Oracle® Communications

Cloud Native Configuration Console User Guide

Release 25.2.200
G48063-01
March 2026

Oracle Communications Cloud Native Configuration Console User Guide, Release 25.2.200

G48063-01

Contents

1 Introduction

2 CNC Console Features

3 Logging into CNC Console

4 Configuring CNC Console IAM

6 Accessing NF Configurations Through Curl and Postman

7 CNC Console Metrics

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
A-CNCC Core	Agent CNC Console is a CNCC Core instance which manages local NF(s) and local OCCNE common services(s). A-CNCC is managed by M-CNCC. A-CNCC contains A-CNCC Core Ingress Gateway. A-CNCC has no IAM component. A-CNCC is also known as A-CNCC Core or aCncc Core.
AD	Active Directory
ASM	Aspen Service Mesh
BSF	Oracle Communications Cloud Native Core, Binding Support Function
CNC Console	Oracle Communications Cloud Native Configuration Console
cnDBTier	Oracle Communications Cloud Native Core, cnDBTier
CNE	Oracle Communications Cloud Native Core, Cloud Native Environment
CNI	Container Network Interface
CNLB	Cloud Native Load Balancer
CRD	Custom Resource Definitions
CRUD Operations	CREATE, READ, UPDATE, DELETE
CS	Common Service
ECDSA	Elliptic Curve Digital Signature Algorithm
EIR	Equipment Identity Register
GRR	Geo Replication Recovery
HTTPS	Hypertext Transfer Protocol Secure
IAM	Identity Access Management
Instance	NF or CNE common service managed by either M-CNCC Core or A-CNCC Core.
KPI	Key Performance Indicator
LCM	Lifecycle Management
LDAP	Lightweight Directory Access Protocol
LDAPS	Lightweight Directory Access Protocol (Over SSL)
MC	Multi Cluster. In multi cluster, a single CNCC can manage NF instances that accessess different Kubernetes clusters.
MCMI	Multiple Clusters Multiple Instances
M-CNCC	Manager CNC Console or mCncc is a CNC Console instance which manages multiple A-CNCC and local instances. **Non OCI**: M-CNCC has two components M-CNCC IAM and M-CNCC Core **OCI**: M-CNCC has only M-CNCC Core component. M-CNCC IAM is substituted with OCI IAM.

Table (Cont.) Acronyms

Acronym	Description
M-CNCC Core	Manager CNC Console Core or M-CNCC Core (also known as mCncc Core) is a core component of M-CNCC that provides GUI and API access portal for accessing NF and OCCNE common services. M-CNCC Core contains M-CNCC Core Ingress Gateway and M-CNCC Core back-end microservices.
M-CNCC IAM	Manager CNC Console IAM or M-CNCC IAM (also known as mCncc Iam) is an IAM component of M-CNCC. M-CNCC IAM contains M-CNCC IAM Ingress Gateway and M-CNCC IAM back-end microservices.
M-CNCC Kubernetes cluster	Kubernetes cluster hosting M-CNCC
MO	Mananged Objects
MOS	My Oracle Support
mTLS	Mutual Transport Layer Security
NRF	Oracle Communications Cloud Native Core, Network Repository Function
OCI	Oracle Cloud Infrastructure
OCNADD	Oracle Communications Network Analytics Data Director
OCNF	Oracle Communications Network Function
OSDC	Oracle Software Delivery Cloud
OSO	Oracle Communications Operations Services Overlay
PROVGW	Provisioning Gateway
RBAC	Role Based Access Control
Release Stream	A release stream is a sequence of releases for a product available to the customer. Example of release streams are 24.1.x, 24.2.x, 24.3.x, 25.1.1xx, 25.1.2xx, 25.2.1xx, and so on.
REST API	Representational State Transfer Application Programming Interface
SAML	Security Assertion Markup Language
SBA	Service Based Architecture
SBI	Service Based Interface
SCP	Oracle Communications Cloud Native Core, Service Communication Proxy
SCSI	Single Cluster Single Instance
SEPP	Oracle Communications Cloud Native Core, Security Edge Protection Proxy
Site	Kubernetes Cluster
SSO	Single Sign On
TLS	Transport Layer Security
UDR	Oracle Communications Cloud Native Core, Unified Data Repository
UE	User Equipment
URI	Subscriber Location Function

What's New in This Guide

This section introduces the documentation updates for release 25.2.2xx.
Release 25.2.200 - G48063-01, March 2026

- **Feature Updates:**
  - **CNC Console IAM Backend Upliftment**
    - Updated Figure 4-2 in the Types of Roles in CNC Console section to reflect the latest UI after the CNC Console IAM keycloak updates.
    - Updated the procedure to log into CNC Console and select the desired realm in the Accessing Roles in CNC Console Applications section along with the following screenshots to reflect the latest UI:
      - Figure 4-3
      - Figure 4-4
      - Figure 4-5
    - Updated the procedure to change the CNC Console IAM admin password in the Creating or Updating Admin User Password in CNC Console IAM section along with the following screenshots to reflect the latest UI:
      - Figure 4-6
      - Figure 4-7
      - Figure 4-8
    - Updated the procedure to create or update the CNC Console IAM User password in the Creating or Updating CNC Console Core User Password in CNC Console IAM section and the following screenshots to reflect the latest UI:
      - Figure 4-9
      - Figure 4-10
      - Figure 4-11
    - Updated the procedure to create users in the Creating the Users section along with the following screenshots to reflect the latest UI:
      - Figure 4-14
      - Figure 4-15
      - Figure 4-16
      - Figure 4-17
      - Figure 4-18
      - Figure 4-19
    - Updated the procedure to view the users in the Viewing the Users section along with the following screenshots to reflect the latest UI:
      - Figure 4-20
      - Figure 4-21
      - Figure 4-22

  - Updated the procedure to assign roles to the user in the Assigning Roles to the User section, along with the following screenshots to reflect the latest UI:
    - Figure 4-23
    - Figure 4-24
  - Updated the procedure to configure SAML IDP in CNC Console IAM in the Integrating SAML SSO with CNC Console IAM section, along with the following screenshots to reflect the latest UI:
    - Figure 4-25
    - Figure 4-26
    - Figure 4-27
    - Figure 4-28
    - Figure 4-29
    - Figure 4-30
    - Figure 4-31
    - Figure 4-32
    - Figure 4-33
    - Figure 4-34
    - Figure 4-35
    - Figure 4-36
    - Figure 4-37
    - Figure 4-38
    - Figure 4-39
    - Figure 4-40
    - Figure 4-41
    - Figure 4-42
    - Figure 4-43
    - Figure 4-44
    - Figure 4-45
- Updated the procedure to configure user federation in the Configuring User Federation with CNC Console IAM section, along with the following screenshots to reflect the latest UI:
  - Figure 4-46
  - Figure 4-47
  - Figure 4-48
  - Figure 4-49
  - Figure 4-50
  - Figure 4-51
  - Figure 4-52
  - Figure 4-53

            - Figure 4-54

    - Updated the procedure to set direct access grants available in the Generate Access Tokens section, along with the following screenshots to reflect the latest UI:
        - Figure 6-2
        - Figure 6-3

- **General Updates:**
    - Updated the resolution and table format for all the alerts in the following sections:
        - CNC Console IAM Alerts
        - CNC Console Core Alerts
    - Added the procedure to access cnDBTier resources through CNC Console using curl or Postman in the Accessing cnDBTier Resources section.
    - Added the following images in the Support for CNE Common Services section to reflect the CNE and OSO Common Services cards:
        - Figure 2-1
        - Figure 2-2

1 Introduction

The Cloud Native Configuration Console (CNC Console) is a single-screen solution to configure and manage network functions (NFs).

The CNC Console has the following modules:

- **CNC Console Core** : CNC Console Core acts as User Interface (UI) or Application Programming Interface (API) portal for NFs and Oracle Communications Cloud Native Environment (OCCNE) common services. CNC Console Core module is the part of CNC Console that integrates with other Cloud Native Core Network Functions.
- **CNC Console Identity and Access Management (CNC Console IAM)**: CNC Console IAM acts as a local identity provider and as a broker for an external identity provider. The CNC Console IAM module includes required authentication and authorization procedures, such as creating and assigning roles to users.

Oracle Cloud Infrastructure (OCI)

CNC Console supports deployment on Oracle Cloud Infrastructure (OCI).

Oracle Cloud Infrastructure (OCI) is a set of complementary cloud services that enable you to build and run a range of applications and services in a highly available hosted environment. OCI provides high-performance compute capabilities (as physical hardware instances) and storage capacity in a flexible overlay virtual network that is securely accessible from your on-premises network.

OCI provides a single platform for all Oracle products. Thus, deploying the CNC NFs on OCI allows operational efficiency. Additionally, OCI reduces the Total Cost of Ownership (TCO), improves performance, resource utilization, and persistence.

For more information, see Oracle Cloud Infrastructure Documentation and *Oracle Communications Cloud Native Core OCI Adaptor, NF Deployment on OCI Guide*.

> ⓘ **Note**
>
> The CNC Console Core module is applicable for OCI deployment. CNC Console IAM is not applicable for OCI deployment; instead, OCI IAM is used.

1.1 References

Refer to the following documents for more information:

- *Oracle Communications Cloud Native Core, Cloud Native Environment User Guide*
- *Oracle Communications Cloud Native Core, cnDBTier User Guide*
- *Oracle Communications Cloud Native Core Automated Test Suite User Guide*
- *Oracle Communications Cloud Native Core, OCI Deployment Guide*
- *Oracle Communications Cloud Native Core, OCI Adaptor User Guide*

- *Oracle Communications Cloud Native Core, Certificate Management User Guide*
- *Oracle Communications Network Analytics Data Director User Guide*
- *Oracle Communications Cloud Native Core, Network Function Data Collector User Guide*
- *Oracle Communications Cloud Native Core Release Notes*
- *Oracle Communications Cloud Native Core Licensing Information User Guide*
- *Oracle Communications Cloud Native Core Solution Upgrade Guide*
- *Oracle Communications Cloud Native Core Security Guide*
- *Oracle Communications Cloud Native Configuration Console User Guide*
- *Oracle Communications Cloud Native Configuration Console Network Impact Report*
- *Oracle Communications Cloud Native Configuration Console Troubleshooting Guide*
- *Oracle Communications Cloud Native Configuration Console REST Specification Guide*

2

CNC Console Features

This section describes the features supported by CNC Console.

2.1 Support for Network Functions

CNC Console GUI provides a user interface to configure and manage the following network functions (NFs):

Oracle Communications Cloud Native Core, Binding Support Function (BSF)

BSF provides the following functions:

- Allows PCF users to register, update, and remove the binding information.
- Allows NF consumers to retrieve the binding information.

CNC Console provides an interface to configure global and service parameters in BSF.

For more information about configuring parameters, see the *Configuring BSF Using CNC Console* section in *Oracle Communications Cloud Native Core, Binding Support Function User Guide.*

> ⓘ **Note**
>
> The performance and capacity of the CNC Console system may vary based on the call model, feature or interface configuration, and underlying CNE and hardware environment.

Oracle Communications Network Analytics Data Director (OCNADD)

OCNADD is a Network Data Broker (NDB) in the 5G core network. It receives the network traffic data from various sources such as 5G NFs, Non-5G nodes, and third-party producers, and sends the filtered and consolidated data securely to the subscribed consumers, which are third party consumer applications or platforms.

Data collection is a complex task in a 5G Service Based Architecture (SBA). The data is collected to provide meaningful insights to the customers. OCNADD provides the following benefits:

- Filters, replicates, aggregates, and routes data feeds to subscribed third-party consumers.
- Ensures data security, low latency, and redundancy while collecting and processing the data feeds.
- Enables the Communications Service Providers (CSP) to correlate and transform the acquired data as per their data feed configuration to create comprehensive dashboards and Key Performance Indicators (KPIs). This enables them to achieve meaningful insights about all functions in the 5G network.

This information can be used for monetization, improving service quality, reducing downtime, ensuring easy network scalability, and minimizing losses. OCNADD-generated data feeds are beneficial for monitoring and troubleshooting during a network failure. It is a crucial function, aiding in self-healing networks.

Oracle Communications Cloud Native Core, Network Repository Function (NRF)

NRF provides the following functions:

- Maintains the profiles of the available NF instances and their supported services in the 5G core network.
- Allows consumer NF instances to discover other providers' NF instances.
- Tracks the status of NF instances.
- Provides OAuth2-based access token service for consumer NF authorization.
- Supports specific NF Type selection based on subscriber identity.
- Forwards messages between NRFs.
- Supports georedundancy for service availability.

The NRF interacts with every other NF in the 5G core network and it supports the above functions through the following services:

- Management Service
- Discovery Service
- Access Token Service

CNC Console provides an interface to configure global and service parameters in NRF.

For more information about configuring parameters, see the *Configuring NRF Using CNC Console* section in *Oracle Communications Cloud Native Core, Network Repository Function User Guide.*

Oracle Communications Cloud Native Core, Network Slice Selection Function (NSSF)

Network slices enable users to choose customized network environments with a variety of functionalities and performance standards. These standards may include specific aspects such as mobility, latency, availability, and reliability.

Network slices are distinguished by the features they offer and their network function (NF) optimizations. Each network slice may be associated with a distinct S-NSSAI, reflecting different slice and service types.

Users can deploy multiple instances of network slices that offer the same features, tailored for different groups of User Equipments (UEs). These instances provide different dedicated services as they are provisioned for individual customers.

It is possible for network slices to have unique S-NSSAIs. These S-NSSAIs might share the same slice or service type while having different slice differentiators.

The Network Slice Selection Function (NSSF) is responsible for identifying the network function (NF) associated with a particular slice.

Functions of the Network Function

- Allows selection of customized network slices
- Supports various functionalities such as mobility
- Meets different performance requirements (latency, availability, reliability)
- Differentiates slices by supported features and NF optimizations

- Associates network slices with different S-NSSAIs for slice and service types
- Deploys multiple network slice instances for different UE groups with the same features
- Delivers committed services for specific customers
- Supports unique S-NSSAIs with the same type but different differentiators
- Provides the NSSF with the capability to determine the NF for each slice

NSSF is a functional element that supports the following functionalities:

- NSSF enables the Access and Mobility Management Function (AMF) to perform initial registration and Protocol Data Unit (PDU) session establishment.
- AMF can retrieve NRF, NSI ID, and target AMFs as part of UE initial registration and PDU establishment procedure.
- NSSF uses an NF Service Consumer (AMF) to update the S-NSSAIs that AMF supports and notifies of any changes in the status.
- NSSF selects the network slicing instance (NSI) and determines the authorized Network Slice Selection Assistance Information (NSSAIs) and AMF to serve the UE.
- NSSF interaction with NRF allows retrieving specific NF services to be used for registration request.

NSSF provides the following information when queried by the AMF:

- Allowed NSSAIs
- Configured NSSAIs
- Restricted NSSAIs
- Candidate AMF List (in case of registration)
- Network Slice instance ID (for PDU session establishment)
- Slice-level NRF information (for PDU Connectivity)

NSSF supports the above functions through the following NSSF services:

- NS Selection service (Nnssf_NSSelection): This service is used by an NF Service Consumer (AMF) to retrieve the information related to network slice. It enables network slice selection in the serving Home Public Land Mobile Network (HPLMN).
- NS Availability Service (Nnssf_NSAvailability): This service stores and maintains list of supported S-NSSAIs per TA. It allows NF service Consumer (AMF) to update and subscribe the above data and get notifications for any addition or deletion of supported S-NSSAIs.

Oracle Communications Cloud Native Core, Converged Policy (Policy)

Policy is an NF for policy control decisions and flow-based charging control. It consists of the following functions:

- Policy rules for application and service data flow detection, gating, QoS, and flow based charging to the Session Management Function (SMF).
- Access and Mobility Management related policies to the Access and Mobility Management Function (AMF).
- UE Route Selection Policies (URSP) rules to User Equipment (UE) through AMF.
- Access to subscription information relevant for policy decisions in a Unified Data Repository (UDR).
- Network control for service data flow detection, gating, and Quality of Service (QoS).

- Flow based charging towards the Policy and Charging Enforcement Function (PCEF).
- Receiving session and media related information from Application Function (AF) and informing AF of traffic plane events.
- Provision of Policy and Charging Control (PCC) Rules to Policy and Charging Enforcement Function (PCEF) through the Gx reference point.

Policy supports the above functions through the following services:

- Session Management Service
- Access and Mobility Service
- Policy Authorization Service
- User Equipment (UE) Policy Service
- PCRF Core Service

CNC Console provides an interface to configure policies and manageable objects in Policy.

For more information about configuring parameters, see the *Configuring CNC Policy Using CNC Console* section in *Oracle Communications Cloud Native Core, Policy User Guide.*

Provisioning Gateway (PROVGW)

Oracle Provisioning Gateway (PROVGW) for Subscriber Location Function (SLF) is implemented as a cloud native function. It supports:

- HTTP1.1 over TLS.
- Custom entities or fields in UDR mode over SOAP/XML interface.
- Conversion of requests as defined in SEC.yaml configurations for SOAP/XML interface in UDR mode.
- Auditor functionality in SLF mode.
- OAM interface and configuration APIs to configure Ingress Gateway, Egress Gateway, and other Provisioning Gateway microservices.

It has two modes, which are as follows:

- SLF mode: This mode is applicable when Provisioning Gateway is deployed with UDR for SLF use case. You can configure Provisioning Gateway to connect to multiple segments of SLF, where each segment has two SLFs. This mode offers an HTTP2-based secured REST/JSON interface (through Ingress Gateway API) for SLF data provisioning. It relays the request received by the provisioning client (for example, MTAS) to multiple 5G UDR or SLF segments. The response received from each UDR or SLF segment is then consolidated and the final response is sent to the provisioning system.
  You can deploy multiple Provisioning Gateways in the same segment or across multiple segments, where each one is stateless and does not interact with each other. If one of the Provisioning Gateways goes down, MTAS can use the second Provisioning Gateway instance to continue provisioning SLF data on UDR.
- UDR mode: This mode is applicable when deployed with UDR for converged policy database solution. In this mode, Provisioning Gateway supports SOAP/XML interface, which is similar to 4G UDR.

Oracle Communications Cloud Native Core, Service Communication Proxy (SCP)

SCP provides the following functionalities to other 5G Network Functions (NFs):

- Routing/Selection: Routing rules, refresh cache, and handle application failures and redirects.

    - Dynamic Discovery: The 5G topology is determined from Network Repository Function (NRF) and creation of routing rules.
    - Static Configuration: Enables NF Profiles configuration.
- Load Balancing: Load balancing based on static capacity, NF Type, NF Specific, and NF Priority as mentioned in the NF Profile.
- NF Subscription: Subscription for all NF types.
- Circuit Breaking: Initiated on a per FQDN basis when outstanding transactions exceed a configurable value.
- Message Priority: Message Priority assignment and override based on the 3GPP-SBI-Message-Priority header.
- Congestion and Overload: Uniform load balancing and routing strategy across the network and protects the pod from overload related to various system resources.

CNC Console provides an interface to configure the SCP features.

For more information about configuring parameters, see the *Configuring SCP Using CNC Console* section in *Oracle Communications Cloud Native Core, Service Communication Proxy User Guide*.

Oracle Communications Cloud Native Core, Security Edge Protection Proxy (SEPP)

SEPP supports the following functionalities:

- Protects application layer control plane messages and sensitive data between two NFs belonging to different PLMNs that use the N32 interface to communicate with each other. The N32 interface is used between the SEPPs of a VPLMN and a HPLMN in roaming scenarios. 3GPP has specified N32 to be considered as two separate interfaces: N32-c and N32-f.
    - N32-c is the control plane interface between the SEPPs for performing the initial handshake and negotiating the parameters to be applied for the actual N32 message forwarding.
    - N32-f is the forwarding interface between the SEPPs, that is used for forwarding the communication between the Network Function (NF) service consumer and the NF service producer after applying the application level security protection.
- Provides secure communication of Inter PLMN messages from Consumer NF to Producer NF using TLS protection mode (HTTP over TLS).
- Supports configuration of roaming partner profiles using REST API.
- Performs mutual authentication and negotiation of cipher suites with the SEPP in the roaming partner’s network.
- Handles key management aspects that involve setting up the required cryptographic keys needed for securing messages on the N32 interface between two SEPPs.
- Provides a single point of access and control to internal NFs.
- Validates inbound traffic as to whether it is from an authorized external PLMN.
- Supports cross-layer validation of source and destination addresses and identifiers to provide anti-spoofing capabilities.

CNC Console provides an interface to configure different services in SEPP.

For more information about configuring parameters, see the *Configuring SEPP Using CNC Console* section in *Oracle Communications Cloud Native Core, Security Edge Protection Proxy User Guide.*

Oracle Communications Cloud Native Core, Unified Data Repository (UDR)

Oracle's 5G UDR:

- Leverages a common Oracle Communications Cloud Native Framework.
- Is compliant with 3GPP 29.505 Release 15 specifications UDM.
- Is compliant with 3GPP 29.519 Release 16 (backward compatible with Release 15) specifications for PCF.
- Has tiered architecture providing separation between the connectivity, business logic, and data layers.
- Uses Oracle MySQL NDB Cluster CGE Edition as backend database in the Data Tier.
- Registers with NRF in the 5G network so that the other NFs in the network can discover UDR through NRF.
- Registers UDR with services like DR-SERVICE and GROUP-ID-MAP.

CNC Console provides an interface to configure global and service parameters in UDR.

For more information about configuring parameters, see the *Configuring UDR Using CNC Console* section in *Oracle Communications Cloud Native Core, Unified Data Repository User Guide.*

Oracle Communications Cloud Native Core, Certificate Management (OCCM)

Oracle Communications Cloud Native Core, Certificate Management (OCCM) is an automated system that oversees and maintains the security certificates required for Oracle 5G Network Functions (NFs). These certificates are used to verify and secure communications within the network.

OCCM provides the following benefits:

- **Automatic certificate renewal** prevents network disruptions caused by expired certificates.
- **Constant monitoring** ensures that certificates are always up-to-date and valid.
- **Simplified management** reduces the manual effort needed to track and renew numerous certificates.
- **Compliance with standards** supports the use of separate certificates for different network modes and workflows, as recommended by 3GPP.
- **Supports large-scale networks** by efficiently handling hundreds of certificates across different interfaces and operators.

OCCM works behind the scenes to manage all the security certificates in your 5G network. These certificates are needed for safe communication between different network functions. As your network grows, the number of certificates can become very large and difficult to track by hand. OCCM monitors each certificate and renews them automatically when they are about to expire. This helps your business keep the network running smoothly, without unexpected issues caused by expired certificates.

OCCM supports the following functions:

- Automatically manages and renews certificates for Oracle 5G Network Functions
- Monitors certificates continuously for validity and expiration
- Supports the use of separate certificates for client/server modes and various workflows as recommended by 3GPP

- Handles multiple operator certificates needed for different network interfaces
- Eliminates the risk of network disruption due to missed or expired certificate renewals

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

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

CNC Console supports the following OCCM functionalities:

- Creating certificates based on applicable NF certificate parameters.
- Automatic certificate renewal by OCCM.
- Manual certificate renewal. This can be used in case a certificate is revoked, or when migrating from manual certificate management to automatic certificate management.
- Integration with single or multiple Certificate Authorities (CAs) for signing certificates.

CNC Console provides an interface to configure different services in OCCM. For more information, see *OCCM Supported Features* in the *Oracle Communications Cloud Native Core, Certificate Management User Guide.*

Managing CNC Console Support for NF GUI

Observe

For information on Metrics and KPIs, see CNC Console Metrics, and CNC Console KPIs sections.

Maintain

If you encounter alerts at system or application levels, see CNC Console Alerts section for resolution steps.

In case the alert still persists, perform the following:

1. **Collect the logs:** For more information on how to collect logs, see "CNC Console Logs" in *Oracle Communications Cloud Native Configuration Console Troubleshooting Guide*.
2. **Raise a service request:** See My Oracle Support for more information on how to raise a service request.

2.2 LCM Automation

This feature optimizes the deployment or upgrade steps. This is achieved by automating service account creation that enables you to create user-defined service account automatically without performing any manual steps. The automation includes:

- Creating Service Account: An automated resource service account creation has been introduced in this release to streamline the management of Kubernetes resources through Helm charts.
- Applying NF Alert Configuration rules: CNC Console uses the `oso-alr-config` Helm chart, introduced as part of the OSO package, to apply alert rules using Helm upgrades. While the `oso-alr-config` Helm chart deploys automatically during the OSO installation, CNC Console performs Helm upgrades to apply or update the required alert rules. Both manual and automated configurations are supported.

Managing LCM Automation

This section explains the procedure to enable and configure the feature.

Enable and Configure

Configure this feature using Helm. To enable and configure this feature, see *Oracle Communications Cloud Native Configuration Console Installation, Upgrade, and Fault Recovery Guide.*

Observe

There are no additional metrics or KPIs for this feature.

Maintain

If you encounter alerts at system or application level, see CNC Console Alerts section for resolution steps.

In case the alert still persists, perform the following:

1. **Collect the logs:** For more information on how to collect logs, see CNC Console Logs.
2. **Raise a service request:** See My Oracle Support for more information on how to raise a service request.

2.3 Multiple Admin Support for CNC Console IAM

As CNC Console evolved from a simple interface for managing a single network function (NF) instance with one admin account to supporting multi-cluster deployments catering to multiple NF Instances, including access management for observability applications, there is a need to have support for multiple admin users for managing the users across multiple Customer Operations teams.

As part of this feature, CNC Console IAM is enhanced to support creation and management of multiple Admin Users in IAM using both Internal and External IDPs (SAML and LDAP). All existing measurements, alarms, logging mechanisms, and other features apply to all admin users, similar to existing Core users.

There are default password policies defined for CNC Console IAM users in the default realm. The option to enable these policies is provided during installation.

Managing Multiple Admin Support for CNC Console IAM Deployments

Enable and Configure

During the deployment, a default admin user is created. To create multiple CNC Console IAM admin users, see the 'Post Installation steps' section in the *Oracle Communications Cloud Native Configuration Console Installation, Upgrade, and Fault Recovery Guide.*

Observe

For information on Metrics and KPIs, see CNC Console Metrics, and CNC Console KPIs sections.

Maintain

If you encounter alerts at system or application levels, see CNC Console Alerts section for resolution steps.

In case the alert still persists, perform the following:

1. **Collect the logs:** For more information on how to collect logs, see CNC Console Logs.
2. **Raise a service request:** See My Oracle Support for more information on how to raise a service request.

2.4 Support for cnDBTier APIs in CNC Console

With the implementation of this feature, cnDBTier APIs are integrated into the CNC Console, and NF users can view specific cnDBTier APIs, such as checking the cnDBTier version, status of cnDBTier clusters, and georeplication status on the CNC Console.

The following read-only cnDBTier APIs are available in CNC Console:

- Backup List: Displays the details of stored backups, such as the ID and size of the backup.
- cnDBTier version: Displays the cnDBTier version.
- Database Statistics Report: Displays the number of available databases.
- Georeplication Status:
  - Real-Time Overall Replication Status: Displays the overall replication status in multisite deployments. For example, in a four-site deployment, it provides the replication status between the following sites: site1-site2, site1-site3, site1-site4, site2-site3, site2-site4, and site2-site1. This is applicable for all other sites.
  - Site-Specific Real-Time Replication Status: Displays the site-specific replication status.
- Georeplication Recovery:
  - Update Cluster as Failed: Marks the disrupted cluster as failed.
  - Start Georeplication Recovery: Initiates georeplication recovery.
  - Georeplication Recovery Status: Monitors the georeplication recovery status.
- HeartBeat Status: Displays the connectivity status between the local site and the remote site name to which CNC Console is connected.
- Local Cluster Status: Displays the status of the local cluster.
- On-Demand Backup: Displays the status of initiated on-demand backups.
- cnDBTier Health: Displays the health status of the following services:
  - Replication Health Status: Displays the health status of the replication service. It checks the following:
    - if the replication service is up
    - if the replication service can connect to database
  - Monitor Health Status: Displays the health status of the monitor service. It checks the following:
    - if the monitor service is up
    - if the service can connect to database
    - if the metrics are fetched (the metrics are fetched when the service is up and not when it is down)
  - NDB Health Status: Displays the health status of the NDB service pods like (data pods, sql pods, app-my-sql pods, mgmt pods). It checks the following:
    - if the pod is connected to PVC

    - if the pod's status is up

> **Note**
>
> PVC Health Status attribute is set to NA when some of the database pods are not connected to the PVC.

  - Backup Manager Health Status: Displays the health status of the backup manager service. It checks the following:
    - if the backup manager service is up
    - if the service can connect to database

Managing cnDBTier APIs at CNC Console

Enable
This feature is enabled automatically when cnDBTier is configured as an instance during the CNC Console deployment. However, this feature is NF dependent and must be enabled in their respective NF configurations. cnDBTier APIs can be added for NFs or CNC Console instance. For more information about integrating cnDBTier APIs in CNC Console, see *Oracle Communications Cloud Native Core, cnDBTier User Guide* and for information on how to enable at CNC Console, see *Oracle Communications Cloud Native Configuration Console Installation, Upgrade, and Fault Recovery Guide.*

Observe

For information on Metrics and KPIs, see CNC Console Metrics, and CNC Console KPIs sections.

Maintain

If you encounter alerts at system or application levels, see CNC Console Alerts section for resolution steps.

In case the alert still persists, perform the following:

1. **Collect the logs:** For more information on how to collect logs, see "CNC Console Logs" in *Oracle Communications Cloud Native Configuration Console Troubleshooting Guide*.
2. **Raise a service request:** See My Oracle Support for more information on how to raise a service request.

2.5 Support for CNE Common Services

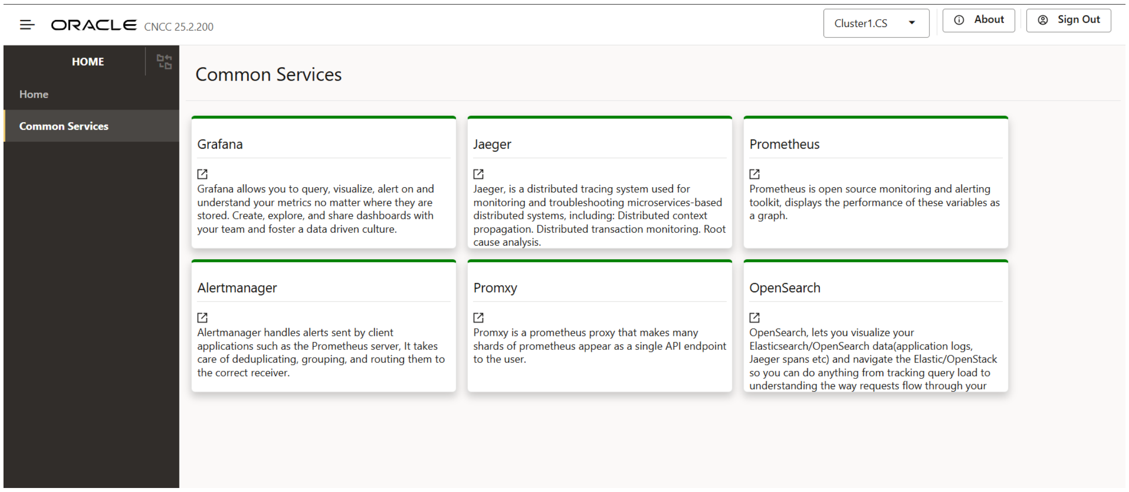
> **Note**
>
> Not applicable for OCI deployment.

CNC Console Common Services GUI provides an option to enable cards (hyperlinks) for CNE Common services and OSO services.

When CNC Console is integrated with common services, it provides an additional layer of security through authentication and authorization for common services that don't have their own authentication mechanism. CNC Console also provides the user a single login for all common services as per your assigned roles.

CNC Console Common Services GUI provides an option to enable cards (hyperlinks) for OCCNE Common services such as Grafana, Kibana, Jaeger, Prometheus, AlertManager, Promxy, OpenSearch, and Jaeger-ES.

Figure 2-1 CNE Common Services



Common Services

Grafana

Grafana allows you to query, visualize, alert on and understand your metrics no matter where they are stored. Create, explore, and share dashboards with your team and foster a data driven culture.

Jaeger

Jaeger, is a distributed tracing system used for monitoring and troubleshooting microservices-based distributed systems, including: Distributed context propagation. Distributed transaction monitoring. Root cause analysis.

Prometheus

Prometheus is open source monitoring and alerting toolkit, displays the performance of these variables as a graph.

Alertmanager

Alertmanager handles alerts sent by client applications such as the Prometheus server, It takes care of deduplicating, grouping, and routing them to the correct receiver.

Promxy

Promxy is a prometheus proxy that makes many shards of prometheus appear as a single API endpoint to the user.

OpenSearch

OpenSearch, lets you visualize your Elasticsearch/OpenSearch data(application logs, Jaeger spans etc) and navigate the Elastic/OpenStack so you can do anything from tracking query load to understanding the way requests flow through your

Table 2-1 CNE Common Services

Common Service Name	Description
Grafana	Grafana enables you to query, visualize, alert on, and understand your metrics, regardless of their storage location. Create, explore, and share dashboards to support a data-driven culture.
Kibana	Kibana lets you visualize your Elasticsearch data and navigate the Elastic Stack, enabling you to track query load and understand the way requests flow through your applications.
Jaeger	Jaeger is a distributed tracing system for monitoring and troubleshooting microservices-based distributed systems. It provides distributed context propagation, transaction monitoring, and root cause analysis.
Prometheus	Prometheus is an open source monitoring and alerting toolkit that displays the performance of variables as a graph.
Alertmanager	Alertmanager handles alerts sent by client applications, such as the Prometheus server. It deduplicates, groups, and routes alerts to the correct receiver.
Promxy	Promxy is a Prometheus proxy that presents multiple Prometheus shards as a single API endpoint.
OpenSearch	OpenSearch lets you visualize your Elasticsearch or OpenSearch data (application logs, Jaeger spans, and more) and navigate the Elastic or OpenStack environment, helping you analyze query load and request flows.

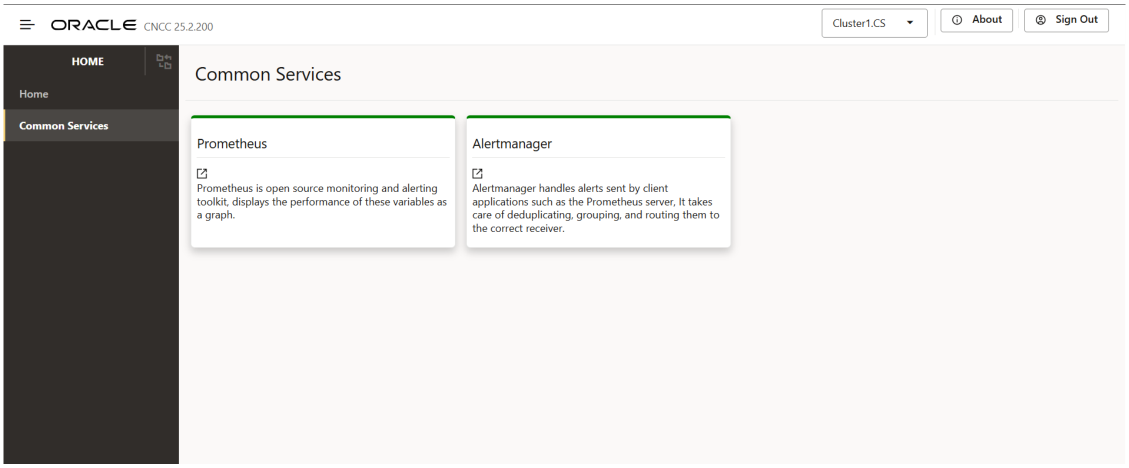
Table 2-1   (Cont.) CNE Common Services

Common Service Name	Description
Jaeger-ES	Jaeger-ES enables you to observe legacy traces stored in Elasticsearch, which is useful for troubleshooting and monitoring microservices-based distributed systems.

OSO Common Services

CNC Console Common Services GUI also provides an option to enable cards (hyperlinks) for OSO services such as Prometheus and AlertManager.

Figure 2-2   OSO Common Services



Prometheus is open source monitoring and alerting toolkit, displays the performance of these variables as a graph.

Alertmanager handles alerts sent by client applications such as the Prometheus server, It takes care of deduplicating, grouping, and routing them to the correct receiver.

Table 2-2   OSO Common Service

OSO Common Service Name	Description
Prometheus	Prometheus is an open source monitoring and alerting toolkit that displays the performance of variables as a graph.
Alertmanager	Alertmanager handles alerts sent by client applications, such as the Prometheus server. It deduplicates, groups, and routes alerts to the correct receiver.

Managing Common Service Support (OSO Cards and CNE Cards)

Enable and Configure

Prometheus and Alertmanager UIs can be accessed using the CNC Console. For more information on accessing Prometheus and Alertmanager UIs using CNC Console, refer to *Oracle Communications Cloud Native Configuration Console User Guide.*

Observe

For information on Metrics and KPIs, see CNC Console Metrics, and CNC Console KPIs sections.

Maintain

If you encounter alerts at system or application levels, see CNC Console Alerts section for resolution steps.

In case the alert still persists, perform the following:

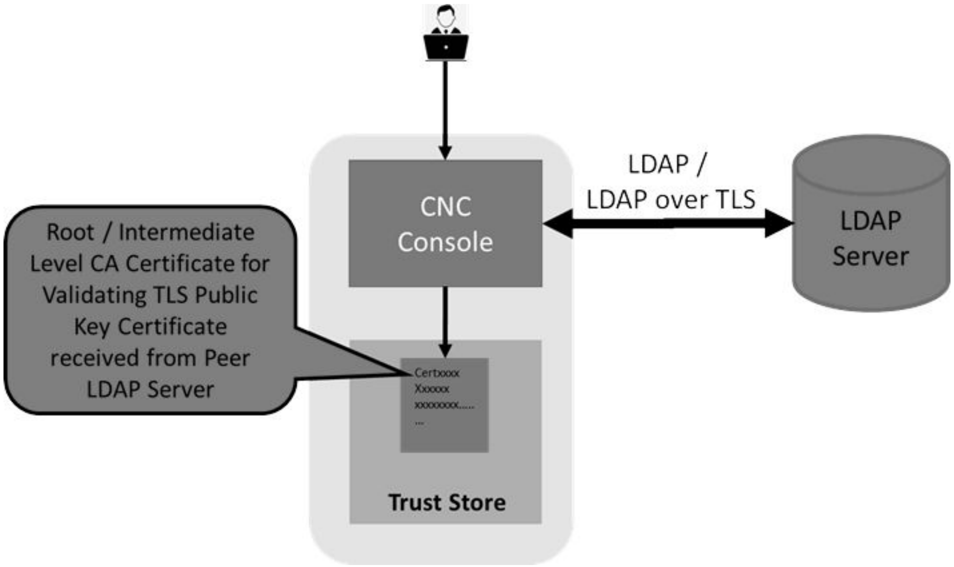
1. **Collect the logs:** For more information on how to collect logs, see "CNC Console Logs" in *Oracle Communications Cloud Native Configuration Console Troubleshooting Guide*.
2. **Raise a service request:** See My Oracle Support for more information on how to raise a service request.

2.6 LDAP Integration

> **Note**
>
> **For OCI:** See the OCI Active Directory Integration section.

The Lightweight Directory Access Protocol (LDAP) is a protocol that defines the technique for accessing the directory data.

Figure 2-3 LDAP


Root / Intermediate Level CA Certificate for Validating TLS Public Key Certificate received from Peer LDAP Server
CNC Console
LDAP / LDAP over TLS
LDAP Server
Certxxxx
Xxxxxx
xxxxxxxx.....
...
Trust Store

The CNC Console IAM is used as an integration platform to connect it into existing Lightweight Directory Access Protocol (LDAP) and Active Directory (AD) servers.

CNC Console IAM can combine existing external user databases having user and credential details. You can integrate the CNC Console IAM to perform validation of these user credentials and pull in the identity information.

CNC Console IAM provides LDAP over TLS support to securely communicate with external LDAP and Active Directory servers.

CNC Console IAM also supports Lightweight Directory Access Protocol Secure (LDAPS) between the client and server to make the communication secure.

Managing LDAP Integration

Enable and Configure

To enable and configure LDAP integration feature, see Integrating CNC Console LDAP Server with CNC Console IAM

Observe

For information on Metrics and KPIs, see CNC Console Metrics, and CNC Console KPIs sections.

Maintain

If you encounter alerts at system or application levels, see CNC Console Alerts section for resolution steps.

In case the alert still persists, perform the following:

1. **Collect the logs:** For more information on how to collect logs, see "CNC Console Logs" in *Oracle Communications Cloud Native Configuration Console Troubleshooting Guide*.
2. **Raise a service request:** See My Oracle Support for more information on how to raise a service request.

2.7 SAML 2.0 Integration

> **Note**
>
> **For OCI:** See the OCI SAML Integration section.

SAML (Security Assertion Markup Language) enables applications to authenticate a user using an identity provider. CNC Console can broker identity providers based on the SAML v2.0 protocol.

Managing SAML 2.0 Integration with CNC Console

Enable and Configure

To enable and configure SAML 2.0 feature integration with CNC Console, see Integrating SAML SSO with CNC Console IAM section.

Observe

For information on Metrics and KPIs, see CNC Console Metrics, and CNC Console KPIs sections.

Maintain

If you encounter alerts at system or application levels, see CNC Console Alerts section for resolution steps.

In case the alert still persists, perform the following:

1. **Collect the logs:** For more information on how to collect logs, see "CNC Console Logs" in *Oracle Communications Cloud Native Configuration Console Troubleshooting Guide* .
2. **Raise a service request:** See My Oracle Support for more information on how to raise a service request.

2.8 Logging Support

Note

Logging for CNC Console IAM is not applicable for OCI deployment.

The CNC Console logs are categorized into the following types:

- Regular logs
- Audit logs
- Security logs

Regular Logs

These logs contain all kinds of error messages, warnings, or other events written within the application which provide logical, high-level information about the application and ongoing events.

Example:

```
{"level": "INFO","message": "Started GatewayApplication in 10.748 seconds
(JVM running for 12.825)"}
{"level": "INFO","message": "Creating plain httpClient"}
{"level": "INFO","message": "Creating plain restTemplate"}
{"level": "ERROR","message": "Can't get cfgs of topic
public.dynamic.datamodel,  exception is:\n
javax.ws.rs.ProcessingException: java.net.ConnectException: Connection
refused (Connection
        refused)"}
```

Audit Logs

These logs contain user related information and the activity within the system.

