5-01.svc.thrust5:8083", certName="NRFTLS-47", certUuid="c0578b02-caab-454a-bd97-422b0e1c575b", container="occm", endpoint="cnc-metrics", helm_sh_chart="occm-25.1.100", 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"}

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	• certUuid • certName • nfType • requestType • status • statusCode • issuerName • caServer

Example	occm_cmp_responses_total{ application="occm", caServer="http://ca90-openssl-mock.occncc-thrust5-01.svc.thrust5:8083", certName="NRFTLS-47", certUuid="c0578b02-caab-454a-bd97-422b0e1c575b", container="occm", endpoint="cnc-metrics", helm_sh_chart="occm-25.1.100", 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"}
	occm_cmp_responses_total{ application="occm", caServer="http://ca90-openssl-mock.occncc-thrust5-01.svc.thrust5:8083", certName="NRFTLS-47", certUuid="c0578b02-caab-454a-bd97-422b0e1c575b", container="occm", endpoint="cnc-metrics", helm_sh_chart="occm-25.1.100", 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"}

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	• certUuid • certName • nfType • issuerName • certPurpose
Example	occm_cmp_identity_cert_expiration_seconds{ application="occm", certName="OCCM-HTTPS-1", certPurpose="HTTPS-RA", certUuid="164dea8d-a54e-4556-81d0-f0a923030416", container="occm", endpoint="cnc-metrics", helm_sh_chart="occm-25.1.100-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

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	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-25.1.100-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

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	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-25.1.100-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

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-25.1.100-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"} 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-25.1.100", namespace="occm-ns", nfType="OCCM", pod="occm-occm-ccfcbbbfb-nl22k", 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-25.1.100", 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-25.1.100", 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	occm_secret_event_total{app_kubernetes_io_application="occm", app_kubernetes_io_component="occm", app_kubernetes_io_engVersion="25.1.100-COCCM-1332-240722084346-939119a5", app_kubernetes_io_instance="occm", app_kubernetes_io_managed_by="Helm", app_kubernetes_io_microservice="occm", app_kubernetes_io_mktgVersion="25.1.100.0.0", app_kubernetes_io_name="occm", app_kubernetes_io_part_of="occm", app_kubernetes_io_vendor="Oracle", app_kubernetes_io_version="25.1.100.0.0", application="occm", belongs="certificate-other", container="occm", endpoint="cnc-metrics", event="deleted", exported_namespace="occm-ns", helm_sh_chart="occm-25.1.100-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"}

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 OccmCmpIdentityCertExpirationMinor

Table 7-2 OccmCmpIdentityCertExpirationMinor

Field	Details
Description	CMP Identity (OCCM) certificate has expired. The certificate {{$labels.certName}} used by {{$labels.nfType}} for {{$labels.certPurpose}} 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	Information that certificate is going to expire within 90 days. The alert is cleared when the certificate is renewed so that the certificate expiry day is below the minor threshold or when the certificate expiry day crosses the major threshold, in this case the alert is raised. Note: The threshold is configurable in the occm_alertingrules_<version>.yaml file. Steps: 1. Check the certificate configuration to renew before the expiry day. 2. If this is unexpected, contact My Oracle Support.

7.2 OccmCmpIdentityCertExpirationMajor

Table 7-3 OccmCmpIdentityCertExpirationMajor

Field	Details
Description	CMP Identity (OCCM) certificate has expired. 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.) OccmCmpIdentityCertExpirationMajor

Field	Details
Recommended Actions	Information that certificate is going to expire within 30 days. The alert is cleared when the certificate is renewed or when the certificate expiry days crosses the critical threshold, in which case the alert is raised. Note: The threshold is configurable in the occm_alertingrules_<version>.yaml file. Steps: 1. Check the certificate configuration to renew before the expiry day. 2. Refer to the application logs on Kibana and filter based on occm service name. Check for ERROR WARNING logs related to the thread exceptions. 3. Perform the resolution steps depending on the failure reason. 4. If this is unexpected, contact My Oracle Support.

7.3 OccmCmpIdentityCertExpirationCritical

Table 7-4 OccmCmpIdentityCertExpirationCritical

Field	Details
Description	CMP Identity (OCCM) certificate has expired. The certificate {{$labels.certName}} used by {{$labels.nfType}} for {{$labels.certPurpose}} will expire within one week.
Summary	namespace: {{$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 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
Recommended Actions	Information that Certificate is going to expire within one week. The alert is cleared when the certificate is renewed. Note: The threshold is configurable in the occm_alertingrules_<version>.yaml file. Steps: 1. Check the certificate configuration to renew before the expiry day. 2. Refer to the application logs on Kibana and filter based on occm service name. Check for ERROR WARNING logs related to the thread exceptions. 3. Perform the resolution steps depending on the failure reason. 4. If this is unexpected, contact My Oracle Support.

7.4 OccmCmpIdentityCertExpired

Table 7-5 OccmCmpIdentityCertExpired

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 {{$labels.certName}} used by {{$labels.nfType}} for {{$labels.certPurpose}} is expired.
Summary	'namespace: {{$labels.namespace}}, podname: {{$labels.pod}}, timestamp: {{ with query "time()" }}{{ . | first | value | humanizeTimestamp }}{{ end }}: The certificate {{$labels.certName}} used by {{$labels.nfType}} for {{$labels.certPurpose}} 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
Recommended Actions	Information that the certificate has expired. The alert is cleared when the certificate is recreated. Steps: 1. Refer to the application logs on Kibana and filter based on occm service name. Check for ERROR WARNING logs related to the thread exceptions. 2. Perform the following steps if recreate fails and certificates are expired: a. Check logs to identify the root cause. The possible cause may be CA connection failure. In this case operator must manually configure the CMP Identity certificate. b. Get the kubernetes secret name corresponding to OCCM key and certificate location from the mapped issuer. This information is present under CMP client authentication options for Other Cert section of the issuer. c. Manually create CMP Identity (OCCM) certificate and update the secret. d. Manual recreation of certificate can be triggered when CA connection resumes. e. To renew expired certificate, see "Expired Certificate Detection" section in *Oracle Communications Cloud Native Core, Certificate Management Troubleshooting Guide.* 3. If this is unexpected, contact My Oracle Support.

7.5 OccmEndEntityCertExpirationMinor

Table 7-6 OccmEndEntityCertExpirationMinor

Field	Details
Description	End Entity (NF) certificate has expired. 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
Recommended Actions	Information that certificate is going to expire within 90 days. The alert is cleared when the certificate is renewed so that the certificate expiry day is below the minor threshold or when the certificate expiry day crosses the major threshold, in this case the alert is raised. Note: The threshold is configurable in the occm_alertingrules_<version>.yaml file. Steps: 1. Check the certificate configuration to renew before the expiry day. 2. If this is unexpected, contact My Oracle Support.

7.6 OccmEndEntityCertExpirationMajor

Table 7-7 OccmEndEntityCertExpirationMajor

Field	Details
Description	End Entity (NF) certificate has expired. 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	Information that Certificate is going to expire within 30 days. The alert is cleared when the certificate is renewed or when the certificate expiry day crosses the critical threshold,, in this case the alert is raised. Note: The threshold is configurable in the occm_alertingrules_<version>.yaml file. Steps: 1. Check the certificate configuration to renew before the expiry day. 2. Refer to the application logs on Kibana and filter based on occm service names. Check for ERROR WARNING logs related to thread exceptions. 3. Perform the resolution steps depending on the failure reason. 4. If this is unexpected, contact My Oracle Support.

7.7 OccmEndEntityCertExpirationCritical

Table 7-8 OccmEndEntityCertExpirationCritical

Field	Details
Description	End Entity (NF) certificate has expired. The certificate {{$labels.certName}} used by {{$labels.nfType}} for {{$labels.certPurpose}} will expire within one week.
Summary	'namespace: {{$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 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
Recommended Actions	Information that Certificate is going to expire within one week. The alert is cleared when the certificate is renewed. Note: The threshold is configurable in the occm_alertingrules_<version>.yaml file. Steps: 1. Check the certificate configuration to renew before the expiry day. 2. Refer to the application logs on Kibana and filter based on occm service names. Check for ERROR WARNING logs related to thread exceptions. 3. Perform the resolution steps depending on the failure reason. 4. If this is unexpected, contact My Oracle Support.

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 {{$labels.certName}} used by {{$labels.nfType}} for {{$labels.certPurpose}} is expired'
Summary	'namespace: {{$labels.namespace}}, podname: {{$labels.pod}}, timestamp: {{ with query "time()" }}{{ . | first | value | humanizeTimestamp }}{{ end }}: The certificate {{$labels.certName}} used by {{$labels.nfType}} for {{$labels.certPurpose}} 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
Recommended Actions	Information that certificate has expired. The alert is cleared when the certificate is recreated. Steps: 1. Refer to the application logs on Kibana and filter based on occm service name. Check for ERROR WARNING logs related to the thread exceptions. 2. Perform the following steps if recreate fails and certificates are expired: a. Check logs to identify the root cause. The possible cause may be CA connection failure. b. As a resolution perform the recreate operation when is CA is accessible. Alert will be cleared once recreation is successful. c. If CA is still down then manually create the End-Entity (NF) certificate and update the details in secret, which is automatically monitored by OCCM. d. Manual recreation of certificate can be triggered when CA connection resumes. e. To renew expired certificate, see "Expired Certificate Detection" section in *Oracle Communications Cloud Native Core, Certificate Management Troubleshooting Guide.*. 3. If this is unexpected, contact My Oracle Support.