Security Logs

These logs contain the header, payload, method, scheme, URI, and other details for all requests and the corresponding responses.

Disabling Security Logs

By default, **Security Log** is enabled for both *CCNC Core* and *CNCC IAM*. To disable, set *securityLogEnabled* flag to **false** in *custom-core_values.yaml* and *custom-iam_values.yaml* files.

```
# CNCC configuration
cncc:
  enabled: false
  enablehttp1: false
  securityLogEnabled: false
```

Log Levels

The log level indicates the level of the logs.

The default log levels for CNC Console Core are as follows:

```
ingress-gateway:
  log:
    level:
      cncc:
        root: WARN
        audit: INFO
        security: INFO
```

The default log levels for CNC Console IAM are as follows:

```
ingress-gateway:
  log:
    level:
      cncc:
        root: WARN
        security: INFO
```

Managing Security Logs and Audit Logs

Enable and Configure

For enabling and configuring Security Logs and User Activity Logs, see "CNC Console Logs" in *Oracle Communications Cloud Native Configuration Console Troubleshooting Guide*.

Observe

For information on Metrics and KPIs, see CNC Console Metrics, and CNC Console KPIs sections.

Maintain

If you encounter alerts at system or application levels, see CNC Console Alerts section for resolution steps.

In case the alert still persists, perform the following:

1. **Collect the logs:** For more information on how to collect logs, see "CNC Console Logs" in *Oracle Communications Cloud Native Configuration Console Troubleshooting Guide*.
2. **Raise a service request:** See My Oracle Support for more information on how to raise a service request.

2.9 Support for Multi Instance and Multicluster Deployment

> **Note**
>
> Not supported for OCI deployment.

Multicluster deployment is a method of deploying an application on or across multiple Kubernetes clusters for improving availability, isolation, and scalability.

Support for Multiple Instances of NFs within a cluster

CNC Console supports multiple instances of NFs within a Kubernetes cluster using Manager CNC Console IAM (M-CNCC IAM), Manager CNC Console Core (M-CNCC Core), and Agent CNC Console Core (A-CNCC Core).

> **Note**
>
> CNC Console supports instance level access control. For more information, see Support for Instance Level Access Control and Types of Roles in CNC Console.

Support for Multicluster Deployment for NFs

CNC Console supports NF deployment across Kubernetes clusters using Manager CNC Console IAM (M-CNCC IAM), Manager CNC Console Core (M-CNCC Core), and Agent CNC Console Core (A-CNCC Core). In a multicluster deployment, CNC Console can manage NFs and OCCNE common services deployed in remote Kubernetes clusters.

Support for multicluster deployment mTLS configuration is added to provide secure communication between CNC Console Manager and Agent.

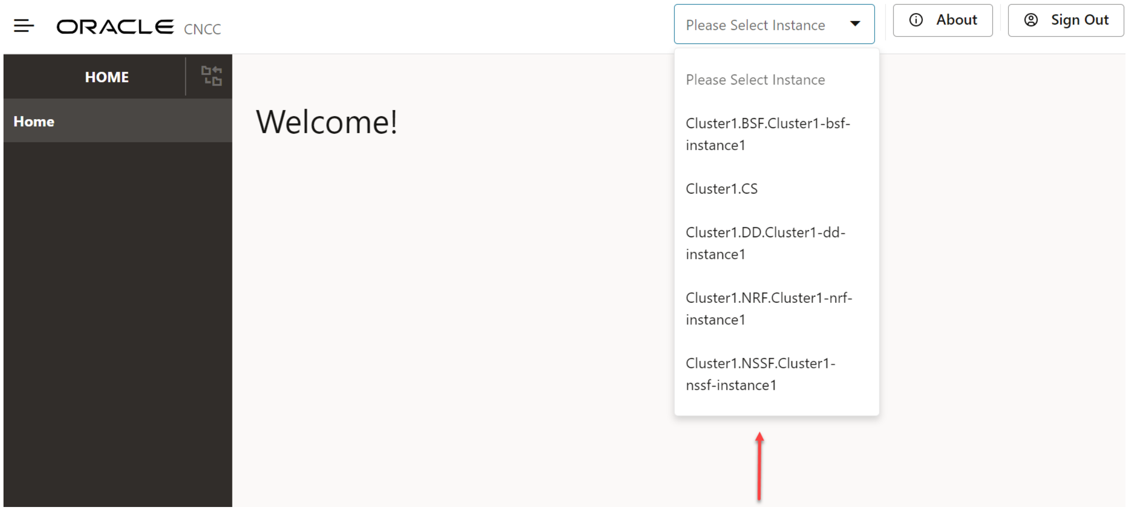
> **Note**
>
> The user must be assigned the cluster role to access multiple clusters. For more information, see Types of Roles in CNC Console.

Selecting the Instance

CNC Console multicluster deployment has introduced a drop-down on header pane for selecting the instance.

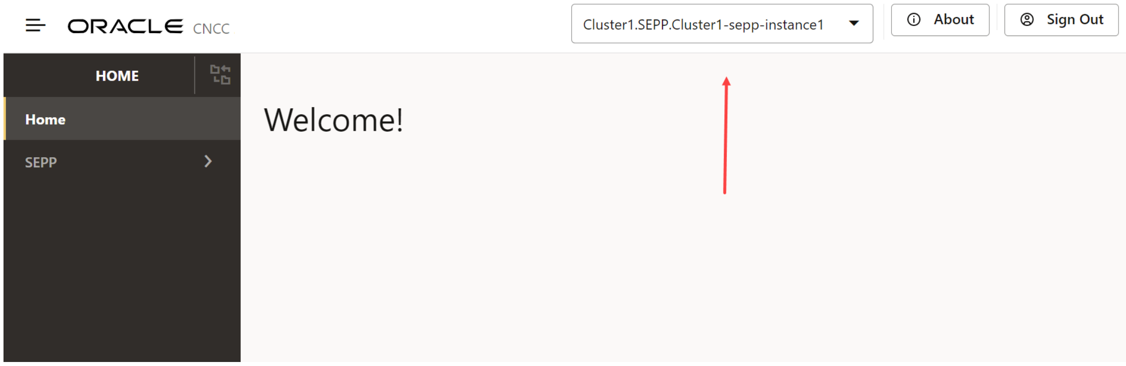
Values configured in M-CNCC Core instances section are displayed in the drop down. The naming convention used for instance drop-down display is <owner>.<type>.<instance id>

Figure 2-4 Selecting the Instance

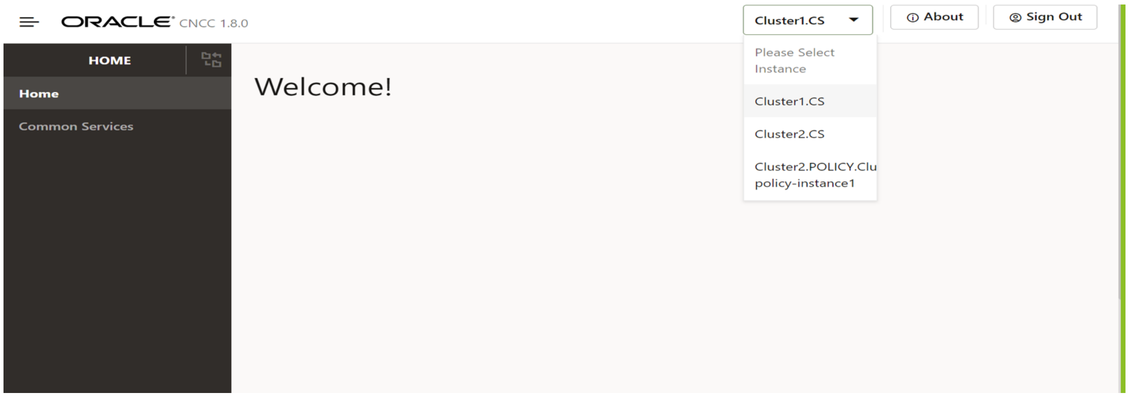

ORACLE CNCC
Please Select Instance
About
Sign Out
HOME
Home
Welcome!
Please Select Instance
Cluster1.BSF.Cluster1-bsf-instance1
Cluster1.CS
Cluster1.DD.Cluster1-dd-instance1
Cluster1.NRF.Cluster1-nrf-instance1
Cluster1.NSSF.Cluster1-nssf-instance1

Once the user selects the required instance from the drop-down, the corresponding menu gets loaded.

Figure 2-5 Loading the Menu



The common service instances configured in the instances section gets displayed in the drop-down.



For more details on multicluster configurations, see *CNC Console Multi-Cluster Configurations* section in *Oracle Communications Cloud Native Configuration Console, Installation, Upgrade, and Fault Recovery Guide.*

2.10 cnDBTier Integration

CNC Console is integrated with cnDBTier in a containerized Oracle Communications Cloud Native Environment. For more information, see *Oracle Communications cnDBTier Installation Guide.*

2.11 Support for TLS

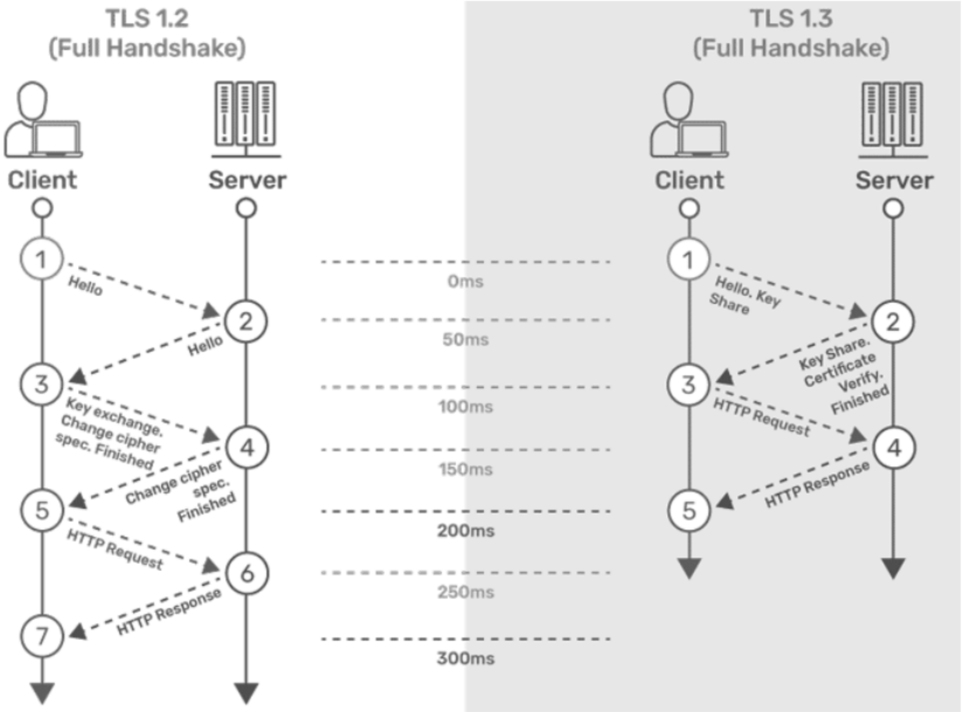
CNC Console uses Hypertext Transfer Protocol Secure (HTTPS) to establish secure connections with Consumer NFs and Producer NFs, respectively. These communication protocols are encrypted using Transport Layer Security (TLS). TLS comprises the following components:

- **Handshake Protocol**: Exchanges the security parameters of a connection. Handshake messages are supplied to the TLS record layer.
- **Record Protocol**: Receives the messages to be transmitted, fragments the data into multiple blocks, secures the records, and then transmits the result. Received data is delivered to higher-level peers.

From Release 24.3.0 onwards, CNC Console supports TLSv1.3 for all Consumer NFs, Producer NFs, Data Director, SBI Interfaces, and interfaces where TLSv1.2 was supported. TLSv1.2 will continue to be supported.

TLS Handshake

This section describes the differences between TLSv1.2 and TLSv1.3 and the advantages of TLSv1.3 over TLSv1.2 and earlier versions.



TLSv1.2

1. The connection or handshake starts when the client sends a “client hello” message to the server. This message consists of cryptographic information such as supported protocols and supported cipher suites. It also contains a random value or random byte string.
2. To respond to the “client hello” message, the server sends the “server hello” message. This message contains the CipherSuite that the server has selected from the options provided by the client. The server also sends its certificate along with the session ID and another random value.
3. The client verifies the certificate sent by the server. When the verification is complete, it sends a byte string and encrypts it using the public key of the server certificate.
4. When the server receives the secret, both the client and server generate a master key along with session keys (ephemeral keys). These session keys are used to symmetrically encrypt the data.
5. The client sends an “HTTP Request” message to the server to enable the server to switch to symmetric encryption using the session keys.
6. To respond to the client’s “HTTP Request” message, the server does the same and switches its security state to symmetric encryption. The server concludes the handshake by sending an HTTP response.
7. The client-server handshake is completed in two round-trips.

TLSv1.3

1. The connection or handshake starts when the client sends a “client hello” message to the server. The client sends the list of supported cipher suites. The client also sends its key share for that particular key agreement protocol.
2. To respond to the “client hello” message, the server sends the key agreement protocol that it has chosen. The “Server Hello” message comprises the server key share, server certificate, and the “Server Finished” message.
3. The client verifies the server certificate, generates keys as it has the key share of the server, and sends the “Client Finished” message along with an HTTP request.
4. The server completes the handshake by sending an HTTP response.

The following digital signature algorithms are supported in TLS handshake:

Table 2-3 Digital Signature Algorithms

Algorithm	Key Size (Bits)	Elliptic Curve (EC)
RS256 (RSA)	2048	NA
	4096 This is the recommended value.	NA
ES256 (ECDSA)	NA	SECP384r1 This is the recommended value.

Comparison Between TLSv1.2 and TLSv1.3

The following table provides a comparison of TLSv1.2 and TLSv1.3:

Table 2-4 Comparison of TLSv1.2 and TLSv1.3

Feature	TLS v1.2	TLS v1.3
TLS Handshake	• The initial handshake was carried out in clear text. • A typical handshake in TLSv1.2 involves the exchange of 5 to 7 packets.	• The initial handshake is carried out along with the key share. • A typical handshake IN TLSv1.3 involves the exchange of up to 3 packets.
Cipher Suites	• Less secure Cipher suites. • Use SHA-256 and SHA-384 hashing – TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 – TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 – TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256 – TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 – TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256	• More secure Cipher suites. • Apart from all the ciphers supported for TLSv1.2, the following additional ciphers are supported for only TLSv1.3: – TLS_AES_128_GCM_SHA256 – TLS_AES_256_GCM_SHA384 – TLS_CHACHA20_POLY1305_SHA256
Round-Trip Time (RTT)	This has a high RTT during the TLS handshake.	This has low RTT during the TLS handshake.
Perfect Forward Secrecy (PFS)	This doesn't support PFS.	TLSv1.3 supports PFS. PFS ensures that each session key is completely independent of long-term private keys, which are keys that are used for an extended period to decrypt encrypted data.
Privacy	This is less secure, as the ciphers used are weak.	This is more secure, as the ciphers used are strong.
Performance	This has high latency and a less responsive connection.	This has low latency and a more responsive connection.

Advantages of TLSv1.3

The TLSv1.3 handshake offers the following improvements over earlier versions:

- All handshake messages after the ServerHello are encrypted.
- It improves efficiency in the handshake process by requiring fewer round trips than TLSv1.2. It also uses cryptographic algorithms that are faster.
- It provides better security than TLSv1.2, addressing known vulnerabilities in the handshake process.
- It eliminates data compression.

The following table describes the TLS versions supported on the client and server sides. The last column indicates which version will be used.

TLS Version Used

When CNC Console is acting as a client or a server, it can support different TLS versions.

The following table provides information about which TLS version will be used when various combinations of TLS versions are present between the server and the client.

Table 2-5 TLS Version Used

Client Support	Server Support	TLS Version Used
TLSv1.2, TLSv1.3	TLSv1.2, TLSv1.3	TLSv1.3
TLSv1.3	TLSv1.3	TLSv1.3
TLSv1.3	TLSv1.2, TLSv1.3	TLSv1.3
TLSv1.2, TLSv1.3	TLS v1.3	TLSv1.3
TLSv1.2	TLSv1.2, TLSv1.3	TLSv1.2
TLSv1.2, TLSv1.3	TLSv1.2	TLSv1.2
TLS v1.3	TLSv1.2	Sends an error message. For more information about the error message, see *Oracle Communications Cloud Native Configuration Console Troubleshooting Guide.*
TLSv1.2	TLSv1.3	Sends an error message. For more information about the error message, see *Oracle Communications Cloud Native Configuration Console Troubleshooting Guide.*

ⓘ **Note**

- If Egress Gateway is deployed with both the versions of TLS that is TLSv1.2 and TLSv1.3, then Egress Gateway as client will send both versions of TLS in the client hello message during the handshake and the server needs to decide which version to be used.
- If Ingress Gateway is deployed with both the version of TLS that is with TLSv1.2 and TLSv1.3, then Ingress Gateway as the server will use the TLS version received from the client in the server hello message during the handshake.
- This feature does not work in ASM deployment.

Managing Support for TLSv1.2 and TLSv1.3

Enable:

This feature can be enabled or disabled at the time of CNC Console deployment using the following Helm parameters:

- **enableIncomingHttps**: This flag is used for enabling/disabling HTTPS/2.0 (secured TLS) in the Ingress Gateway. If the value is set to false, CNC Console will not accept any HTTPS/2.0 (secured) traffic. If the value is set to true, CNC Console will accept HTTPS/2.0 (secured) traffic.
- **enableOutgoingHttps**: This flag is used for enabling/disabling HTTPS/2.0 (secured TLS) in the Egress Gateway. If the value is set to false, CNC Console will not accept any HTTPS/2.0 (secured) traffic. If the value is set to true, CNC Console will accept HTTPS/2.0 (secured) traffic.

For more information on enabling this flag, see the "Customizing CNC Console" section in *Oracle Communications Cloud Native Configuration Console Installation, Upgrade, and Fault Recovery Guide*.

Configure

You can configure this feature using Helm parameters.

The following parameters in the Ingress Gateway and Egress Gateway microservices must be customized to support TLSv1.2 or TLSv1.3:

1. **Generate HTTPS certificates** for both the ingress and egress gateways. Ensure that the certificates are correctly configured for secure communication. After generating the certificates, create a Kubernetes secret for each gateway (egress and ingress). Then, configure these secrets to be used by the respective gateways. For more information about HTTPS configuration, generating certificates, and creating secrets, see the "Configuring Secrets for Enabling HTTPS" section in the *Oracle Communications Cloud Native Configuration Console Installation, Upgrade, and Fault Recovery Guide* .
2. After configuring the secret and applying it to the namespace where CNC Console is deployed, perform the following Helm changes for Ingress and Egress gateways in the `occncc_custom_values_<version>.yaml` file:
   - **Parameters required to support TLSv1.2**:
     - `service.ssl.tlsVersion` indicates the TLS version.
     - `cipherSuites` indicates supported cipher suites.
     - `allowedCipherSuites` indicates allowed cipher suites.
   - **Parameters required to support TLSv1.3**:
     - `service.ssl.tlsVersion` indicates the TLS version.
     - `cipherSuites` indicates the supported cipher suites.
     - `allowedCipherSuites` indicates the allowed cipher suites.
     - `clientDisabledExtension` is used to disable the extension sent by messages originating from clients during the TLS handshake with the server.
     - `serverDisabledExtension` is used to disable the extension sent by messages originating from servers during the TLS handshake with the client.
     - `tlsNamedGroups` is used to provide a list of values sent in the `supported_groups` extension. These are comma-separated values.
     - `clientSignatureSchemes` is used to provide a list of values sent in the `signature_algorithms` extension.

   For more information about configuring the values of the above-mentioned parameters, see the "Customizing CNC Console" section in the *Oracle Communications Cloud Native Configuration Console Installation, Upgrade, and Fault Recovery Guide.*
3. Save the `occncc_custom_values_<version>.yaml` file.
4. Install CNC Console. For more information about the installation procedure, see the *Oracle Communications Cloud Native Configuration Console Installation, Upgrade, and Fault Recovery Guide* .
5. Run Helm upgrade if you are enabling this feature after CNC Console deployment. For more information about the upgrade procedure, see the *Oracle Communications Cloud Native Configuration Console Installation, Upgrade, and Fault Recovery Guide* .

> **Note**
>
> - CNC Console does not prioritize cipher suites based on priority. To select a cipher based on priority, you must list the cipher suites in decreasing order of priority.
> - CNC Console does not prioritize supported groups based on priority. To select a supported group based on priority, you must list the supported group values in decreasing order of priority.
> - If you want to provide values for the `signature_algorithms` extension using the `clientSignatureSchemes` parameter, the following comma-separated values must be provided to deploy the services:
>   - rsa_pkcs1_sha512
>   - rsa_pkcs1_sha384
>   - rsa_pkcs1_sha256
>
>   > **Note**
>   >
>   > By default, it is `null`.
>
> - The mandatory extensions as listed in RFC 8446 cannot be disabled using the `clientDisabledExtension` attribute on the client or using the `serverDisabledExtension` attribute on the server side. The following is the list of the extensions that cannot be disabled:
>   - supported_versions
>   - key_share
>   - supported_groups
>   - signature_algorithms
>   - pre_shared_key

Observe

Metrics

There are no new metrics for this feature.

For more information about metrics, see CNC Console Metrics section.

KPIs

There are no new KPIs for this feature.

Alerts

There are no new alerts for this feature.

Maintain

To resolve any alerts at the system or application level, see CNC Console Alerts section. If the alerts persist, perform the following:

1. **Collect the logs**: For more information on how to collect logs, see *Oracle Communications Cloud Native Core, Network Slice Selection Function Troubleshooting Guide*.

2. **Raise a service request**: See My Oracle Support for more information on how to raise a service request.

2.11.1 TLS 1.3 Support for Kubernetes API Server Communication

In a Kubernetes-based 5G Core deployment, CNC Console communicates with the Kubernetes API server (Kube-Api-Server) to retrieve the secrets and ConfigMap information.

With the implementation of this feature, CNC Console supports TLSv1.3 in addition to TLSv1.2 for establishing secure communication with the Kubernetes API server.

> **Note**
>
> - This enhancement does not support both TLSv1.3 and TLSv1.2 concurrently. All the pods can either communicate through TLSv1.3 or TLSv1.2 with Kubernetes API server.
> - An application TLS server configured with TLSv1.3 requires a corresponding CNE version that also supports TLSv1.3.
> - When secrets are volume mounted, changes may not be detected immediately within the configured reload period. Kubelet may take a few seconds to a few minutes (typically 1 to 3 minutes) to reflect updates in the volumes. Once the changes appear in the volumes, they are detected in the next scheduler run, and the necessary actions are triggered.
> - When `tlsVersionSupportForKubeApiServer.enabled` is true, the following can be observed:
>   - Helm installation and upgrade fails if:
>   - invalid TLS versions are configured.
>   - configured cipher suites are not compliant with the configured TLS versions or if no cipher suites are configured.
>   - Mounting secrets from different namespaces is not supported, and all secrets must be located in the same namespace where the Ingress Gateway microservices are deployed.

Managing TLS 1.3 Support for Kubernetes API Server Communication

Enable

This feature is disabled by default at the time of CNC Console deployment.

Perform the following configuration to enable this feature using the Helm:

1. Open the occncc_custom_values_<version>.yaml file.
2. Set the value of `global.tlsVersionSupportForKubeApiServer.enabled` to true under the Global parameters section.
3. Configure the following parameters appropriately:
   a. `global.tlsVersionSupportForKubeApiServer.kubeApiServerTlsVersion` to define the TLS version to be used under the Global parameters section.
   b. `global.tlsVersionSupportForKubeApiServer.cipherSuites` to define the ciphersuites used for the configured TLS version under the Global parameters section.

4. For more information about the above parameters, see the “Customizing CNC Console” in *Oracle Communications Cloud Native Configuration Console Installation, Upgrade, and Fault Recovery Guide.*
5. Save the file.
6. Install CNC Console. For more information about installation procedure, see *Oracle Communications Cloud Native Configuration Console Installation, Upgrade, and Fault Recovery Guide*.
7. Run Helm upgrade if you are enabling this feature after CNC Console deployment. For more information about upgrade procedure, see *Oracle Communications Cloud Native Configuration Console Installation, Upgrade, and Fault Recovery Guide*.

Observability

Metrics

The following metrics are added for this feature:

- oc_ingressgateway_incoming_tls_connections{InstanceIdentifier=~".*iam_ingressgateway"}
- oc_ingressgateway_incoming_tls_connections{InstanceIdentifier=~".*mcore_ingressgateway"}
- oc_ingressgateway_incoming_tls_connections{InstanceIdentifier=~".*acore_ingressgateway"}
- security_cert_x509_expiration_seconds{InstanceIdentifier=~".*iam_ingressgateway"}
- security_cert_x509_expiration_seconds{InstanceIdentifier=~".*mcore_ingressgateway"}
- security_cert_x509_expiration_seconds{InstanceIdentifier=~".*acore_ingressgateway"}

For more information about metrics, see CNC Console Metrics section.

Alerts

There are no alerts related to this feature.

KPIs
There are no KPIs related to this feature.

Maintain

To resolve any alerts at the system or application level, see CNC Console Alerts section. If the alerts persist, perform the following:

1. **Collect the logs**: For more information on how to collect logs, see *Oracle Communications Cloud Native Core, Network Slice Selection Function Troubleshooting Guide*.
2. **Raise a service request**: See My Oracle Support for more information on how to raise a service request.

2.12 TLS Support with Automated Certificate Management

Introduction

In CNC Console 24.2.x and earlier, X.509 and Transport Layer Security (TLS) certificates were managed manually. When multiple instances of CNC Console were deployed in a 5G network, certificate management, such as certificate creation, renewal, removal, and so on, became tedious and error-prone.

Starting with CNC Console 24.3.x, you can integrate CNC Console with Oracle Communications Cloud Native Core, Certificate Management (OCCM) to support automation of certificate lifecycle management. OCCM manages TLS certificates stored in Kubernetes secrets by integrating with Certificate Authority (CA) using the Certificate Management Protocol Version 2 (CMPv2) protocol in the Kubernetes secret. OCCM obtains and signs TLS certificates within the CNC Console namespace. For more information about OCCM, see Oracle Communications Cloud Native Core, Certificate Management User Guide.

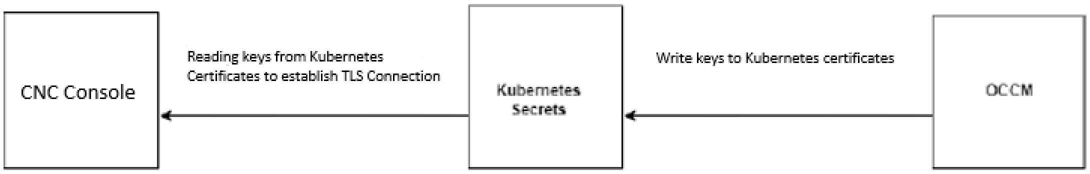
Support for OCCM

This feature enables CNC Console to create, recreate, and renew TLS certificates using OCCM. Certificate validity is monitored for auto-renewal. For information about enabling HTTPS, see "Configuring Secrets for Enabling HTTPS" in Oracle Communications Cloud Native Core, Cloud Native Core Console Installation, Upgrade, and Fault Recovery Guide.

The below diagram indicates that OCCM writes the keys to the certificates and CNC Console reads these keys to establish a TLS connection.

- M-CNCC IAM TLS certificates
- M-CNCC Core TLS certificates
- A-CNCC Core TLS Certificates

Figure 2-6 Establishing Connection Between CNC Console and TLS Using OCCM Key


CNC Console
Reading keys from Kubernetes Certificates to establish TLS Connection
Kubernetes Secrets
Write keys to Kubernetes certificates
OCCM

Install Guide Considerations

Upgrade: When CNC Console is deployed with OCCM, follow the specific upgrade procedure. For more information, see "Upgrade Automated Certificate Lifecycle Management Through OCCM" in *Oracle Communications Cloud Native Configuration Console Installation, Upgrade, and Fault Recovery Guide.*

• **Rollback**: For more information on migrating the secrets from CNC Console to OCCM, see "Rollback from Automated Certificate Management to Manual Certificate Management" in *Oracle Communications Cloud Native Configuration Console Installation, Upgrade, and Fault Recovery Guide.*

Maintain

If you encounter any OCCM-specific alerts, see the "OCCM Alerts" section in *Oracle Communications Cloud Native Core, Certificate Management User Guide.*
If you encounter alerts at system or application levels, see CNC Console Alerts section for resolution steps.

In case the alert still persists, perform the following:

1. **Collect the logs and Troubleshooting Scenarios**: For more information on how to collect logs and troubleshooting information, see *Oracle Communications Cloud Native Configuration Console Troubleshooting Guide.*

2. **Raise a service request:** See My Oracle Support for more information on how to raise a service request.

2.13 Service Mesh Communication

> **Note**
>
> Not applicable for OCI deployment.

CNC Console leverages the Istio or Envoy service mesh (Aspen Service Mesh) for all internal and external communication. The service mesh integration provides Console-NF communication and allows API gateway to co-work with service mesh. The service mesh integration supports these services by deploying a special sidecar proxy in the environment to intercept all network communication between microservices.

For more information, see *CNC Console Configuration to Support ASM and OSO* in *Oracle Communications Cloud Native Configuration Console Installation and Upgrade Guide*.

2.14 Support for Dual Stack Networking

Using the dual-stack mechanism, applications or NFs can establish connections with pods and services in a Kubernetes cluster using IPv4 or IPv6 or both simultaneously. Dual stack provides:

- Coexistence strategy that allows hosts to reach IPv4 and IPv6 simultaneously.
- IPv4 and IPv6 allocation to the Kubernetes clusters during cluster creation. This allocation is applicable for all Kubernetes resources unless explicitly specified during cluster creation.

Kubernetes cluster can have either IPv4 preferred or IPv6 preferred IP address configuration.

IP Address Allocation to Pods

IP address allocation to pods depends on the IP address preference set in the Kubernetes cluster. Pods do not have the ability to choose an IP address. Consider the following example of a pod deployed in an IPv4 preferred infrastructure. Here, if the Kubernetes cluster has IPv4 preferred configuration, both IPv4 and IPv6 are allocated to the pod, but the primary IP address is IPv4. Example:

```
IP:                  10.xxx.xxx.xxx
IPs:
  IP:              10.xxx.xxx.xxx
  IP:              fd00::1:cxxx:bxxx:8xxx:xxxx
Controlled By:  ReplicaSet/cncc-iam-ingress-gateway-577d8c7d66
Containers:
.....
```

IP Address Allocation to Services

IP address allocation to all the console services, depends on the `cnccDeploymentMode` Helm parameter configuration. This Helm parameter automatically configures IP Family Policy and IP Families attributes to the console services. For more information about `cnccDeploymentMode`, see the "Customizing CNC Console" section in the *Oracle Communications Cloud Native Configuration Console Installation, Upgrade, and Fault Recovery Guide.*

You can customize the IP address allocation to services based on the cnccDeploymentMode Helm parameter. Services route the traffic to the destination endpoints based on this configuration. If the cnccDeploymentMode Helm parameter is set to IPv4, then IPv4 is allocated to services, and services use IPv4 pod IPs to send the traffic to endpoints.

The following table describes how IP address allocation, IP Family Policy, and IP Families vary based on the cnccDeploymentMode Helm parameter configuration for services:

Table 2-6 IP Address Allocation

Infrastructure Preference	Application Preference (cnccDeploymentMode Helm Parameter)	IP Family Policy Attribute	IP Families Attribute	Pod IP	Service IP	Endpoints
IPv4 Preferred	IPv4	SingleStack	IPv4	IPv4,IPv6	IPv4	IPv4
IPv6 Preferred	IPv4	SingleStack	IPv4	IPv6,IPv4	IPv4	IPv4
IPv4 Preferred	IPv6	SingleStack	IPv6	IPv4,IPv6	IPv6	IPv6
IPv6 Preferred	IPv6	SingleStack	IPv6	IPv6,IPv4	IPv6	IPv6
IPv4 Preferred	IPv4_IPv6 (IPv4Preferred)	RequiredDualStack	IPv4 Preferred	IPv4,IPv6	IPv4,IPv6	IPv4
IPv6 Preferred	IPv4_IPv6 (IPv4Preferred)	RequiredDualStack	IPv4 Preferred	IPv6,IPv4	IPv6,IPv4	IPv4
IPv4 Preferred	IPv6_IPv4 (IPv6Preferred)	RequiredDualStack	IPv6 Preferred	IPv4,IPv6	IPv4,IPv6	IPv6
IPv6 Preferred	IPv6_IPv4 (IPv6Preferred)	RequiredDualStack	IPv6 Preferred	IPv6,IPv4	IPv6,IPv4	IPv6

The following columns are used in the table above:

- Infrastructure Preference: This is the Kubernetes cluster deployment. It can have either IPv4 preferred or IPv6 preferred.
- cnccDeploymentMode Helm Parameter: This parameter is configured to set the preference of IP address allocation to the CNC Console services. It has the following values:
  - IPv4: The Ingress Gateway service will be single stack with IPv4 only. It uses IPv4 pod IPs to establish connections.
  - IPv6: The Ingress Gateway service will be single stack with IPv6 only. It uses IPv6 pod IPs to establish connections.
  - IPv4_IPv6: The Ingress Gateway service will be dual stack with both IPv4 and IPv6 addresses. It uses IPv4 pod IPs to establish connections.
  - IPv6_IPv4: The Ingress Gateway service will be dual stack with both IPv4 and IPv6 addresses. It uses IPv6 pod IPs to establish connections.
  - ClusterPreferred: Default value configured for cnccDeploymentMode, All console services will be single stack with IPv4 or IPv6 based on the infrastructure preference.
- IP Family Policy Attribute: This attribute is automatically configured based on the cnccDeploymentMode Helm parameter configuration. It has the following values:
  - SingleStack: It can allocate only a single IP address to services based on the IP Families attribute configuration. For example, if the IP Family Policy attribute is set to SingleStack and IP Families is set to IPv4, then IPv4 is allocated to services.
  - RequiredDualStack: It can allocate both the IP addresses to services based on the IP Families attribute configuration. For example, if the IP Family Policy attribute is set to

RequiredDualStack and IP Families is set to IPv6 preferred, then both IPv6 and IPv4 are allocated to services with IPv6 preferred. If the infrastructure is not dual stack, then Helm upgrade or install will fail.

- IP Families Attribute: This attribute is automatically configured based on the cnccDeploymentMode Helm parameter. It has the following values:
  - IPv4
  - IPv6
  - IPv4 Preferred
  - IPv6 Preferred
- Pod IP: Primary IP address allocated to pods in the Kubernetes cluster.
- Service IP: Primary IP address allocated to services in the Kubernetes cluster.
- Endpoints: Selected pod IP addresses based on the primary service IP.

Example of a service deployed with IPv4 preferred IP address:

```
IP Family Policy:    RequireDualStack
IP Families:         IPv4,IPv6
IP:                  10.xx.xx.xx
IPs:                 10.xx.xx.xx,fd00:x:x:x::xxxx
```

Example of a service deployed with IPv6 preferred IP address:

```
IP Family Policy:     RequireDualStack
IP Families:          IPv6,IPv4
IP:                   fd00:x:x:x::xx
IPs:                  fd00:x:x:x::xxxx,10.xx.xx.xxx
```

Enable

This feature is disabled by default. You can enable this feature by configuring the `cnccDeploymentMode` Helm parameter appropriately.

2.15 CNC Console GUI Session Timeout

The duration (in seconds) before a CNC Console GUI session times out and the user has to log in again can be configured using the `ingress-gateway.cncc.core.sessionTimeout` parameter in the custom_values.yaml file.

By default, the session timeout value is set to 1800 seconds. However, you can set it to any value between 300 and 7200 seconds.

Note

The value of the CNC Console IAM SSO session idle timeout configuration is not considered for CNC Console Core session management.

2.16 Support for Instance Level Access Control

With this feature, CNC Console supports access management at the instance level. This enables CNC Console to enforce restrictions on users based on the instances assigned to them. With instance-based access restriction, CNC Console can prevent users from accessing instances to which they are not entitled. This capability is in addition to the existing RBAC capabilities.

Instance level access control is supported for the following deployment scenarios:

- Non-OCI deployments
- OCI Deployments

CNC Console allows associating instance roles to the users. This feature can be controlled using the `instanceLevelAuthorizationEnabled` Helm parameter. When this feature is enabled, user must have the instance level role to access the instance. For more information on instance level role, see Types of Roles in CNC Console.

When this feature is enabled for the first time, the INSTANCE_ALL role is assigned to all existing users. To control instance-level access, the operator must assign instance specific roles and remove the INSTANCE_ALL role from users.

2.17 Network Policies

> **Note**
>
> Not applicable for OCI deployment.

Network Policies are an application-centric construct that allow you to specify how a pod communicates with various network entities. They create pod-level rules to control communication between cluster pods and services, and determine which pods and services can access one another inside a cluster.

Previously, the pods under CNC Console deployment could be contacted by any other pods in the Kubernetes cluster without any restrictions. Now, network policies provide namespace-level isolation, which allows secure communications to and from CNC Console with rules defined in the respective network policies. Network policies enforce access restrictions for all the applicable data flows, except communication from Kubernetes node to pod for invoking container probe. For example, CNC Console internal microservices cannot be contacted directly by any other pods.

Managing Support for Network Policies

Enable

To use this feature, network policies must be applied to the namespace where CNC Console is deployed.

Configure

You can configure this feature using Helm. For information about Configuring Network Policy for CNC Console Deployment, see *Cloud Native Configuration Console Installation, Upgrade, and Fault Recovery Guide*.

Observe

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

2.18 Traffic Segregation

This feature provides end-to-end traffic segregation to CNC Console 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 each CNC Console 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 CNC Console 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.

This 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 CNC Console.

Prerequisites

The CNLB feature is only available in CNC Console 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 alternative 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. CNC Console supports two types of CNLB NADs:

1. **Ingress Network Attachment Definitions**
   Ingress NADs are used to handle inbound traffic only. This traffic enters the CNLB application through an external interface service IP address and is routed internally using interfaces within CNLB networks.
   - Naming Convention: `nf-<service_network_name>-int`
2. **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`

Managing Ingress and Egress Traffic Segregation

Enable:

This feature is disabled by default. To enable this feature, you must configure the network attachment annotations in the custom values file.

Configuration

For more information about Traffic Segregation configuration, see the "Installing CNC Console Package" section in the *Oracle Communications Cloud Native Configuration Console Installation, Upgrade, and Fault Recovery Guide.*

Observe

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

Maintain

To resolve any alerts at the system or application level, see CNC Console Alerts section. If the alerts persist, perform the following:

1. Collect the logs: For more information on how to collect logs, see Oracle Communications Cloud Native Core, Network Slice Selection Function Troubleshooting Guide.
2. Raise a service request: See My Oracle Support for more information on how to raise a service request.

3 Logging into CNC Console

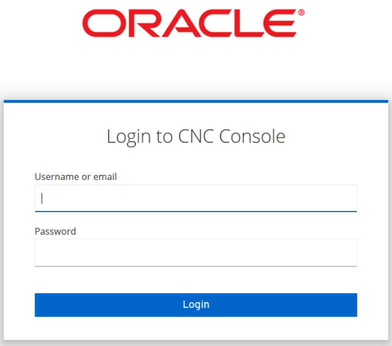
Logging into CNC Console GUI for Non-OCI Deployment

Use the following procedure to sign in to CNC Console:

1. Open any browser.
2. Enter the URL: ***http://<host name>:<port number>***.
   Here, *host name* is cncc-mcore-ingress-ip and *port number* is cncc-mcore-ingressport.

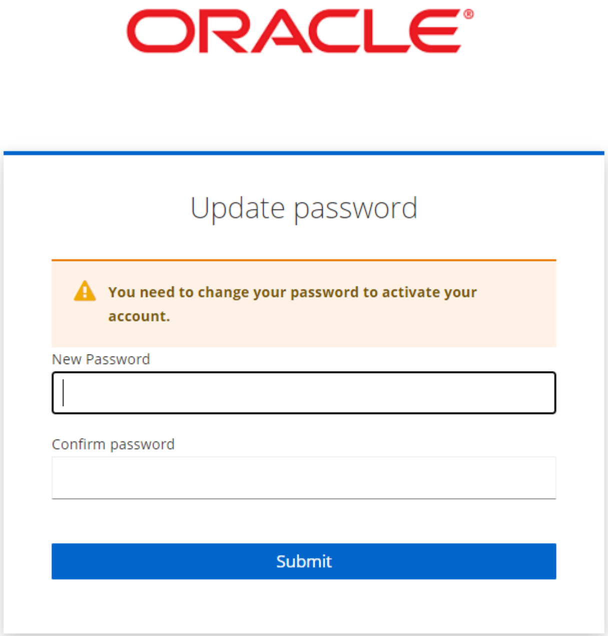
   The following screen appears:

   Figure 3-1 CNC Console Login

3. Enter valid credentials.
4. Click **Login**. The welcome page of the CNC Console interface appears.
5. After the first login, you will be prompted to change your password.

Figure 3-2 Update Password



Update password

You need to change your password to activate your account.

New Password

Confirm password

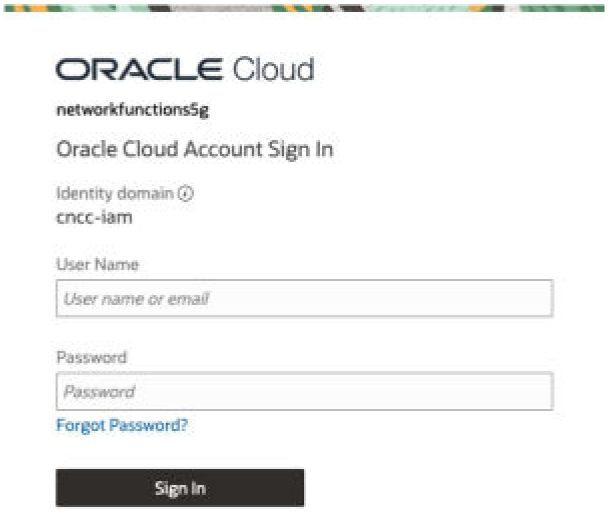
> **Note**
>
> To configure CNC Console IAM, see the Configuring CNC Console IAM section.

Logging into CNC Console GUI in OCI Deployment

Use the following procedure to sign in to CNC Console

1. Open any browser.
2. Enter the M-CNCC Core URL: ***http://<host name>:<port number>***.
3. Enter valid credentials.

Figure 3-3 Login page



ORACLE Cloud

networkfunctions5g

Oracle Cloud Account Sign In

Identity domain ⓘ
cncc-iam

User Name
User name or email

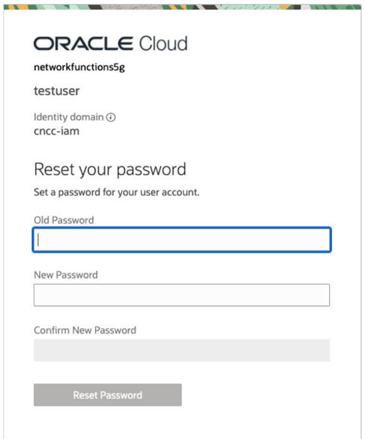
Password
Password

Forgot Password?

Sign In

4. Click **Sign In**.
5. After signing in, you will be prompted to change your password.

Figure 3-4 Reset password



ORACLE Cloud

networkfunctions5g

testuser

Identity domain ⓘ
cncc-iam

Reset your password

Set a password for your user account.

Old Password

New Password

Confirm New Password

Reset Password

6. Click **Reset Password**.
7. You are redirected to the CNC Console Core welcome page.

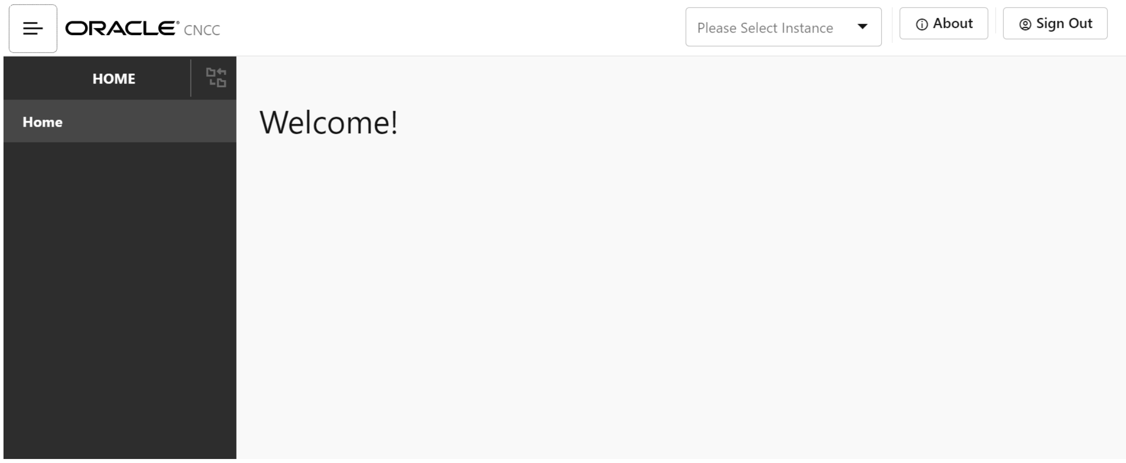
> **Note**
>
> If any issues are identified while accessing the CNC Console GUI, see the *Oracle Communications Cloud Native Configuration Console Troubleshooting Guide.*

3.1 CNC Console Dashboard

This section provides an overview of the CNC Console Dashboard.

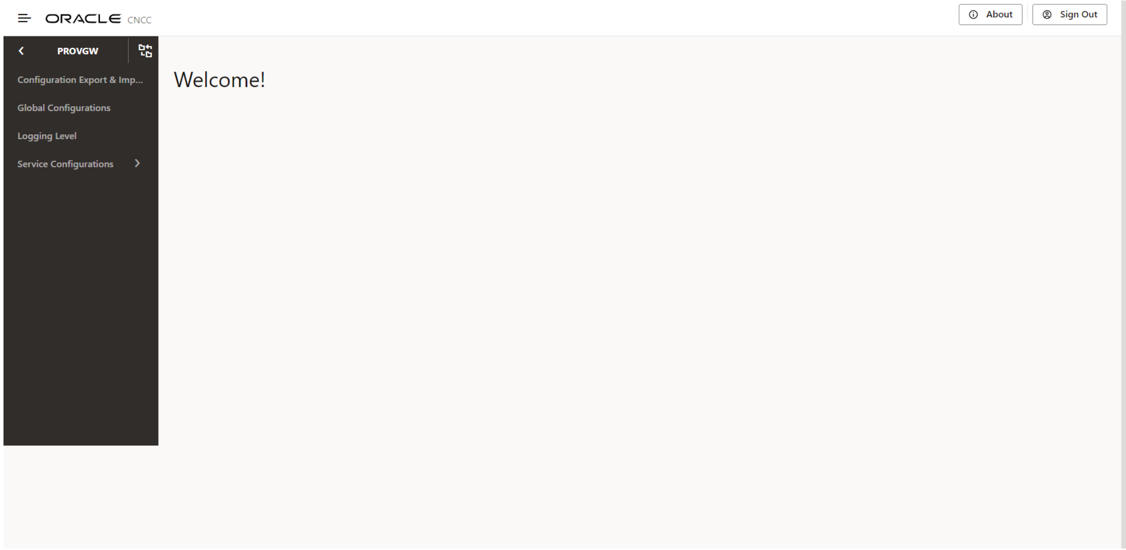
After you sign in, the CNC Console welcome page appears:

Figure 3-5   CNC Console Welcome Page



- **Top Ribbon** – The top ribbon contains the following:
  - **Instance selection drop-down list** – Select the NF instance from the list.
  - **About** – Displays the product name and version of the interface.
  - **Sign Out** – Signs out from CNC Console.
- **Left Pane- NFs and Configurations:**
  The left pane displays the selected network function and its configurations.
- **Right Pane- Details View:**
  The right pane displays the configurable parameters for the selected NFs.
- **Other dashboard options:**

Figure 3-6   Dashboard Options



- The **Menu Hierarchy** button displays the navigation path from the home screen to the current menu item.

- The **Application Navigation** allows you to collapse the left pane and display the screen in full.
- The **Back Icon** allows the user to navigate to the Home menu. The screen does not get refreshed automatically. You must click **Home** or the NF menu to view the updated screen.

3.2 Viewing cnDBTier APIs in CNC Console

You must enable the CNCC instance to view the cnDBTier menu. For more information, see the NF Instance Configuration With cnDBTier Menu Enabled section in the *Oracle Communications Cloud Native Configuration Console Installation, Upgrade, and Fault Recovery Guide*.

Perform the following procedure to view the cnDBTier APIs on the CNC Console:

> ⓘ **Note**
>
> The following cnDBTier APIs are read-only.

Selecting the CNC Console Instance

1. Log in to CNC Console and navigate to the **Select Instances** drop-down.
2. Select the CNC Console instance.
3. From the left navigation pane, click the NF or CNC Console tab, and then click the **cnDBTier** tab. A list of cnDBTier APIs is displayed.

Backup List

Perform the following procedure to view the list of completed backups:

- From the left navigation pane, click the NF or CNC Console tab, and then click the **cnDBTier** tab.
- Click **Backup List** to view the list of completed backups, including backup ID, backup size, and creation timestamp.
  The **Backup List** screen is displayed.

Table 3-1 Backup List

Fields	Description
Backup Details	This field displays information such as the backup ID, backup size, and backup creation timestamp.
Site Name	This field displays the name of the current site to which NF is connected.
Backup ID	This field displays the ID of the stored backup.
Backup Size (bytes)	This field displays the size of the stored backup.
Creation Timestamp	This field displays the time recorded when the backup was stored.

cnDBTier Version

Use the following procedure to view the version:

1. From the left navigation pane, click the NF or CNC Console tab, and then click the **cnDBTier** tab.
2. Click the **cnDBTier Version** to view the version.

Table 3-2 cnDBTier Version Attributes

Fields	Description
cnDBTier Version	This field displays the cnDBTier version.
NDB Version	This field displays the network database (NDB) version.

Database Statistics Report

Use the following procedure to view the available databases.

1. From the left navigation pane, click the NF or CNC Console tab, and then click the **cnDBTier** tab.
2. Click the **Database Statistics Report** to view the available databases.

Table 3-3 Database Statistics Report

Fields	Description
Database Count	This field displays the number of available databases.
Database Tables Count	This field displays the available database names and their table counts.
Database Name	This field displays the database name.
Table Count	This field displays the table count for each database.
Database Table Rows Count	This field displays the table rows present in each table.

Click the **View** button next to the database name to view the **View Database Table Rows Count** screen.

Table 3-4 View Database Table Rows Count

Fields	Description
Database Name	This field displays the database name.
Tables	This field displays the table names and the corresponding rows in each table.
Table Name	This field displays the table name.
Row Count	This field displays the table rows present in each table.

Georeplication Status

Use the following procedure to view the local site and remote site name to which CNC Console is connected.

1. From the left navigation pane, click the NF or CNC Console tab, and then click the **cnDBTier** tab.
2. Click **Georeplication Status** to view the local site and remote site name to which NF is connected.

Table 3-5 Georeplication Status

Fields	Description
Local Site Name	This field displays the local site name to which NF is connected.
Remote Site Name	This field displays the remote site name.
Replication Status	This field displays the replication status with the corresponding sites.
Seconds Behind Remote Site	This field displays the number of seconds that the last record read by the local site is behind the latest record written by the remote site for all the replication groups.

a. Click the **View** icon in the **Actions** menu to view the **View Georeplication Status** screen.

Table 3-6 Georeplication Status

Fields	Description
Replication Group Delay	This field displays the number of seconds that the last record read by the local site is behind the latest record written by the remote site for individual replication groups.
Replication Channel Group Id	This field displays the ID of the replication channel group.

b. Click the **View** icon to view the **Replication Group Delay** attributes.

Table 3-7 View Replication Group Delay

Fields	Description
Channel Details	This field displays the channel details such as Remote Replication IP and Role.
Remote Replication IP	This field displays the IP of the remote replication channel.
Role	This field displays the role of the replication channel IP.

Georeplication Recovery

Use the following procedure to mark sites as failed, perform georeplication recovery, and monitor the status:

1. From the left navigation pane, click the NF or CNC Console tab, and then click the **cnDBTier** tab.
2. Click **Georeplication Recovery** to view the failed sites, perform georeplication recovery, and monitor its status.
3. The **Georeplication Recovery** API menu is displayed.
4. Click **Update Cluster As Failed** to view the failed clusters. The Update Cluster As Failed page is displayed.

Table 3-8 Update Cluster As Failed

Fields	Description
Cluster Names	This field displays a list of cnDBTier clusters that can be marked as failed.

Table 3-8 (Cont.) Update Cluster As Failed

Fields	Description
Failed Cluster Names	This field displays a cnDBTier cluster that is marked as failed.

5. Click **Update Cluster**. The selected cluster name is updated in the **Failed Cluster Names** field.
6. Click **Start Georeplication Recovery** to initiate georeplication recovery. The Start Georeplication Recovery page is displayed.

Table 3-9 Start Georeplication Recovery

Fields	Description
Failed Cluster Name	This field displays a list of all the clusters that have been marked as failed.
Backup Cluster Name (Optional)	This field displays a list of all the healthy clusters. If no cluster is selected, the system uses the first available healthy cluster for the backup.

To start georeplication recovery, select the name of the failed cluster from the **Failed Cluster Name** field, and then click **Start Georeplication Recovery.**

7. Click **Georeplication Recovery Status** to monitor the status of the clusters. The Georeplication Recovery Status page is displayed.

Table 3-10 Georeplication Recovery Status

Fields	Description
Local Cluster Name	This field displays the name of the local cluster.
Georeplication Recovery Status Details	This field displays the details of the georeplication recovery status of cnDBTier clusters.
Cluster Name	This field displays the clusters by name.
Georeplication Recovery Status	This field displays the current georeplication recovery status of the corresponding cluster.

For more information on georeplication recovery using cnDBTier APIs, see *Oracle Communications Cloud Native Core, cnDBTier Installation, Upgrade, and Fault Recovery Guide*.

HeartBeat Status

Perform the following procedure to view the connectivity between local site and remote site name to which NF is connected.

1. From the left navigation pane, click the **NF** tab, and then click the **cnDBTier** tab.
2. Click the **HeartBeat Status** to view the connectivity between local site and remote site name to which NF is connected.

Table 3-11 HeartBeat Status Details

Fields	Description
Site Name	This field displays the name of the current site to which NF is connected.
HeartBeat Details	This field displays information such as the remote site name, heartbeat status, heartbeat lag, and replication channel group id.
Remote Site Name	This field displays the remote site name.
Heartbeat Status	This field displays the connectivity status with corresponding sites.
Heartbeat Lag	This field displays the lag or latency in seconds it took to synchronize between sites.
Replication Channel Group Id	This field displays the ID of the replication channel group.

Local Cluster Status

Perform the following procedure to view the local cluster status for the current site.

1. From the left navigation pane, click the **NF** tab, and then click the **cnDBTier** tab.
2. Click the **Local Cluster Status** to view the local cluster status for the current site:

Table 3-12 Local Cluster Status

Fields	Description
Site Name	This field displays the name of the current site to which NF is connected.
Cluster Status	This field displays the local cluster status for the current site.

On Demand Backup

Use the following procedure to view the available databases.

1. From the left navigation pane, click the **NF** tab, and then click the **cnDBTier** tab.
2. Click the **On Demand Backup** to create a new backup and view the status of initiated on-demand backups.

Table 3-13 On Demand Backup Details

Fields	Description
Site Name	This field displays the name of the current site to which NF is connected.
DR Status	This field displays the status of DR.
Backup ID	This field displays the ID of the stored backup.
Backup Status	This field displays the status of backup.
Remote Transfer Status	This field displays the status of remote transfer.
Initiate Backup	This field displays whether the backup is initiated.

a. Click the **Edit** button.
The **Edit** On Demand Backup screen appears.

> ⓘ **Note**
>
> The **Edit** mode is available only for Initiate Backup.

b. Enable the **Initiate Backup** option, then click **Save**.
   A confirmation message, 'Saved successfully,' appears.
c. Click **Cancel** to navigate back to the On Demand Backup screen.
d. Click **Refresh** to reload the On Demand Backup screen.

cnDBTier Health

Use the following procedure to view the health status of the microservices:

1. From the left navigation menu, navigate to **NF** and then click **cnDBTier** tab.
   The **cnDBTier** page is displayed.
2. Click **cnDBTier Health** to view the health status of the microservices, such as replication, backup manager, monitor services, and NDB services.
   The **cnDBTier Health** page is displayed.
   - Click the **Backup Manager Health Status** to view the health status of the backup manager.
     The **Backup Manager Health Status** page is displayed.

> ⓘ **Note**
>
> The following APIs are read-only.

Table 3-14 Backup Manager Health Status

Fields	Description
Service Name	This attribute displays the service name of the backup manager microservice.
Service Status	This attribute displays the service status of the backup manager microservice. Possible values are UP and DOWN.
DB Connection Status	This attribute displays the database connection status of the backup manager microservice. Possible values are UP and DOWN.
Overall Backup Manager Service Health	This attribute displays the overall health status of the backup manager microservice. Possible values are UP and DOWN.
Backup Executor Health Status	This attribute displays the following information like node ID and DB connection status of the backup executor.
Node Id	This attribute displays the ID of the node.
DB Connection Status	This attribute displays the backup executor database connection status with the nodes. Possible values are UP and DOWN.

   - Click the **Monitor Health Status** to view the health status of the services.
     The **Monitor Health Status** page is displayed.

> **Note**
>
> The following APIs are read-only.

Table 3-15 Monitor Health Status details

Attribute	Description
Service Name	This attribute displays the service name of the monitor microservice.
DB Connection Status	This attribute displays the database connection status of the monitor microservice. Possible values are UP and DOWN.
Metric Scrape Status	This attribute displays the status of the metric scrape, that is if the metrics are fetched or not. If the metrics are fetched then the service is up and vice versa. Possible values are UP and DOWN.
Overall Monitor Service Health	This attribute displays the overall health status of the monitor microservice. Possible values are UP and DOWN.

- Click the **NDB Health Status** to view the health status of the network database. The **NDB Health Status** page is displayed.

> **Note**
>
> The following APIs are read-only.

Table 3-16 NDB Health Status details

Attribute	Description
Local Site Name	This attribute displays the name of the current site. For example, site 1, site 2.
NDB Health Status Details	This attribute displays the health status of the network database like name of the NDB service, status of the service, health status of PVC.
Service Name	This attribute displays the service name. For example, ndbmgmd-0, ndbmtd-0, ndbmyappsqld-1, ndbmysqld-2.
Service Status	This attribute displays the status of the service. Possible values are UP and DOWN.
PVC Health Status	This attribute displays the health status of the PVC. Possible values are UP, DOWN, and NA. **Note**: This attribute is set to NA when some of the database pods are not connected to the PVC.

- Click the **Replication Health Status** to view the health status of the replication sites. The **Replication Health Status** page is displayed.

> ⓘ **Note**
>
> The following APIs are read-only.

Table 3-17 Replication Health Status details

Attribute	Description
Local Site Name	This attribute displays the name of the current site (site 1, site 2).
Health Status Details	This attribute displays the health status details of the local site like replication service name, replication service status, database connection status of the replication service, and the overall health status of the replication microservices. The number of rows in this table varies depending on the type of deployment (for example, two-site, three-site deployments).
Service Name	This attribute displays the name of the available replication service.
Service Status	This attribute displays the status of the available replication service. Possible values are UP and DOWN.
DB Connection Status	This attribute displays the database connection status of the replication microservice. Possible values are UP and DOWN.
Overall Replication Service Health	This attribute displays the overall health status of the replication microservice. Possible values are UP and DOWN.

4
Configuring CNC Console IAM

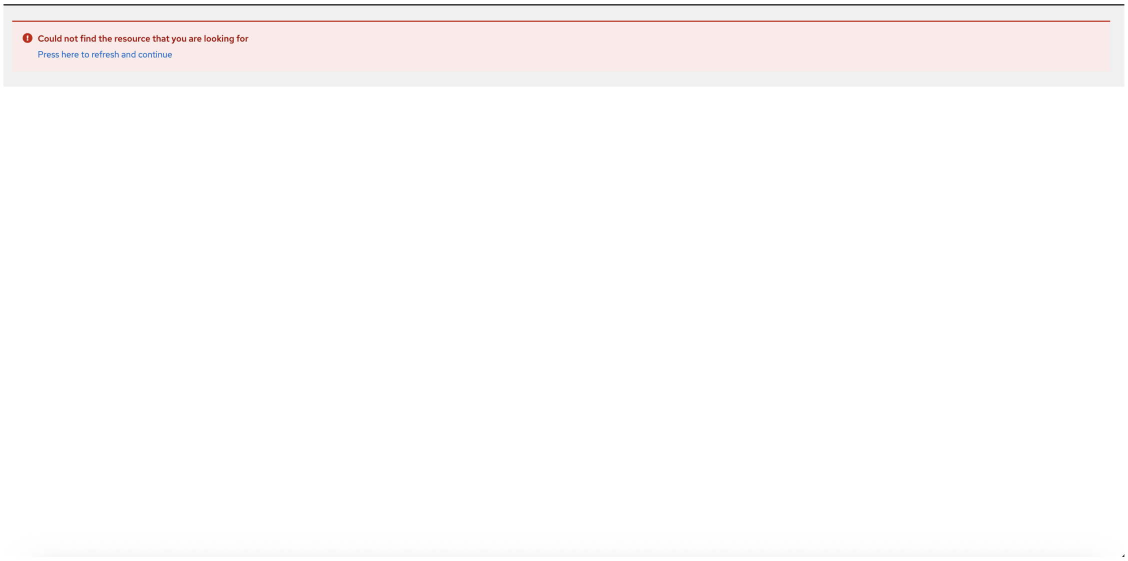
> **Note**
>
> Not applicable for OCI deployment.

This section provides details on how to configure CNC Console IAM.

Restricted Actions on CNC Console IAM

You can only perform those actions that are listed in the *Oracle Communications Cloud Native Configuration Console User Guide* using CNC Console IAM. The following error message appears if you attempt a restricted action:

Figure 4-1 Restrict Access



Could not find the resource that you are looking for

Press here to refresh and continue

Click the **Press here to refresh and continue** link to reload CNC Console IAM.

4.1 Role-Based Access Control in CNC Console IAM

Role-Based Access Control (RBAC) is one of the main methods for advanced access control.

It enables you to restrict network access to authorized users based on their assigned roles.

Role

A Role is a collection of permissions that you can apply to users. Roles are defined according to the authority and responsibility of the users within the organization. Using roles makes it easier to add, remove, and update permissions for users.

Composite Role

A composite role is a collection of one or more additional roles grouped together.

4.1.1 Types of Roles in CNC Console

Role-Based Access Control (RBAC) is managed by the Identity and Access Management (IAM) functionality in CNC Console IAM.

The following roles are predefined in the CNC Console IAM:

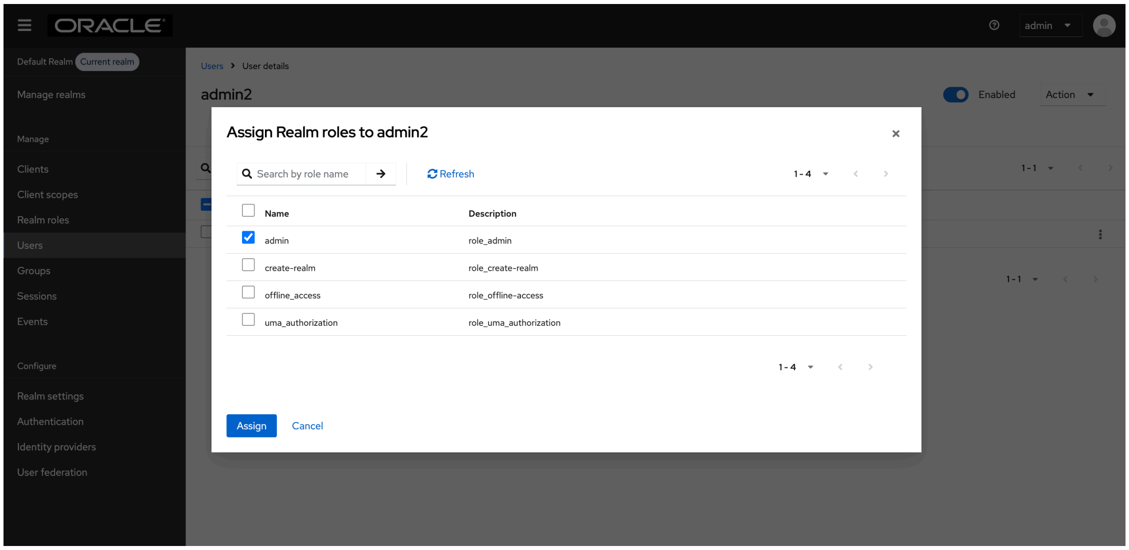
Roles for CNC Console IAM Admin Users

Admin Role

In CNC Console IAM, a user in the default realm must be assigned the Admin role to gain administrative privileges.

An admin user has the necessary permissions to modify settings related to other admin users.

Figure 4-2 Assign Realm roles to admin



Assign Realm roles to admin2

	Name	Description
☑	admin	role_admin
□	create-realm	role_create-realm
□	offline_access	role_offline-access
□	uma_authorization	role_uma_authorization

Roles for CNC Console Core Users

Read and Write Roles:

- **NF Roles**
  The user assigned this role can perform read and write operations for the assigned NFs. NF level roles are classified into:
  - **<NF>_READ:** With this permission, the assigned user can perform the read operation for NFs.
    For example, if the user has the POLICY_READ role, then the user can only read configurations of any MOs within the Policy, and cannot write, update, or delete any record.
  - **<NF>_WRITE:** With this permission, the assigned user can perform create, read, update, and delete operations for NFs. For example, if the user has POLICY_WRITE then the user can read, write, update, or delete any MOs configurations within the NF.

> **Note**
>
> CNCC_READ/WRITE roles are also included under NF roles.

- **Common Services Roles**
  Role: CS_WRITE

  The user assigned this role has access to all the common services and can perform create, read, update, and delete (CRUD) operations.
  The user can read, add, update, or delete MOs configurations for all common services such as Grafana, Kibana, Jaeger, Prometheus, Alertmanager, Promxy, OpenSearch, and Jaeger-ES supported by CNC Console application. For example, if user has CS_WRITE then, the user can read, write, update, or delete any MOs configurations in common services.
- **Admin Roles**
  Role: ADMIN

  The user assigned this role has access to all resources (NF resources and CS resources) within the CNC Console application.

  The user can create, read, update, and delete MOs configurations for all NFs and CSs supported by CNC Console. For example, if the user has ADMIN, then they can read, create, update, or delete any MO configurations of any NFs and CSs supported by CNC Console.

Cluster or Site Roles
In case of multicluster deployments, in addition to ADMIN, <NF>_READ , <NF>_WRITE, or CS_WRITE roles, the user must be assigned a cluster role that corresponds to the cluster in which they are accessing a particular NF or CS. The name of the cluster role must match with role name given in Helm configuration in `global.mCnccCores.role/global.mCnccCores.id` or `global.aCnccs.role/global.aCnccs.id` for M-CNCC and A-CNCC respectively. From 24.2.0 onwards, Cluster roles will be automatically created by Helm hooks.

The user can access all NFs or CS in that cluster. For example, if a user has the Cluster1 role, and ADMIN, <NF>_READ, <NF>_WRITE, or CS_WRITE then they can access all the NFs or VS in Cluster1.

Instance Role

The operator can enable or disable this feature using the `global.instanceLevelAuthorizationEnabled` flag in Helm configuration. By default, this flag is set as **false**. Instance-level roles allow users to have access to specific NF instances. A user can be associated with a single or multiple instance roles. Instance role name must match the instance ID of that particular instance given in Helm configuration in `global.instances[i].id`. Instance roles are automatically created by Helm hooks.

The user can only access one NF or CS. For example, if the user has Cluster1-SCP-instance1 role, ADMIN, SCP_READ, or SCP_WRITE role, and Cluster1 role in case it is a multicluster deployment, then they can only access Cluster1-SCP-instance1.

Role: INSTANCE_ALL

INSTANCE_ALL is a catch-all role for all instance level roles. A user with INSTANCE_ALL role can access all instances, provided they have ADMIN, <NF>_READ, or <NF>_WRITE role, and a cluster-level role if it is a multicluster deployment. This role is automatically assigned to all local users during the first upgrade when this feature is enabled. If the operator wants to restrict a user to a particular instance, then they have to unassign INSTANCE_ALL role and assign any of the instance-level roles to the user.

The user can access all NF and CS instances. For example, if a user has INSTANCE_ALL role, then they can access all NF or CS instances, provided they have ADMIN, <NF>_READ, or <NF>_WRITE role and a cluster-level role in case of a multicluster deployment.

The following table describes the roles that must be assigned to a user to grant them access to NF or CS configurations:

Table 4-1 Accessing NF or CS Configurations

Multicluster Flag (global.isMultiClusterDeployment)	Instance Role Flag (global.instanceLevelAuthorizationEnabled)	Roles Required
Enabled	Enabled	Cluster-level role, instance-level role, and NF-level role
Enabled	Disabled	Cluster Level role and NF Level role
Disabled	Enabled	Instance Level role and NF Level role
Disabled	Disabled	NF Level role

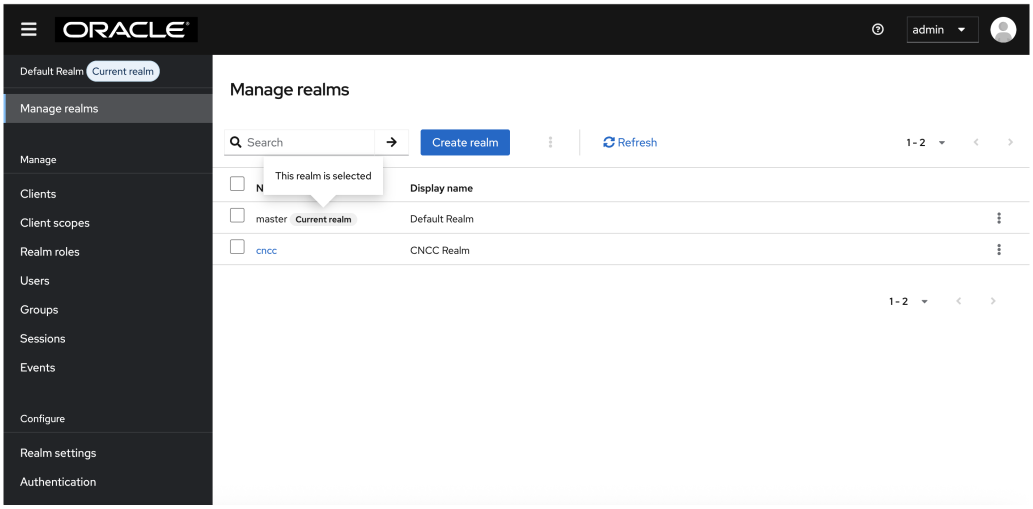
> **Note**
>
> For more information on how to assign roles to a user, see *Oracle Communications Cloud Native Configuration Console Installation, Upgrade, and Fault Recovery Guide.*

4.1.2 Accessing Roles in CNC Console Applications

Viewing the Roles

1. Log into CNC Console IAM using Admin credentials.
   Select the appropriate realm based on the users whose roles you want to view. To switch realms, go to **Manage Realm** and select the desired realm from the list of available realms.
   a. To view roles in the default realm, select realm as shown below:

Figure 4-3 Viewing Realm

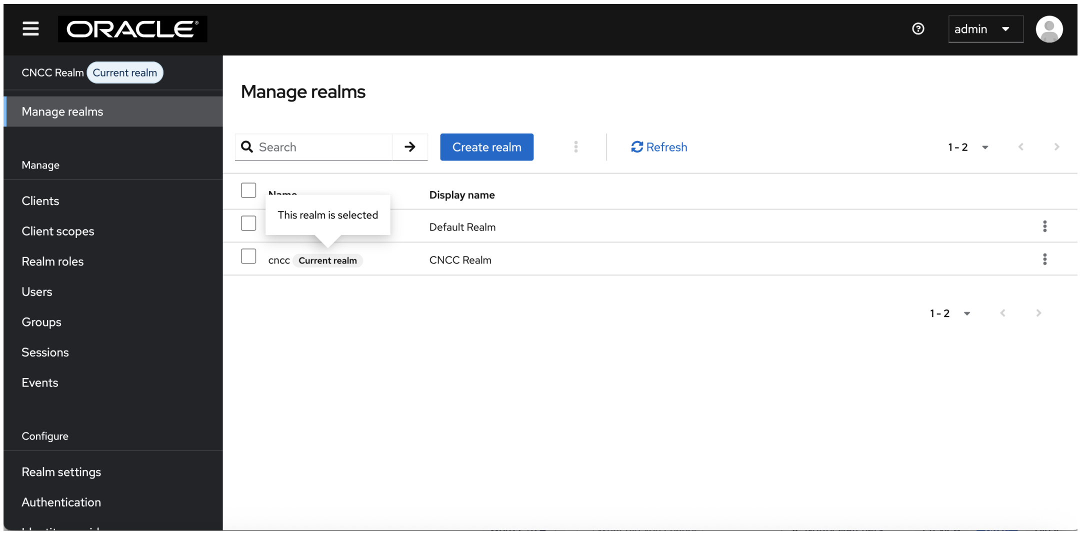


Manage realms

N	Display name
master Current realm	Default Realm
cncc	CNCC Realm

b. To view roles in the CNCC realm, select the **cncc** realm. The following screen appears:

Figure 4-4 Viewing cncc Realm



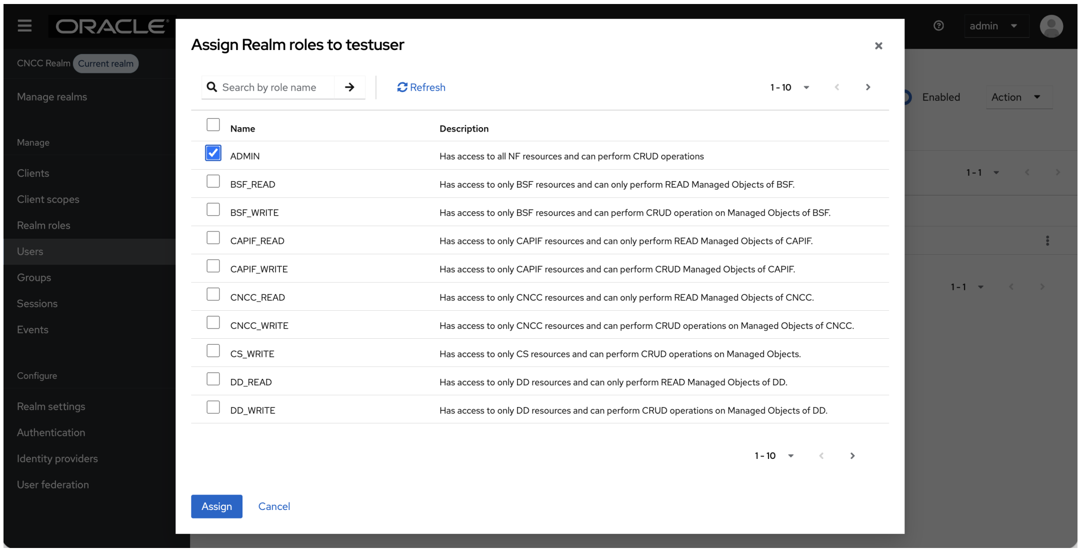
	Name	Display name
□		Default Realm
□	cncc Current realm	CNCC Realm

In the example below, the **cncc** realm is selected to view the available CNCC realm roles.

Follow similar step as outlined below in the default realm to view the available roles for CNCC IAM admin users.

2. To access or view the available roles, click **Realm Roles** on the left pane. The defined roles are displayed on the right pane. Here, the **cncc realm** is selected.

Figure 4-5 Realm roles



□	Name	Description
☑	ADMIN	Has access to all NF resources and can perform CRUD operations
□	BSF_READ	Has access to only BSF resources and can only perform READ Managed Objects of BSF.
□	BSF_WRITE	Has access to only BSF resources and can perform CRUD operation on Managed Objects of BSF.
□	CAPIF_READ	Has access to only CAPIF resources and can only perform READ Managed Objects of CAPIF.
□	CAPIF_WRITE	Has access to only CAPIF resources and can perform CRUD Managed Objects of CAPIF.
□	CNCC_READ	Has access to only CNCC resources and can only perform READ Managed Objects of CNCC.
□	CNCC_WRITE	Has access to only CNCC resources and can perform CRUD operations on Managed Objects of CNCC.
□	CS_WRITE	Has access to only CS resources and can perform CRUD operations on Managed Objects.
□	DD_READ	Has access to only DD resources and can only perform READ Managed Objects of DD.
□	DD_WRITE	Has access to only DD resources and can perform CRUD operations on Managed Objects of DD.

Note

To know more about roles, see Role Based Acess Control in CNC Console IAM.

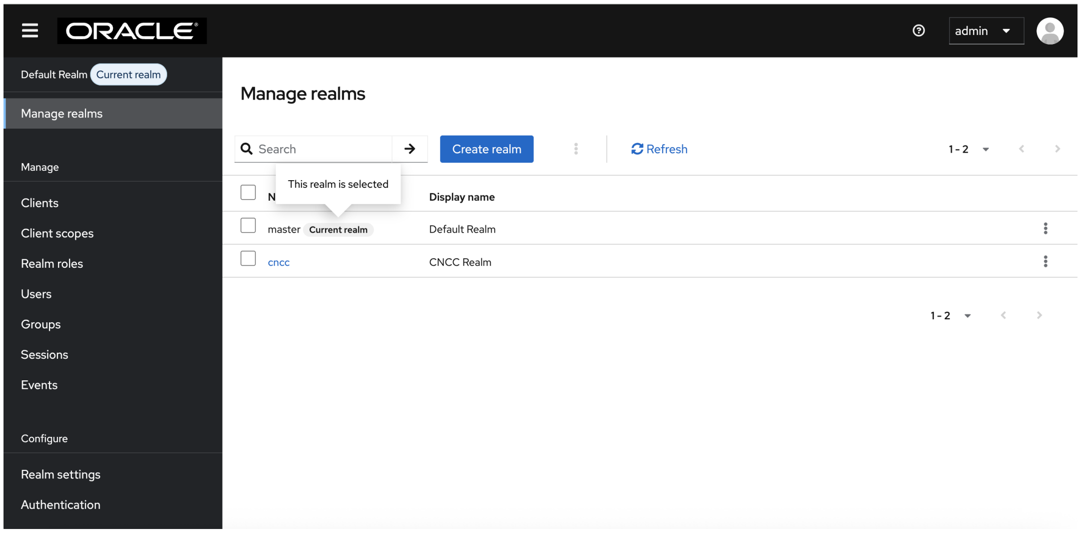
4.1.3 Creating or Updating Admin User Password in CNC Console IAM

This section describes how to create or update the admin password in CNC Console IAM.

CNC Console provides support to change the CNC Console IAM password. To update password:

1. Log in to CNC Console IAM with your login credentials. Go to **Manage realms** and select the default realm from the list of available realms.

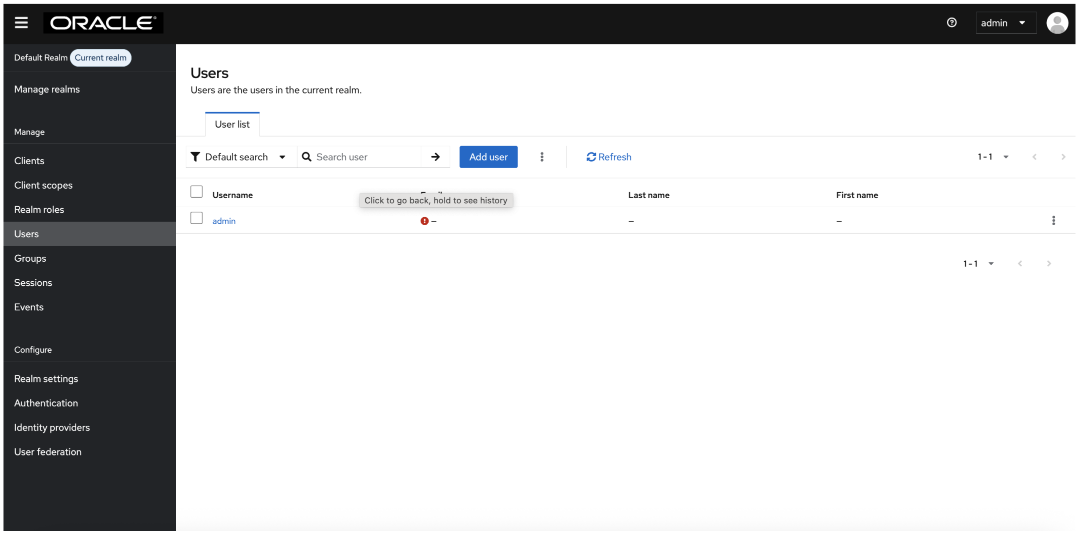
Figure 4-6 Log in to CNC Console IAM



N	Display name
master Current realm	Default Realm
cncc	CNCC Realm

2. Select the **Users** tab to see all the users in the realm. Click the user for which you want to change password.

Figure 4-7 Users



Users

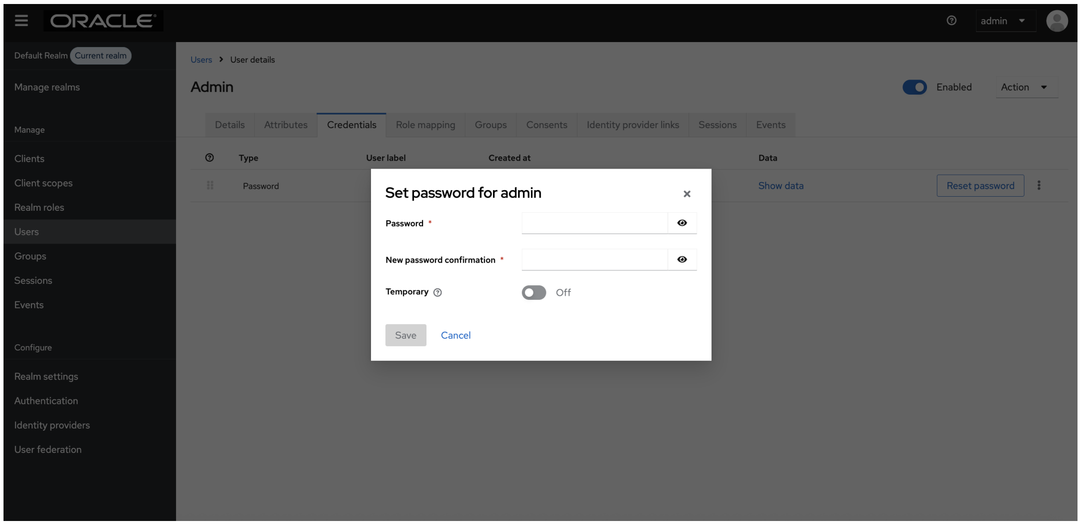
Users are the users in the current realm.

Username	E	Last name	First name
admin	–	–	–

3. Under the **Credentials** tab, click **Reset Password**.

4. Enter the new password in the **Password** field and enter the new password again in the **New Password Confirmation** field.
5. Set **Temporary** to **Off** and click **Save**.