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.
Recommended Actions	The alert is cleared when the occm service is available. Steps: 1. Check the orchestration logs of occm service and check for liveness or readiness probe failures. 2. Refer to the application logs on Kibana and filter based on occm service names. Check for ERROR WARNING logs related to thread exceptions. 3. Depending on the failure reason, take the resolution steps. 4. In case the issue persists, contact My Oracle Support.

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

Table 7-11 (Cont.) OccmMemoryUsageMinorThreshold

Field	Details
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.
Recommended Actions	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. Note: The threshold is configurable in the occm_alertingrules_<version>.yaml file. Steps: 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. Depending on the failure reason, take the resolution steps. 3. If this is unexpected, contact My Oracle Support.

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	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 Note: The threshold is configurable in the occm_alertingrules_<version>.yaml file. Steps: 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. Depending on the failure reason, take the resolution steps. 3. If this is unexpected, contact My Oracle Support.

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.
Recommended Actions	The alert gets cleared when the memory utilization falls below the Critical Threshold.Note : The threshold is configurable in the alerts.yaml Note: The threshold is configurable in the occm_alertingrules_<version>.yaml file. Steps: 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. Depending on the failure reason, take the resolution steps. 3. If this is unexpected, contact My Oracle Support.

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
Recommended Actions	Information regarding CPU usage If it is above 70% The alert gets cleared when the CPU usage falls below the Minor Threshold. Note: The threshold is configurable in the occm_alertingrules_<version>.yaml file. Steps: 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. Depending on the failure reason, take the resolution steps. 3. If this is unexpected, contact My Oracle Support.

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	Information that the rate of certificate failure due to CMP command execution error has crossed the threshold. 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. Note: The threshold is configurable in the occm_alertingrules_<version>.yaml file. Steps: 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. Depending on the failure reason, take the resolution steps. 3. If this is unexpected, contact My Oracle Support.

7.15 OccmCMPFailureMajor

Table 7-16 OccmCMPFailureMajor

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	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	occm_cmp_responses_total
Recommended Actions	Information that the rate of certificate failure due to CMP command execution error has crossed the threshold. The alert is cleared when the rate of certificate failure due to CMP command execution error falls below the Major threshold or when the error rate crosses the Critical threshold, in which case the OccmCMPFailureCritical alert is raised. Note: The threshold is configurable in the occm_alertingrules_<version>.yaml file. Steps: 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. Depending on the failure reason, take the resolution steps. 3. If this is unexpected, contact My Oracle Support.

7.16 OccmCMPFailureCritical

Table 7-17 OccmCMPFailureCritical

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	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
Recommended Actions	Information that the rate of certificate failure due to CMP command execution error has crossed the threshold. The alert is cleared when the rate of certificate failure due to CMP command execution error falls below the Critical threshold. Note: The threshold is configurable in the occm_alertingrules_<version>.yaml file. Steps: 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. Depending on the failure reason, take the resolution steps. 3. If this is unexpected, contact My Oracle Support.

7.17 OccmFailureMinor

Table 7-18 OccmFailureMinor

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	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	Information that the rate of OCCM errors has crossed the threshold. 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. Note: The threshold is configurable in the occm_alertingrules_<version>.yaml file. Steps: 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. Depending on the failure reason, take the resolution steps. 3. If this is unexpected, contact My Oracle Support.

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
Recommended Actions	Information that the rate of OCCM errors has crossed the threshold. 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. Note: The threshold is configurable in the occm_alertingrules_<version>.yaml file. Steps: 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. Depending on the failure reason, take the resolution steps. 3. If this is unexpected, contact My Oracle Support.

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
Recommended Actions	Information that the rate of certificate failure due to CMP command execution error has crossed the threshold. The alert is cleared when the rate of certificate failure due to CMP command execution error falls below the Critical threshold. Note: The threshold is configurable in the occm_alertingrules_<version>.yaml file. Steps: 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. Depending on the failure reason, take the resolution steps. 3. If this is unexpected, contact My Oracle Support.

7.20 OccmInputSecretModifyMajor

Table 7-21 OccmInputSecretModifyMajor

Field	Details
Description	**Input secret is modified by non-OCCM user** The Secret {{$labels.secret}} in {{$labels.secretNamespace}} is modified by non-occm user, which is used by {{$labels.name}}.'
Summary	'namespace: {{$labels.namespace}}, podname: {{$labels.pod}}, timestamp: {{ with query "time()" }}{{ . | first | value | humanizeTimestamp }}{{ end }}: The Secret {{$labels.secret}} in {{$labels.secretNamespace}} is modified by non-occm user, which is used by {{$labels.name}} and {{$labels.type}}.'
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	Information that the input secret is modified by non-OCCM user. Steps: 1. Check input secrets for any modifications. 2. See the alert label for the namespace and to see which secret alert is triggered. 3. Update input secrets with correct data, if require. 4. If this is unexpected, contact My Oracle Support.

7.21 OccmOutputSecretModifyMinor

Table 7-22 OccmOutputSecretModifyMinor

Field	Details
Description	**Output secret is modified by non-OCCM user** The Secret {{$labels.secret}} in {{$labels.secretNamespace}} is modified by non-occm user, which is used by {{$labels.name}}.'
Summary	'namespace: {{$labels.namespace}}, podname: {{$labels.pod}}, timestamp: {{ with query "time()" }}{{ . | first | value | humanizeTimestamp }}{{ end }}: The Secret {{$labels.secret}} in {{$labels.secretNamespace}} is modified by non-occm user, which is used by {{$labels.name}} and {{$labels.type}}.'
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
Recommended Actions	Information that the output secret is modified by non-OCCM user. Steps: 1. Check output secrets for any modifications. 2. Automatic recreation will be triggered if certificate which is modified does not match with cert config. 3. Updatation of validity will be done, if the modified certificate validation is successful with certification configuration. No recreation will be triggered in this case. 4. If this is unexpected, contact My Oracle Support.

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
Recommended Actions	Information that the Kubernetes resources (secret or namespace) are deleted by non-OCCM user. Steps: 1. Check output secrets for any deletion. 2. Automatic recreation of certificate will be triggered, if secret is deleted. 3. if namespace is deleted, then automatic recreation of certificate does not happen and the operator must delete the certificate configuration from the OCCM which are associated with that namespace. 4. If this is unexpected, contact My Oracle Support.

7.23 OCCM Alert and MIB Configuration in Prometheus

CNE supporting Prometheus HA

This section describes the measurement based Alert rules configuration for OCCM in Prometheus. You must use the updated `occm_alerting_rules_promha_<version>.yaml file`.

Run the following command to create ot update the PrometheusRule resource specified in the alert YAML file:

```
$ kubectl apply -f occm_alerting_rules_promha_<version>.yaml
```

Disabling Alerts

This section describes the procedure to disable the alerts in OCCM. To disable alerts:

1. Edit `occm_alerting_rules_promha_<version>.yaml` file to remove specific alert.

2. Remove complete content of the specific alert from the `occm_alerting_rules_promha_<version>.yaml` file.
   For example, ff you want to remove OccmServiceDown alert, remove the complete content:

```
## ALERT SAMPLE START##
- alert: OccmServiceDown
      annotations:
        description: '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'
      expr: absent(up{pod=~".*occm.*", namespace="occm-ns"}) or
(up{pod=~".*occm.*", namespace="occm-ns"}) == 0
      labels:
        severity: critical
        oid: "1.3.6.1.4.1.323.5.3.54.1.2.7004"
        namespace: ' {{ $labels.namespace }} '
        podname: ' {{$labels.pod}} '
## ALERT SAMPLE END##
```

3. Perform Alert configuration.

Validating Alerts

Configure and Validate Alerts in Prometheus Server. Refer to OCCM Alert Configuration for procedure to configure the alerts.

After configuring the alerts in Prometheus server, a user can verify that by following steps:

1. Open the Prometheus server from your browser using the <IP>:<Port>
2. Navigate to Status and then Rules
3. Search OCCM. OCCMAlerts list is displayed.

Note

If you are unable to see the alerts, it means that the alert file has not loaded in a format which the Prometheus server accepts. Modify the file and try again.

Configuring SNMP-Notifier

Configure the IP and port of the SNMP trap receiver in the SNMP Notifier using following procedure:

1. Run the following command to edit the deployment:

```
kubectl edit deploy <snmp_notifier_deployment_name> -n <namespace>
```

   Example:

```
$ kubectl edit deploy occne-snmp-notifier -n occne-infra
```

2. Edit the destination as follows:

   ```
   --snmp.destination=<destination_ip>:<destination_port>
   ```

   Example:

   ```
   --snmp.destination=10.75.203.94:162
   ```

MIB Files for OCCM

There are two MIB files which are used to generate the traps. The user need to update these files along with the Alert file in order to fetch the traps in their environment.

- `occm_mib_tc_<version>.mib`: This is considered as OCCM top level mib file, where the Objects and their data types are defined
- `occm_mib_<version>.mib`: This file fetches the Objects from the top level mib file and based on the Alert notification, these objects can be selected for display.

Note

MIB files are packaged along with OCCM CSAR package. Download the file from MOS. For more information, see *Oracle Communications Cloud Native Core, Certificate Management Installation, Upgrade, and Fault Recovery Guide.*

8 OCCM KPIs

This section describes the KPIs available for OCCM.

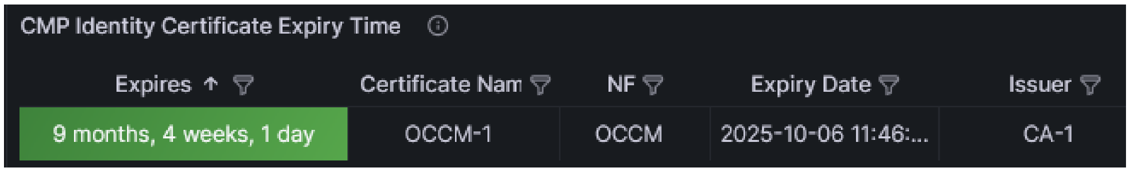
8.1 CMP Identity (OCCM) Certificate Expiry Time

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 Nam	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 <= 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

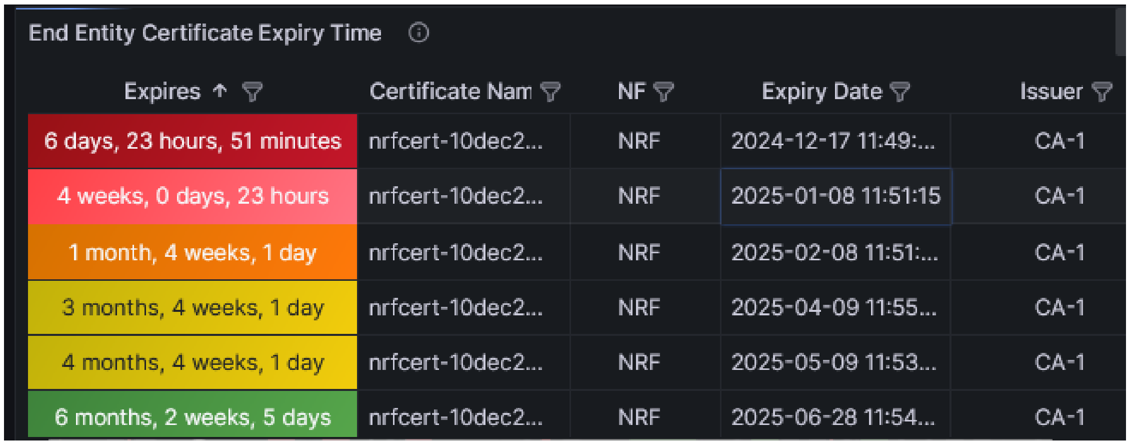
Green :- Certificates not expiring sooner

8.2 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{namespace="$namespace"} * 1000 != 0 Expires column: ((occm_end_entity_cert_expiration_seconds{namespace="$namespace"} != 0)-time())*1000

OCCM KPI Dashboard



End Entity Certificate Expiry Time

Expires	Certificate Nam	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

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

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

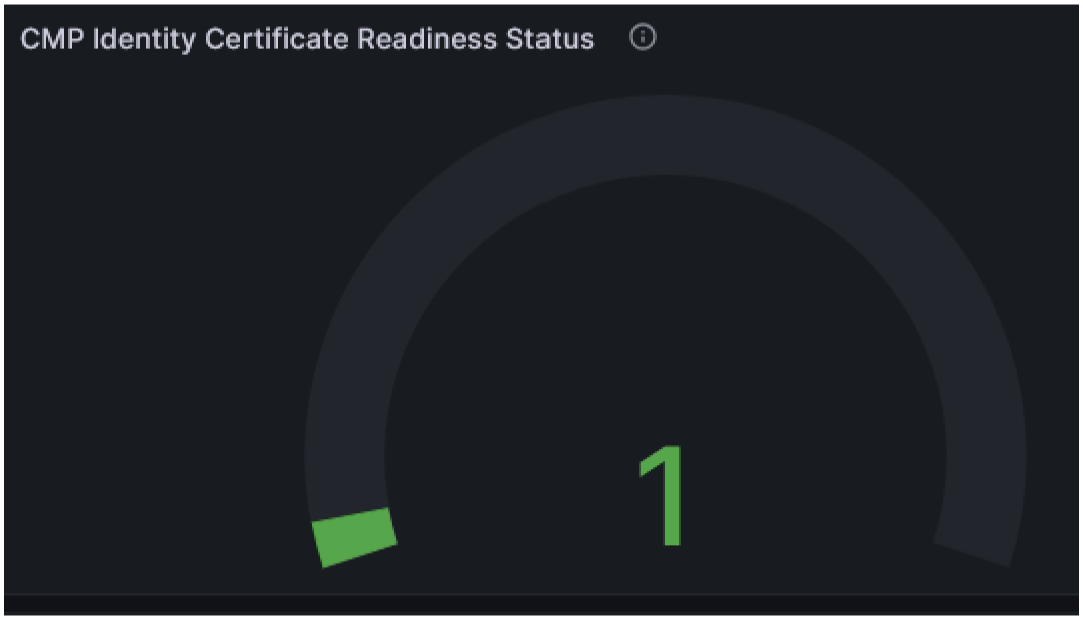
Table 8-3 CMP Identity (OCCM) Certificate Readiness Status

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

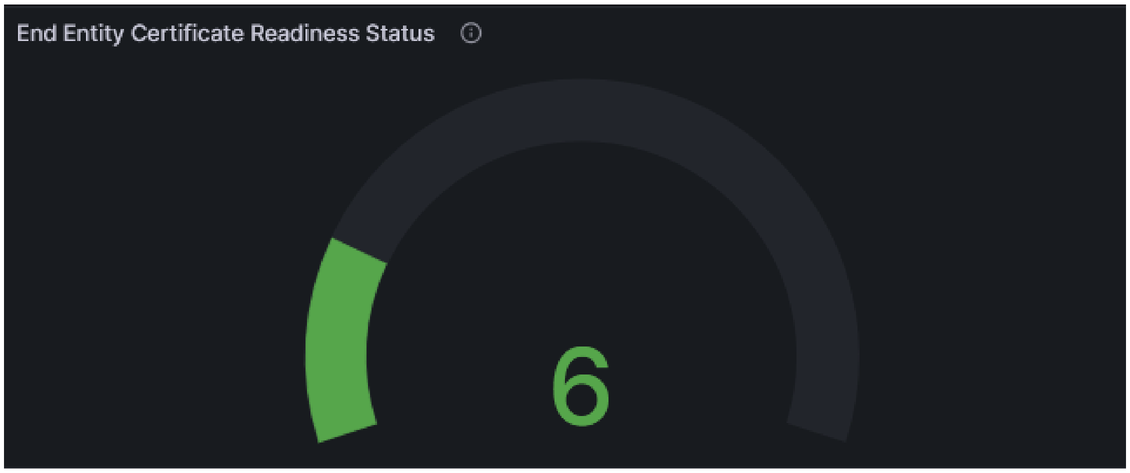
Table 8-4 End Entity (NF) Certificate Readiness Status

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{namespace="$namespace"} == 1) (Color:Orange)
	Ready:count(occm_end_entity_cert_status{namespace="$namespace"} == 2) (Color:Green)
	Failed:count(occm_end_entity_cert_status{namespace="$namespace"} == 3) (Color:Red)
	Waiting:count(occm_end_entity_cert_status{namespace="$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. 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]))

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. 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]))

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. sum(rate(container_cpu_usage_seconds_total{image!="",namespace="$namespace", pod=~"occm-.*."}[2m])) by(pod)

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	occm_cert_request_status_total{namespace="occm-ns", errorReason= "ERR_SECRET_FAILED"}

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	occm_cert_request_status_total{namespace="occm-ns", errorReason= "ERR_SECRET_EXIST"}

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	occm_cert_request_status_total{namespace="occm-ns", operationType="RENEW"}

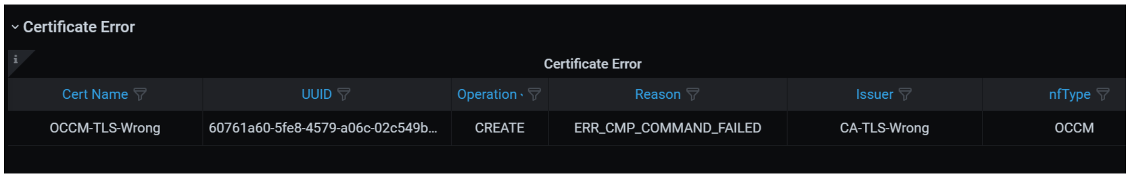
8.21 Certificate Error and Warnings

Table 8-21 Certificate Error and Warnings

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

OCCM KPI Dashboard

Figure 8-5 Certificate Error and Warnings



Certificate Error

Cert Name	UUID	Operation	Reason	Issuer	nfType
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

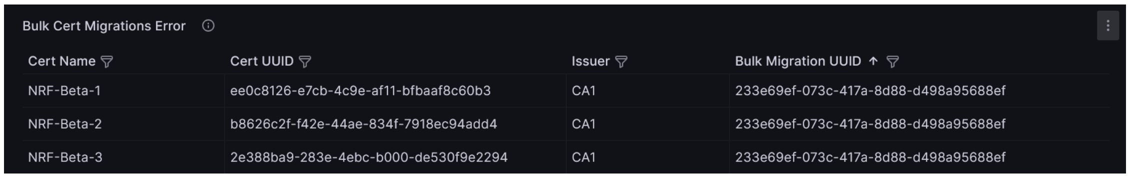
8.22 Bulk Certificate Migration Error

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.