Figure 4-8 Credentials



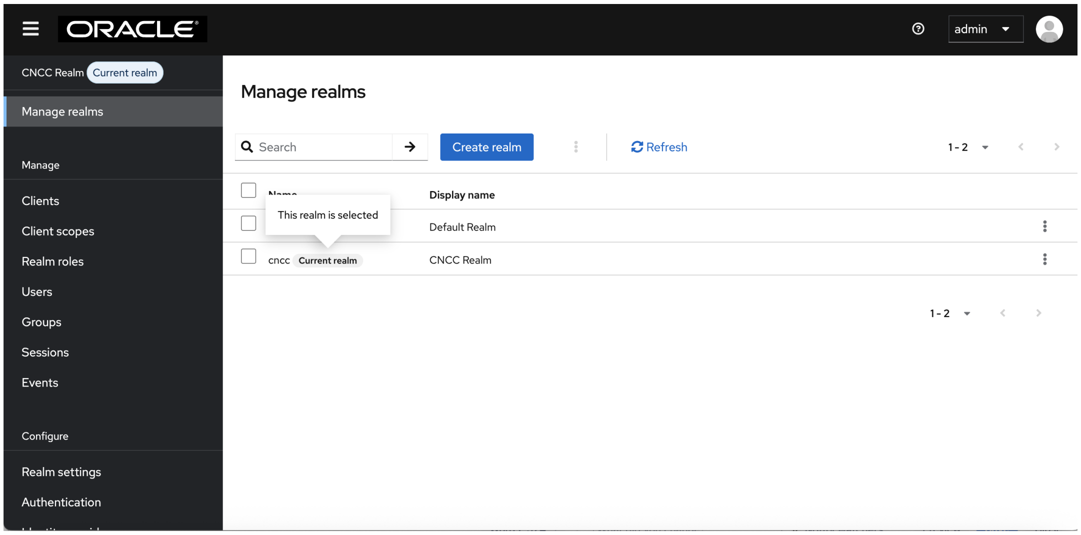
4.1.4 Creating or Updating CNC Console Core User Password in CNC Console IAM

This section describes how to create or update the user password in CNC Console.

Perform the following steps to create or update the user password:

1. Login to CNC Console, go to **Manage realms**, and select the **cncc** realm from the list of available realms.

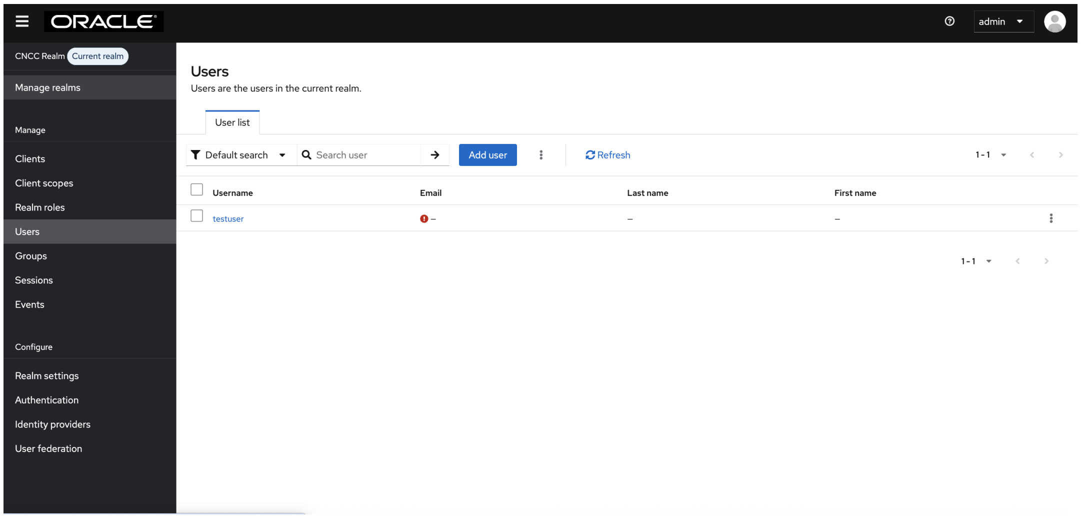
Figure 4-9 Realm Settings



Name	Display name
	Default Realm
cncc Current realm	CNCC Realm

2. Click **Users** on the left pane to view all users. Click the **Username** of the user to update the credentials.

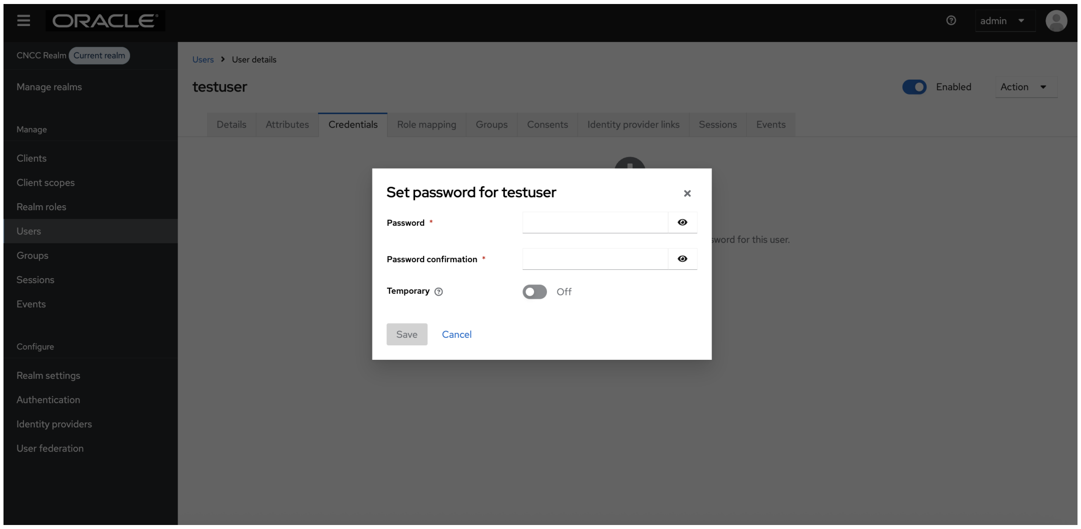
Figure 4-10 Users



Username	Email	Last name	First name
testuser	–	–	–

3. Under the **Credentials** tab, click **Set Password**.
4. Enter the new password in the **Password** field and enter the new password again in the **New Password Confirmation** field.
5. Set **Temporary** to **Off** and click **Save**.

Figure 4-11 Credentials



Set password for testuser

Password *

Password confirmation *

Temporary: Off

4.1.5 Password Policies for CNC Console Users

The following password policies are enabled by default for all CNCC Console users.

These password policies are disabled for CNC Console IAM users by default but can be enabled by setting the flag `global.enableDefaultAdminPasswordPolicy` to **true** in the `occncc_custom_values_<version>.yaml` file.

Table 4-2 Password Policies for CNC Console Users

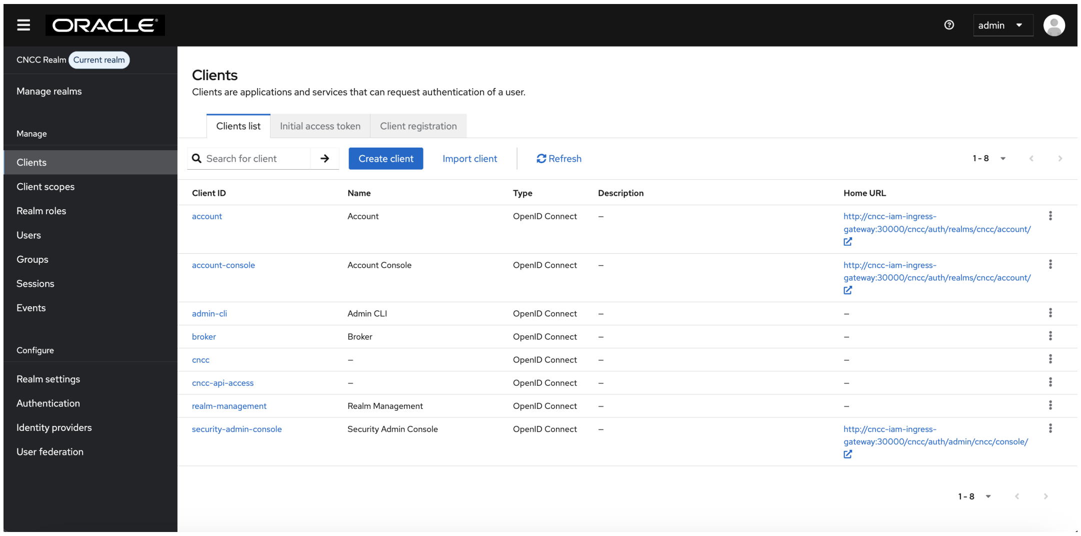
Policy	Description	Value
Expire Password	The number of days the password is valid before a new password is required.	30
Special Characters	The minimum number of special characters required in the password string.	1
Uppercase Characters	The minimum number of uppercase characters required in the password string.	1
Lowercase Characters	The minimum number of lowercase characters required in the password string.	1
Digits	The minimum number of numerical digits required in the password string.	1
Not Recently Used	Prevents a recently used password from being reused.	5
Not Username	The password cannot match the username.	ON

4.2 Configuring the CNC Console Redirection URL

After successfully deploying CNC Console IAM, the administrator must perform the following steps to configure the CNC Console redirection URL:

1. Log in to CNC Console IAM with your login credentials. Go to **Manage realms** and select the **cncc** realm from the list of available realms.
2. On the left pane, select **Clients** and on the right pane select the **cncc** Client ID.

Figure 4-12 Clients Screen



Clients

Clients are applications and services that can request authentication of a user.

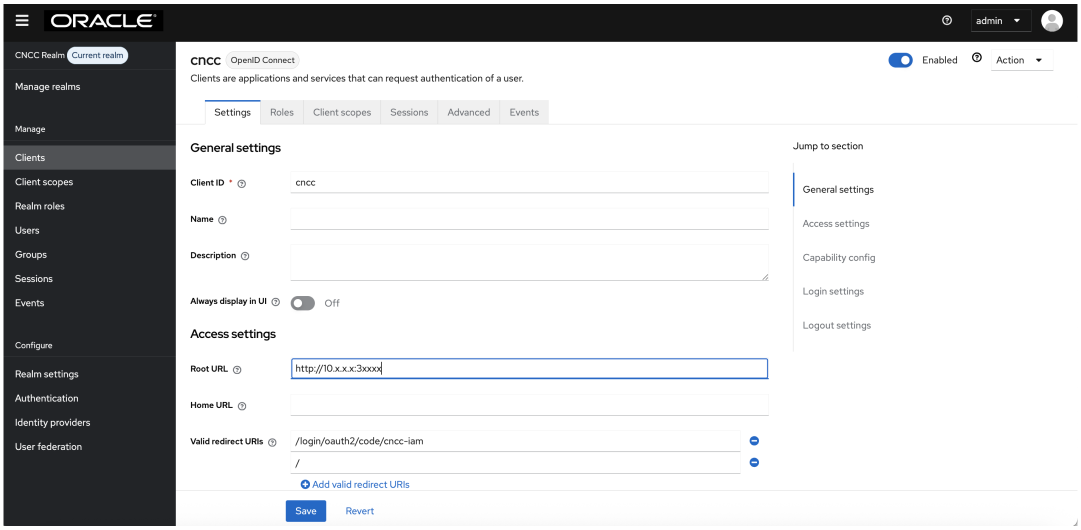
Client ID	Name	Type	Description	Home URL
account	Account	OpenID Connect	–	http://cncc-iam-ingress-gateway:30000/cncc/auth/realms/cncc/account/
account-console	Account Console	OpenID Connect	–	http://cncc-iam-ingress-gateway:30000/cncc/auth/realms/cncc/account/
admin-cli	Admin CLI	OpenID Connect	–	–
broker	Broker	OpenID Connect	–	–
cncc	–	OpenID Connect	–	–
cncc-api-access	–	OpenID Connect	–	–
realm-management	Realm Management	OpenID Connect	–	–
security-admin-console	Security Admin Console	OpenID Connect	–	http://cncc-iam-ingress-gateway:30000/cncc/auth/admin/cncc/console/

3. Enter CNC Console Core Ingress URL in the **Root URL** field and click **Save.**

```
<scheme>://<cncc-mcore-ingress IP/FQDN>:<cncc-core-ingress Port>
```

> **Note**
>
> **Valid Redirect URIs** are populated automatically; only the root URL needs to be configured as part of the post-installation procedure.

Figure 4-13 Redirection URL



Clients are applications and services that can request authentication of a user.

General settings

Client ID *: cncc

Name:

Description:

Always display in UI: Off

Access settings

Root URL: http://10.x.x.x:3xxxx

Home URL:

Valid redirect URIs: /login/oauth2/code/cncc-iam
/

4.3 Users in CNC Console IAM

Users can be created in both the default (master) and CNC Console realms.

A user created in the default realm will have administrative privileges, enabling them to log in to CNC Console IAM and perform various tasks related to user management, authentication, authorization, and system configuration.

A user created in the CNC Console realm can log in to the CNC Console Core GUI and access Network Functions (NF) and Common Services, depending on the roles assigned to them.

This section includes:

- Creating the users
- Viewing the Users
- Assigning the roles to the users

> **Note**
>
> For the details on setting or updating the admin password, see Creating or Updating Admin User Password in CNC Console IAM.

> **Note**
>
> For the details about setting or updating the user password, see Creating or Updating CNC Console Core User Password in CNC Console IAM.

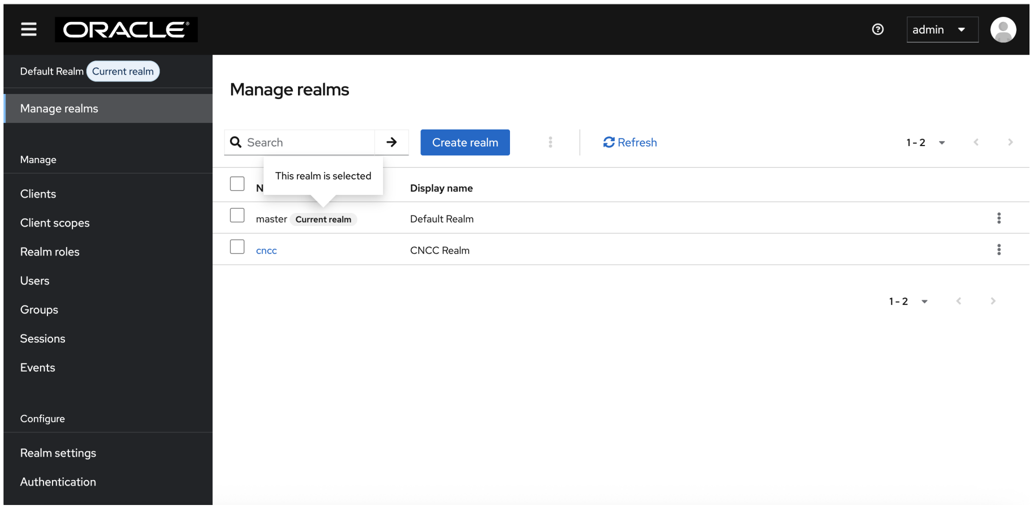
> **Note**
>
> In CNC Console IAM, the default realm refers to **master** realm. Users created in default (**master**) realm refers to CNC Console IAM admin users and users created in **cncc** realm refers to CNC Console users.

4.3.1 Creating the Users

Perform the following procedure to create users:

1. Log in to CNC Console IAM and select the appropriate realm based on where you want to create users. To switch realms, go to **Manage realms** and select the desired realm from the list of available realms.
   a. To create users in the default realm, select the default realm. The following screen appears:

Figure 4-14 default realm

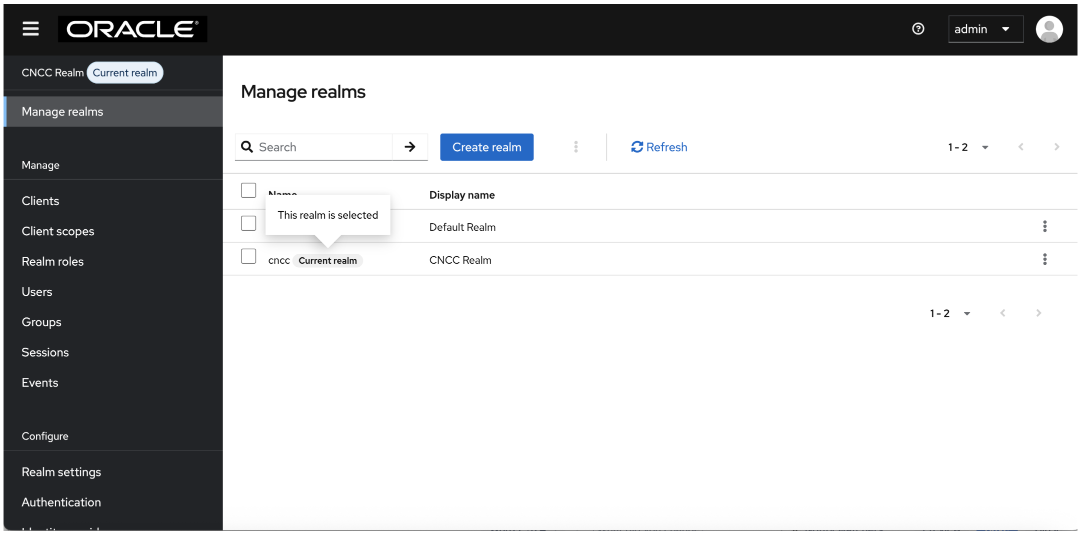


Manage realms

N	Display name
master Current realm	Default Realm
cncc	CNCC Realm

   b. To create users in the cncc realm, select the **cncc** realm. The following screen appears:

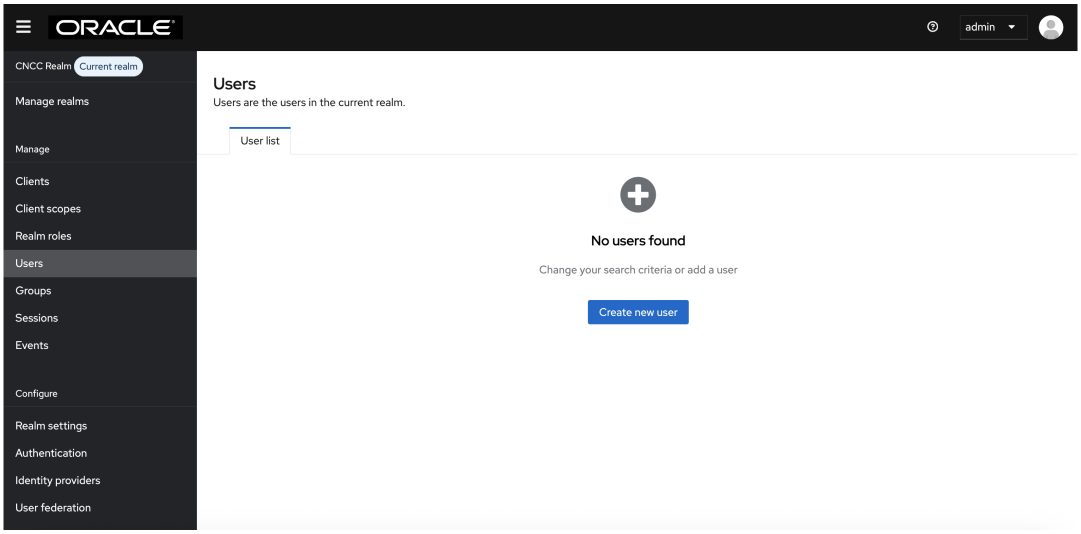
Figure 4-15   cncc realm



In the example below, the **cncc** realm is selected to create CNC Console users, as users have access to CNC Console. Follow similar steps as outlined below in the default realm to create CNC Console IAM admin users.

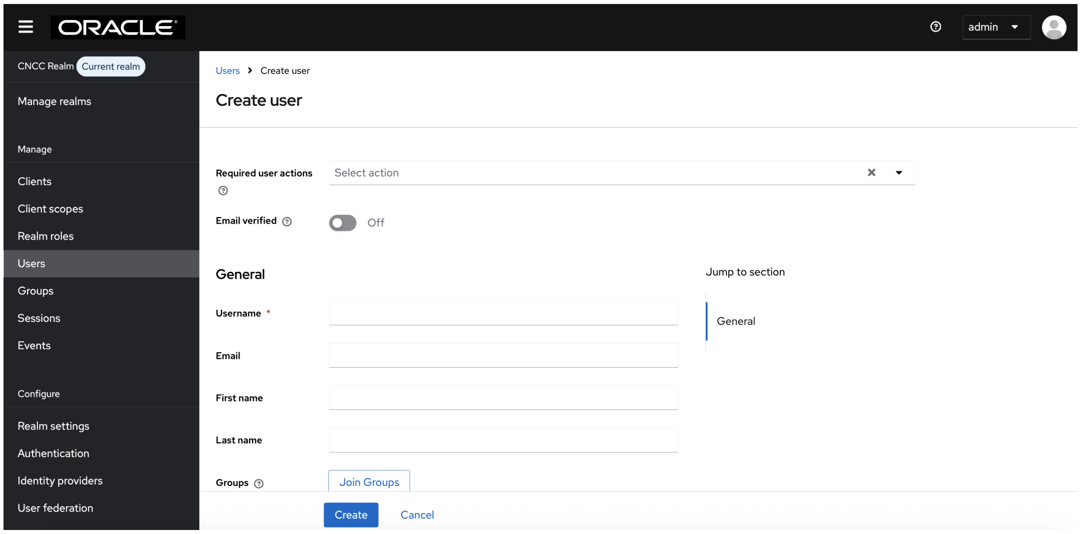
2. Click **Users** on the left navigation bar and click **Create new user**.

Figure 4-16   Add User



3. The **Create user** screen appears. Add the user details and click **Create**.

Figure 4-17 Create user



Create user

Required user actions: Select action

Email verified: Off

General

Username *:

Email:

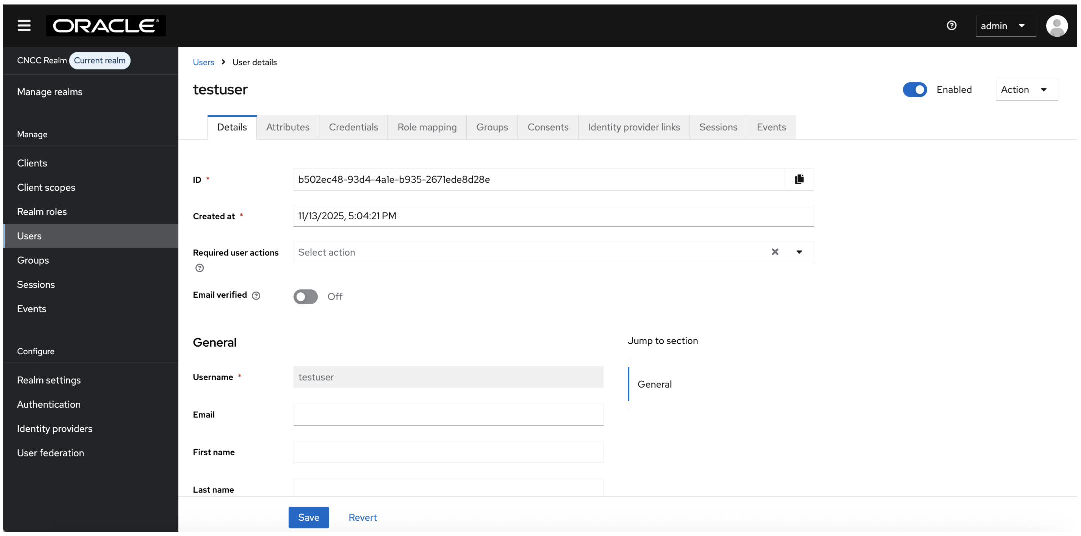
First name:

Last name:

Groups:

4. The user has been created and the user details screen appears.

Figure 4-18 New user details



testuser

Enabled

ID *: b502ec48-93d4-4a1e-b935-2671ede8d28e

Created at *: 11/13/2025, 5:04:21 PM

Required user actions: Select action

Email verified: Off

General

Username *: testuser

Email:

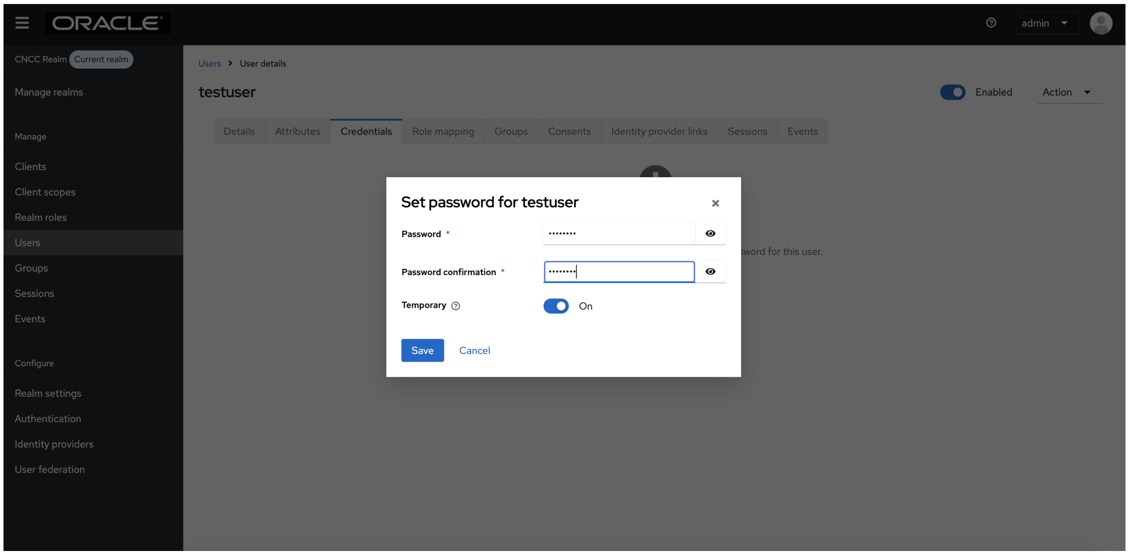
First name:

Last name:

5. Go to the **Credentials** tab and click **Set Password** to set the password for that user.
6. Set **Temporary** to **On** to prompt the user to change their password when they log in for the first time to the CNC Console UI.

> **Note**
>
> It is recommended to enable the **Temporary** flag for security.

Figure 4-19 Set Password



Set password for testuser

Password * ••••••••

Password confirmation * ••••••••

Temporary On

Save Cancel

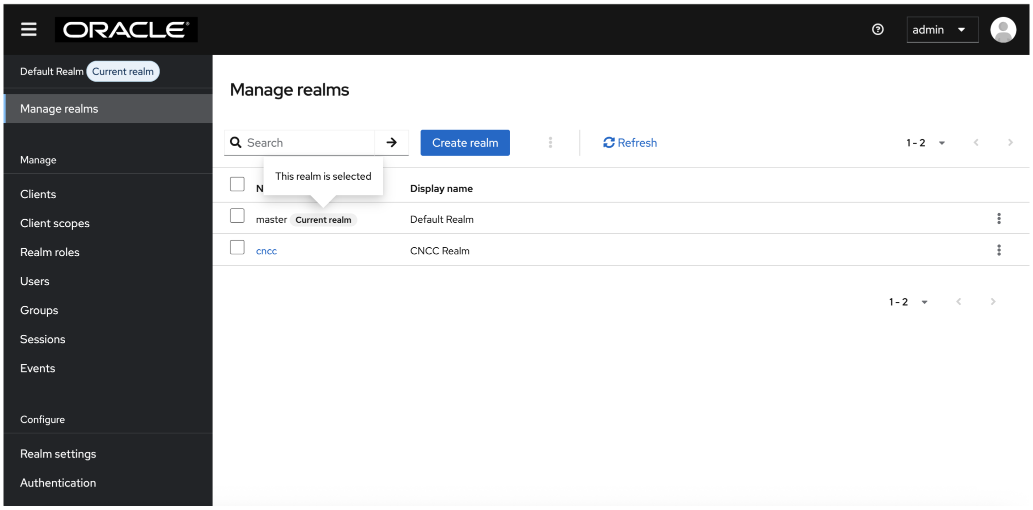
> **Note**
>
> Setting the **Temporary** flag **ON** prompts the user change the password when logging in to the CNC Console for the first time.

4.3.2 Viewing the Users

Perform the following procedure to view users:

1. Log in to CNC Console IAM and select the appropriate realm based on how users want to view. To switch realms, go to **Manage Realm** and select the desired realm from the list of available realms.
   a. To view the users with administrative privileges, select the default realm.

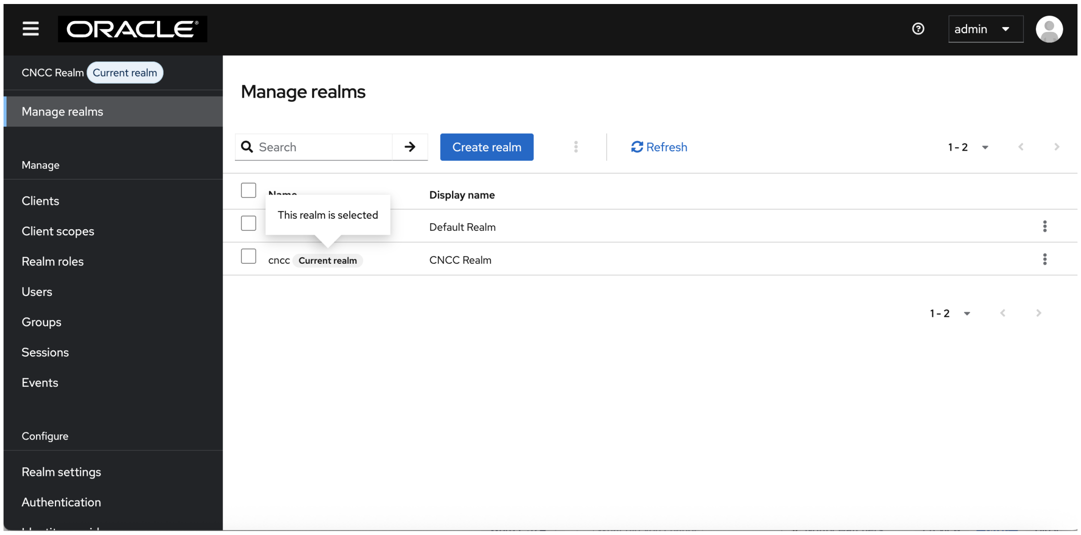
Figure 4-20 default realm



Name	Display name
master Current realm	Default Realm
cncc	CNCC Realm

b. To view users with access to CNCC, select the **cncc** realm. The following screen appears:

Figure 4-21 CNCC Realm

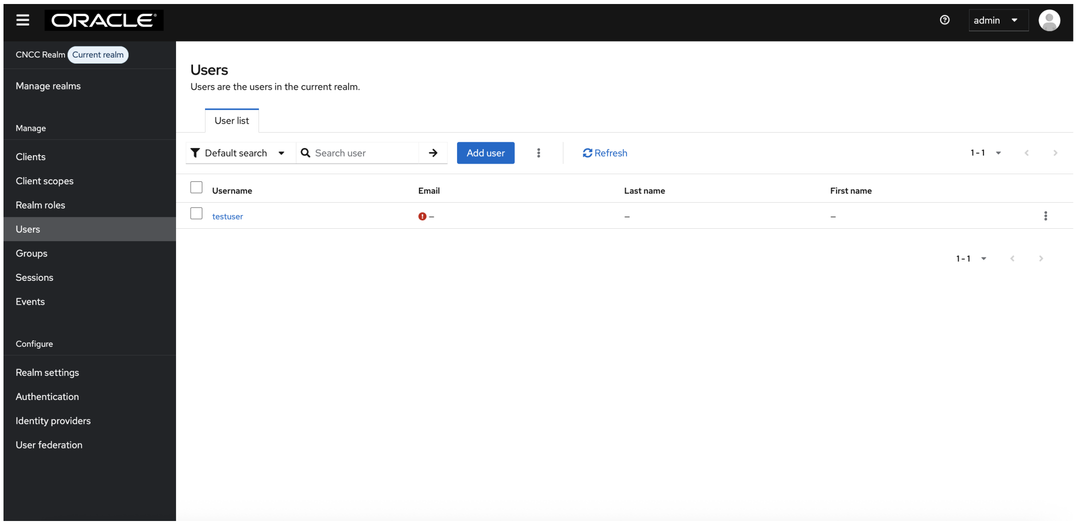


Name	Display name
	Default Realm
cncc Current realm	CNCC Realm

In the example below, the **cncc** realm is selected to view users with access to **CNCC**. Follow similar steps as outlined below in the default realm to view CNCC IAM admin users.

2. Select **Users** on the left-side navigation bar. A list of users will be displayed.

Figure 4-22 View All Users



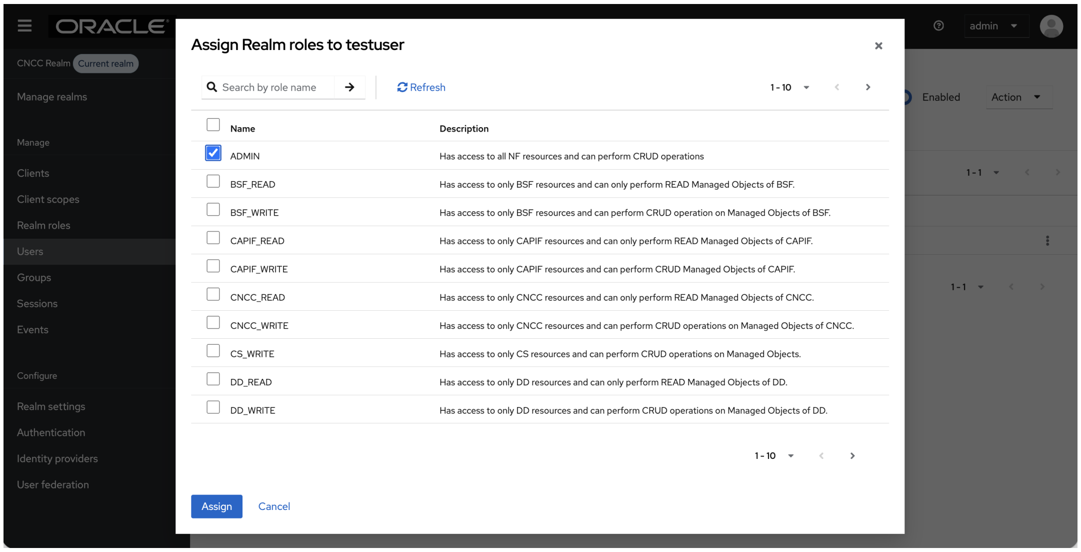
Username	Email	Last name	First name
testuser	–	–	–

4.3.3 Assigning Roles to the User

Perform the following procedure to assign roles to the user:

1. Select a user to assign roles. Navigate to the **Role Mappings** tab, click **Assign Role**, and then choose **Realm Roles** from the drop-down menu.

Figure 4-23 Assign Realm Roles to User



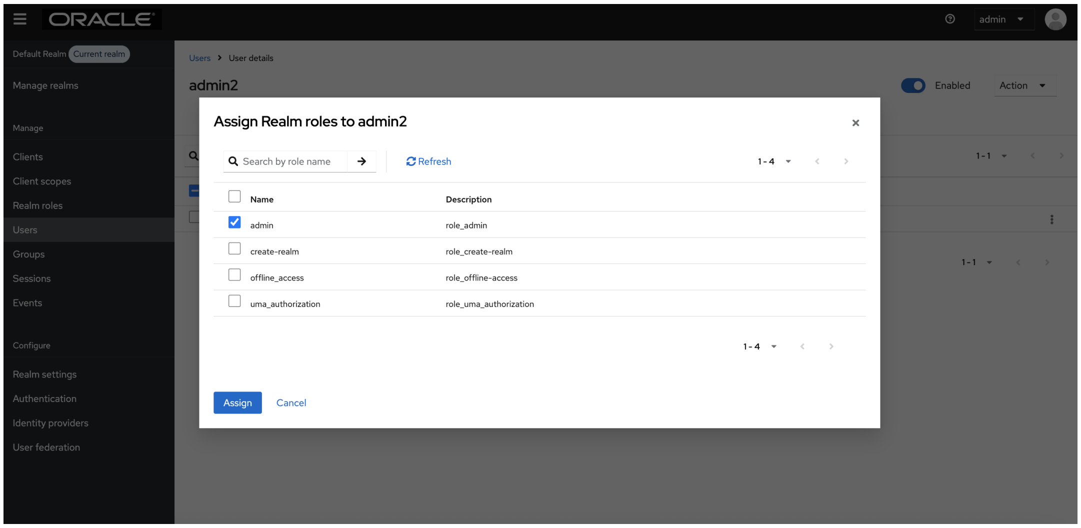
Assign Realm roles to testuser

	Name	Description
☑	ADMIN	Has access to all NF resources and can perform CRUD operations
□	BSF_READ	Has access to only BSF resources and can only perform READ Managed Objects of BSF.
□	BSF_WRITE	Has access to only BSF resources and can perform CRUD operation on Managed Objects of BSF.
□	CAPIF_READ	Has access to only CAPIF resources and can only perform READ Managed Objects of CAPIF.
□	CAPIF_WRITE	Has access to only CAPIF resources and can perform CRUD Managed Objects of CAPIF.
□	CNCC_READ	Has access to only CNCC resources and can only perform READ Managed Objects of CNCC.
□	CNCC_WRITE	Has access to only CNCC resources and can perform CRUD operations on Managed Objects of CNCC.
□	CS_WRITE	Has access to only CS resources and can perform CRUD operations on Managed Objects.
□	DD_READ	Has access to only DD resources and can only perform READ Managed Objects of DD.
□	DD_WRITE	Has access to only DD resources and can perform CRUD operations on Managed Objects of DD.

> **Note**
>
> You must change the number of entries displayed per page from the pagination drop-down to **100 per page** to view all entries.

2. Select the checkbox for the roles you wish to assign to the user and click **Assign** at the bottom to save the changes.
3. For users created in the default realm, ensure that the admin role is assigned to the newly created user from the list of available roles.

Figure 4-24 Assign Realm Roles to Admin



Assign Realm roles to admin2

	Name	Description
☑	admin	role_admin
□	create-realm	role_create-realm
□	offline_access	role_offline-access
□	uma_authorization	role_uma_authorization

4.4 CNC Console SAML SSO Integration

4.4.1 Integrating SAML SSO with CNC Console IAM

Overview

Security Assertion Markup Language (SAML) is an open standard that allows identity providers (IdP) to pass authorization credentials to service providers (SP). The identity provider authenticates the user and returns the assertion information about the authenticated user and the authentication event to the application. Using SSO, if the user tries to access any other application that uses the same identity provider for user authentication, the user does not need to log in again. This is the principle of SSO (Single Sign-on).

> **Note**
>
> To enable SAML identity provider authentication for user login, ensure that the CNC Console is deployed using the secure HTTPS protocol.

> **Note**
>
> CNC Console supports SAML 2.0.

CNC Console SAML SSO Flow Overview

This section provides an overview of how a user can configure the SAML SSO flow in CNC Console IAM.

The following are the high-level steps; the detailed procedure is explained in the section "Configuring SAML Identity Provider in CNC Console IAM.".

1. Configure the External SAML IdP in CNC Console IAM
2. Configuration required in External SAML IdP
   a. Make sure the required user details are present in the External SAML IdP. In your external IdP, create user accounts that will access the CNC Console.

      > **Note**
      >
      > Do not create the user in CNC Console IAM; we suggest to create user in external IDP only for SAML SSO flow

   b. Define Roles in the External SAML IdP and Assign Them to Users. Within the external IdP, create roles or groups and assign them to the appropriate users.

      > **Note**
      >
      > No role assignment is required in CNC Console IAM; roles should only be assigned in external IDP for SAML SSO Flow

3. Map SAML IdP Roles to CNC Console IAM Roles.
4. Log in to CNC Console using Single Sign On (SSO) Option.

> **Note**
>
> Log in to CNC Console only by clicking the Single Sign-On (SSO) option displayed on the login screen. This will redirect to the external IdP. Do not use CNC Console user credentials configured in CNC Console IAM to log in.

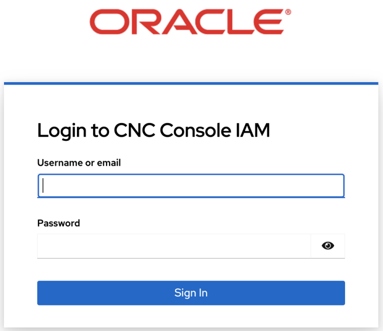
Configuring SAML Identity Provider in CNC Console IAM

Perform the following procedure to configure SAML identity provider

1. Log in to CNC Console IAM Console using admin credentials provided during CNC Console IAM installation.

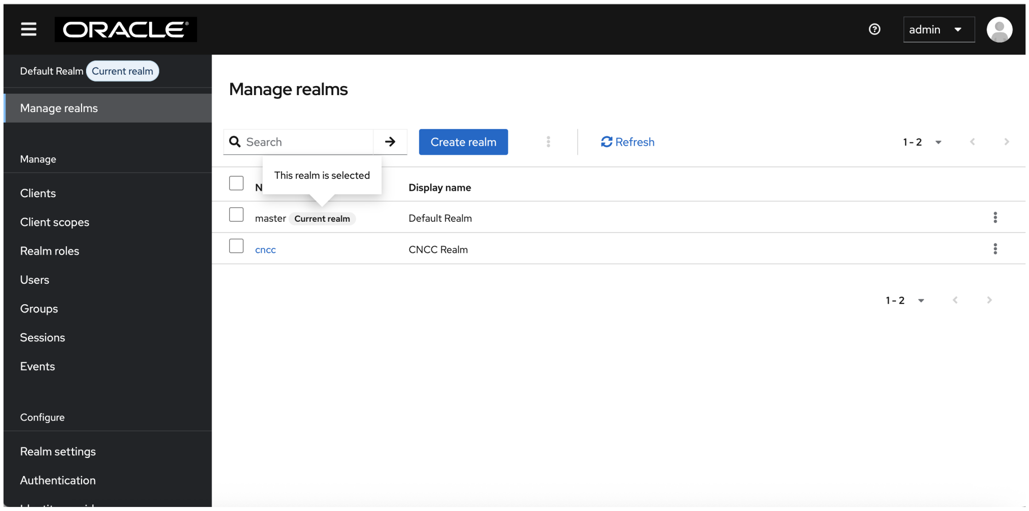
   ```
   http://<cncc-iam-ingress-extrenal-ip>:<cncc-iam-ingress-service-port>
   Example: http://cncc-iam-ingress-gateway.cncc.svc.cluster.local:30085/
   ```

   Figure 4-25 Login screen

2. Select the appropriate realm based on where you want to enable authentication through an identity provider. To switch realms, go to **Manage Realm** and select the desired realm from the list of available realms:
   a. To enable authentication for CNC Console IAM admin users, choose the default realm.