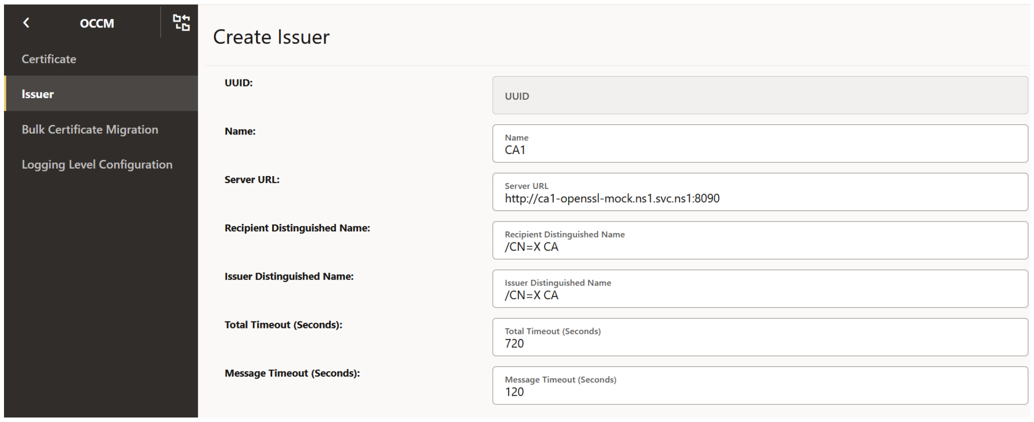
A.1 Certificate Configuration Examples

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. **Create the Issuer**:
   The following screenshots provide a sample configuration for creating the issuer using CNC Console GUI

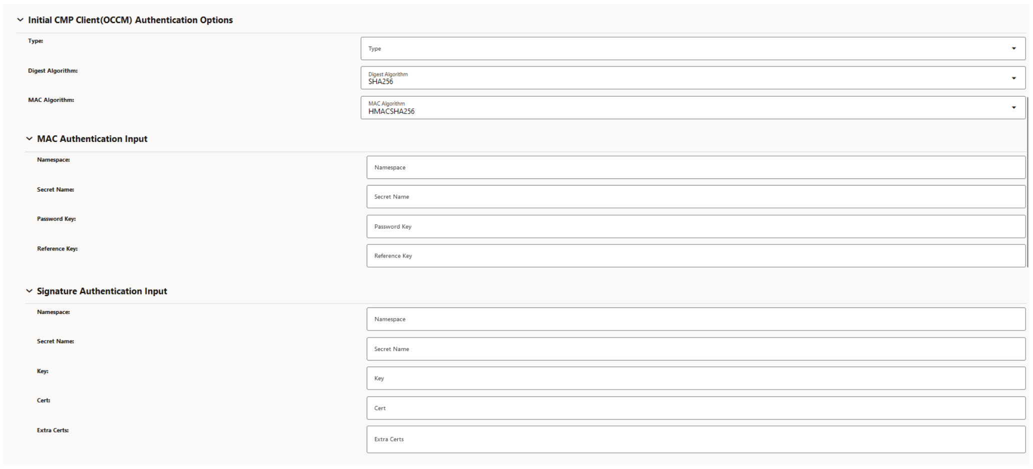
   a. **Figure 7 Create Issuer**



Create Issuer

UUID:	UUID
Name:	Name CA1
Server URL:	Server URL http://ca1-openssl-mock.ns1.svc.ns1:8090
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. **Figure 8 Initial CMP Client (OCCM) Authentication Options**



Initial CMP Client(OCCM) Authentication Options

Type:	Type
Digest Algorithm:	Digest Algorithm SHA256
MAC Algorithm:	MAC Algorithm HMACSHA256

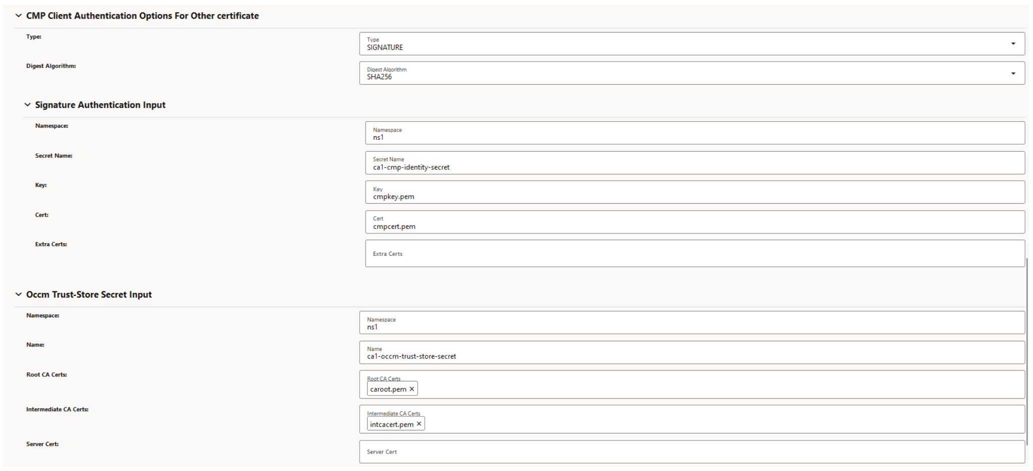
MAC Authentication Input

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. **Figure 9 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

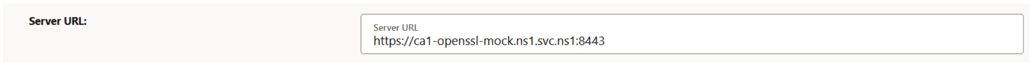
Namespace:	Namespace: ns1
Secret Name:	Secret Name: ca1-cmp-identity-secret
Key:	Key: cmpkey.pem
Cert:	Cert: cmpcert.pem
Extra Certs:	Extra Certs

Occm Trust-Store Secret Input

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

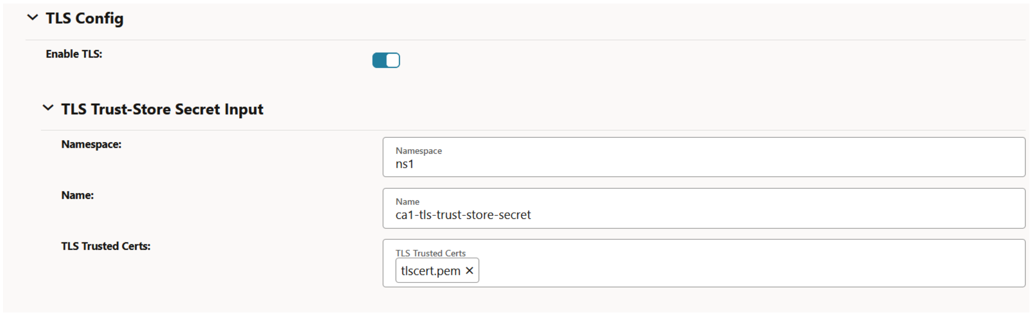
d. To enable HTTPS communication, provide HTTPS scheme in the server URL field and provide the TLS trust store certificates under TLS config.

Figure 10 HTTPS Scheme



Server URL:	Server URL: https://ca1-openssl-mock.ns1.svc.ns1:8443

Figure 11 Enable TLS Config



TLS Config

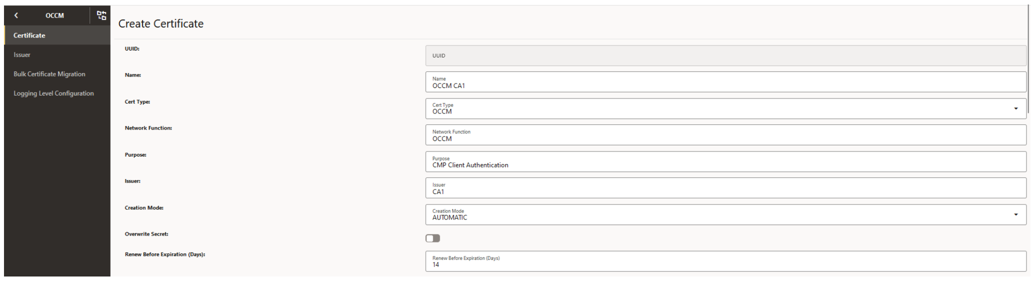
Enable TLS:	●

TLS Trust-Store Secret Input

Namespace:	Namespace: ns1
Name:	Name: ca1-tls-trust-store-secret
TLS Trusted Certs:	TLS Trusted Certs: tlscert.pem ×

2. **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.