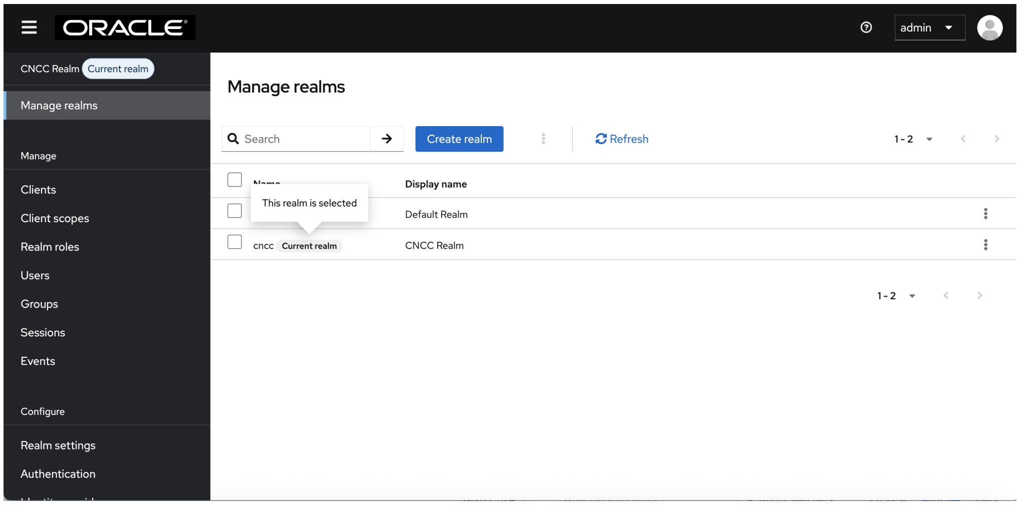
Figure 4-26 default realm



N	Display name
master Current realm	Default Realm
cncc	CNCC Realm

b. To enable authentication for CNC Console users, choose the **cncc** realm.

Figure 4-27 cncc realm



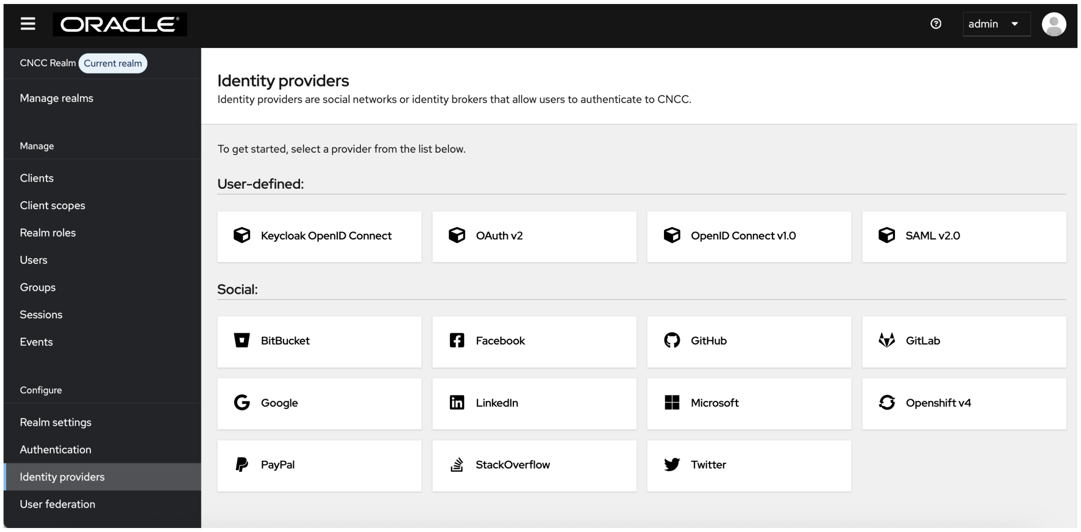
Name	Display name
	Default Realm
cncc Current realm	CNCC Realm

In the following example, the **cncc Realm** is selected to enable authentication through an identity provider to CNC Console.

Follow similar steps as outlined below in the default Realm to enable authentication through an identity provider to CNC Console IAM.

3. Select the **cncc** realm, and click **Identity providers** on the left navigation bar to open the **Identity providers** screen appears.

Figure 4-28 Identity Provider Screen



Identity providers

Identity providers are social networks or identity brokers that allow users to authenticate to CNCC.

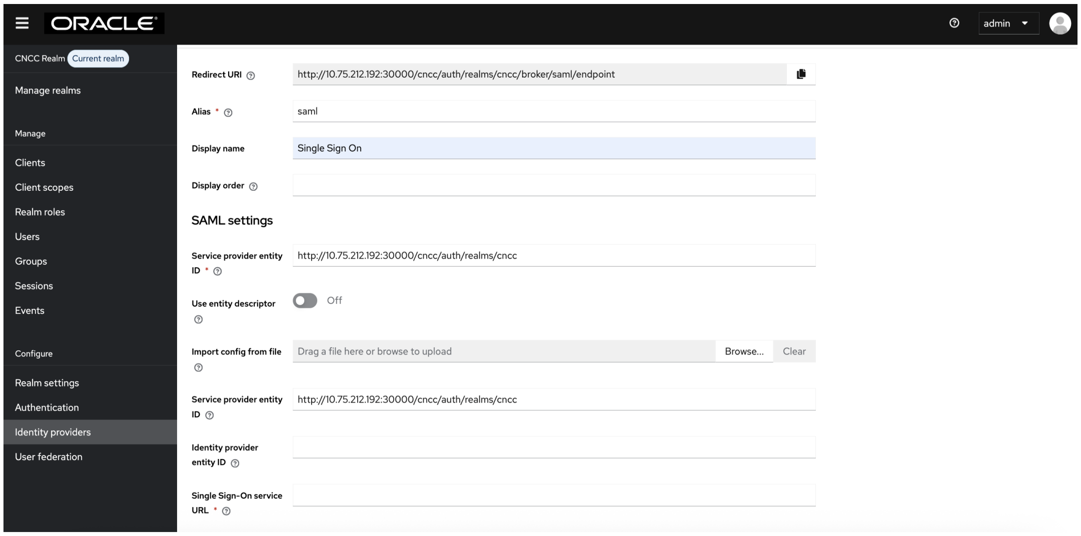
To get started, select a provider from the list below.

User-defined:

Social:

4. Select **SAML v2.0** from the available options to open the **Add SAML Provider** form and configure your SAML IdP parameters.

Figure 4-29 SAML Settings



Redirect URI: http://10.75.212.192:30000/cncc/auth/realms/cncc/broker/saml/endpoint

Alias *: saml

Display name: Single Sign On

Display order:

SAML settings

Service provider entity ID *: http://10.75.212.192:30000/cncc/auth/realms/cncc

Use entity descriptor: Off

Import config from file: Drag a file here or browse to upload

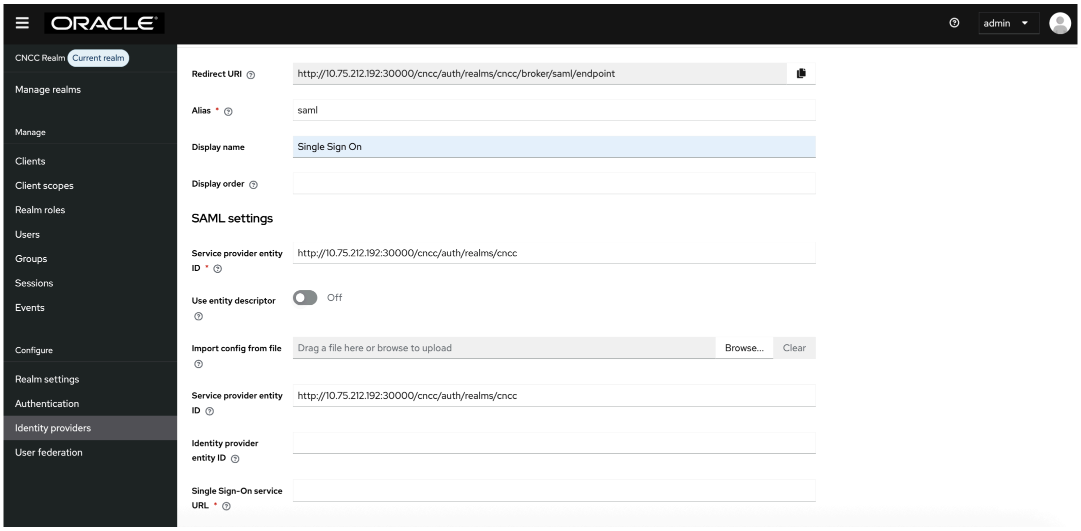
Service provider entity ID: http://10.75.212.192:30000/cncc/auth/realms/cncc

Identity provider entity ID:

Single Sign-On service URL *:

5. Give an appropriate name for the **Display Name** field.
6. To import the metadata file exported from SAML client in the IdP, disable the **Use Entity descriptor** flag, and click **Browse** in the **Import from config file** field to upload the file.

Figure 4-30   Import from Config File



Redirect URI: http://10.75.212.192:30000/cncc/auth/realms/cncc/broker/saml/endpoint

Alias *: saml

Display name: Single Sign On

Display order:

SAML settings

Service provider entity ID *: http://10.75.212.192:30000/cncc/auth/realms/cncc

Use entity descriptor: Off

Import config from file: Drag a file here or browse to upload

Service provider entity ID: http://10.75.212.192:30000/cncc/auth/realms/cncc

Identity provider entity ID:

Single Sign-On service URL *:

Click **Import** and **Save.** The other required fields populate automatically.

Perform the following procedure to configure the IdP manually, if you are facing difficulty in importing the metadata file from the IdP Client:

a. Navigate to the **Identity providers** screen and click **SAML v2.0**.

b. Set the value of **Single Sign-On Service URL** to the URL of the preferred IdP. Example: `<IP/FQDN>:<PORT>/auth/realms/master/protocol/saml` (URI for their preferred IdP where SAML AuthnRequest will be sent).

c. Set the value of **Single Logout Service URL**. Example: `<IP/FQDN>:<PORT>/auth/realms/master/protocol/saml` (URI for their preferred IdP where logout requests must be sent).

d. If the IdP supports *HTTP POST*binding methods, enable **HTTP-POST Binding Response, HTTP-POST Binding Logout** and **HTTP-POST Binding for AuthnRequest** flags. By default, HTTP-Redirect will be used.

e. If the IdP is sending signed Assertions, set **Want Assertions Signed** to **ON**.

f. Set **Validate Signature** to **ON.**

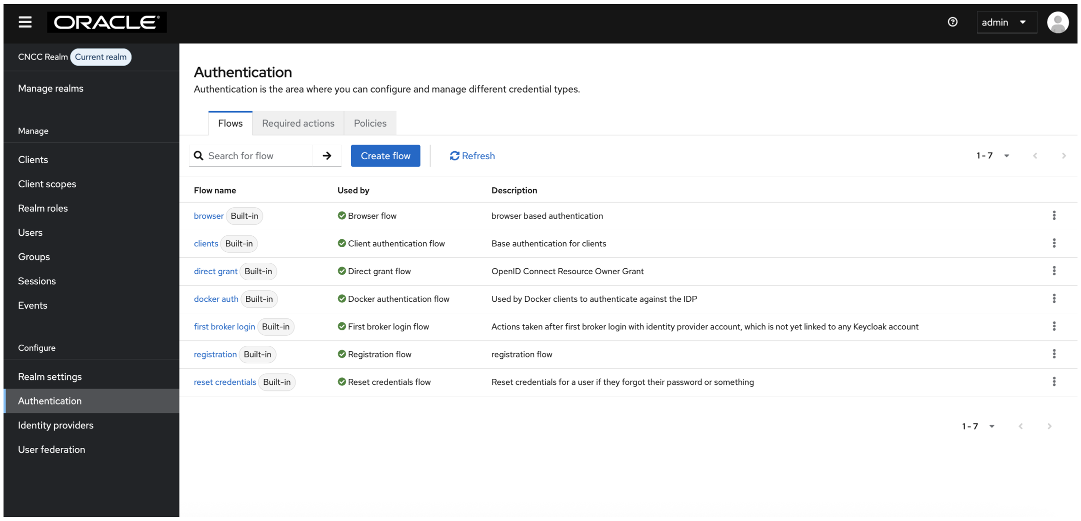
g. Provide value for **Validating X509 Certificates** (If you are using keycloak as an IdP, use the certificate from **master** realm-> Realm Settings-> Keys).

h. Click **Add**.

IdP is now configured manually.

7. To create custom **First Login Flow**, click **Authentication** tab on the left pane. The **Authentication** screen appears.

Figure 4-31 Import from



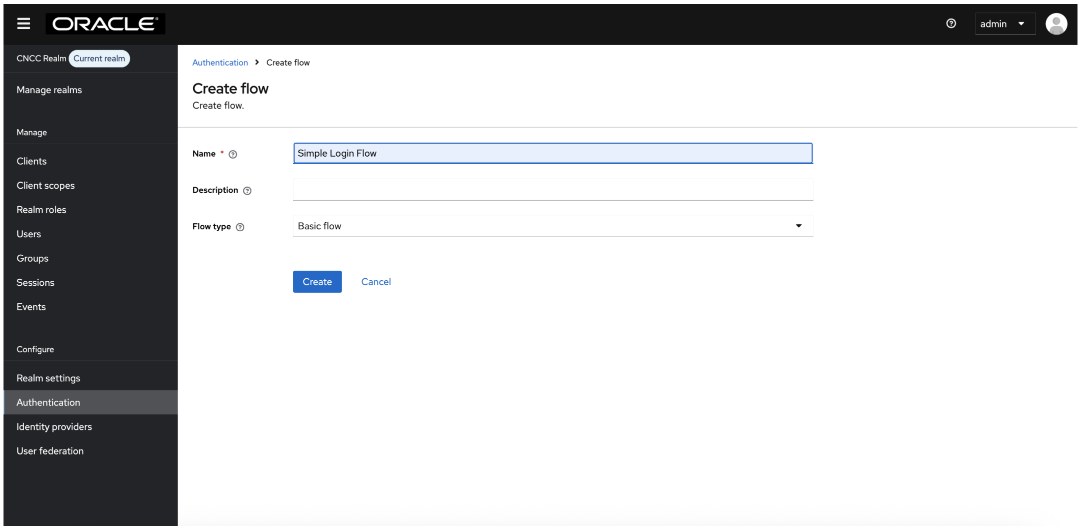
Authentication

Authentication is the area where you can configure and manage different credential types.

Flow name	Used by	Description
browser Built-in	✓ Browser flow	browser based authentication
clients Built-in	✓ Client authentication flow	Base authentication for clients
direct grant Built-in	✓ Direct grant flow	OpenID Connect Resource Owner Grant
docker auth Built-in	✓ Docker authentication flow	Used by Docker clients to authenticate against the IDP
first broker login Built-in	✓ First broker login flow	Actions taken after first broker login with identity provider account, which is not yet linked to any Keycloak account
registration Built-in	✓ Registration flow	registration flow
reset credentials Built-in	✓ Reset credentials flow	Reset credentials for a user if they forgot their password or something

8. Click **Create Flow** on the right pane. The **Create Flow** screen appears.

Figure 4-32 Create Flow



Create flow

Create flow.

Name * : Simple Login Flow

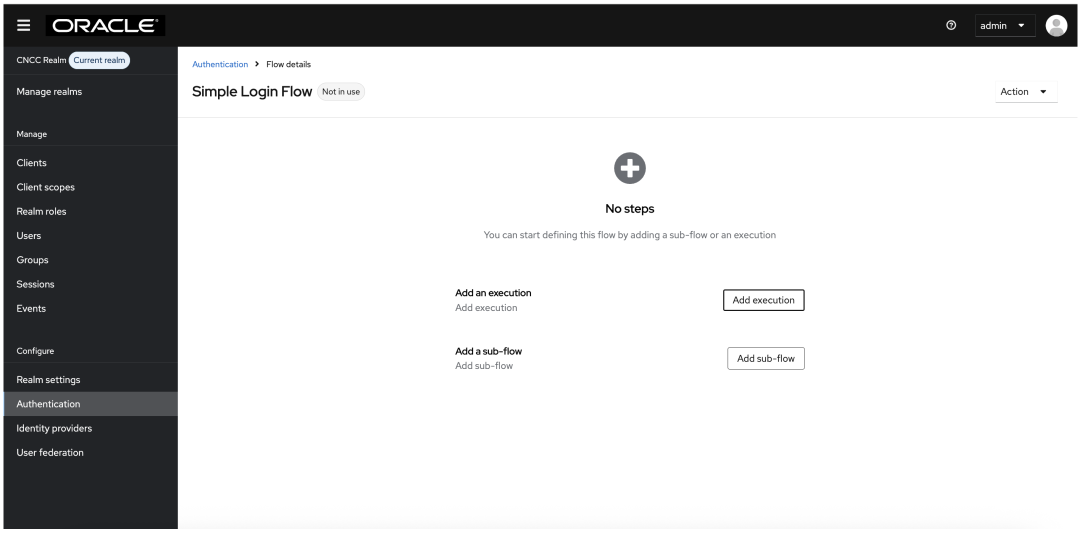
Description :

Flow type : Basic flow

Enter the appropriate name and click **Create**.

9. The **Simple Login Flow** screen appears. Click **Add execution** on the right pane.

Figure 4-33 Simple Login Flow



Simple Login Flow Not in use

No steps

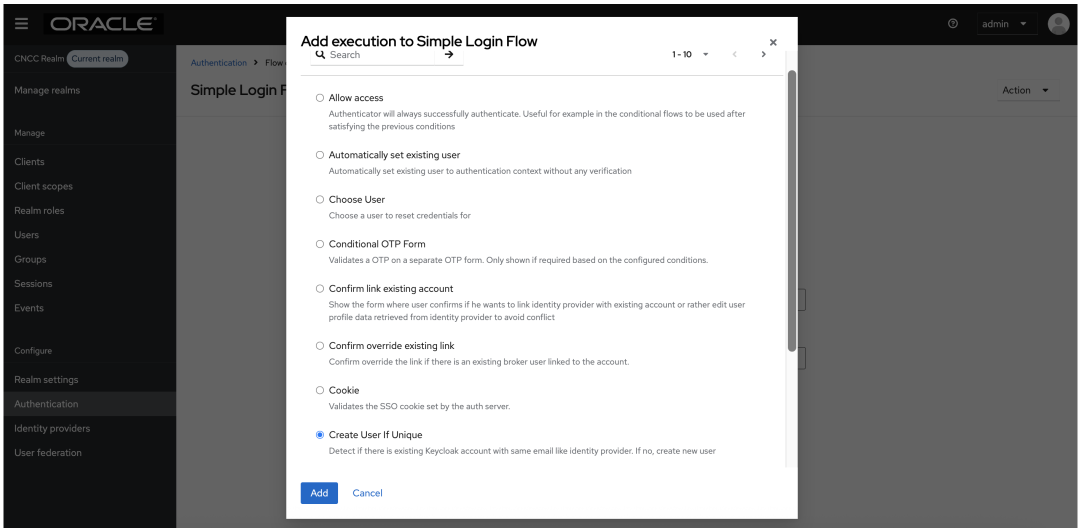
You can start defining this flow by adding a sub-flow or an execution

Add an execution
Add execution

Add a sub-flow
Add sub-flow

10. Select **Create User If Unique**, and click **Add**.

Figure 4-34 Add Execution to Simple Login Flow



Add execution to Simple Login Flow

○ Allow access
Authenticator will always successfully authenticate. Useful for example in the conditional flows to be used after satisfying the previous conditions

○ Automatically set existing user
Automatically set existing user to authentication context without any verification

○ Choose User
Choose a user to reset credentials for

○ Conditional OTP Form
Validates a OTP on a separate OTP form. Only shown if required based on the configured conditions.

○ Confirm link existing account
Show the form where user confirms if he wants to link identity provider with existing account or rather edit user profile data retrieved from identity provider to avoid conflict

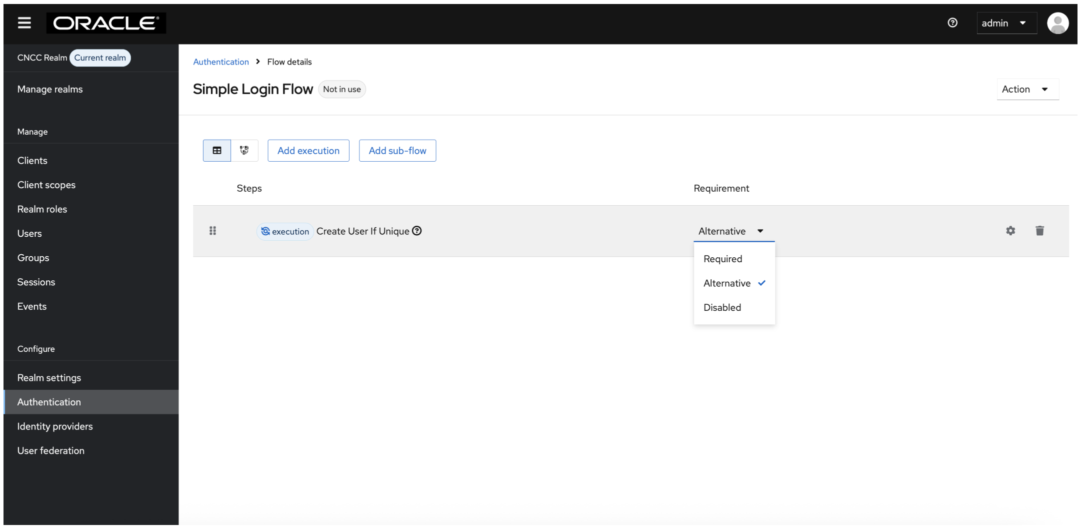
○ Confirm override existing link
Confirm override the link if there is an existing broker user linked to the account.

○ Cookie
Validates the SSO cookie set by the auth server.

● Create User If Unique
Detect if there is existing Keycloak account with same email like identity provider. If no, create new user

11. You will be redirected to **Authentication** page. From **Requirement** section, select **Alternative**.

Figure 4-35 Authentication

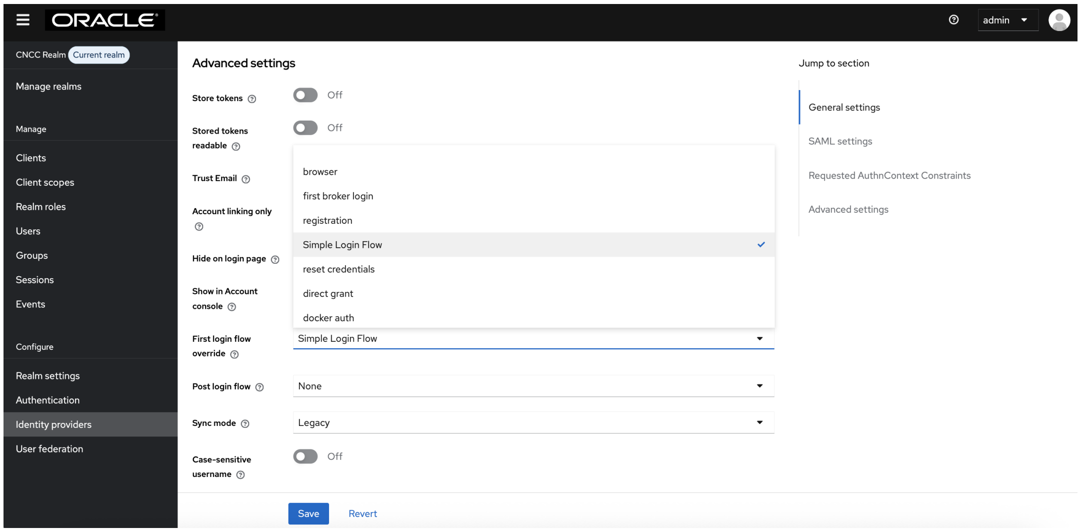


Simple Login Flow Not in use

Steps	Requirement
execution Create User If Unique	Alternative

12. Navigate to the **Identity Provider** settings and select your custom flow from the **First Login Flow Override** dropdown. The completed form appears:

Figure 4-36 Advanced Settings Page



Advanced settings

Store tokens	Off
Stored tokens readable	Off
Trust Email	
Account linking only	
Hide on login page	
Show in Account console	
First login flow override	Simple Login Flow
Post login flow	None
Sync mode	Legacy
Case-sensitive username	Off

13. Click **Save**.
The above screen appears. Map the SAML IdP roles with CNC Console IAM API roles.

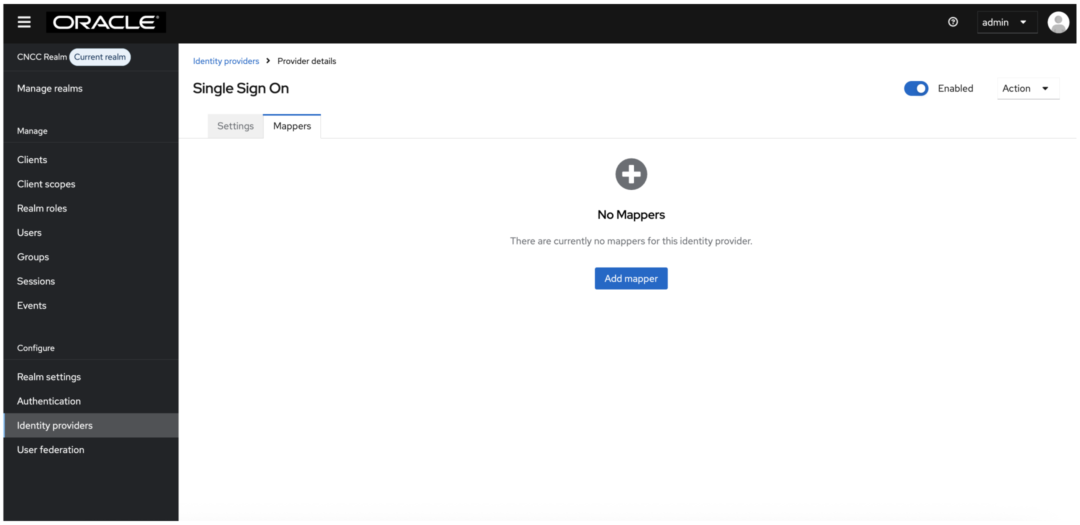
> **Note**
>
> **CNC Console IAM(SP) Configuration in IdP**
>
> In a SAML based SSO Implementation, the IdP needs to send SAML assertions towards a Service Provider (CNC Console IAM in this case) endpoint.
>
> Use the following CNC Console endpoint in the IdP:
>
> ```
> http://<IP/FQDN>:<PORT>/cncc/auth/realms/cncc/broker/saml/endpoint
> ```
>
> Example:
>
> ```
> http://cncc-iam-ingress-gateway.cncc.svc.cluster.local:30085/cncc/auth/
> realms/cncc/broker/saml/endpoint
> ```

Mapping SAML IdP roles with CNC Console IAM API roles

Perform the following procedure to map SAML IdP roles with CNC Console IAM API roles:

1. After saving SAML IdP configurations in CNC Console IAM, select **Identity providers** on the left pane and click the name of your identity provider. Click **Mappers** tab on the right pane. Click **Add Mapper.**

   Figure 4-37 Single Sign On

2. The **Add Identity Provider Mapper** screen appears.
   a. Give an appropriate name for the Identity Provider Mapper in the **Name** field.
   b. select **SAML Attribute to Role** from the **Mapper Type** drop-down.
   c. Enter the **Attribute Name.**

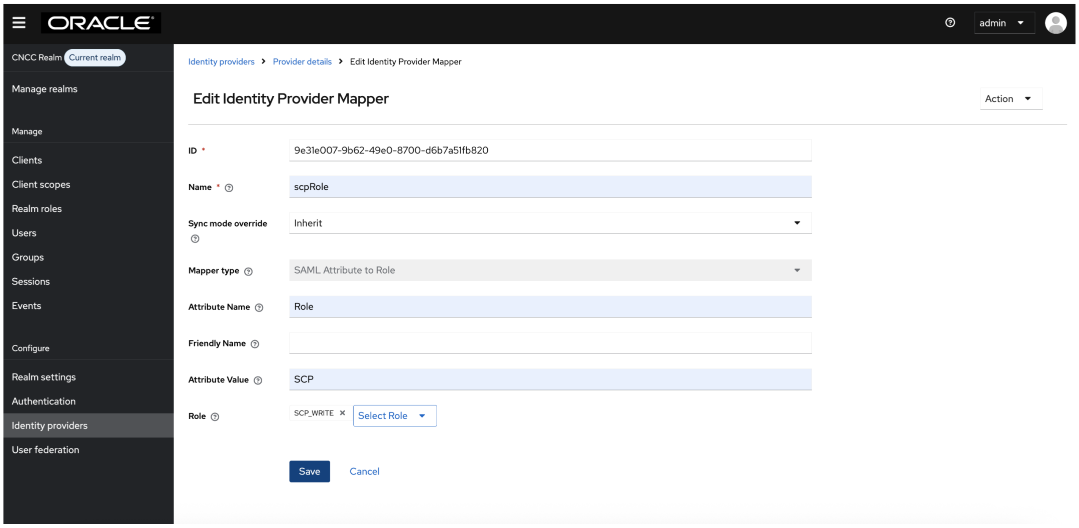
> **Note**
>
> The attribute value (role) may not be the same as given in example, it is specific to which SAML IdP customer is using. For the exact attribute value, refer to your SAML IdP documentation or support, or you can find it in the SAML response from your SAML IdP.

d. Enter the **Attribute Value** as the one of the roles added in SAML IdP. For example: 'NRF', 'SCP', etc.

e. Click **Select Role** to select the API roles to be enabled for this mapping.

f. Click **Assign**. Then click **Save**.

Figure 4-38 Example Values for scpRole Mapper for cncc realm



Edit Identity Provider Mapper

ID *: 9e31e007-9b62-49e0-8700-d6b7a51fb820

Name *: scpRole

Sync mode override: Inherit

Mapper type: SAML Attribute to Role

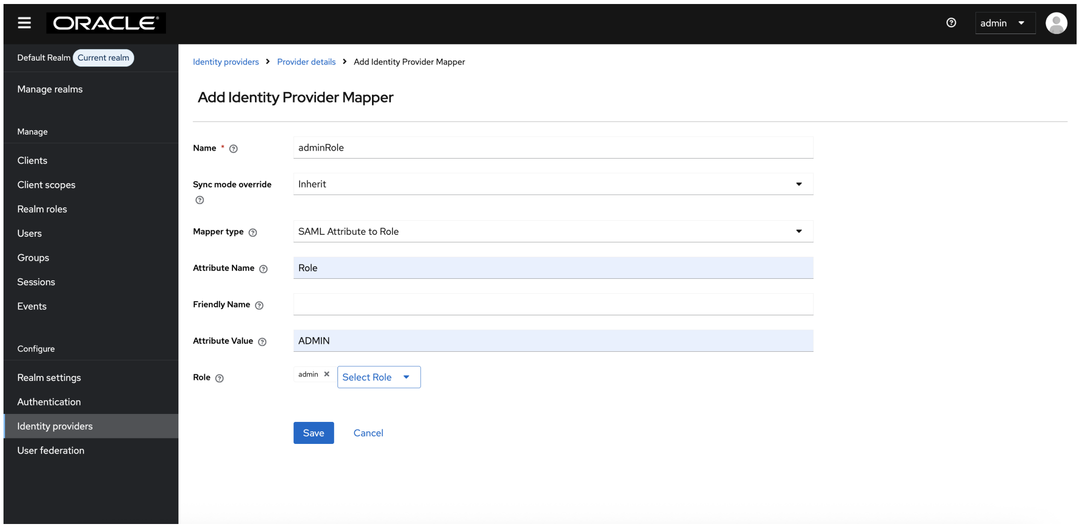
Attribute Name: Role

Friendly Name:

Attribute Value: SCP

Role: SCP_WRITE

Figure 4-39 Example Values for admin role mapper for default realm



Add Identity Provider Mapper

Name *: adminRole

Sync mode override: Inherit

Mapper type: SAML Attribute to Role

Attribute Name: Role

Friendly Name:

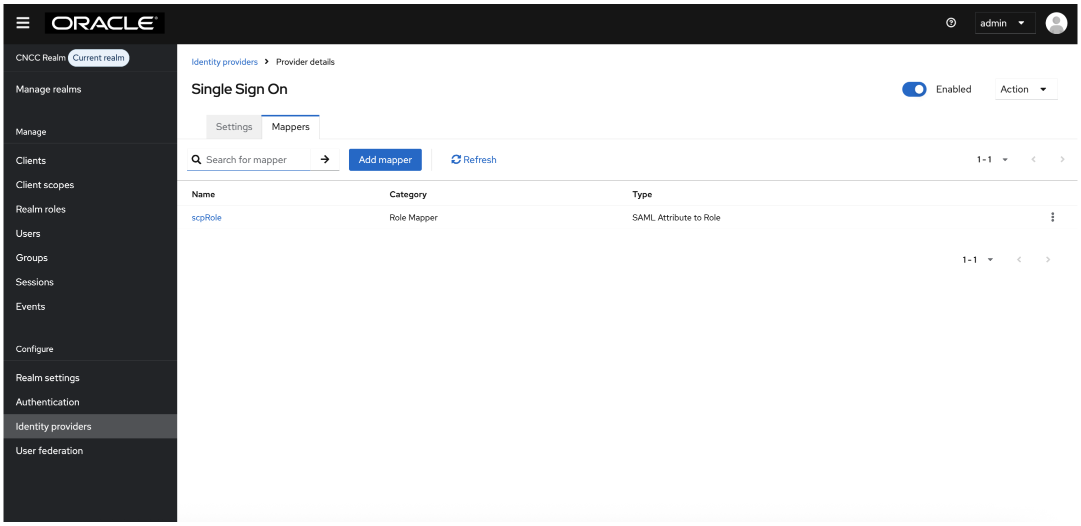
Attribute Value: ADMIN

Role: admin

3. Once the mappers are created, they appear in the list as shown following screen shot. If there are future requirements to map additional attributes, similar mappers can be created

by following the same steps. For user profile attribute mappings, see Additional Attribute Mappings (Optional User Profile Attribute Mapping).

Figure 4-40   Single Sign On



Name	Category	Type
scpRole	Role Mapper	SAML Attribute to Role

Additional Attribute Mappings (Optional User Profile Attribute Mapping)

In addition to role mappings, CNC Console IAM supports importing user profile attributes from the Identity Provider (IdP) during SAML authentication. This allows attributes such as **User Name**,**First Name**, **Last Name**, and **Email Address** to be automatically populated in CNC Console IAM based on SAML assertions provided by the IdP.

If your IdP includes these attributes, you can configure corresponding attributes in the CNC Console IAM SAML client to map them to the CNC Console user profile.

Sample SAML attribute statement (for illustration only):

> **Note**
>
> The attribute names listed below are examples. Your IdP may use different attribute names as per its own standards. The attribute name shown by the IdP is the one that needs to be selected during mapping.

```
<saml:Attribute Name="uid" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-
format:basic">
   <saml:AttributeValue xsi:type="xs:string">user_saml</saml:AttributeValue>
</saml:Attribute>

<saml:Attribute Name="FirstName"
NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:basic">
   <saml:AttributeValue xsi:type="xs:string">User</saml:AttributeValue>
</saml:Attribute>

<saml:Attribute Name="LastName"
NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:basic">
   <saml:AttributeValue xsi:type="xs:string">Saml</saml:AttributeValue>
</saml:Attribute>
```

```
<saml:Attribute FriendlyName="email" Name="urn:oid:1.2.840.113549.1.9.1"
NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:basic">
   <saml:AttributeValue xsi:type="xs:string”>user_saml@oracle.com</
saml:AttributeValue>
</saml:Attribute>
```

Perform the following procedure to configure additional attribute mapping in CNC Console IAM:

1. Log in to CNC Console IAM Administration Console.
2. Select cncc realm
3. Navigate to **Identity Providers** and select your SAML provider.
4. Go to the **Mappers** tab and click **Add mapper.**
5. Provide a name to the mapper.
6. Set the **Mapper Type** to **User Attribute Importer**.
7. Set Name Format to: ATTRIBUTE_FORMAT_BASIC (default)
8. In **Attribute Name**, enter the SAML attribute name as provided by the IdP (for example, `uid`, `FirstName`, `LastName`, `email`, or your IdP’s equivalent).
9. In **User Attribute**, select `username`, `firstName`, `lastName`, and `email` from the dropdown, or if these attributes are already available in CNC Console IAM and do not need to be created, set **Sync Mode** to **Inherit** (default)
10. Click **Save**.

Repeat these steps for each attribute (`username`, `firstName`, `lastName`, or `email`) that you want to map.

The following is a sample summary for attribute mapping:

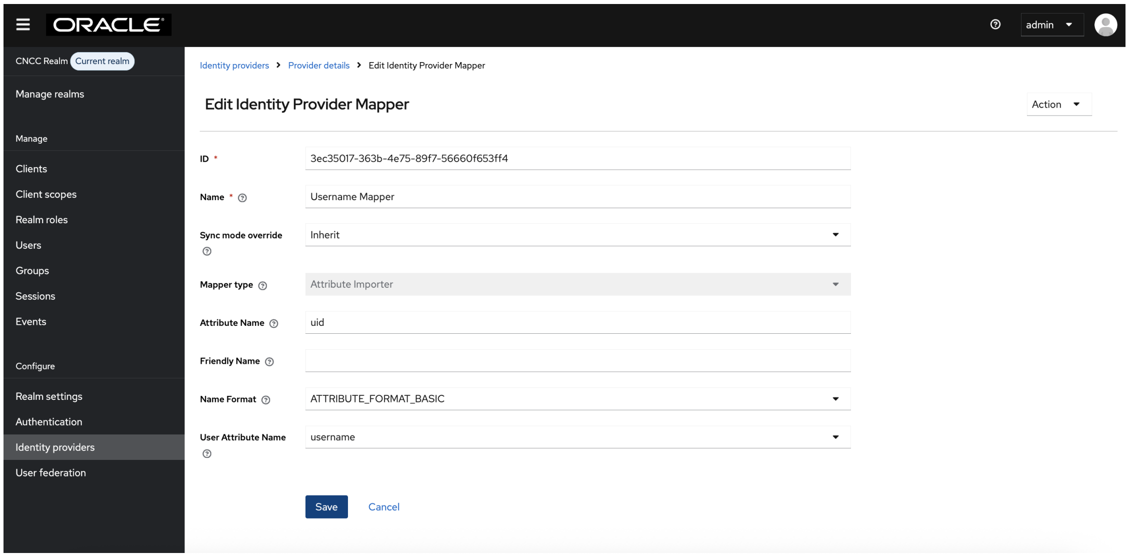
Table 4-3 Sample Attribute Mapping Summary

SAML Attribute from IdP	CNC Console User Attribute (Drop down)	Purpose
`uid`	`username`	Populates the user name in CNC Console UI or User Profile
`FirstName`	`firstName`	Populates the first name in CNC Console UI or user profile
`LastName`	`lastName`	Populates the last name in CNC Console UI or user profile
`email`	`email`	Populates the email address in CNC Console UI or user profile

> **Note**
>
> The IdP must include the attributes in the SAML response for the mapping to work. If attribute names differ from the example, use the exact attribute names configured at your IdP. No CNC Console application-level changes are required. All configurations are done in the CNC Console IAM GUI.