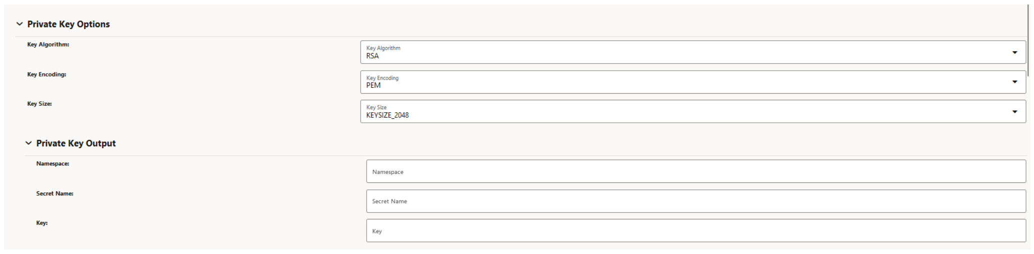
a. **Figure 12 Create CMP Identity (OCCM) Certificate**



Create Certificate

UUID:	UUID
Name:	Name: OCCM CA1
Cert Type:	Cert Type: OCCM
Network Function:	Network Function: OCCM
Purpose:	Purpose: CMP Client Authentication
Issuer:	Issuer: CA1
Creation Mode:	Creation Mode: AUTOMATIC
Overwrite Secret:	○
Renew Before Expiration (Days):	Renew Before Expiration (Days): 14

b. **Figure 13 Private Key Options**



Private Key Options

Key Algorithm: RSA

Key Encoding: PEM

Key Size: KEYSIZE_2048

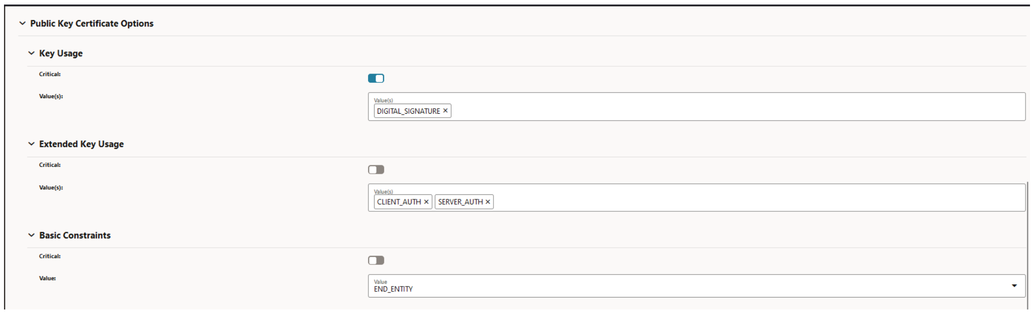
Private Key Output

Namespace: Namespace

Secret Name: Secret Name

Key: Key

c. **Figure 14 Public Key Certificate Options**



Public Key Certificate Options

Key Usage

Critical: ●

Value(s): DIGITAL_SIGNATURE ×

Extended Key Usage

Critical: ○

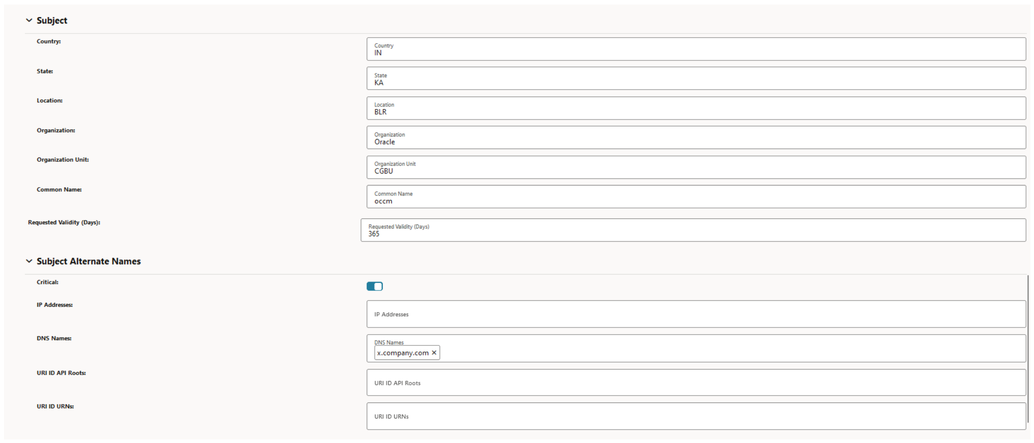
Value(s): CLIENT_AUTH × SERVER_AUTH ×

Basic Constraints

Critical: ○

Value: END_ENTITY

d. **Figure 15 Subject and Subject Alternate Name**



Subject

Country: IN

State: KA

Location: BLR

Organization: Oracle

Organization Unit: CGBU

Common Name: occm

Requested Validity (Days): 365

Subject Alternate Names

Critical: ●

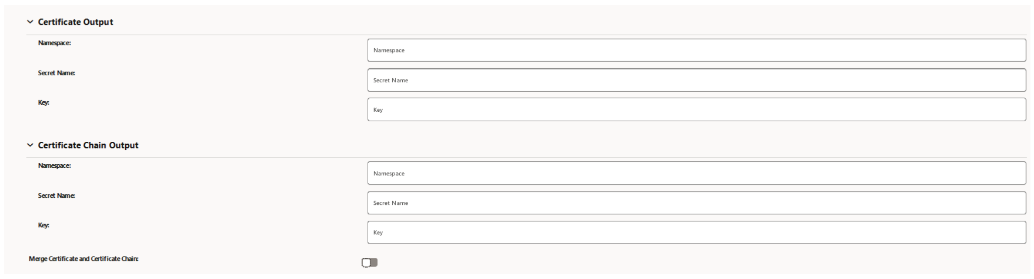
IP Addresses: IP Addresses

DNS Names: x.company.com ×

URI ID API Roots: URI ID API Roots

URI ID URNs: URI ID URNs

e. **Figure 16 Certificate Output and Certificate Chain Output**



Certificate Output

Namespace: Namespace

Secret Name: Secret Name

Key: Key

Certificate Chain Output

Namespace: Namespace

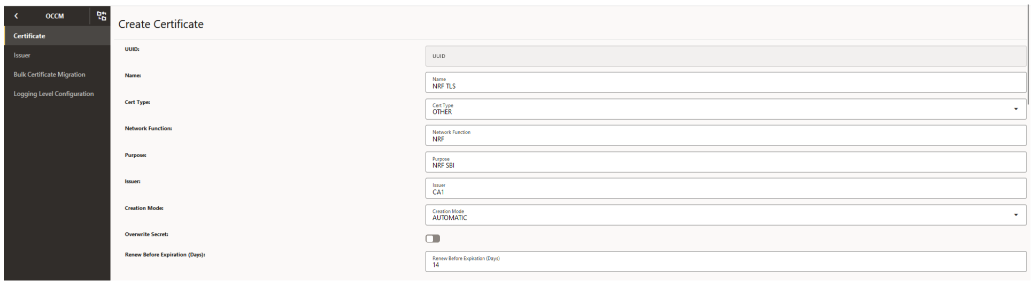
Secret Name: Secret Name

Key: Key

Merge Certificate and Certificate Chain: ○

3. **Create End Entity (NF) Certificate: (PEM encoding):**
The following screenshots provide a sample configuration for creating NF Certificate using CNC Console GUI.

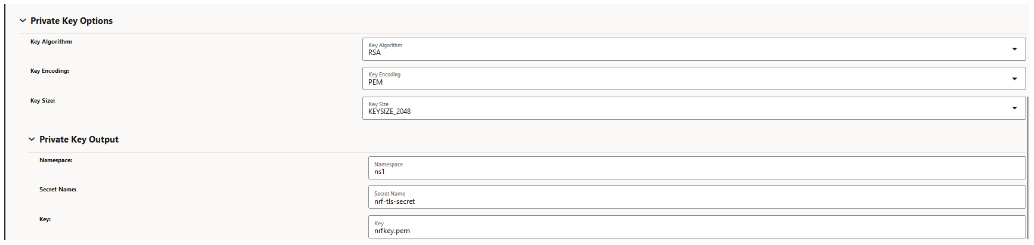
a. **Figure 17 Create End Entity (NF) Certificate**



Create Certificate

UUID:	UUID
Name:	NRF TLS
Cert Type:	OTHER
Network Function:	NRF
Purpose:	NRF SBI
Issuer:	CA1
Creation Mode:	AUTOMATIC
Overwrite Secret:	
Renew Before Expiration (Days):	14

b. **Figure 18 Private Key Options**



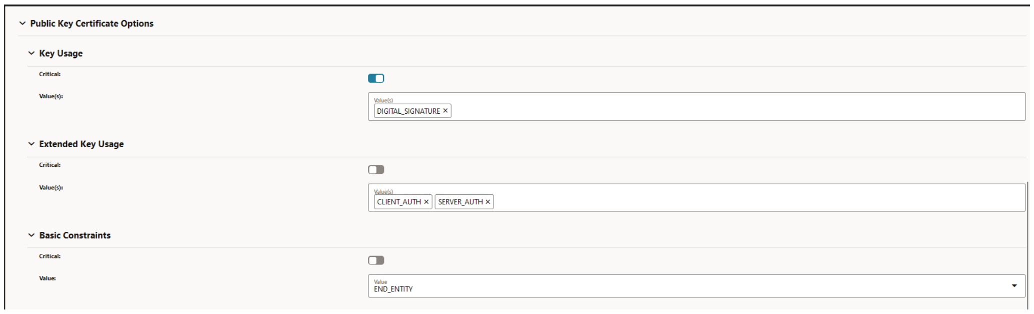
Private Key Options

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

Private Key Output

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

c. **Figure 19 Public Key Options**



Public Key Certificate Options

Key Usage

Critical:	✓
Value(s):	DIGITAL_SIGNATURE

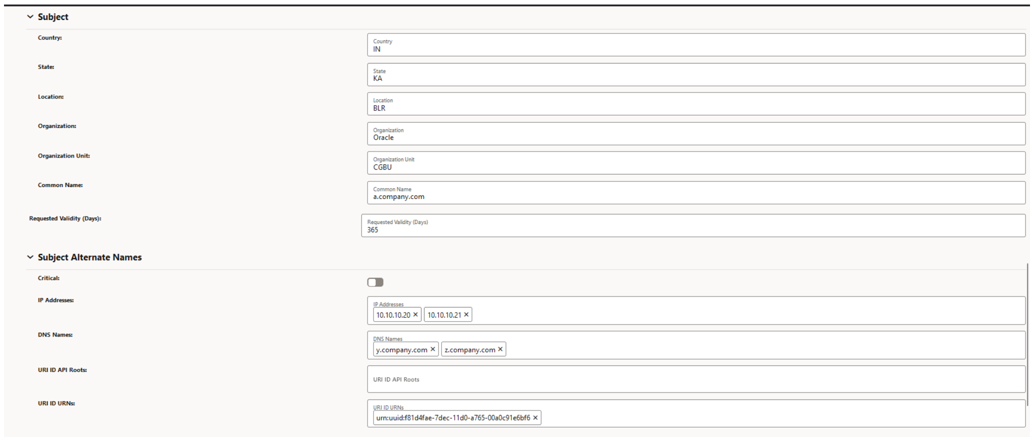
Extended Key Usage

Critical:	
Value(s):	CLIENT_AUTH, SERVER_AUTH

Basic Constraints

Critical:	
Value:	END_ENTITY

d. **Figure 20 Subject and Subject Alternate Names**



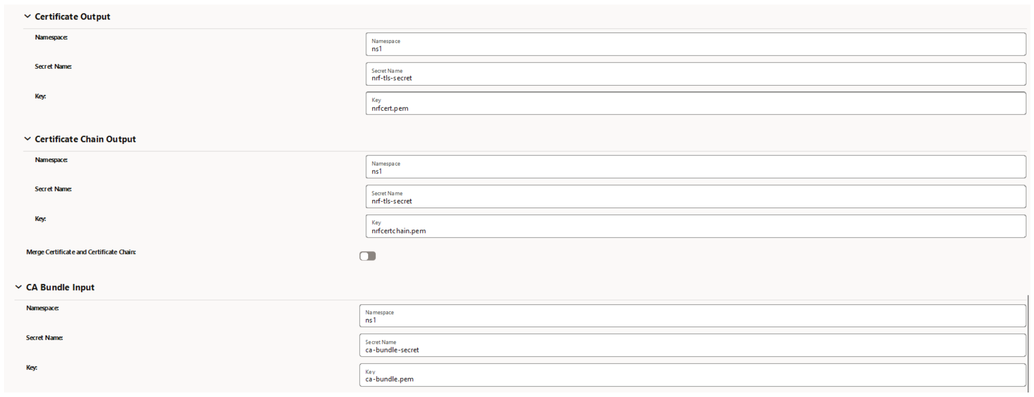
Subject

Country:	IN
State:	KA
Location:	BLR
Organization:	Oracle
Organization Unit:	CGBU
Common Name:	a.company.com
Requested Validity (Days):	365

Subject Alternate Names

Critical:	
IP Addresses:	10.10.10.20, 10.10.10.21
DNS Names:	y.company.com, z.company.com
URI ID API Roots:	
URI ID URNs:	urn:uuid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6

e. **Figure 21 Certificate Output**



Certificate Output

Namespace:	ns1
Secret Name:	nrf-tls-secret
Key:	nrfcert.pem

Certificate Chain Output

Namespace:	ns1
Secret Name:	nrf-tls-secret
Key:	nrfcertchain.pem

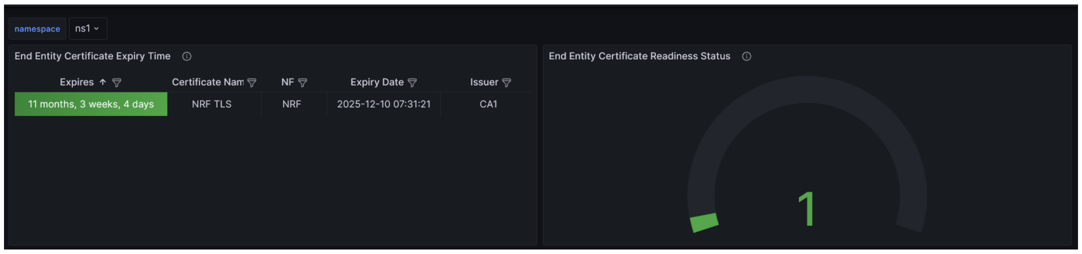
Merge Certificate and Certificate Chain:

CA Bundle Input

Namespace:	ns1
Secret Name:	ca-bundle-secret
Key:	ca-bundle.pem

4. **Check Grafana Dashboard**
   Check the grafana dashboard to view the certificates created.

   Figure 22 End Entity (NF) Sample Grafana Dasboard



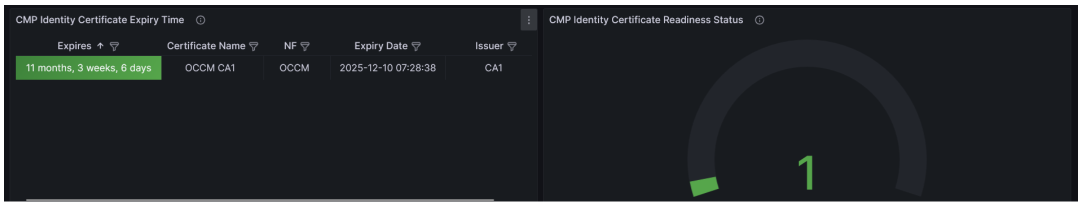
namespace ns1

End Entity Certificate Expiry Time

Expires	Certificate Nam	NF	Expiry Date	Issuer
11 months, 3 weeks, 4 days	NRF TLS	NRF	2025-12-10 07:31:21	CA1

End Entity Certificate Readiness Status

   Figure 23 CMP Identity (OCCM) Sample Grafana Dasboard



CMP Identity Certificate Expiry Time

Expires	Certificate Name	NF	Expiry Date	Issuer
11 months, 3 weeks, 6 days	OCCM CA1	OCCM	2025-12-10 07:28:38	CA1

CMP Identity Certificate Readiness Status

   The screenshot shows 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.

5. **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: LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCkXXXXXXXXXX
  nrfcertchain.pem: LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tXXXXXXXXXXX
  nrfkey.pem: LS0tLS1CRUdJTiBQUklWQVRFIEtFWS0tLS0tCk1XXXXXXXXXXX
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:
      XXXXXXXXX
    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:
         XXXXXXXXX