Sample screen shots for mappers are as follows:

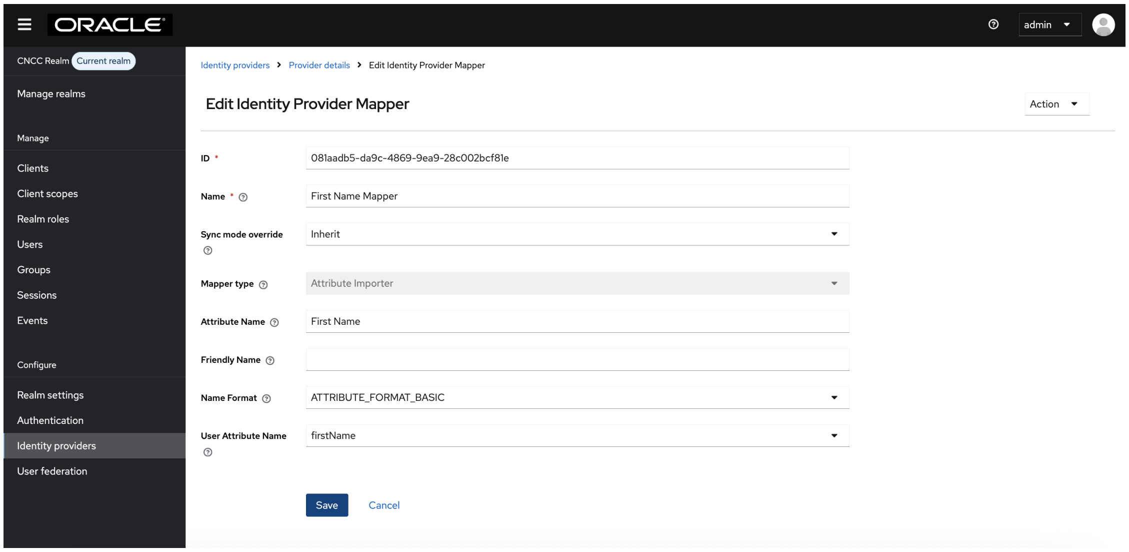
Figure 4-41 Mapper for Username



Edit Identity Provider Mapper

ID *	3ec35017-363b-4e75-89f7-56660f653ff4
Name *	Username Mapper
Sync mode override	Inherit
Mapper type	Attribute Importer
Attribute Name	uid
Friendly Name	
Name Format	ATTRIBUTE_FORMAT_BASIC
User Attribute Name	username

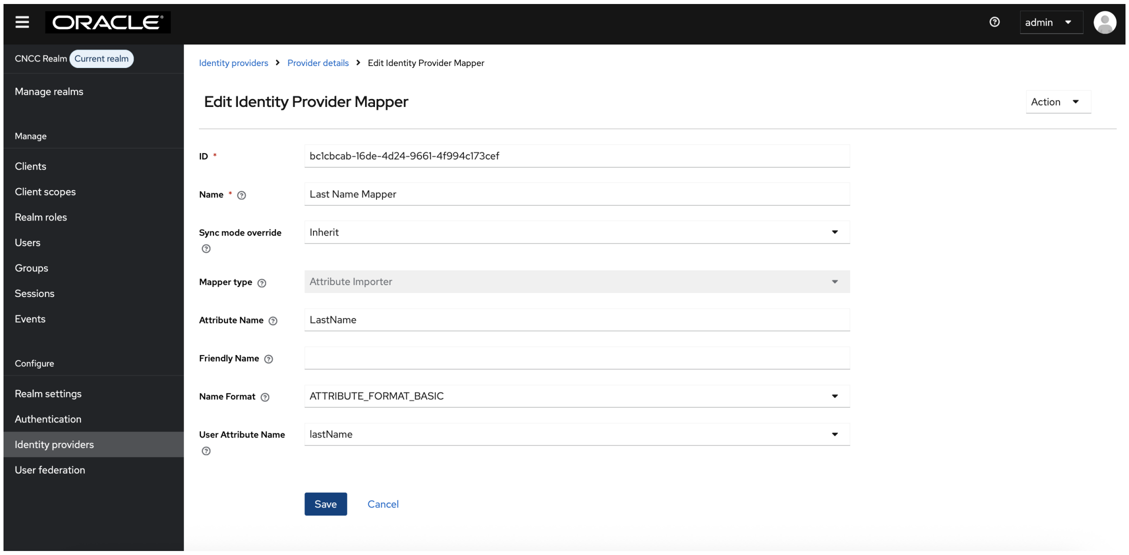
Figure 4-42 Mapper for First Name



Edit Identity Provider Mapper

ID *	081aadb5-da9c-4869-9ea9-28c002bcf81e
Name *	First Name Mapper
Sync mode override	Inherit
Mapper type	Attribute Importer
Attribute Name	First Name
Friendly Name	
Name Format	ATTRIBUTE_FORMAT_BASIC
User Attribute Name	firstName

Figure 4-43 Mapper for Last Name



Edit Identity Provider Mapper

ID *: bc1cbcab-16de-4d24-9661-4f994c173cef

Name *: Last Name Mapper

Sync mode override: Inherit

Mapper type: Attribute Importer

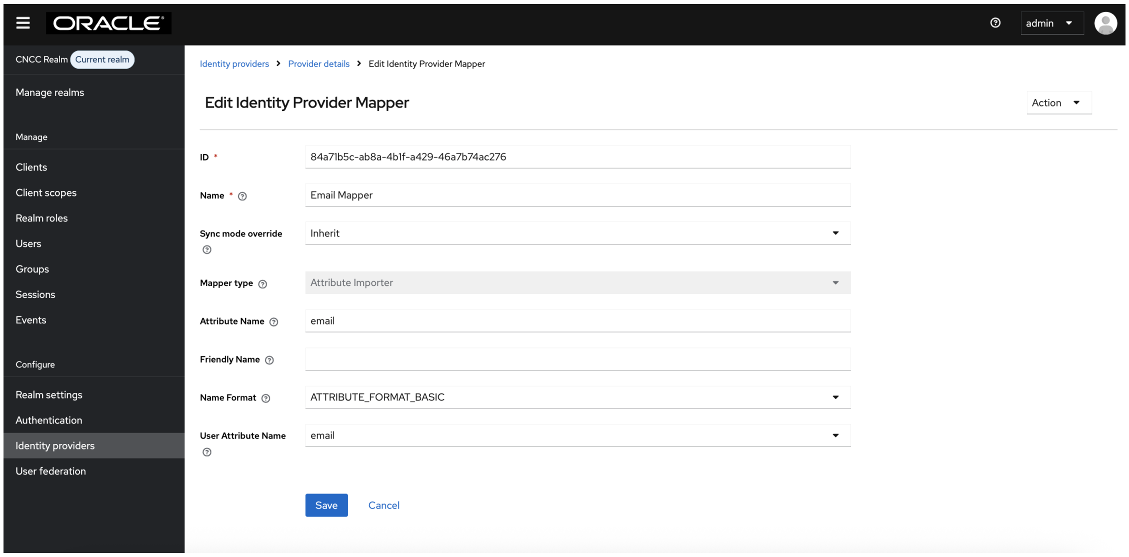
Attribute Name: LastName

Friendly Name:

Name Format: ATTRIBUTE_FORMAT_BASIC

User Attribute Name: lastName

Figure 4-44 Mapper for Email



Edit Identity Provider Mapper

ID *: 84a71b5c-ab8a-4b1f-a429-46a7b74ac276

Name *: Email Mapper

Sync mode override: Inherit

Mapper type: Attribute Importer

Attribute Name: email

Friendly Name:

Name Format: ATTRIBUTE_FORMAT_BASIC

User Attribute Name: email

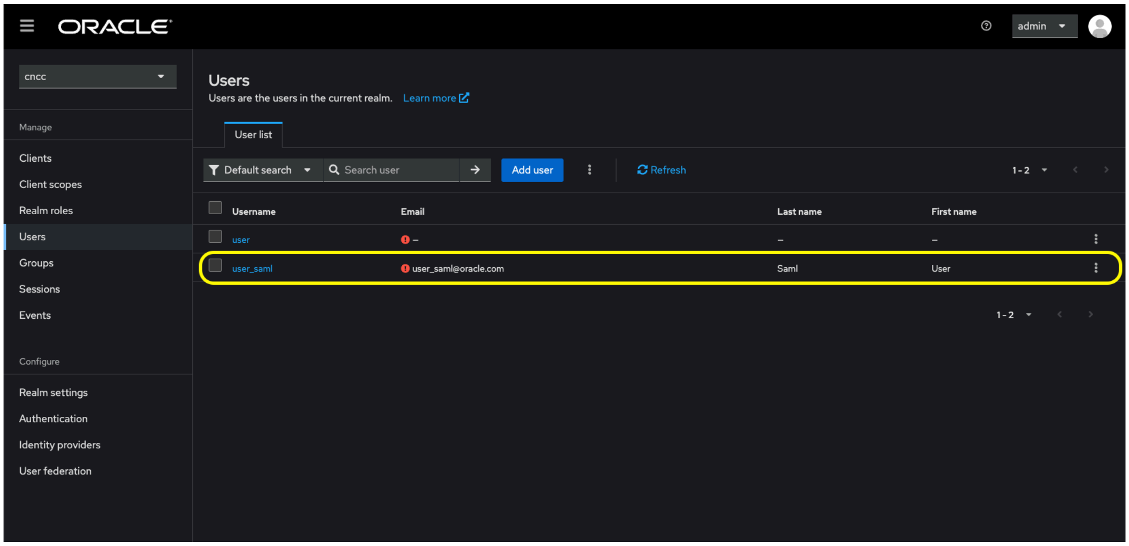
Verification

When these mappers are configured during the first SAML login,

- CNC Console IAM will automatically populate `username`, `firstName`, `lastName`, and `email` for the user.
- No additional manual updates are required in M-CNCC Core or M-CNCC IAM.

For example,

Figure 4-45 SAML User in CNC Console IAM After Adding Mappers



Users

Users are the users in the current realm. Learn more

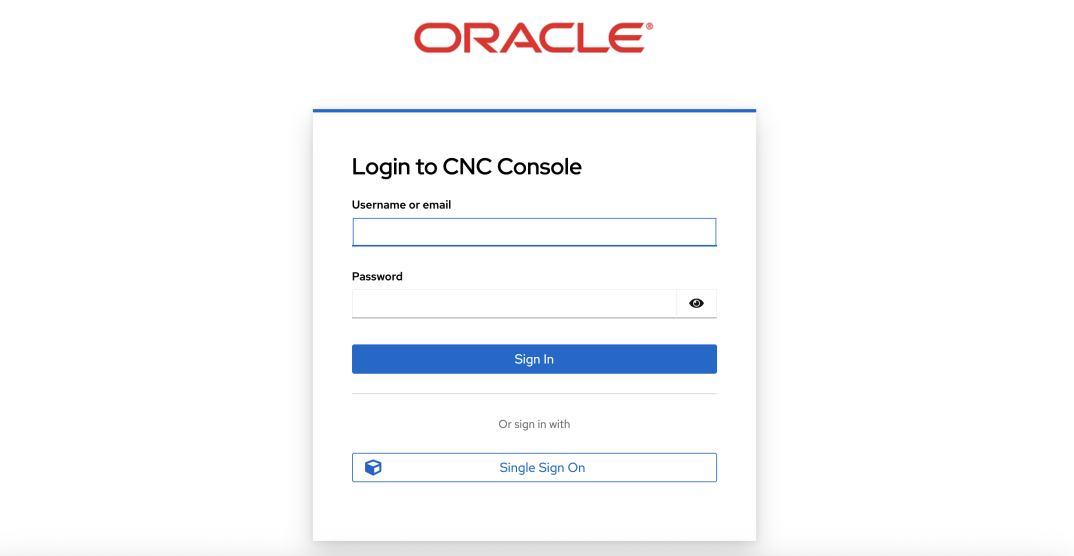
Username	Email	Last name	First name
user	–	–	–
user_saml	user_saml@oracle.com	Saml	User

Accessing CNC Console Core Application

Perform the following procedure to access the CNC Console application:

1. Log in to CNC Console Core, and browse to the application using hostname and port. The user is redirected to CNC Console IAM (broker).

   ```
   http://<cncc-core-ingress-extrenal-ip>:<cncc-iam-ingress-service-port>
   Example: http://cncc-core-ingress-gateway.cncc.svc.cluster.local:30075/
   ```

2. Click **Single Sign On** to authenticate using SAML SSO. The user is redirected to SAML IdP log in. Enter user details to access CNC Console Core application.

4.5 Integrating CNC Console LDAP Server with CNC Console IAM

Overview

The CNC Console IAM can be used as an integration platform to connect to existing LDAP and Active Directory servers.

User Federation in CNC Console IAM lets you sync users and groups from LDAP and Active Directory servers and assign roles.

CNC Console IAM provides an option to configure a secured connection URL to your LDAP store.

For example: `ldaps://myhost.com:636'

CNC Console IAM uses SSL to communicate with the LDAP server. The truststore must be properly configured on the CNC Console IAM server side, otherwise CNC Console IAM cannot trust the SSL connection to LDAP.

Sample LDAP LDIF File
This is a sample ldap-ldif file that ldap is loaded with for importing LDAP users and groups to CNC C Core.

```
dn: dc=oracle,dc=org
objectclass: top
objectclass: domain
objectclass: extensibleObject
dc: oracle

dn: ou=groups,dc=oracle,dc=org
objectclass: top
objectclass: organizationalUnit
ou: groups

dn: ou=people,dc=oracle,dc=org
objectclass: top
objectclass: organizationalUnit
ou: people

dn: uid=ben,ou=people,dc=oracle,dc=org
objectclass: top
objectclass: person
objectclass: organizationalPerson
objectclass: inetOrgPerson
cn: Ben Alex
sn: Alex
uid: ben
userPassword: benspass

dn: uid=bob,ou=people,dc=oracle,dc=org
objectclass: top
objectclass: person
objectclass: organizationalPerson
```

```
objectclass: inetOrgPerson
cn: Bob Hamilton
sn: Hamilton
uid: bob
userPassword: bobspass

dn: uid=joe,ou=people,dc=oracle,dc=org
objectclass: top
objectclass: person
objectclass: organizationalPerson
objectclass: inetOrgPerson
cn: Joe Smeth
sn: Smeth
uid: joe
userPassword: joespass

dn: cn=admin,ou=groups,dc=oracle,dc=org
objectclass: top
objectclass: groupOfUniqueNames
cn: admin
uniqueMember: uid=ben,ou=people,dc=oracle,dc=org
ou: admins

dn: cn=scp,ou=groups,dc=oracle,dc=org
objectclass: top
objectclass: groupOfUniqueNames
cn: scp
uniqueMember: uid=ben,ou=people,dc=oracle,dc=org
uniqueMember: uid=joe,ou=people,dc=oracle,dc=org
ou: scpusers

dn: cn=nrf,ou=groups,dc=oracle,dc=org
objectclass: top
objectclass: groupOfUniqueNames
cn: nrf
uniqueMember: uid=ben,ou=people,dc=oracle,dc=org
uniqueMember: uid=bob,ou=people,dc=oracle,dc=org
ou: nrfusers
```

The above data will be used as a reference to integrate LDAP with CNC Console IAM to import users into the cncc realm.

Sample LDAP LDIF File
This is a sample `ldap-ldif` file used to import LDAP users and groups into CNC Console IAM.

```
dn: dc=oracle,dc=org
objectclass: top
objectclass: domain
objectclass: extensibleObject
dc: oracle

dn: ou=groups,dc=oracle,dc=org
objectclass: top
objectclass: organizationalUnit
ou: groups
```

```
dn: ou=people,dc=oracle,dc=org
objectclass: top
objectclass: organizationalUnit
ou: people

dn: uid=ben,ou=people,dc=oracle,dc=org
objectclass: top
objectclass: person
objectclass: organizationalPerson
objectclass: inetOrgPerson
cn: Ben Alex
sn: Alex
uid: ben
userPassword: benspass

dn: uid=bob,ou=people,dc=oracle,dc=org
objectclass: top
objectclass: person
objectclass: organizationalPerson
objectclass: inetOrgPerson
cn: Bob Hamilton
sn: Hamilton
uid: bob
userPassword: bobspass


dn: cn=admin,ou=groups,dc=oracle,dc=org
objectclass: top
objectclass: groupOfUniqueNames
cn: admin
uniqueMember: uid=ben,ou=people,dc=oracle,dc=org
uniqueMember: uid=bob,ou=people,dc=oracle,dc=org
ou: admins
```

The above data can be used as a reference to integrate LDAP with CNC Console IAM for importing users into the default realm.

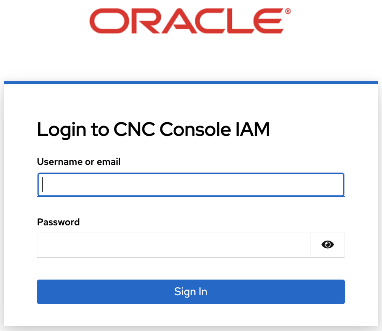
4.5.1 Configuring User Federation with CNC Console IAM

This section provides information about configuring user federation with CNC Console IAM (LDAP Server integration).

To configure user federation:

1. Log in to the CNC Console IAM console using admin credentials provided during CNC Console IAM installation.

Figure 4-46 Login Screen



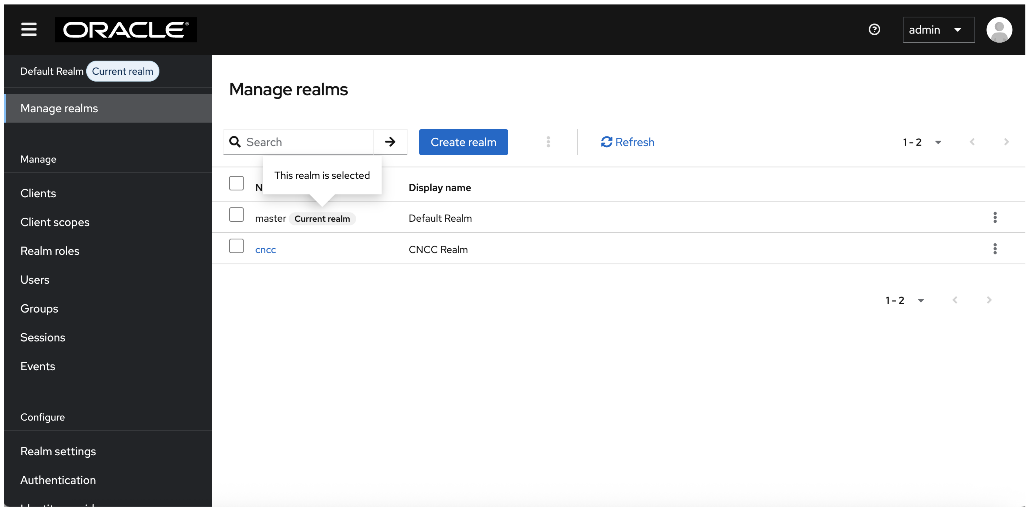
Login to CNC Console IAM

Username or email

Password

2. Select the appropriate realm based on where you want to import users. To switch realms, go to **Manage Realm** and select the desired realm from the list of available realms:
   a. To grant LDAP users access to CNC Console IAM, choose the default realm.

Figure 4-47 default realm

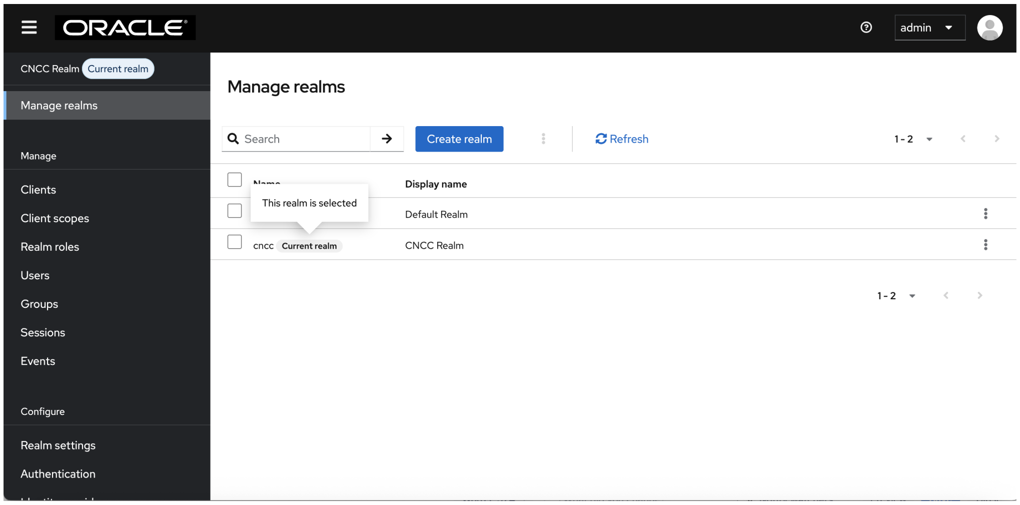


Manage realms

N	Display name
master Current realm	Default Realm
cncc	CNCC Realm

   b. To grant LDAP users access to CNC Console Core, choose the **cncc** realm.

Figure 4-48 cncc realm



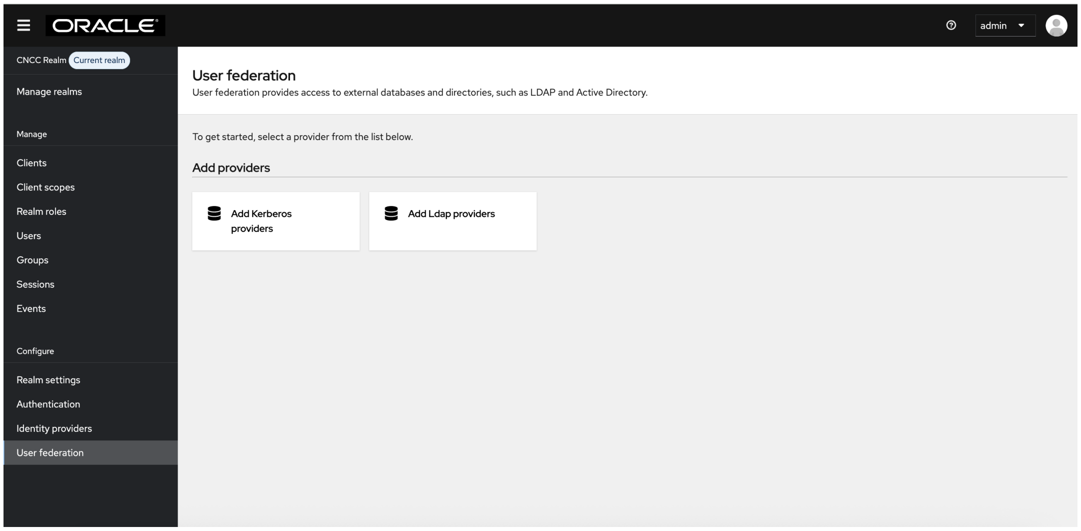
Manage realms

	Name	Display name
□		Default Realm
□	cncc Current realm	CNCC Realm

In the example below, we are selecting the **cncc Realm** to import users, as we want LDAP users to have access to CNC Console Core. Follow similar steps as outlined below in the default realm to import users, ensuring that LDAP users have access to CNC Console IAM.

3. Click **User Federation** on the left pane. The **User Federation** screen appears.

Figure 4-49 User Federation



User federation

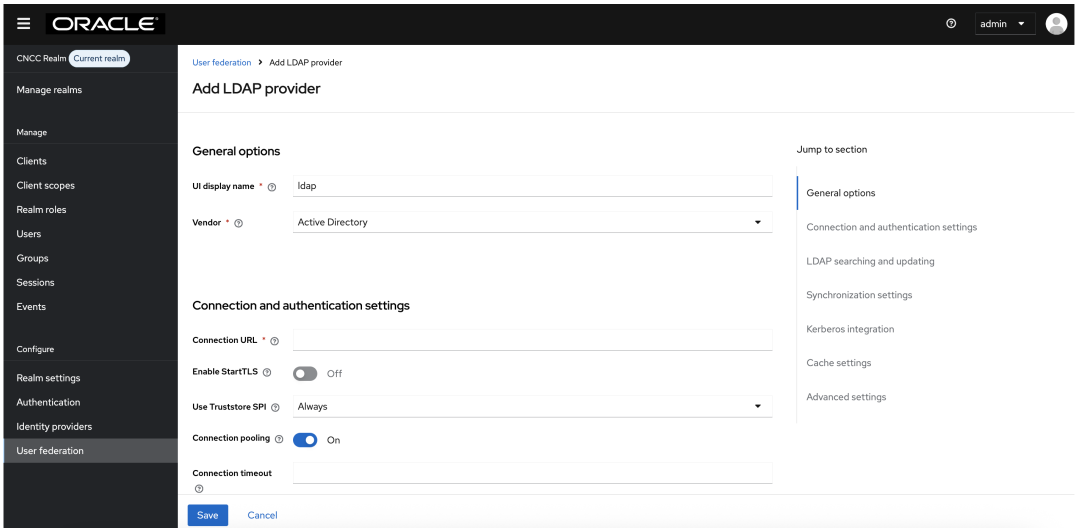
User federation provides access to external databases and directories, such as LDAP and Active Directory.

To get started, select a provider from the list below.

Add providers

4. Click **Add LDAP providers**. The following page will automatically open a form to fill in your LDAP connection parameters. The form will be initially empty as shown below:

Figure 4-50 Add LDAP providers



Add LDAP provider

General options

UI display name *: ldap

Vendor *: Active Directory

Connection and authentication settings

Connection URL *:

Enable StartTLS: Off

Use Truststore SPI: Always

Connection pooling: On

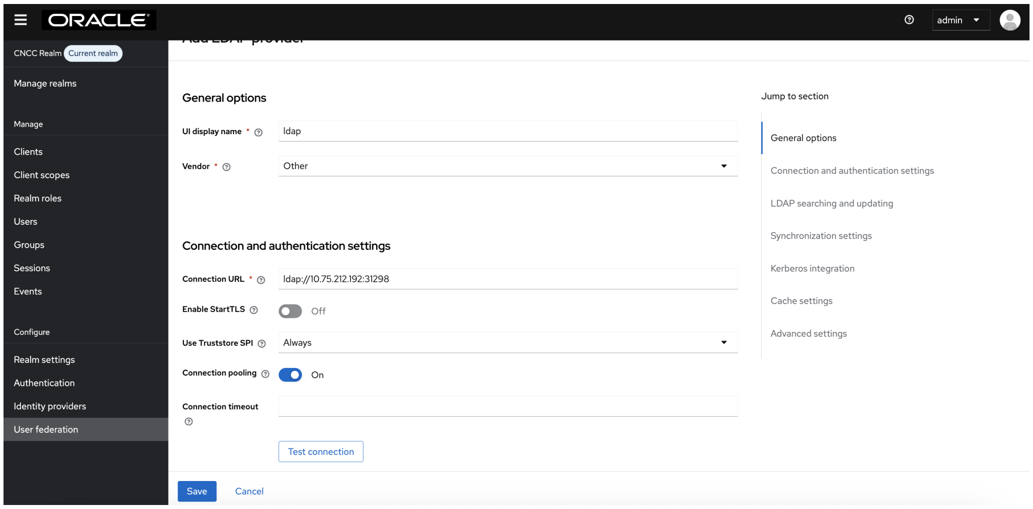
Connection timeout:

5. Enter the values for the following fields:
   - **UI Display Name:** Enter the display name.
   - **Vendor:** Enter the LDAP server provider name for the company.

> **Note**
>
> This usually populates the defaults for many fields. However, in case the user has a different setup than the defaults, the correct values must be provided. Based on the current setup, select **Other** from the drop-down list.

   - Provide your company LDAP server details in the **Connection URL** field, in the same way as you provided for `ldap-ldif` file already. That is, the connection URL (hostname prefixed with ldap:// OR when LDAP Secure connection enabled (LDAPS) hostname prefix should be ldaps://), and the port.

Figure 4-51 General Options



General options

UI display name *: ldap

Vendor *: Other

Connection and authentication settings

Connection URL *: ldap://10.75.212.192:31298

Enable StartTLS: Off

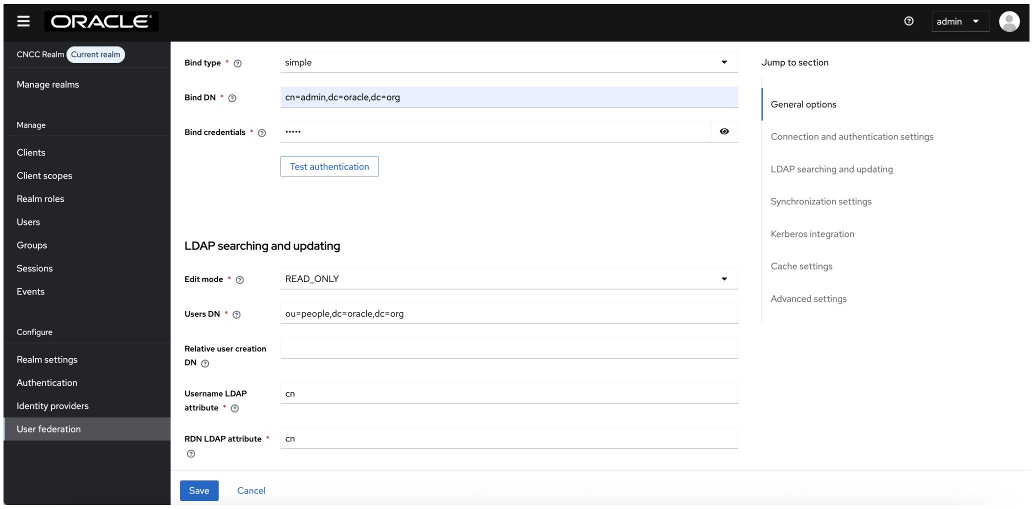
Use Truststore SPI: Always

Connection pooling: On

Connection timeout:

- If your LDAP is secured then select **simple** from the **Bind type** drop-down, and add the admin bind username and password, or select the Bind type as **none**. Sample data for the field Bind DN: "cn=admin,dc=oracle,dc=org".
- Click **Test Connection** and **Test Authentication**. Both of these tests should be successful.
- From the **Edit Mode** drop-down list, select **READ_ONLY**.
- In most cases, the **UUID LDAP attribute** value is set as entry UUID. If you do not have a suitable value, use an alternate unique identifier.
- Click **Test Connection** and **Test Authentication.**

Figure 4-52 LDAP searching and updating



Bind type * simple
Bind DN * cn=admin,dc=oracle,dc=org
Bind credentials * •••••

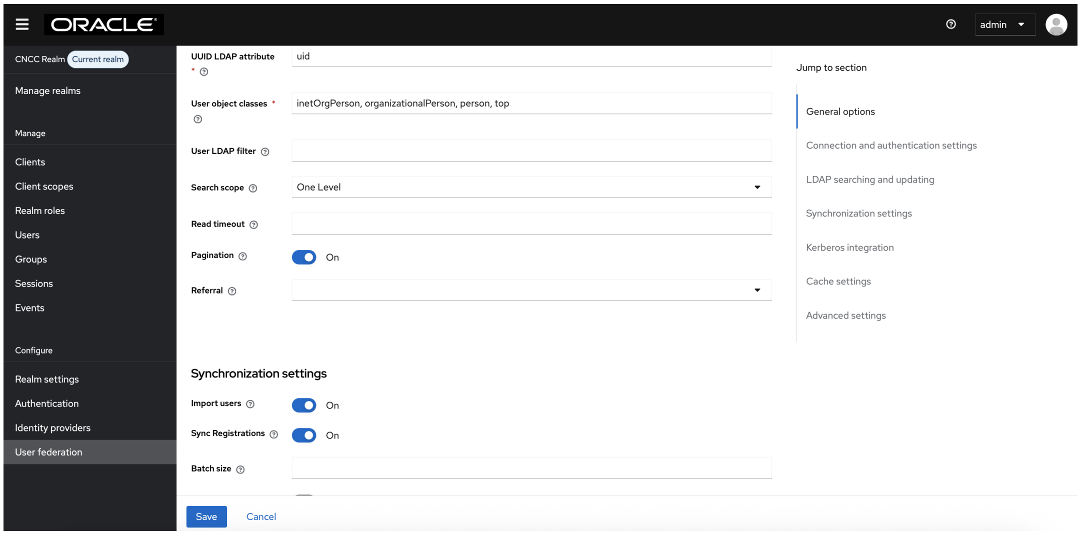
LDAP searching and updating

Edit mode * READ_ONLY
Users DN * ou=people,dc=oracle,dc=org
Relative user creation DN
Username LDAP attribute * cn
RDN LDAP attribute * cn

- The default setting for **Import Users** is **ON**. Change it to **OFF** to disable user sync.
- Set the cache policy to **NO_CACHE**.

6. After populating the required fields, the following screen appears:

Figure 4-53 Synchronization settings



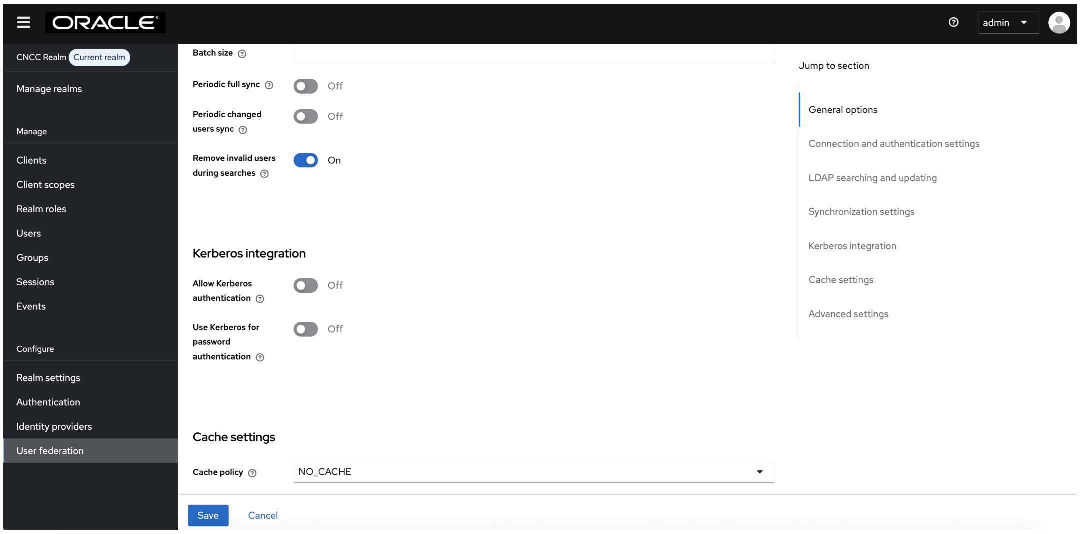
UUID LDAP attribute * uid
User object classes * inetOrgPerson, organizationalPerson, person, top
User LDAP filter
Search scope One Level
Read timeout
Pagination On
Referral

Synchronization settings

Import users On
Sync Registrations On
Batch size

7. Click **Save**.

Figure 4-54 Kerberos Integration



Batch size

Periodic full sync: Off

Periodic changed users sync: Off

Remove invalid users during searches: On

Kerberos integration

Allow Kerberos authentication: Off

Use Kerberos for password authentication: Off

Cache settings

Cache policy: NO_CACHE

Save Cancel

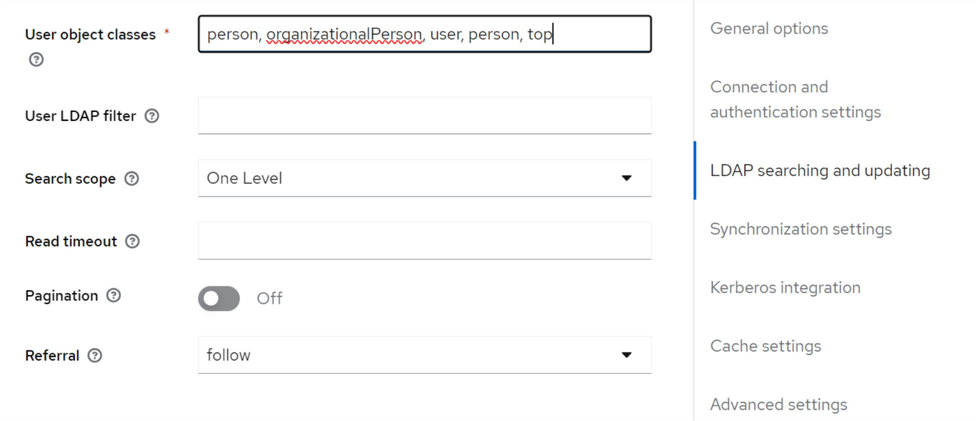
Note

Enabling and Disabling the Manage DSA IT Control in LDAP Requests: CNC Console IAM allows the user to enable or disable Manage DSA IT Control in the LDAP requests sent from CNC Console IAM pods to the LDAP server.

Manage DSA IT Control is enabled by default as **Referral** is set to **Ignore** in the **User Federation Setup**.

You can disable this by setting **Referral** to **Follow**.

Figure 4-55 Enabling and Disabling the Manage DSA IT Control in LDAP Requests



User object classes *: person, organizationalPerson, user, person, top

User LDAP filter:

Search scope: One Level

Read timeout:

Pagination: Off

Referral: follow

4.5.2 Grouping the LDAP Mapper and Assigning the Roles

When an LDAP Federation provider is created, CNC Console IAM provides a set of built-in mappers for this provider. Users can change the set and create a new mapper, or update and delete existing ones.

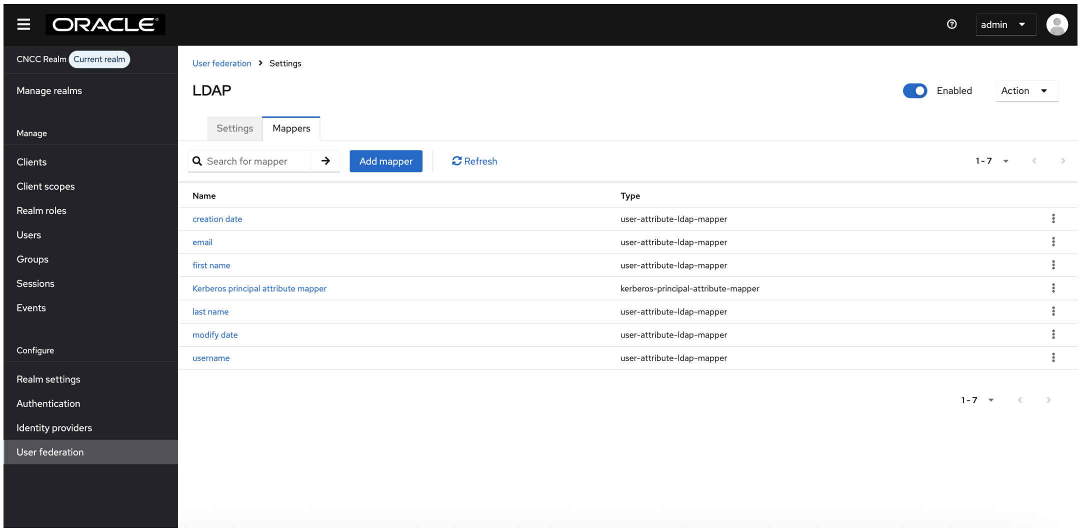
Group Mapper

The Group Mapper allows you to configure group mappings from LDAP into CNC Console IAM group mappings. Group mapper can be used to map LDAP groups from a particular branch of an LDAP tree into groups in CNC Console IAM. It also propagates user-group mappings from LDAP into user-group mappings in CNC Console IAM.

Perform the following procedure to add the group mapper and assign the roles:

1. Under **Configure** in the left pane, click **User Federation**. Click **ldap** and select the **Mappers** tab, and then click **Add Mapper**.

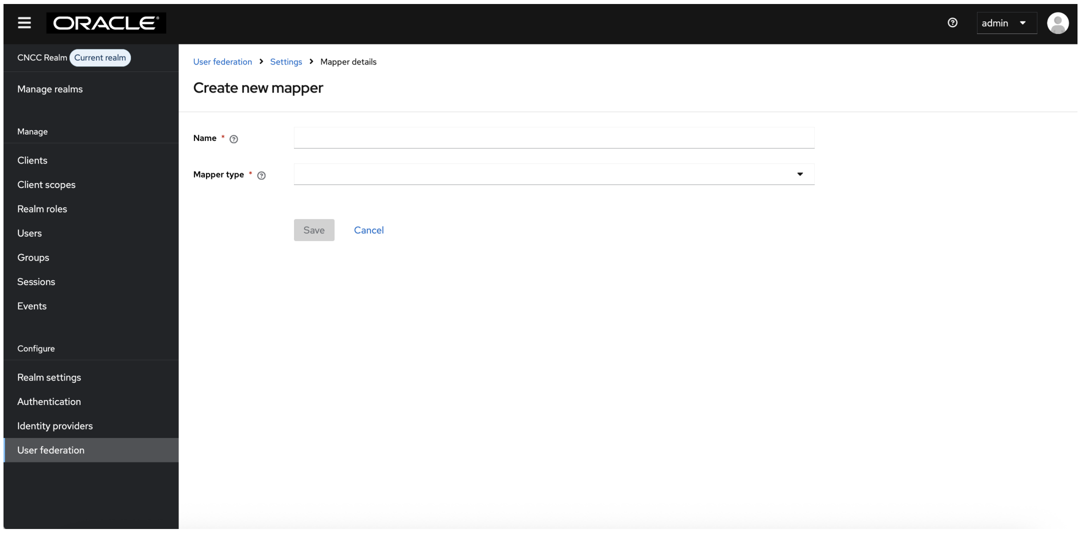
Figure 4-56   LDAP Mapper Page



Name	Type
creation date	user-attribute-ldap-mapper
email	user-attribute-ldap-mapper
first name	user-attribute-ldap-mapper
Kerberos principal attribute mapper	kerberos-principal-attribute-mapper
last name	user-attribute-ldap-mapper
modify date	user-attribute-ldap-mapper
username	user-attribute-ldap-mapper

2. The **Create New Mapper** page appears. Give an appropriate name for the field **Name.** Select group-ldap-mapper from the **Mapper Type** drop-down menu. Click **Save.**

Figure 4-57 User Federation Mapper Page

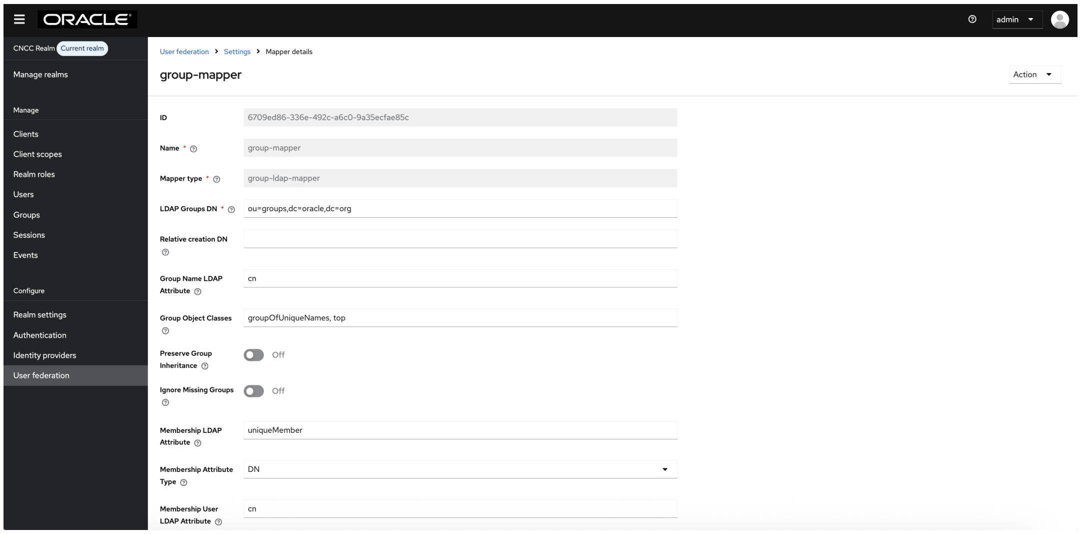


Create new mapper

Name *	
Mapper type *	

The following screen appears:

Figure 4-58 LDAP Mapper Filled Form



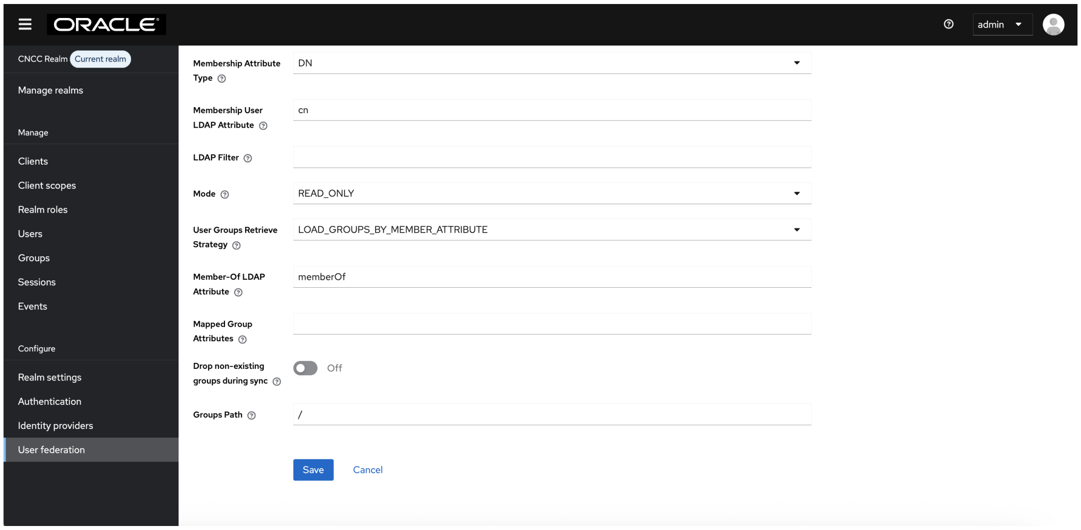
group-mapper

ID	6709ed86-336e-492c-a6c0-9a35ecfae85c
Name *	group-mapper
Mapper type *	group-ldap-mapper
LDAP Groups DN *	ou=groups,dc=oracle,dc=org
Relative creation DN	
Group Name LDAP Attribute	cn
Group Object Classes	groupOfUniqueNames, top
Preserve Group Inheritance	Off
Ignore Missing Groups	Off
Membership LDAP Attribute	uniqueMember
Membership Attribute Type	DN
Membership User LDAP Attribute	cn

> **Note**
>
> When selected, default values will be set by CNC Console IAM. However, you must change some values based on your LDAP records.

3. Click **Save**.

Figure 4-59 Save



Membership Attribute Type: DN

Membership User LDAP Attribute: cn

LDAP Filter:

Mode: READ_ONLY

User Groups Retrieve Strategy: LOAD_GROUPS_BY_MEMBER_ATTRIBUTE

Member-Of LDAP Attribute: memberOf

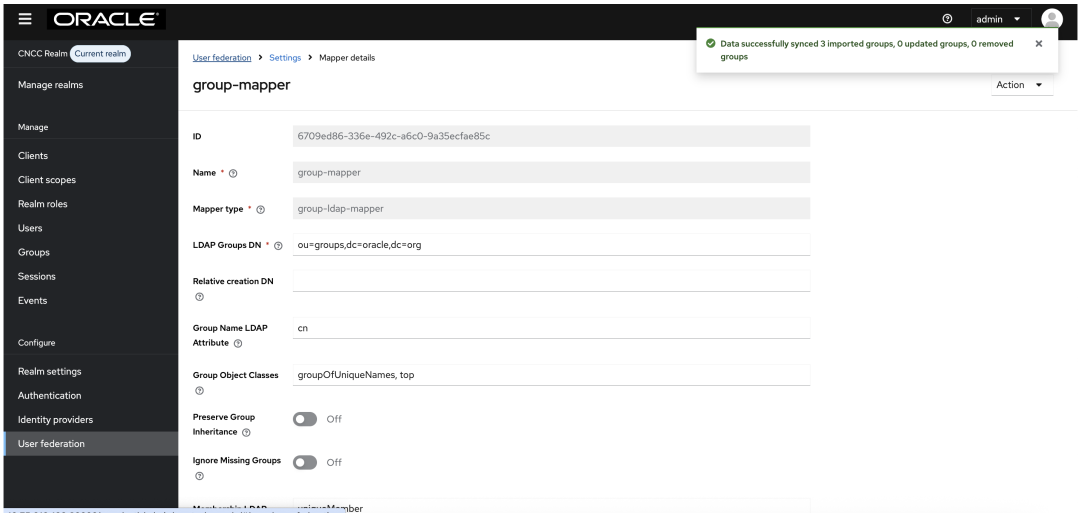
Mapped Group Attributes:

Drop non-existing groups during sync: Off

Groups Path: /

4. Click the name of your mapper. Under the Action menu, click **Sync LDAP Groups to Keycloak**. The success message appears with the number of groups imported and so on.

Figure 4-60 Group Mapper



Data successfully synced 3 imported groups, 0 updated groups, 0 removed groups

group-mapper

ID: 6709ed86-336e-492c-a6c0-9a35ecfae85c

Name *: group-mapper

Mapper type *: group-ldap-mapper

LDAP Groups DN *: ou=groups,dc=oracle,dc=org

Relative creation DN:

Group Name LDAP Attribute: cn

Group Object Classes: groupOfUniqueNames, top

Preserve Group Inheritance: Off

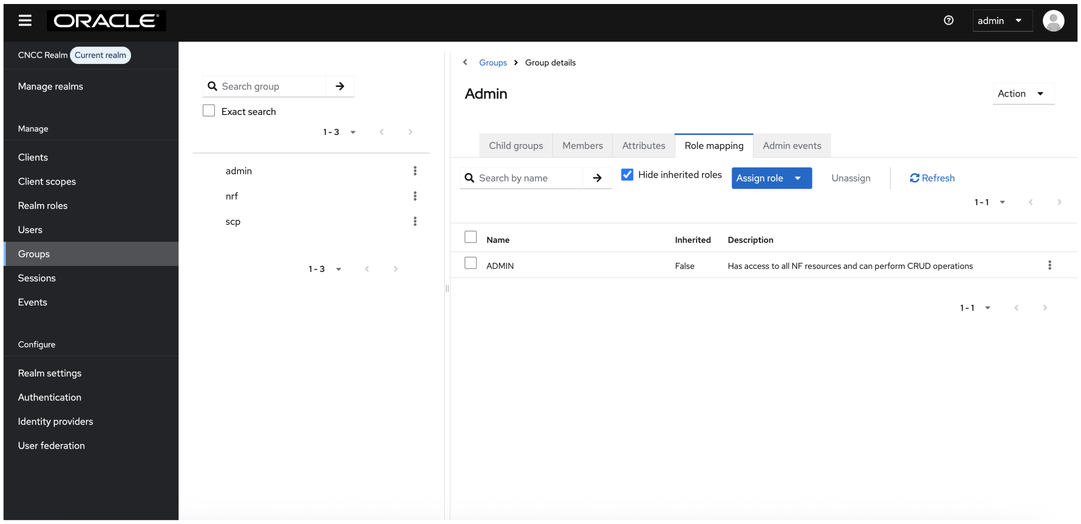
Ignore Missing Groups: Off

Membership LDAP: uniqueMember

> **Note**
>
> If this step fails, then you might need to look through the troubleshooting section and check the CNC Console IAM logs in debug mode. See the "CNC Console Logs" section in *Oracle Communication Cloud Native Configuration Console Troubleshooting Guide* for further details.

5. Select the **Groups** in the left pane to view all groups.

Figure 4-61 Groups

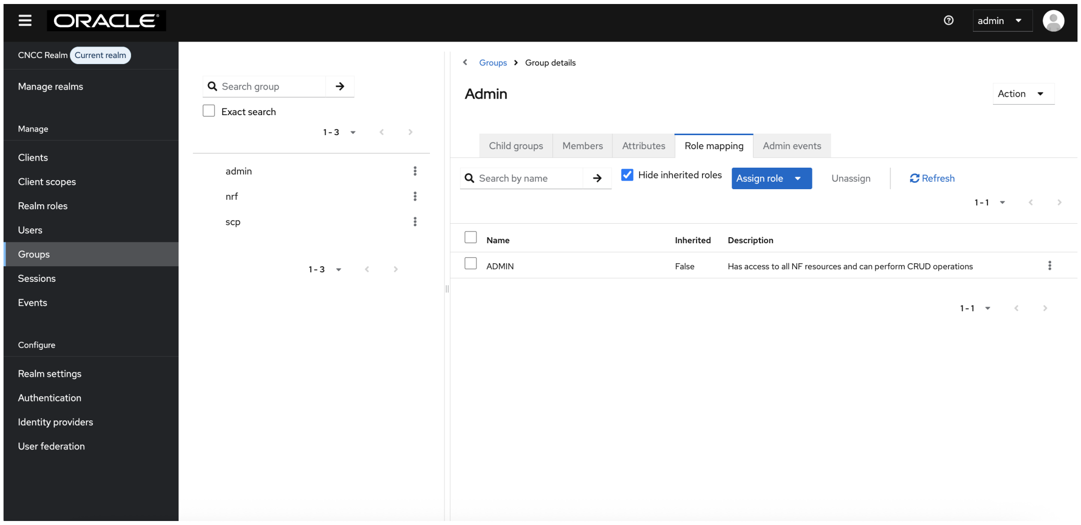


Name	Inherited	Description
ADMIN	False	Has access to all NF resources and can perform CRUD operations

6. Click any group and click **Edit**. The following tabs appear: **Child groups, Attributes, Role Mappings,** and **Members**.
   - Select the **Role Mapping** tab to view a list of roles that are predefined in CNC Console IAM.
   - Select one or more roles from **Available Roles** and assign it to the group.
   - When you are done, you can test authentication and authorization by logging into the CNC Console GUI.

   For example, in the **cncc** realm, if the group admin is assigned the **ADMIN** role, any user belonging to the admin group automatically inherits the Admin role, granting them full access to all NF resources supported by the CNC Console.

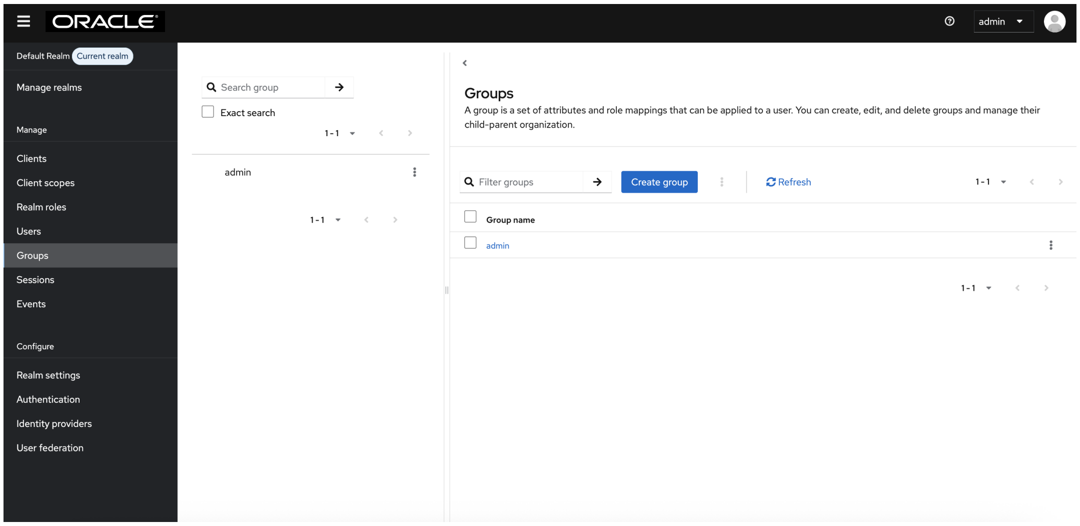
Figure 4-62 Role mapping to LDAP Group



Name	Inherited	Description
ADMIN	False	Has access to all NF resources and can perform CRUD operations

   - In the default realm, if the group admin is assigned the **ADMIN** role, any user belonging to the admin group automatically inherits the Admin role, granting full permissions to perform all operations and manage the realm.

Figure 4-63 CNC Console IAM Role mapping to LDAP Group for default Realm



Groups

A group is a set of attributes and role mappings that can be applied to a user. You can create, edit, and delete groups and manage their child-parent organization.

> **Note**
>
> - When the user password is updated from CNC Console IAM and sent to LDAP, it is always sent in plain text. This is different from updating the password to the built-in CNC Console IAM database, where hashing and salting is applied to the password before it is sent to the DB. In the case of LDAP, CNC Console IAM relies on the LDAP server to provide hashing and salting to passwords.
> - Most LDAP servers (Microsoft Active Directory, RHDS, FreeIPA) provide this by default. Some servers (OpenLDAP, ApacheDS) may store the passwords in plain text by default, and the user must explicitly enable password hashing for these servers.

5

Configuring OCI IAM

5.1 OCI SAML Integration

Overview

SAML (Security Assertion Markup Language) enables applications to authenticate a user using an identity provider. The identity provider authenticates the user and returns the assertion information about the authenticated user and the authentication event to the application. If the user tries to access any other application that uses the same identity provider for user authentication, the user does not need to log in a second time and will be granted access. This is the principle of SSO (Single Sign-On).

Adding a SAML Identity Provider in OCI IAM

OCI IAM Console allows you to add a SAML 2.0 identity provider (IdP) to an identity domain, so that authenticated users from the IdP can access Oracle Cloud Infrastructure resources and cloud applications.

For more information, see the "Managing a SAML Identity Provider" section in Oracle Cloud Infrastructure Documentation.

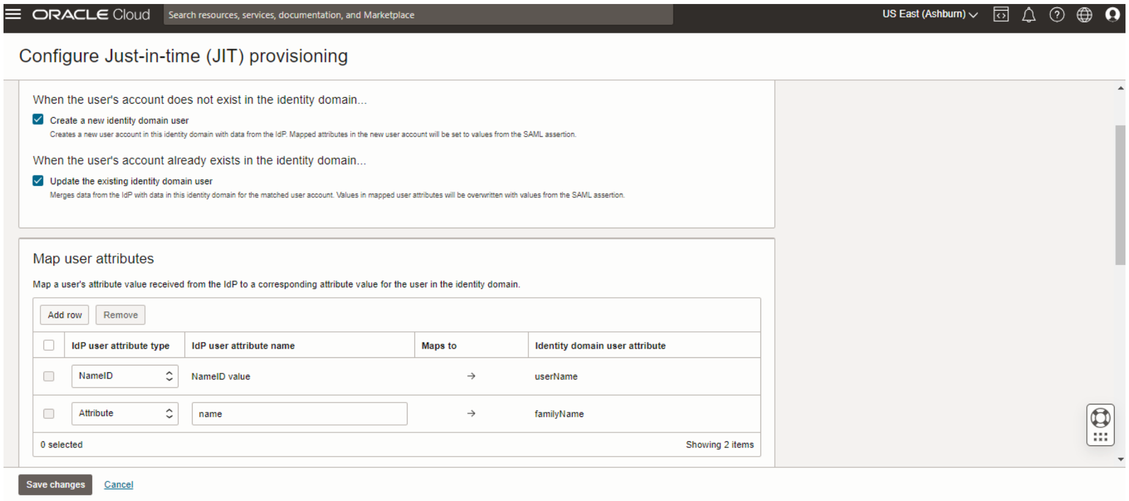
> ⓘ **Note**
>
> - After following the steps from the above guide, the SAML IdP is configured, but any user created in SAML IdP must also be created in OCI IAM to allow a successful login.
> - To log in directly to CNC Console Core using SAML IdP, SAML JIT (Just-In-Time) must be configured.
> - Role mappings are also configured as a part of JIT (Just-In-Time).

JIT (Just-In-Time) Configuration in OCI IAM

OCI IAM allows setting up a SAML identity provider (IdP) that uses Just-In-Time (JIT) provisioning for an identity domain.

For information on assigning identity providers to policy, see the "Adding a SAML Just-in-Time Identity Provider" section in Oracle Cloud Infrastructure Documentation.

Figure 5-1   JIT Provisioning



Configure Just-in-time (JIT) provisioning

When the user's account does not exist in the identity domain...

☑ Create a new identity domain user

Creates a new user account in this identity domain with data from the IdP. Mapped attributes in the new user account will be set to values from the SAML assertion.

When the user's account already exists in the identity domain...

☑ Update the existing identity domain user

Merges data from the IdP with data in this identity domain for the matched user account. Values in mapped user attributes will be overwritten with values from the SAML assertion.

Map user attributes

Map a user's attribute value received from the IdP to a corresponding attribute value for the user in the identity domain.

□	IdP user attribute type	IdP user attribute name	Maps to	Identity domain user attribute
□	NameID	NameID value	→	userName
□	Attribute	name	→	familyName

0 selected   Showing 2 items

Configuration Example:

The **NameID** value to **userName** is the default mapping, and the **name** to **familyName** mapping is configured by the user.

In this configuration, OCI IAM looks for the **name** attribute in the assertions coming from SAML IdP, as follows:

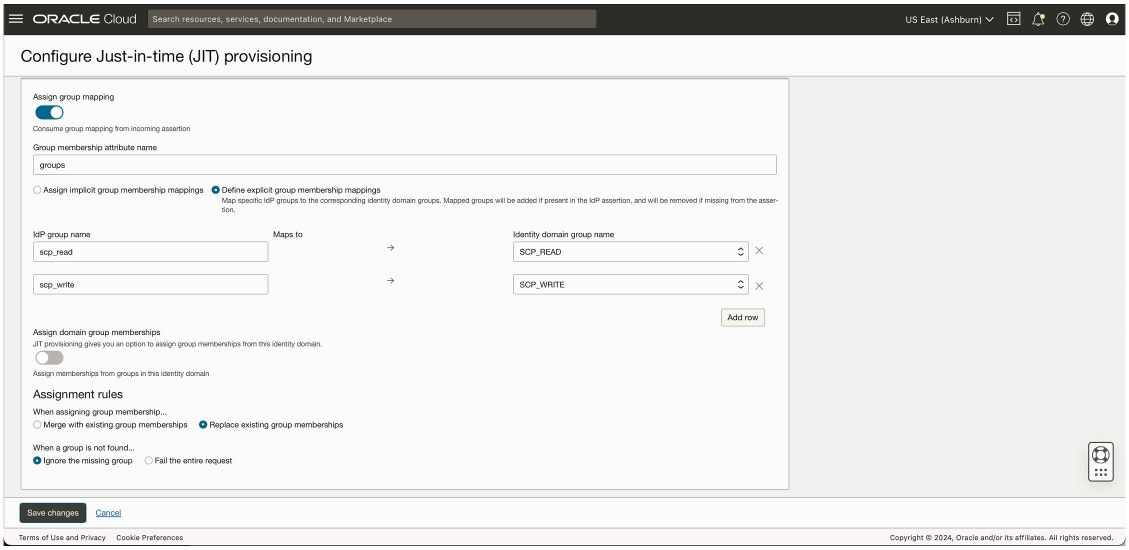
```
curl --location --request PUT 'http://{host}:{port}/occm-config/v1/occm/
logging' \
--header 'oc-cncc-id: Cluster1' \
--header 'oc-cncc-instance-id: Cluster1-occm-instance1' \
--header 'Authorization: Bearer eyJhbG...Yh8bJI_Owc_nb_hA' \
--header 'Content-Type: application/json' \
--data-raw '{
    "appLogLevel":"INFO",
    "packageLogLevel":[
    {
        "packageName":"root",
        "logLevelForPackage":"ERROR"
        }
    ]
    }'<saml:AttributeStatement>
      <saml:Attribute FriendlyName="name"
                      Name="name"
                      NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-
format:basic">
        <saml:AttributeValue xmlns:xs="http://www.w3.org/2001/XMLSchema"
                             xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"
                             xsi:type="xs:string">SamlUser</
saml:AttributeValue>
      </saml:Attribute>
```

> **Note**
>
> The user must ensure that the SAML IdP is populating these attributes in the assertions; otherwise, JIT configuration will not work, and SAML SSO authentication will fail.

In role mapping, since OCI IAM reads roles as groups, we have assigned 'Group membership attribute name' as **groups**.

After this, CNC Console (OCI IAM) Groups can be mapped to the IdP Groups as shown in the following screenshot:

Figure 5-2 Sample Role Mapping



Configure Just-in-time (JIT) provisioning

Assign group mapping

Consume group mapping from incoming assertion

Group membership attribute name

groups

○ Assign implicit group membership mappings ● Define explicit group membership mappings

Map specific IdP groups to the corresponding identity domain groups. Mapped groups will be added if present in the IdP assertion, and will be removed if missing from the assertion.

IdP group name	Maps to	Identity domain group name
scp_read	→	SCP_READ
scp_write	→	SCP_WRITE

Add row

Assign domain group memberships

JIT provisioning gives you an option to assign group memberships from this identity domain.

Assign memberships from groups in this identity domain

Assignment rules

When assigning group membership...

○ Merge with existing group memberships ● Replace existing group memberships

When a group is not found...

● Ignore the missing group ○ Fail the entire request

Save changes
Cancel
Terms of Use and Privacy
Cookie Preferences
Copyright © 2024, Oracle and/or its affiliates. All rights reserved.

Activating or Deactivating an Identity Provider

For information on activating or deactivating an identity provider, see the "Activating or Deactivating an Identity Provider" section in Oracle Cloud Infrastructure Documentation.

Assigning Identity Providers to the Policy

For information on assigning identity providers to policy, see the "Assigning Identity Providers to the Policy" section in Oracle Cloud Infrastructure Documentation.

Testing an Identity Provider

For information on testing an identity provider, see the "Testing an Identity Provider "section in Oracle Cloud Infrastructure Documentation.

Updating an Identity Provider

For information on updating an identity provider, see the "Updating an Identity Provider" section in Oracle Cloud Infrastructure Documentation.

5.2 OCI Password Policy

Overview

Password policies let you define a set of criteria for user passwords in an identity domain in IAM. The criteria are enforced when a user creates their password for an identity domain.

These password policies govern the passwords associated with users created in a particular identity domain, or the passwords of users in a user group of a particular domain.

Managing Password Policies in OCI IAM

For information on managing password policies in OCI IAM, see the Managing Password Policies section in Oracle Cloud Infrastructure Documentation.

Default Password Policy

If no custom policy is defined, then the OCI default password policy (Custom) will be applied.

ⓘ **Note**

If required, OCI recommends the user to review each criterion, and modify the existing password policy or create a new password policy according to their own requirements.

Policy Criteria	Description	Value
Password length (minimum)	The minimum number of characters that the password must contain.	12
Password length (maximum)	The maximum number of characters a password can contain. A password can’t exceed 500 characters.	40
Expires after (days)	The number of days until the password expires. Setting this option to 0 means that the password never expires.	120
Account lock threshold	The number of consecutive, unsuccessful login attempts into the identity domain after which the user account is locked. If you enter 0, then the user's account is never locked.	5
Enable automatic account unlock	Automatically unlocks the locked user accounts after the configured duration has passed.	Enabled

Automatically unlock account after (minutes)	The amount of time (in minutes), after which locked user accounts unlock automatically. You can set a value ranging between 5 minutes and 24 hours.	30
Previous passwords remembered	The number of unique new passwords that a user must use before a previously used password can be reused.	4
Alphabetic (minimum)	The number of alphabetical characters that the password must contain.	0
Numeric (minimum)	The number of numeric characters that the password must contain.	1
Special (minimum)	The number of special characters that the password must contain.	0
Lowercase (minimum)	The number of lowercase characters that the password must contain.	1
Uppercase (minimum)	The number of uppercase characters that the password must contain.	1
Unique (minimum)	The number of unique characters that the password must contain. Increasing the number of unique characters in a password can increase password strength by avoiding repetitive sequences that are easily guessed.	0
Repeated (maximum)	The number of repeated characters that are allowed for the password. Limiting the use of repeating characters in a password provides extra security by preventing users from creating passwords that are easy to guess, such as the same character repeated several times.	0
Starts with an alphanumeric character	Select this checkbox to force the first character of all passwords to be an alphanumeric character.	No

Required characters		The alphanumeric or special characters that the password must contain are separated by commas.	None
Password must not contain	The user's first name	Prevents the user's first name from being used as all or part of the password.	Enabled
	The user's last name	Prevents the user's last name from being used as all or part of the password.	Enabled
	The username	Prevents the username from being used as all or part of the password.	Enabled
Characters not allowed		The alphanumeric or special characters that aren’t allowed in the password are separated by commas.	None
Whitespaces		Prevents whitespace characters from being used as part of a password. A whitespace character is a character that represents horizontal space. For example, for the display name of John Smith, the space between “John” and “Smith” is a whitespace character.	None
Dictionary words		Screens all passwords for words that can be found in a dictionary and prohibits those words.	None

5.3 OCI Groups

Overview

Access management for resources is a critical function for any organization. Role-based access control (RBAC) helps you manage who has access to resources, what they can do with those resources and the areas they can access.

Role-based access control (RBAC)

RBAC restricts network access based on a person's role within an organization, and has become one of the main methods for advanced access control. The roles in RBAC refer to the levels of access that employees have to the network.

Role

A role is a collection of permissions that you can apply to users. Using roles makes it easier to add, remove, and adjust permissions than assigning permissions to users individually. As your user base increases in scale and complexity, roles become particularly useful.

> **Note**
>
> Roles are called Groups in OCI IAM.

5.3.1 CNC Console Groups

In CNC Console, RBAC is controlled by Oracle Cloud Infrastructure Identity Access Management (OCI IAM).

Groups related to CNC Console applications are defined in OCI IAM.

> **Note**
>
> All the Groups (NF Group, Instance Group, and INSTANCE_ALL) are automatically created at OCI-IAM during installation.

Read and Write Groups

NF Groups:
A user assigned this group can perform read and write operations for the assigned NFs. NF level groups are classified into:

- **<NF>_READ:** With this permission, the assigned user can perform the read operation for NFs.
  For example, if the user has the POLICY_READ group, then the user can only read configurations of any MOs within the Policy and cannot write, update, or delete any record within the Policy and cannot write or update or delete any record.
- **<NF>_WRITE:** With this permission, the assigned user can perform create, read, update, and delete operations for NFs. For example, if the user has POLICY_WRITE, then the user can read, write, update, or delete any MO configurations within the NF.

> **Note**
>
> CNCC_READ and CNCC_WRITE groups are also included under NF groups.

Instance Group

From release 24.2.0 onwards, CNC Console has introduced the instance-level group. The operator can enable or disable this feature using the `global.instanceLevelAuthorizationEnabled` flag in the Helm configuration. By default, this flag is set to **false**.

Instance-level groups allow users to have access to specific NF instances. A user can be associated with a single or multiple instance groups. The name of the instance group must match the instance ID of that instance given in Helm configuration under `global.instances[i].id`. Instance groups are automatically created by Helm hooks.

A user can only access one NF. For example, if the user has Cluster1-SCP-instance1 group, ADMIN, SCP_READ, or SCP_WRITE group, and Cluster1 group in case it is a multicluster deployment, then they can only access Cluster1-SCP-instance1.

INSTANCE_ALL

INSTANCE_ALL is a catch-all group for all instance-level groups. A user with INSTANCE_ALL and <NF>_READ or <NF>_WRITE groups has access to all instances. If this feature is enabled, the INSTANCE_ALL group is automatically assigned to all local users for the first upgrade. If the operator wants to restrict a user to a particular instance, then they have to unassign INSTANCE_ALL group and assign any of the instance-level groups to the user.

The user can access all NF instances. For example, if a user has `INSTANCE_ALL` group, then they can access all NF instances, provided they have ADMIN, <NF>_READ, or <NF>_WRITE group.

5.3.2 Managing Groups in OCI IAM

For information on managing groups in OCI IAM, see the "Managing Groups" section in Oracle Cloud Infrastructure Documentation.

5.3.3 Import and Export of Groups in OCI IAM

> **Note**
>
> For information on transferring data, see the "Transferring Data" section in Oracle Cloud Infrastructure Documentation.

5.3.3.1 Import Groups

For information on importing groups, see the "Importing Groups" section in Oracle Cloud Infrastructure Documentation.

5.3.3.2 Export Groups

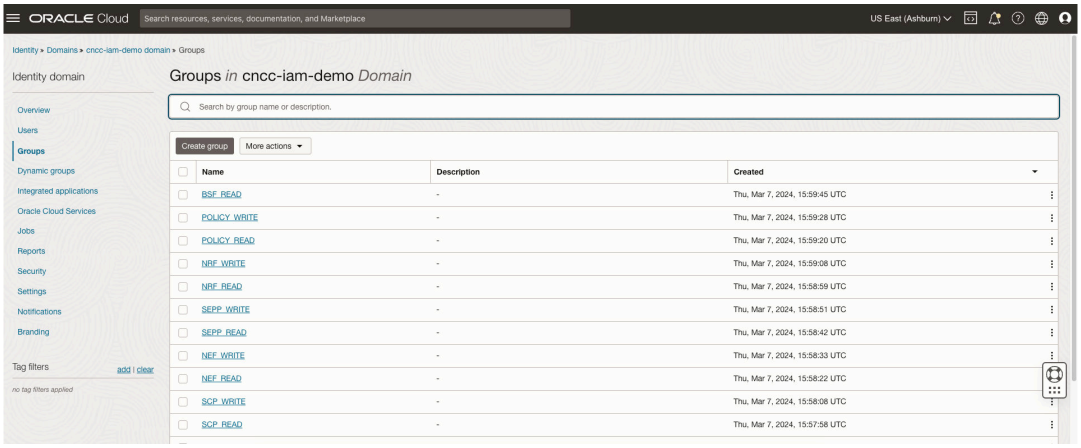
For information on exporting groups, see the "Exporting Groups" section in Oracle Cloud Infrastructure Documentation.

5.3.4 View Groups

To view groups:

1. Log in to **OCI Console**.
2. Open the navigation menu, and click **Identity & Security**. Under **Identity**, click **Groups**. A list of the groups in your tenancy appears.

Figure 5-3 View Groups



Name	Description	Created
BSF_READ	-	Thu, Mar 7, 2024, 15:59:45 UTC
POLICY_WRITE	-	Thu, Mar 7, 2024, 15:59:28 UTC
POLICY_READ	-	Thu, Mar 7, 2024, 15:59:20 UTC
NRF_WRITE	-	Thu, Mar 7, 2024, 15:59:08 UTC
NRF_READ	-	Thu, Mar 7, 2024, 15:58:59 UTC
SEPP_WRITE	-	Thu, Mar 7, 2024, 15:58:51 UTC
SEPP_READ	-	Thu, Mar 7, 2024, 15:58:42 UTC
NEF_WRITE	-	Thu, Mar 7, 2024, 15:58:33 UTC
NEF_READ	-	Thu, Mar 7, 2024, 15:58:22 UTC
SCP_WRITE	-	Thu, Mar 7, 2024, 15:58:08 UTC
SCP_READ	-	Thu, Mar 7, 2024, 15:57:58 UTC

5.3.5 Assigning Groups to User Using CSV

Prerequisite

The following prerequisites are required for assigning groups to users using a CSV file:

1. CNC Groups must be created within OCI IAM.
2. Users must be present within the OCI IAM.

Procedure

Perform the following procedure to assign groups to users using CSV:

1. Export the existing groups.
2. Update the users in User Members column of the `groups.csv` file.

   > **Note**
   >
   > For multiple users use semicolons (;) to separate the users.
   > For example, Robin; David

3. Save the CSV file.
4. Import the groups using `groups.csv`.

5.4 OCI Users

5.4.1 Managing Users

For information on managing users in OCI deployment, see the "Managing Users" section in Oracle Cloud Infrastructure Documentation.

5.4.2 Managing User Credentials in OCI IAM

For information on managing user credentials, see the "Changing Users Password" section in Oracle Cloud Infrastructure Documentation.

5.4.3 Import and Export of User in OCI IAM

This section describes the procedure to import and export users in OCI IAM.

> **Note**
>
> For more details, see the "Transferring Data" section in Oracle Cloud Infrastructure Documentation.

5.4.3.1 Import Users

For information on importing users, see the "Importing Users" section in Oracle Cloud Infrastructure Documentation.

5.4.3.2 Export User

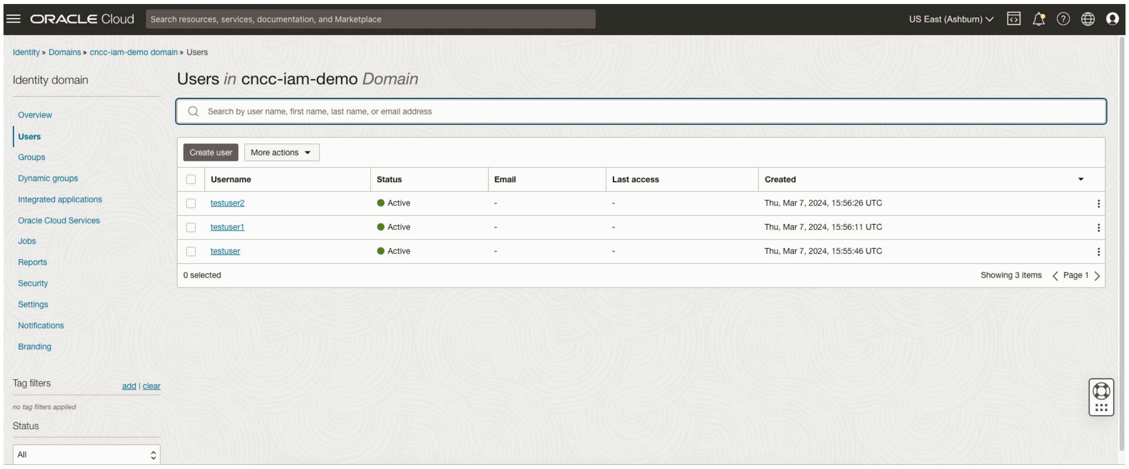
For information on exporting users, see the "Exporting Users" section in Oracle Cloud Infrastructure Documentation.

5.4.4 View Users in OCI IAM

To view users:

1. Log in to **OCI Console**.
2. Open the navigation menu and click **Identity & Security**. Under **Identity**, click **Users**. A list of users in your tenancy appears.

Figure 5-4 View Users



Users *in* cncc-iam-demo *Domain*

□	Username	Status	Email	Last access	Created
□	testuser2	● Active	-	-	Thu, Mar 7, 2024, 15:56:26 UTC
□	testuser1	● Active	-	-	Thu, Mar 7, 2024, 15:56:11 UTC
□	testuser	● Active	-	-	Thu, Mar 7, 2024, 15:55:46 UTC

0 selected

Showing 3 items < Page 1 >

5.5 OCI Active Directory Integration

Overview

Active Directory (AD) stores information about objects on the network and makes this information easy for administrators and users to find and use. Active Directory uses a structured data store as the basis for a logical and hierarchical organization of directory information.

Prerequisites

The following prerequisites are required for OCI active directory integration:

1. Running Active Directory.
2. Groups must be created in Active Directory following the CNC Console group name format, and users must be assigned to the groups.

> **Note**
>
> To create CNC Console groups, see OCI Groups.

5.5.1 Setting Up a Microsoft Active Directory Bridge

For information on how to install the active directory bridge, see the "Setting Up a Microsoft Active Directory Bridge section" in the Oracle Cloud Infrastructure Documentation.

5.5.2 Setting Up A Microsoft Active Directory Bridge With SSL Enabled

For information on setting up a Microsoft Active Directory bridge with SSL enabled, see the "Setting Up a Microsoft Active Directory Bridge" section in the Oracle Cloud Infrastructure Documentation.

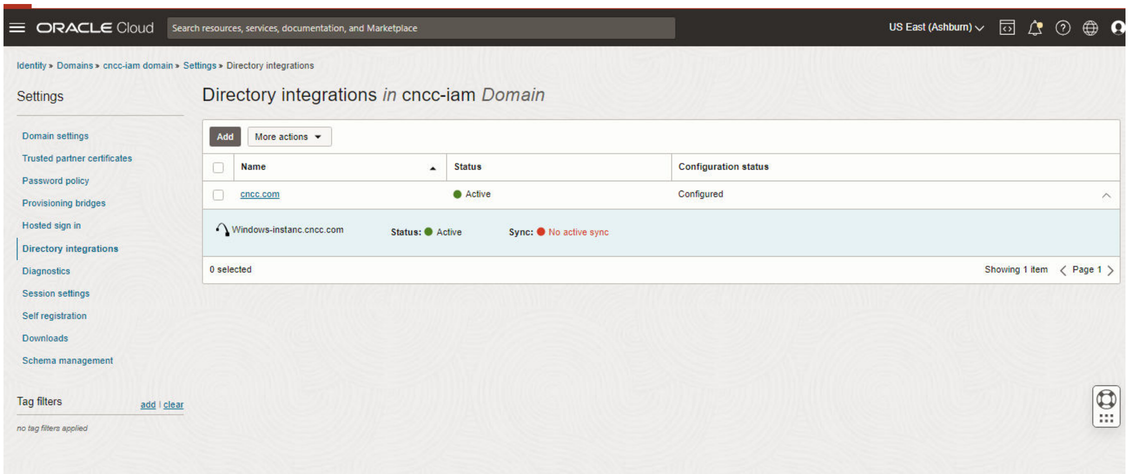
> **Note**
>
> You must enable the **Use SSL** flag while providing the Active Directory credentials.

5.5.3 Importing Users and Groups From Active Directory

Prerequisite
You must perform the following steps before importing users and groups:

1. Active Directory Bridge must be configured.
2. Active Directory Bridge must be in an active state.

Figure 5-5   Directory integration in cncc-iam Domain



Directory integrations *in* cncc-iam *Domain*

□	Name	Status	Configuration status
□	cncc.com	● Active	Configured

Windows-instanc.cncc.com   **Status:** ● Active   **Sync:** ● No active sync

0 selected   Showing 1 item   Page 1

> **Note**
>
> If Active Directory bridge is not configure or the status is inactive, see the "Setting Up a Microsoft Active Directory Bridge" section in the Oracle Cloud Infrastructure Documentation.

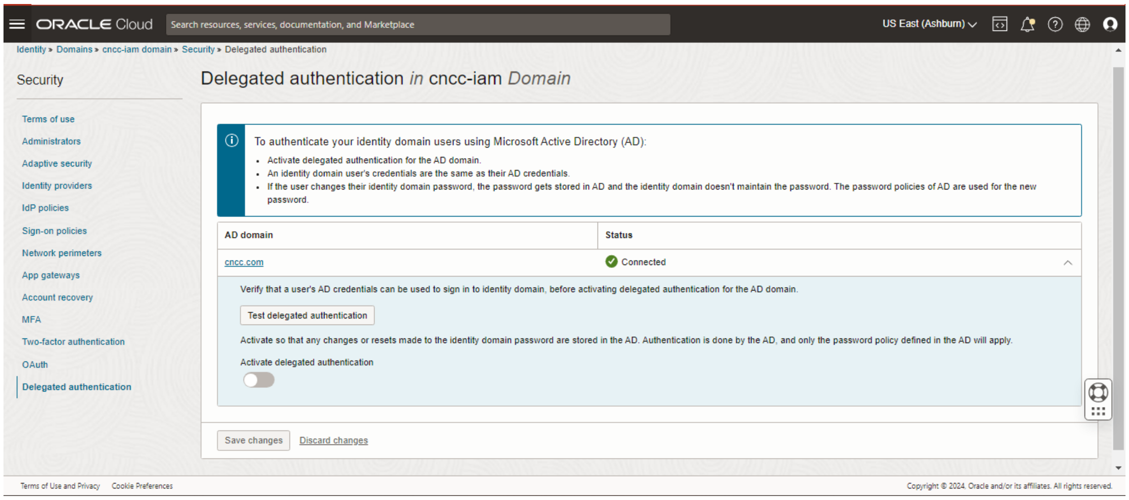
Procedure
To import users from Active Directory, see the "Importing Users from Active Directory" section in the Oracle Cloud Infrastructure Documentation.

5.5.4 Setting up Delegated Authentication

To set up delegated authentication:

1. Log in to **OCI Console**.
2. Open the navigation menu and click **Identity & Security**. Under **Identity**, click **Domains**.
3. Click **Security.**
4. Click **Delegated Authentication.**
5. Click **Test Delegated Authentication.**
6. Provide active directory user credentials.
7. Enable **Activate Delegated Authentication**.

Figure 5-6 Delegated Authentication in CNC Console IAM Domain



Delegated authentication *in* cncc-iam *Domain*

To authenticate your identity domain users using Microsoft Active Directory (AD):

- Activate delegated authentication for the AD domain.
- An identity domain user's credentials are the same as their AD credentials.
- If the user changes their identity domain password, the password gets stored in AD and the identity domain doesn't maintain the password. The password policies of AD are used for the new password.

AD domain	Status
cncc.com	✓ Connected

Verify that a user's AD credentials can be used to sign in to identity domain, before activating delegated authentication for the AD domain.

Test delegated authentication

Activate so that any changes or resets made to the identity domain password are stored in the AD. Authentication is done by the AD, and only the password policy defined in the AD will apply.

Activate delegated authentication: ○ (off)

Save changes Discard changes

6

Accessing NF Configurations Through Curl and Postman

6.1 For Non OCI Deployment

This section describes how CNC Console accesses NF resources through curl or Postman.

6.1.1 Generate Access Tokens

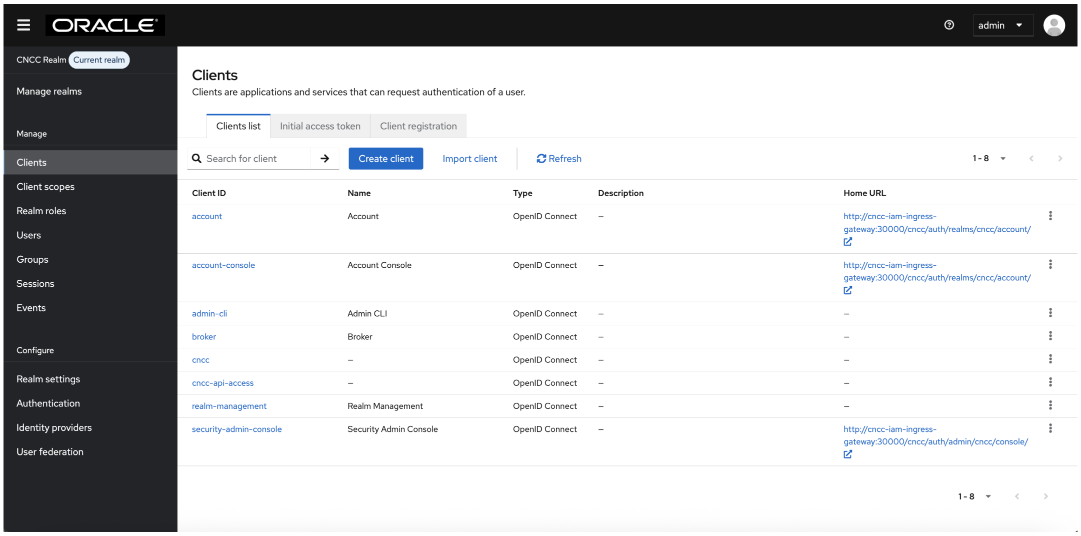
CNC Console IAM provides a REST API for generating and refreshing access tokens.

You must use the `cncc-api-access` client to access NF resources through REST APIs. For security reasons, **Direct Access Grants Enabled** is set to **OFF** by default.