        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-
```

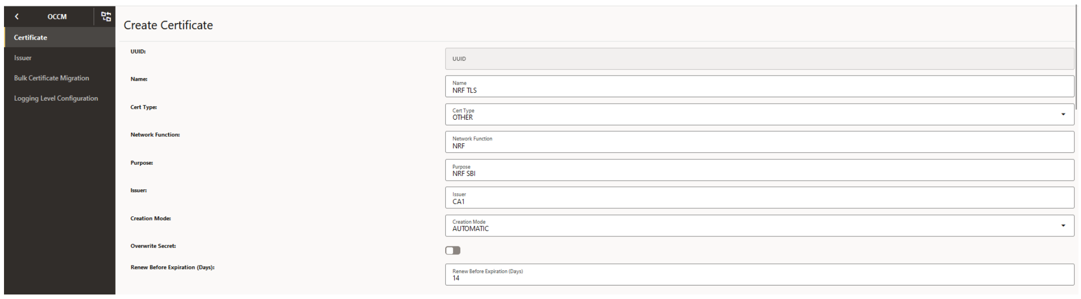
```
a765-00a0c91e6bf6
      X509v3 Subject Key Identifier:
        2B:0D:XXXXXXXXXXXX
      X509v3 Authority Key Identifier:
        20:03:XXXXXXXXXXX
  Signature Algorithm: sha256WithRSAEncryption
  Signature Value:
   XXXXXXXXXXXXXXXXXXXXXXX
```

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 24 Certificate Metadata

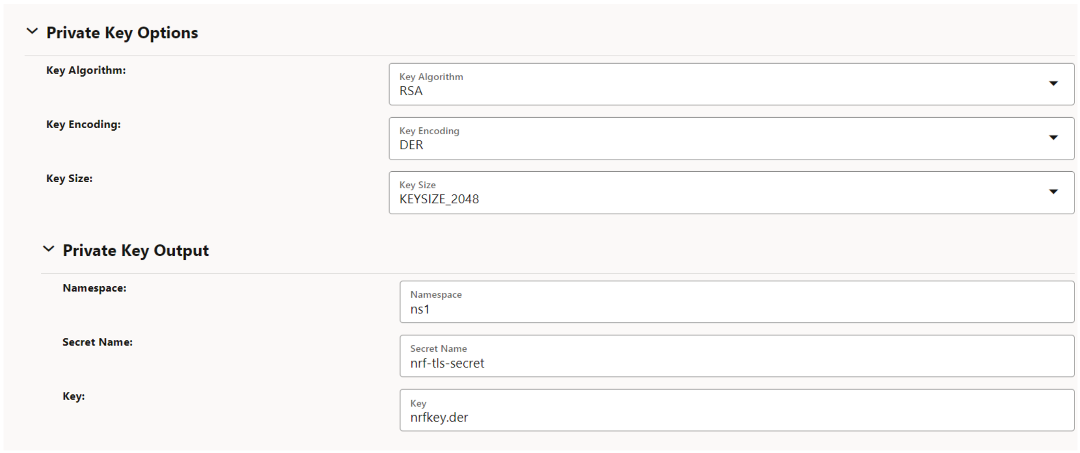


Create Certificate

UUID:	UUID
Name:	Name: NRF TLS
Cert Type:	Cert Type: OTHER
Network Function:	Network Function: NRF
Purpose:	Purpose: NRF SBI
Issuer:	Issuer: CA1
Creation Mode:	Creation Mode: AUTOMATIC
Overwrite Secret:	
Renew Before Expiration (Days):	Renew Before Expiration (Days): 14

2. Private Key Options

Figure 25 Private Key Options



Private Key Options

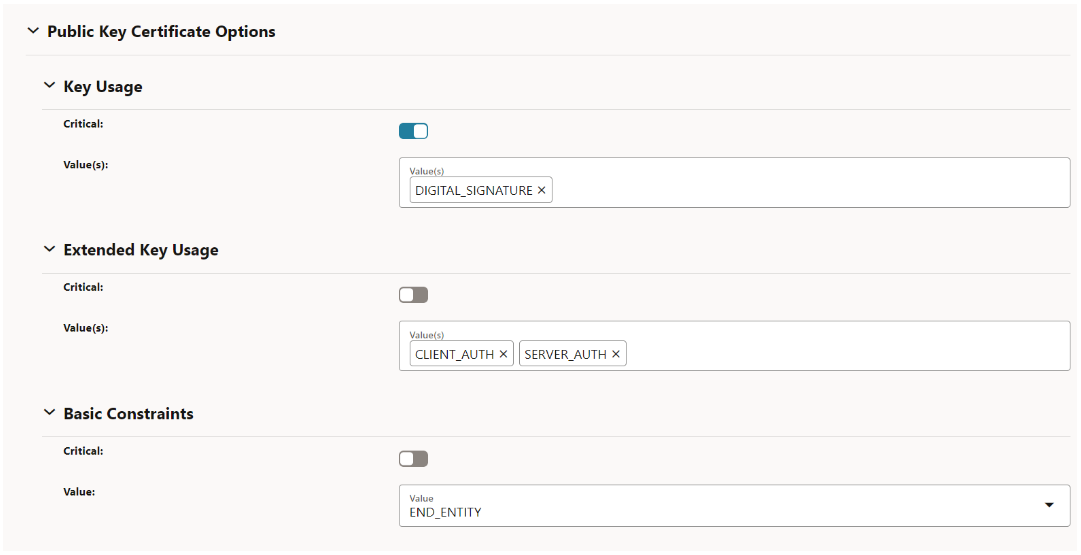
Key Algorithm:	Key Algorithm: RSA
Key Encoding:	Key Encoding: DER
Key Size:	Key Size: KEYSIZE_2048

Private Key Output

Namespace:	Namespace: ns1
Secret Name:	Secret Name: nrf-tls-secret
Key:	Key: nrfkey.der

3. Public Key Certificate Options

Figure 26 Public Key Certificate Options



Public Key Certificate Options

Key Usage

Critical:	on
Value(s):	DIGITAL_SIGNATURE

Extended Key Usage

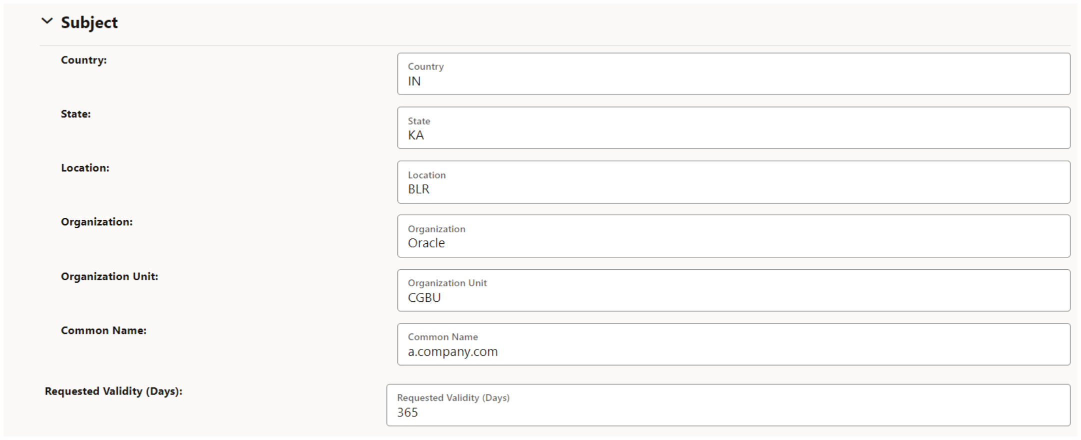
Critical:	off
Value(s):	CLIENT_AUTH, SERVER_AUTH

Basic Constraints

Critical:	off
Value:	END_ENTITY

4. Subject

Figure 27 Subject

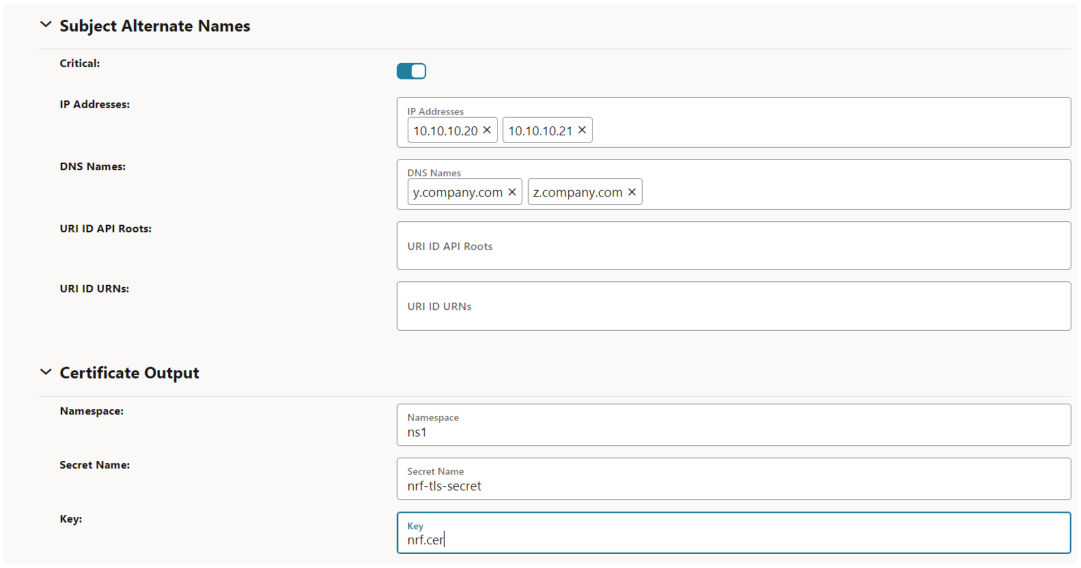


Subject

Country:	IN
State:	KA
Location:	BLR
Organization:	Oracle
Organization Unit:	CGBU
Common Name:	a.company.com
Requested Validity (Days):	365

5. Subject Alternate names

Figure 28 Subject Alternate names



Subject Alternate Names

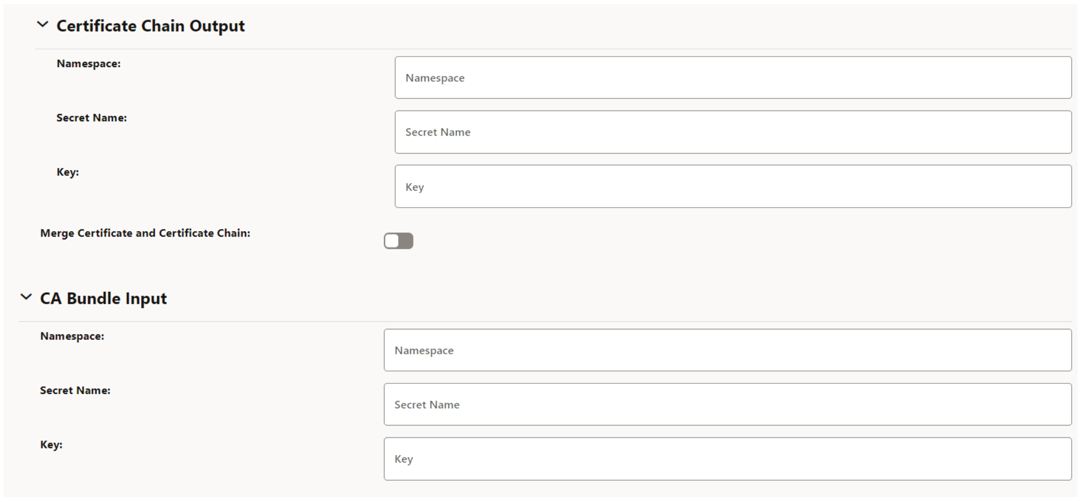
Critical:	On
IP Addresses:	10.10.10.20 ✕ 10.10.10.21 ✕
DNS Names:	y.company.com ✕ z.company.com ✕
URI ID API Roots:	URI ID API Roots
URI ID URNs:	URI ID URNs

Certificate Output

Namespace:	ns1
Secret Name:	nrf-tls-secret
Key:	nrf.cer

6. Optional Certificate chain output and CA bundle input fields

Figure 29 Optional Certificate chain output and CA bundle input fields



Certificate Chain Output

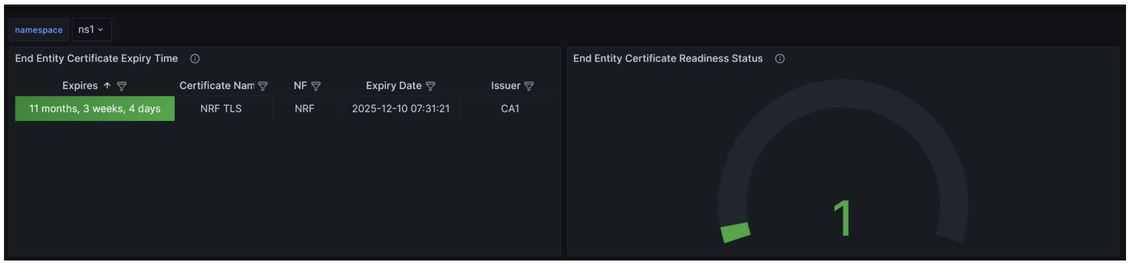
Namespace:	Namespace
Secret Name:	Secret Name
Key:	Key
Merge Certificate and Certificate Chain:	Off

CA Bundle Input

Namespace:	Namespace
Secret Name:	Secret Name
Key:	Key

Check Grafana dashboard

Figure 30 End Entity (NF) Sample Grafana Dasboard



namespace ns1

End Entity Certificate Expiry Time

Expires	Certificate Nam	NF	Expiry Date	Issuer
11 months, 3 weeks, 4 days	NRF TLS	NRF	2025-12-10 07:31:21	CA1

End Entity Certificate Readiness Status

1

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: MIIDrTCCApWgAwIBXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

  nrfkey.der: MIIEogIBAAKXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
kind: Secret

metadata:

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

  name: nrf-tls-secret

  namespace: ns1

  resourceVersion: "346496359"

  uid: 2dbdb2d7-313d-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
```

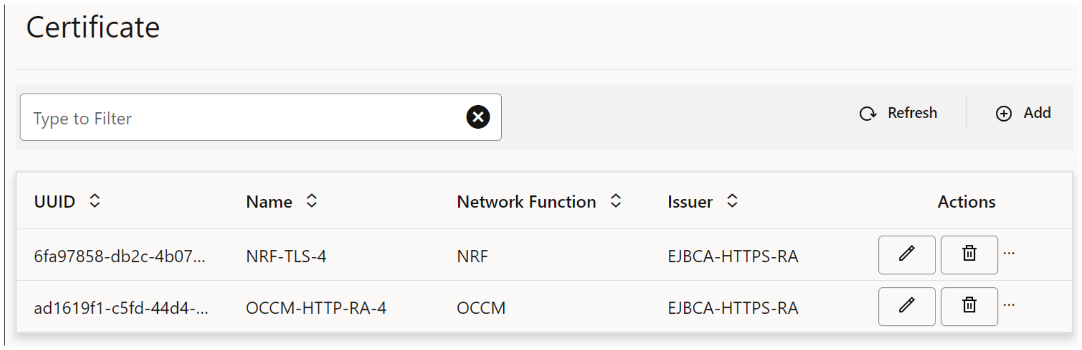
A.1.2 Recreating Certificates - Sample Configuration

This section describes the sequence of steps to be performed to recreate certificates when OCCM or NF certificate configuration has been accepted.

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 **Edit** under **Actions** for the certificate you want to recreate.

Figure 31 Certificate Page



Certificate

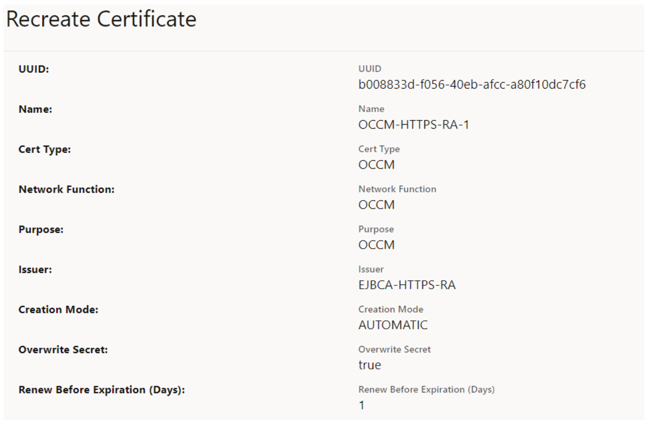
Type to Filter

Refresh Add

UUID	Name	Network Function	Issuer	Actions
6fa97858-db2c-4b07...	NRF-TLS-4	NRF	EJBCA-HTTPS-RA	
ad1619f1-c5fd-44d4-...	OCCM-HTTP-RA-4	OCCM	EJBCA-HTTPS-RA	

The **Recreate Certificate** page appears. The configurations on this page are not editable.

Figure 32 Recreate Certificate Page

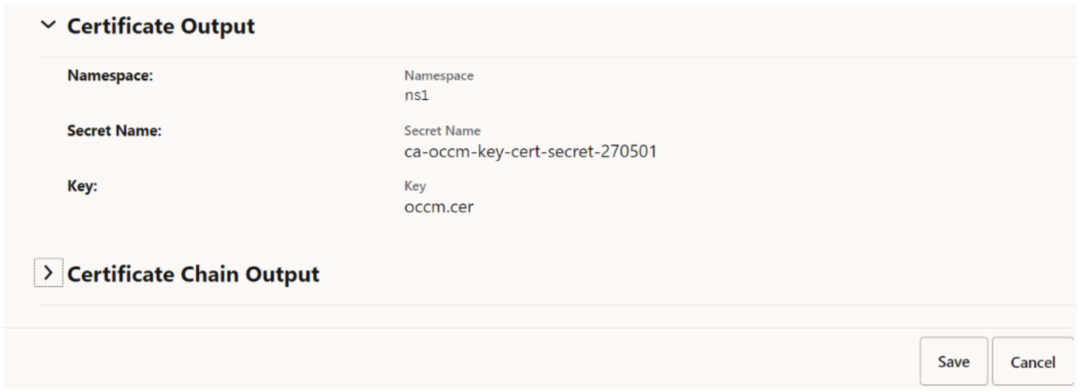


Recreate Certificate

UUID:	UUID b008833d-f056-40eb-afcc-a80f10dc7cf6
Name:	Name OCCM-HTTPS-RA-1
Cert Type:	Cert Type OCCM
Network Function:	Network Function OCCM
Purpose:	Purpose OCCM
Issuer:	Issuer EJBCA-HTTPS-RA
Creation Mode:	Creation Mode AUTOMATIC
Overwrite Secret:	Overwrite Secret true
Renew Before Expiration (Days):	Renew Before Expiration (Days) 1

4. On the **Recreate Certificate** page, click **Save** to trigger the recreate request.

Figure 33 Click Save



Certificate Output

Namespace:	Namespace ns1
Secret Name:	Secret Name ca-occm-key-cert-secret-270501
Key:	Key occm.cer

Certificate Chain Output

Save Cancel

5. When the recreate certificate request has been submitted, verify if the Kubernetes secret specified under private key output and certificate output has been recreated. Run the following command to verify the Kubernetes secret:

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

A sample response is as follows:

```
data:
  nrf.cer:
LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUQ4ekNDQWx1Z0F3SUJBZ0lVSlgvNlBsVXF
haEJaYUVOcm.....
  nrfkey.pem: MHcCAQEEIHtK36V377+977+9akke77+9Xe+/ve+/vQMcHe+/
vRXvv73vv70n77+9VO+/vVPvv73vv70RcE4577+9CgYIKu+/v.....
kind: Secret
metadata:
  creationTimestamp: "2024-05-03T11:05:08Z"
  name: nrf-tls-secret03052402
  namespace: ns1
  resourceVersion: "219805879"
  uid: 7e0d4bbf-291f-4fd2-a3d6-d42b8eff1994
type: Opaque
```

6. Check the Grafana dashboard to view the recreated certificate.