Perform the following steps to set **Direct Access Grants Enabled** to ON:

1. Log in to CNC Console IAM with valid credentials.
2. Go to **Manage realms** and click the **cncc** realm from the list of available realms.
3. On the right pane, click **Clients**. The following screen appears:

Figure 6-1 Clients



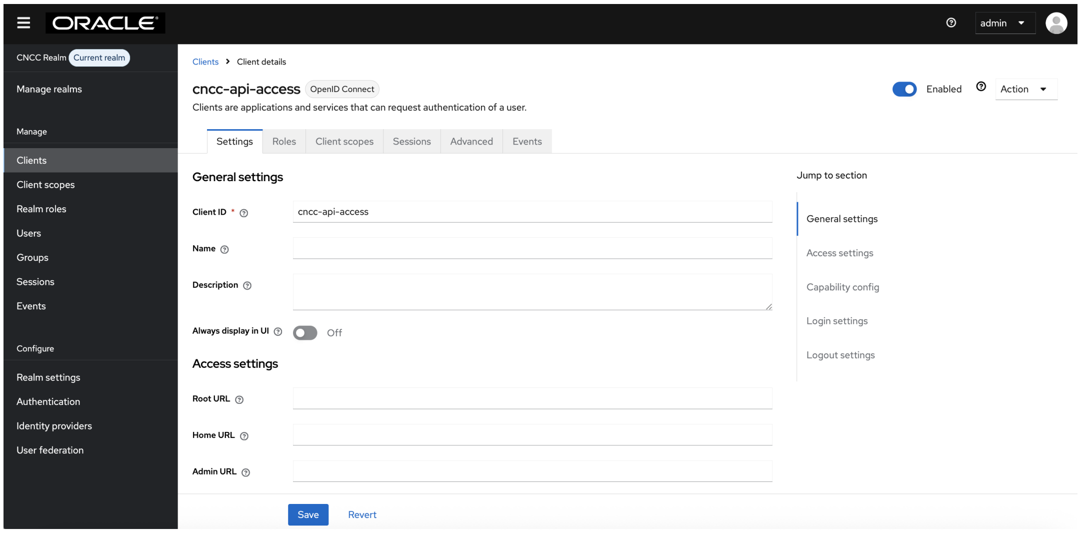
Clients

Clients are applications and services that can request authentication of a user.

Client ID	Name	Type	Description	Home URL
account	Account	OpenID Connect	–	http://cncc-iam-ingress-gateway:30000/cncc/auth/realms/cncc/account/
account-console	Account Console	OpenID Connect	–	http://cncc-iam-ingress-gateway:30000/cncc/auth/realms/cncc/account/
admin-cli	Admin CLI	OpenID Connect	–	–
broker	Broker	OpenID Connect	–	–
cncc	–	OpenID Connect	–	–
cncc-api-access	–	OpenID Connect	–	–
realm-management	Realm Management	OpenID Connect	–	–
security-admin-console	Security Admin Console	OpenID Connect	–	http://cncc-iam-ingress-gateway:30000/cncc/auth/admin/cncc/console/

4. Click **cncc-api-access**. The following screen appears:

Figure 6-2 cncc-api-access



cncc-api-access

Clients are applications and services that can request authentication of a user.

General settings

Client ID * : cncc-api-access

Name:

Description:

Always display in UI: Off

Access settings

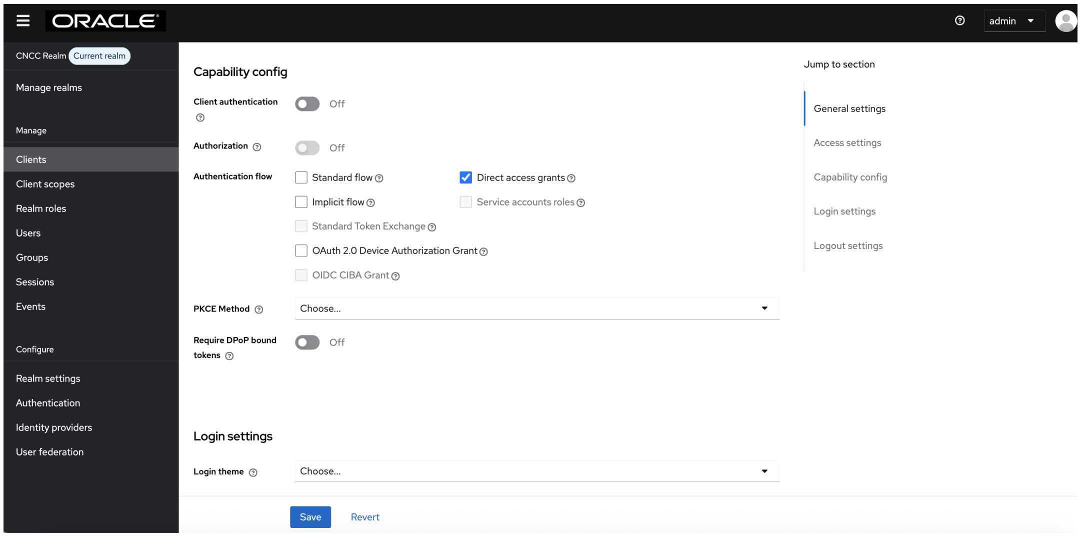
Root URL:

Home URL:

Admin URL:

5. Navigate to the **Capability config** section in the **Settings** tab and select the **Direct Access Grants** checkbox. Click **Save**.

Figure 6-3 Direct Access Grants



Capability config

Client authentication: Off

Authorization: Off

Authentication flow:
- □ Standard flow
- ☑ Direct access grants
- □ Implicit flow
- □ Service accounts roles
- □ Standard Token Exchange
- □ OAuth 2.0 Device Authorization Grant
- □ OIDC CIBA Grant

PKCE Method: Choose...

Require DPoP bound tokens: Off

Login settings

Login theme: Choose...

Perform the following procedure to generate the access tokens:

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

   For example:

   `http://10.75.182.79:8080/cncc/auth/realms/cncc/protocol/openid-connect/token`

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

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

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

3. The curl command to access the token is as follows:

```
  curl --location --request POST 'http://${cncc-iam-ingress-extrenal-ip}:$
{cncc-iam-ingress-service-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=<password value>' \
--data-urlencode 'client_id=cncc-api-access'
```

4. In response, you will get an **access_token** and a **refresh_token**. The response is as follows:

```
{
    "access_token":
"eyJhbGciOiJSUzI1NiIsI................._LcCZYwDQJJTloj2PJ8y1WjO9l2Q",
    "expires_in": 300,
    "refresh_expires_in": 1800,
    "refresh_token":
"eyJhbGciOiJIUzI1NiIs ................ldUIiwia2lkIiA6ICI3YTFlYvKPF-ZIg",
    "token_type": "bearer",
    "not-before-policy": 0,
    "session_state": "6c42d978-14ac-4793-a1e3-789cfbdb2b74",
    "scope": "email profile"
}
```

> **Note**
>
> M-CNCC IAM IP or FQDN which is used to generate access token, and M-CNCC IAM IP or FQDN which is specified in `custom-cncc_values.yaml` must match.

6.1.2 Refresh Access Tokens

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. 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'
```

```
Example:
'client_id': 'cncc-api-access',
'refresh_token':
'eyJhbGciOiJIUzI1NiIs ................ldUIiwia2lkIiA6ICI3YTFlYvKPF-ZIg',
'grant_type': 'refresh_token'
```

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

6.1.3 Accessing NF Resources

Perform the following procedure to access NF resources:

To access NF resources, you must use the access token in every request to an NF resource by placing it in the Authorization header.

The following headers must be included while sending the API request:

- Authorization: The access token should 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.

> ⓘ **Note**
>
> In case of Policy and BSF NFs, additional API prefixes are needed to access NF resources through console and these prefixes differ from one endpoint to another.
>
> Few of the additional API prefixes are the following:
>
> - Policy: "/policyapi"
> - BSF: "/bsfapi"
>
>   For other prefixes and for more information please refer to respective NF documentation.

Using the CNC Console IAM API, the following headers must be passed in curl or postman request while accessing NF resource:

```
curl --location --request GET 'http://${cncc-mcore-ingress-external-ip}:$
{cncc-mcore-ingress-service-port}/<NF API URI>' \

--header 'oc-cncc-id: <oc-cncc-id-value>' \

--header 'oc-cncc-instance-id: <oc-cncc-instance-id-value>' \

--header 'Authorization: Bearer <token>'
```

For example, using SCP Canary Release API, the following headers must be passed in curl or postman request while accessing NF resource:

```
curl --location --request GET 'http://${cncc-mcore-ingress-external-ip}:$
{cncc-mcore-ingress-service-port}/ocscp/scpc-configuration/v1/canaryrelease '
\

--header 'oc-cncc-id: Cluster1' \

--header 'oc-cncc-instance-id: Cluster1-scp-instance1' \

--header 'Authorization: Bearer <token>'
```

6.1.4 Accessing cnDBTier Resources

This section describes the procedure to access cnDBTier resources through CNC Console using curl or Postman. The authentication process to obtain the access token remains the same as described previously. The same NF instance parameters, `oc-cncc-id` and `oc-cncc-instance-id`, should be used for cnDBtier API access.

> **Note**
>
> To access cnDBtier APIs, users must be assigned the appropriate NF specific roles.
>
> For example, access to CNC Console cnDBTier resources requires the **CNCC_READ**/**CNCC_WRITE** roles, while access to SCP cnDBTier resources requires the **SCP_READ**/**SCP_WRITE** roles.
>
> Similarly, for any other integrated NF, ensure that relevant NF specific roles are assigned to the user.

To access cnDBtier resources, use the access token obtained as detailed above and include it in the **Authorization** header for every API request. The required headers are:

- Authorization: The access token should 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

Run the following command to access cnDBtier API:

```
curl --location --request GET 'http://${cncc-mcore-ingress-external-ip}:$
{cncc-mcore-ingress-service-port}/<cnDBtier API URL>' \
  --header 'oc-cncc-id: <oc-cncc-id-value>' \
  --header 'oc-cncc-instance-id: <oc-cncc-instance-id-value>' \
  --header 'Authorization: Bearer <token>'
```

For example, accessing the **CNCC** cnDBtier API:

```
curl --location --request GET 'http://${cncc-mcore-ingress-external-ip}:$
{cncc-mcore-ingress-service-port}/ocdbtier/version' \
  --header 'oc-cncc-id: Cluster1' \
  --header 'oc-cncc-instance-id: Cluster1-cncc-instance1' \
  --header 'Authorization: Bearer <token>'
```

For example, accessing the **SCP** cnDBtier API:

```
curl --location --request GET 'http://${cncc-mcore-ingress-external-ip}:$
{cncc-mcore-ingress-service-port}/ocdbtier/version' \
  --header 'oc-cncc-id: Cluster1' \
  --header 'oc-cncc-instance-id: Cluster1-scp-instance1' \
  --header 'Authorization: Bearer <token>'
```

> **Note**
>
> Ensure that the values of `oc-cncc-id` and `oc-cncc-instance-id` reflect the target CNC Console or NF(for example: SCP, NRF etc) instance you are accessing.

6.2 For OCI Deployment

Overview

OCI IAM provides a secure option for direct API access to CNC Console resources by providing an OCI IAM access token.

This section provides details on the following token generation and CNC Console-NF Resource access.

- Access tokens
- Refresh tokens
- NF Resource Access

6.2.1 Access Token Generation Using User Credentials

Access Token Generation Using User Credentials

Acquire the access token from OCI IAM by triggering the following POST request:

```
curl --location --request POST 'https://<oci-iam-domain-url>/oauth2/v1/token'
\
--header 'Content-Type: application/x-www-form-urlencoded' \
--header 'Authorization: Basic <Base64 encoding of client credentials>' \
--data-urlencode 'grant_type=password' \
--data-urlencode 'username=testuser' \
--data-urlencode 'password=••••••' \
--data-urlencode 'scope=urn:opc:idm:__myscopes__ offline_access'
```

The attributes used in this request are:

<table>
<tr><th>Attribute</th><th>Description</th></tr>
<tr><td>&lt;oci-iam-domain-url&gt;</td><td>OCI IAM Domain URL. For more information, see the Identity Access Management section in the <i>Oracle Communication Cloud Native Configuration Console Installation, Upgrade, and Fault Recovery Guide.</i></td></tr>
<tr><td>&lt;Base64 encoding of client credentials&gt;</td><td>
<ol>
<li>For information on getting clientId and clientSecret, see the Access ClientId and ClientSecret for Confidential Application section in <i>Oracle Communication Cloud Native Configuration Console Installation, Upgrade, and Fault Recovery Guide.</i></li>
<li>Base64 encoding of client credentials
<table>
<tr><td>language</td><td>bash</td></tr>
<tr><td>title</td><td>Command to encode client credentials in base64</td></tr>
</table>
Format:
<pre>'clientId:clientSecret'</pre>
Command:
<pre>echo -n 'clientId:clientSecret' |
base64</pre>
For Example:
<pre>echo -n
'asdfxxxxq3rF:Q3r4fsdxxxxxxxfv' |
base64
YXNkZnEzckY6UTNyNGZzZGZ2</pre>
</li>
</ol>
</td></tr>
<tr><td>grant_type</td><td>The way to gets an access token.
<b>Value:</b> password</td></tr>
<tr><td>username</td><td>Username of the OCI IAM user used for login.</td></tr>
<tr><td>password</td><td>Password of the OCI IAM user used for login.</td></tr>
<tr><td>scope</td><td>The way to limit the amount of access that is granted to an access token.
<b>Values:</b>
<ol>
<li><b>urn:opc:idm:__myscopes__</b></li>
<li><b>offline_access</b></li>
</ol></td></tr>
</table>

In response, we'll get an *access_token* and *refresh_token*.

```
{
    "access_token":
"eyJhbGciOiJSUzI1NiIsI................._LcCZYwDQJJTloj2PJ8y1WjO9l2Q",
    "expires_in": 300,
    "token_type": "bearer",
    "refresh_token": 
"AgAgYWEyMzQ5MGM4YTRj.................FEgVm3XXS_y05UzUHIwrdlyQtsc="
}
```

6.2.2 Access Token Generation Using Refresh Token

Access token generation using Refresh Token

Acquire the access token from OCI IAM by triggering the following POST request:

```
curl --location --requestPOST  'https://<oci-iam-domain-url>/oauth2/v1/token' 
\
--header 'Content-Type: application/x-www-form-urlencoded' \
--header 'Authorization: Basic <Base64 encoding of client credentials in the 
format "clientId:clientSceret">' \
--data-urlencode 'grant_type=refresh_token' \
--data-urlencode 'refresh_token=<refresh_token>'
```

The attributes used in this request are these

Attribute	Description
<oci-iam-domain-url>	OCI IAM Domain URL. For information on identity and access management, see the OCI Identity and Access Management section in the *Oracle Communication Cloud Native Configuration Console Installation, Upgrade, and Fault Recovery Guide.*.

<table>
<tr><td>&lt;Base64 encoding of client credentials&gt;</td><td>1. For information on getting clientId and clientSecret, see the Access ClientId and ClientSecret for Confidential Application section in <i>Oracle Communication Cloud Native Configuration Console Installation, Upgrade, and Fault Recovery Guide.</i>
2. Base64 encoding of client credentials
<table><tr><td>language</td><td>bash</td></tr><tr><td>title</td><td>Command to encode client credentials in base64.</td></tr></table>Format:
<code>'clientId:clientSecret'</code>
Command:
<code>echo -n 'clientId:clientSecret' | base64</code>
For example:
<code>echo -n 'asdfxxxxq3rF:Q3r4fsdxxxxxxxfv' | base64</code>
<code>YXNkZnEzckY6UTNyNGZzZGZ2</code></td></tr>
<tr><td>grant_type</td><td>The way to get an access token.
<b>Value</b>: refresh_token</td></tr>
<tr><td>refresh_token</td><td>The actual refresh_token received while generating access_token as part of Access Token generation using User Credentials.</td></tr>
</table>

In response, we'll get an *access_token* and *refresh_token.*

```
{
    "access_token":
"eyJhbGciOiJSUzI1NiIsI................._LcCZYwDQJJTloj2PJ8y1WjO9l2Q",
    "expires_in": 300,
    "token_type": "bearer",
    "refresh_token":
"AgAgYWEyMzQ5MGM4YTRj.................FEgVm3XXS_y05UzUHIwrdlyQtsc="
}
```

Note

By default, access tokens expire after one hour. This expiry period can be changed in the configuration of the trusted application you configured in OCI IAM. Once your access token expires, you will need to refresh it. You can use the refresh token that was provided to you with your access token.

6.2.3 NF Resource Access Through CNC Console

NF Resource Access via CNC Console

Trigger the following request to access NF resource via CNC Console:

```
curl --location --request GET 'http://
<cncc_mcore_igw_url>:<cncc_mcore_igw_port>/<nf_resource_path>' \
--header 'oc-cncc-id: <oc-cncc-id>' \
--header 'oc-cncc-instance-id: <oc-cncc-instance-id>' \
--header 'Authorization: Bearer <oci_iam_access_token>'
```

For example:

```
curl --location --request GET 'http://
<cncc_mcore_igw_url>:<cncc_mcore_igw_port>/ocscp/scpc-configuration/v1/
canaryrelease ' \
--header 'oc-cncc-id: Cluster1' \
--header 'oc-cncc-instance-id: Cluster1-scp-instance1' \
--header 'Authorization: Bearer
eyJhbGciOiJSUzI1NiIsI................._LcCZYwDQJJTloj2PJ8y1WjO9l2Q'
```

The attributes used in this request are these

Attribute	Description
<oci-iam-access-token>	OCI IAM Access token of the OCI IAM User. See Access Token generation using User Credentials.
<cncc_mcore_igw_url>	CNC Console Manager Ingress Gateway URL. See the Accessing M-CNCC Core section in the *Oracle Communication Cloud Native Configuration Console Installation, Upgrade, and Fault Recovery Guide.*
<oc-cncc-id>	Unique M-CNCC ID per site or cluster (**global.self.cnccId**). See the CNC Console Instance Configuration Options section in the *Oracle Communication Cloud Native Configuration Console Installation, Upgrade, and Fault Recovery Guide.*
<oc-cncc-instance-id>	Unique Instance ID of NF per site or cluster (**global.instances**). See the CNC Console Instance Configuration Options section in the *Oracle Communication Cloud Native Configuration Console Installation, Upgrade, and Fault Recovery Guide.*
<nf_resource_path>	Request path to NF resource.

6.2.4 cnDBTier Resource Access Through CNC Console

(Required) <Enter a short description here.>

This section describes the procedure to access cnDBTier resources via the CNC Console using curl or Postman. The authentication process to obtain the access token remains the same as described previously. The same NF instance parameters, **oc-cncc-id** and **oc-cncc-instance-id**, should be used for cnDBTier API access.

> **Note**
>
> To access cnDBTier APIs, users must be assigned the appropriate NF-specific roles. For example, access to CNC Console cnDBTier resources requires the **CNCC_READ**/**CNCC_WRITE** roles, while access to SCP cnDBTier resources requires the **SCP_READ**/**SCP_WRITE** roles.
>
> Similarly, for any other integrated NF, ensure that relevant NF-specific roles are assigned to the user.

Curl command to access cnDBTier API:

```
curl --location --request GET 'http://
<cncc_mcore_igw_url>:<cncc_mcore_igw_port>/<cnDBtier API URL>' \
  --header 'oc-cncc-id: <oc-cncc-id-value>' \
  --header 'oc-cncc-instance-id: <oc-cncc-instance-id-value>' \
  --header 'Authorization: Bearer <oci_iam_access_token>'
```

For example, accessing the **CNCC** cnDBTier API:

```
curl --location --request GET 'http://
<cncc_mcore_igw_url>:<cncc_mcore_igw_port>/ocdbtier/version' \
  --header 'oc-cncc-id: Cluster1' \
  --header 'oc-cncc-instance-id: Cluster1-cncc-instance1' \
  --header 'Authorization: Bearer
eyJhbGciOiJSUzI1NiIsI................._LcCZYwDQJJTloj2PJ8y1WjO9l2Q'
```

For example, accessing the **SCP** cnDBTier API:

```
curl --location --request GET 'http://${cncc-mcore-ingress-external-ip}:$
{cncc-mcore-ingress-service-port}/ocdbtier/version' \
  --header 'oc-cncc-id: Cluster1' \
  --header 'oc-cncc-instance-id: Cluster1-scp-instance1' \
  --header 'Authorization: Bearer
eyJhbGciOiJSUzI1NiIsI................._LcCZYwDQJJTloj2PJ8y1WjO9l2Q'
```

> **Note**
>
> Ensure that the values of **oc-cncc-id** and **oc-cncc-instance-id** reflect the target CNCC or NF(for example: SCP, NRF etc) instance you are accessing.

7

CNC Console Metrics

This section provides the information about CNC Console Metrics.

Table 7-1 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.

> **Note**
>
> Two sample dashboards are provided - one supporting CNE without Prometheus Operator, and one supporting CNE Prometheus HA Operator.
>
> - CNC Console Metric Dashboard file for CNE without Prometheus Operator: occncc_metric_dashboard_<version>.json
> - CNC Console Metric Dashboard file with CNE Prometheus HA Operator: occncc_metric_dashboard_promha_<version>.json

> **Note**
>
> Prometheus HA supported CNE tags and labels are renamed as shown in below table
>
> | CNE without Prometheus Operator | Prometheus HA supported CNE |
> |---|---|
> | kubernetes_namespace | namespace |
> | kubernetes_pod_name | pod |
> | container_name | container |
>
> Consider updating tags mentioned in following sections of the document as suggested above for Prometheus HA supported CNE
>
> - CNC Console IAM Metrics
> - CNC Console Core Metrics
> - CNC Console KPIs
> - CNC Console Alerts

Dimension Description

The following table describes the different types of metric dimensions:

Table 7-2 CNC Console Dimension Descriptions

Dimension	Description	Values
Method	Http method	GET, PUT, POST, DELETE, PATCH
Route_Path	Path predicate that matched the current request	NF specific routes
InstanceIdentifier	Identifier for ingress gateway	cncc-iam_ingressgateway, cncc-mcore_ingressgateway, cncc-acore_ingressgateway
InstanceId	ID for NF Instances	Cluster1-scp-instance1, Cluster1-nrf-instance1
ResourcePath	Http url	
ResourceType	NF type being accessed	SCP, NRF...
UserId	Id of the user	
UserName	Name of the user	

7.1 CNC Console IAM Metrics

This section provides the information about the CNC Console IAM Metrics.

> **Note**
>
> Not applicable for OCI deployment.

7.1.1 M-CNCC IAM Requests

Table 7-3 M-CNCC IAM Requests

Field	Description
Metric Details	Total number of requests received by M-CNCC IAM
Metric Filter	oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*iam_ingressgateway"} **For OCI:** *Not Applicable*
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.1.2 M-CNCC IAM Response

Table 7-4 M-CNCC IAM Response

Field	Description
Metric Details	Total number of responses sent by M-CNCC IAM
Metric Filter	oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingress gateway"} **For OCI:** *Not Applicable*
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.1.3 M-CNCC IAM Success Responses

Table 7-5 M-CNCC IAM Success Responses

Field	Description
Metric Details	Total number of success responses (2xx) for M-CNCC IAM
Metric Filter	oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingress gateway",Status=~"2.*"} **For OCI:** *Not Applicable*
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.1.4 M-CNCC IAM 5xx Responses

Table 7-6 M-CNCC IAM 5xx Responses

Field	Description
Metric Details	Total number of error responses (5xx) for M-CNCC IAM

Table 7-6 (Cont.) M-CNCC IAM 5xx Responses

Field	Description
Metric Filter	oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",Status=~"5.*"} **For OCI:** *Not Applicable*
Dimensions	- Host - Method - Route_Path - InstanceIdentifier - InstanceId - ResourcePath - ResourceType - UserId - UserName
Metric Type	Counter

7.1.5 M-CNCC IAM 4xx Responses

Table 7-7 M-CNCC IAM 4xx Responses

Field	Description
Metric Details	Total number of error responses (4xx) for M-CNCC IAM
Metric Filter	oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",Status=~"4.*"} **For OCI:** *Not Applicable*
Dimensions	- Host - Method - Route_Path - InstanceIdentifier - InstanceId - ResourcePath - ResourceType - UserId - UserName
Metric Type	Counter

7.1.6 M-CNCC IAM Error Responses

Table 7-8 M-CNCC IAM Error Responses

Field	Description
Metric Details	Total number of error responses for M-CNCC IAM
Metric Filter	oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",Status=~"4.*|5.*"} **For OCI:** *Not Applicable*

Table 7-8 (Cont.) M-CNCC IAM Error Responses

Field	Description
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.1.7 M-CNCC IAM Access Token Request

Table 7-9 M-CNCC IAM Access Token Request

Field	Description
Metric Details	Total number of access token requests received for M-CNCC IAM
Metric Filter	oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*iam_ingressgateway",ResourcePath="/cncc/auth/realms/master/protocol/openid-connect/token"} *Group by user*: (CNE without Prometheus Operator) sum by(Status,Method,kubernetes_namespace,ResourceType,UserId,UserName,pod) (oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*iam_ingressgateway",ResourcePath="/cncc/auth/realms/master/protocol/openid-connect/token"}) *Group by user*: (CNE with Prometheus HA Operator) sum by(Status,Method,namespace,ResourceType,UserId,UserName,pod) (oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*iam_ingressgateway",ResourcePath="/cncc/auth/realms/master/protocol/openid-connect/token"}) **For OCI:** *Not Applicable*
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.1.8 M-CNCC IAM Access Token Granted

Table 7-10 M-CNCC IAM Access Token Granted

Field	Description
Metric Details	Total number of access token granted for M-CNCC IAM
Metric Filter	oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",ResourcePath="/cncc/auth/realms/master/protocol/openid-connect/token",Status="200 OK"} *Group by user*: (CNE without Prometheus Operator) sum by(Status,Method,kubernetes_namespace,ResourceType,UserId,UserName,kubernetes_pod_name) (oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",ResourcePath="/cncc/auth/realms/master/protocol/openid-connect/token",Status="200 OK"}) *Group by user*: (CNE with Prometheus HA Operator) sum by(Status,Method,namespace,ResourceType,UserId,UserName,pod) (oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",ResourcePath="/cncc/auth/realms/master/protocol/openid-connect/token",Status="200 OK"}) **For OCI:** *Not Applicable*
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.1.9 M-CNCC IAM Access Token Not Granted

Table 7-11 M-CNCC IAM Access Token Not Granted

Field	Description
Metric Details	Total number of access token not granted for M-CNCC IAM

Table 7-11 (Cont.) M-CNCC IAM Access Token Not Granted

Field	Description
Metric Filter	oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",ResourcePath="/cncc/auth/realms/master/protocol/openid-connect/token",Status=~"4.*|5.*"} *Group by user*: (CNE without Prometheus Operator) sum by(Status,Method,kubernetes_namespace,ResourceType,UserId,UserName,kubernetes_pod_name) (oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",ResourcePath="/cncc/auth/realms/master/protocol/openid-connect/token",Status=~"4.*|5.*"}) *Group by user*: (CNE with Prometheus HA Operator) sum by(Status,Method,namespace,ResourceType,UserId,UserName,pod) (oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",ResourcePath="/cncc/auth/realms/master/protocol/openid-connect/token",Status=~"4.*|5.*"}) **For OCI:** *Not Applicable*
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.1.10 M-CNCC IAM User Login Failure Responses

Table 7-12 M-CNCC IAM User Login Failure Responses

Field	Description
Metric Details	Total number of user login failure at M-CNCC IAM

Table 7-12 (Cont.) M-CNCC IAM User Login Failure Responses

Field	Description
Metric Filter	oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",ResourcePath="/cncc/auth/realms/master/login-actions/authenticate",Status="200 OK"} *Group by user*: (CNE without Prometheus Operator) sum by(Status,Method,kubernetes_namespace,ResourceType,UserId,UserName,kubernetes_pod_name)(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",ResourcePath="/cncc/auth/realms/master/login-actions/authenticate",Status="200 OK"}) *Group by user*: (CNE with Prometheus HA Operator) sum by(Status,Method,namespace,ResourceType,UserId,UserName,pod)(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",ResourcePath="/cncc/auth/realms/master/login-actions/authenticate",Status="200 OK"}) **For OCI:** *Not Applicable*
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.1.11 M-CNCC IAM Ingress Gateway incoming TLS connections

Table 7-13 M-CNCC IAM Ingress Gateway incoming TLS connections

Field	Description
Metric Details	Number of TLS connections received on the M-CNCC IAM Ingress Gateway and their negotiated versions. The version can be TLS 1.2 or TLS 1.3.
Metric Filter	oc_ingressgateway_incoming_tls_connections{InstanceIdentifier=~".*iam_ingressgateway"}
Dimensions	• NegotiatedTLSVersion • Host • Direction • InstanceIdentifier
Metric Type	Gauge

7.1.12 M-CNCC IAM Ingress Gateway Certificate Expiration Seconds

Table 7-14 M-CNCC IAM Ingress Gateway Certificate Expiration Seconds

Field	Description
Metric Details	Time to certificate expiry in epoch seconds.
Metric Filter	security_cert_x509_expiration_seconds{InstanceIdentifier=~".*iam_ingressgateway"}
Dimensions	• app • chart • endpoint • container • namespace • pod • serialNumber • subject • CN (CommonName) • O (Organization) • L (Locality) • ST (State or ProvinceName) • C (CountryName)
Metric Type	Histogram

7.2 M-CNCC Core Metrics

This section provides the information about the M-CNCC Core Metrics:

7.2.1 M-CNCC Core Requests

Table 7-15 M-CNCC Core Requests

Field	Description
Metric Details	Total number of requests received by M-CNCC Core
Metric Filter	oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*mcore_ingressgateway"} **For OCI**: oc_ingressgateway_http_requests_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway"}.sum()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.2.2 M-CNCC Core Responses

Table 7-16   M-CNCC Core Responses

Field	Description
Metric Details	Total number of responses for M-CNCC Core requests
Metric Filter	oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway"} **For OCI**: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*mcore_ingressgateway"}.sum()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.2.3 M-CNCC Core Success Responses

Table 7-17   M-CNCC Core Success Responses

Field	Description
Metric Details	Total number of success responses (2xx) for CNCC Core requests
Metric Filter	oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*core_ingressgateway",Status=~"2.*"} **For OCI**: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*mcore_ingressgateway",Status=~"2*",ResourcePath=~"*",k8Namespace="cncc-ns"}.sum()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.2.4 M-CNCC Core 5xx Responses

Table 7-18 M-CNCC Core 5xx Responses

Field	Description
Metric Details	Total number of error responses (5xx) for M-CNCC Core requests
Metric Filter	oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"5.*"} **For OCI**: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Status=~"5*",ResourcePath=~"*",k8Namespace="cncc-ns"}.sum()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.2.5 M-CNCC Core 4xx Responses

Table 7-19 M-CNCC Core 4xx Responses

Field	Description
Metric Details	Total number of error responses (4xx) for M-CNCC Core requests
Metric Filter	oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"4.*"} **For OCI**: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Status=~"4*",ResourcePath=~"*",k8Namespace="cncc-ns"}.sum()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.2.6 M-CNCC Core Error Responses

Table 7-20 M-CNCC Core Error Responses

Field	Description
Metric Details	Total number of error responses sent for M-CNCC Core requests
Metric Filter	oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"4.*|5.*"} **For OCI**: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Status=~"4*|5*",ResourcePath=~"*",k8Namespace="cncc-ns"}.sum()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.2.7 M-CNCC Core Access Token Request

Table 7-21 M-CNCC Core Access Token Request

Field	Description
Metric Details	Total number of access token requests received for M-CNCC Core
Metric Filter	oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*iam_ingressgateway",ResourcePath="/cncc/auth/realms/cncc/protocol/openid-connect/token"} *Group by user*: (CNE without Prometheus Operator) sum by(Status,Method,kubernetes_namespace,ResourceType,UserId,UserName,kubernetes_pod_name)(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*iam_ingressgateway",ResourcePath="/cncc/auth/realms/cncc/protocol/openid-connect/token"}) *Group by user*: (CNE with Prometheus HA Operator) sum by(Status,Method,namespace,ResourceType,UserId,UserName,pod)(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*iam_ingressgateway",ResourcePath="/cncc/auth/realms/cncc/protocol/openid-connect/token"}) **For OCI:** *Not Applicable*

Table 7-21 (Cont.) M-CNCC Core Access Token Request

Field	Description
Dimensions	- Host - Method - Route_Path - InstanceIdentifier - InstanceId - ResourcePath - ResourceType - UserId - UserName
Metric Type	Counter

7.2.8 M-CNCC Core Access Token Granted Responses

Table 7-22 M-CNCC Core Access Token Granted Responses

Field	Description
Metric Details	Total number of access token granted for M-CNCC Core
Metric Filter	oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",ResourcePath="/cncc/auth/realms/cncc/protocol/openid-connect/token",Status="200 OK"} *Group by user*: (CNE without Prometheus Operator) sum by(Status,Method,kubernetes_namespace,ResourceType,UserId,UserName,kubernetes_pod_name) (oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",ResourcePath="/cncc/auth/realms/cncc/protocol/openid-connect/token",Status="200 OK"}) *Group by user*: (CNE with Prometheus HA Operator) sum by(Status,Method,namespace,ResourceType,UserId,UserName,pod) (oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",ResourcePath="/cncc/auth/realms/cncc/protocol/openid-connect/token",Status="200 OK"}) **For OCI:** *Not Applicable*
Dimensions	- Host - Method - Route_Path - InstanceIdentifier - InstanceId - ResourcePath - ResourceType - UserId - UserName
Metric Type	Counter

7.2.9 M-CNCC Core Access Token Not Granted Responses

Table 7-23 M-CNCC Core Access Token Granted Responses

Field	**Description**
Metric Details	Total number of access token not granted for M-CNCC Core
Metric Filter	oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",ResourcePath="/cncc/auth/realms/cncc/protocol/openid-connect/token",Status=~"4.*|5.*"} *Group by user*: (CNE without Prometheus Operator) sum by(Status,Method,kubernetes_namespace,ResourceType,UserId,UserName,kubernetes_pod_name) (oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",ResourcePath="/cncc/auth/realms/cncc/protocol/openid-connect/token",Status=~"4.*|5.*"}) *Group by user*: (CNE with Prometheus HA Operator) sum by(Status,Method,namespace,ResourceType,UserId,UserName,pod) (oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",ResourcePath="/cncc/auth/realms/cncc/protocol/openid-connect/token",Status=~"4.*|5.*"}) **For OCI:** *Not Applicable*
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.2.10 M-CNCC Core User Login Failure Responses

Table 7-24 M-CNCC Core User Login Failure Responses

Field	**Description**
Metric Details	Total number of user login failure at M-CNCC Core

Table 7-24 (Cont.) M-CNCC Core User Login Failure Responses

Field	Description
Metric Filter	oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",ResourcePath="/cncc/auth/realms/cncc/login-actions/authenticate",Status="200 OK"} *Group by user*: (CNE without Prometheus Operator) sum by(Status,Method,kubernetes_namespace,ResourceType,UserId,UserName,kubernetes_pod_name) (oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",ResourcePath="/cncc/auth/realms/cncc/login-actions/authenticate",Status="200 OK"}) *Group by user*: (CNE with Prometheus HA Operator) sum by(Status,Method,namespace,ResourceType,UserId,UserName,pod) (oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",ResourcePath="/cncc/auth/realms/cncc/login-actions/authenticate",Status="200 OK"}) **For OCI:** *Not Applicable*
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.2.11 M-CNCC Core User Authorization Failure Responses

Table 7-25 M-CNCC Core User Authorization Failure Responses

Field	Description
Metric Details	Total number of authorization failure responses while accessing NF services at M-CNCC Core

Table 7-25 (Cont.) M-CNCC Core User Authorization Failure Responses

Field	Description
Metric Filter	oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status="403 FORBIDDEN", ResourceType!="UNKNOWN"} *Group by user*: (CNE without Prometheus Operator) sum by(Status,Method,kubernetes_namespace,ResourceType,UserId,UserName,kubernetes_pod_name) (oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status="403 FORBIDDEN", ResourceType!="UNKNOWN"}) *Group by user*: (CNE with Prometheus HA Operator) sum by(Status,Method,namespace,ResourceType,UserId,UserName,pod) (oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status="403 FORBIDDEN", ResourceType!="UNKNOWN"}) **For OCI**: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Status="403 FORBIDDEN",ResourceType!="UNKNOWN"}.groupBy(Status,Method,namespace,ResourceType,UserId,UserName,nodeName).sum()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.2.12 M-CNCC Core BSF Requests

Table 7-26 M-CNCC Core BSF Requests

Field	Description
Metric Details	Total number of requests received by M-CNCC Core for BSF

Table 7-26 (Cont.) M-CNCC Core BSF Requests

Field	Description
Metric Filter	oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/bsfapi/.*"} Multiple Route_path: oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/bsfapi/.*|.*/oc-bsf-configuration/.*|.*/bsf/.*"} **For OCI**: oc_ingressgateway_http_requests_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",ResourcePath=~"*bsfapi*|*oc-bsf-configuration*|*bsf*"}.sum()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.2.13 M-CNCC Core BSF Responses

Table 7-27 M-CNCC Core BSF Responses

Field	Description
Metric Details	Total number of responses sent by M-CNCC Core for BSF
Metric Filter	oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status="200 OK",Method="GET",ResourcePath=~".*/bsfapi/.*"} Multiple ResourcePath: oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status="200 OK",Method="GET",ResourcePath=~".*/bsfapi/.*|.*/oc-bsf-configuration/.*|.*/bsf/.*"} **For OCI**: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Status=~"2*",ResourcePath=~"*bsfapi*|*oc-bsf-configuration*|*bsf*",k8Namespace="cncc-ns"}.sum()

Table 7-27 (Cont.) M-CNCC Core BSF Responses

Field	Description
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.2.14 M-CNCC Core DD Requests

Table 7-28 M-CNCC Core DD Requests

Field	Description
Metric Details	Total number of requests received by M-CNCC Core for DD
Metric Filter	oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/ocnadd/.*|.*/ocnaddapi/.*"} **For OCI**: oc_ingressgateway_http_requests_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Route_path=~"*ocnadd*|*ocnaddapi*"}.sum()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.2.15 M-CNCC Core DD Responses

Table 7-29 M-CNCC Core DD Responses

Field	Description
Metric Details	Total number of responses sent by M-CNCC Core for DD

Table 7-29 (Cont.) M-CNCC Core DD Responses

Field	Description
Metric Filter	oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status="200 OK",Method="GET",ResourcePath=~".*/ocnadd/.*|.*/ocnaddapi/.*"} **For OCI**: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Status=~"2*",Route_path=~"*ocnadd*|*ocnaddapi*",k8Namespace="cncc-ns"}.sum()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.2.16 M-CNCC Core PROVGW Requests

Table 7-30 M-CNCC Core PROVGW Requests

Field	Description
Metric Details	Total number of requests received by M-CNCC Core for PROVGW
Metric Filter	oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~\".*provgw-config.*|.*provgw.*\"} **For OCI**: oc_ingressgateway_http_requests_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Route_path=~"*provgw-config*|*provgw*"}.sum()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.2.17 M-CNCC Core PROVGW Responses

Table 7-31 M-CNCC Core PROVGW Responses

Field	Description
Metric Details	Total number of responses sent by CNCC Core for PROVGW
Metric Filter	oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status="200 OK",Method="GET",ResourcePath=~".*provgw-config.*|.*provgw.*"} **For OCI**: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Status=~"2*",Route_path=~"*provgw-config*|*provgw*",k8Namespace="cncc-ns"}.sum()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.2.18 M-CNCC Core NRF Responses

Table 7-32 M-CNCC Core NRF Responses

Field	Description
Metric Details	Total number of responses sent by M-CNCC Core for NRF
Metric Filter	Single Route_path: oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status="200 OK",Method="GET",ResourcePath=~".*/nrf-configuration/v1/.*"} Multiple Route_path: oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status="200 OK",Method="GET",ResourcePath=~".*/nrf-configuration/.*|.*/nrf-state-data/.*|.*/ocnrf-swagger/.*|.*/nrf-status-data/.*|.*/nrf/nf-common-component/.*|.*/nrf-configuration/v1/.*"} **For OCI**: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Status=~"2*",Route_path=~"*nrf-configuration/v1*|*nrf-state-data*|*ocnrf-swagger*|*nrf-status-data*|*nrf/nf-common-component*",k8Namespace="cncc-ns"}.sum()

Table 7-32 (Cont.) M-CNCC Core NRF Responses

Field	Description
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.2.19 M-CNCC Core NRF Requests

Table 7-33 M-CNCC Core NRF Requests

Field	Description
Metric Details	Total number of requests received by M-CNCC Core for NRF
Metric Filter	oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/nrf-configuration/v1/.*"} Multiple Route_path: oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/nrf-configuration/v1/.*|.*/nrf-state-data/.*|.*/ocnrf-swagger/.*|.*/nrf-status-data/.*|.*/nrf/nf-common-component/.*"} **For OCI**: oc_ingressgateway_http_requests_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Route_path=~"*nrf-configuration/v1*|*nrf-state-data*|*ocnrf-swagger*|*nrf-status-data*|*nrf/nf-common-component*"}.sum()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.2.20 M-CNCC Core NSSF Requests

Table 7-34 M-CNCC Core NSSF Requests

Field	Description
Metric Details	Total number of requests received by M-CNCC Core for NSSF

Table 7-34 (Cont.) M-CNCC Core NSSF Requests

Field	Description
Metric Filter	oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/nnssf-configuration/.*|.*/nssf/.*"} **For OCI**: oc_ingressgateway_http_requests_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Route_path=~"*nnssf-configuration*|*nssf*"}.sum()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.2.21 M-CNCC Core NSSF Responses

Table 7-35 M-CNCC Core NSSF Responses

Field	Description
Metric Details	Total number of responses sent by M-CNCC Core for NSSF
Metric Filter	oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status="200 OK",Method="GET",ResourcePath=~".*/nnssf-configuration/.*|.*/nssf/.*"} **For OCI**: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Status=~"2*",Route_path=~"*nnssf-configuration*|*nssf*",k8Namespace="cncc-ns"}.sum()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.2.22 M-CNCC Core POLICY Requests

Table 7-36 M-CNCC Core POLICY Requests

Field	Description
Metric Details	Total number of requests received by M-CNCC Core for POLICY
Metric Filter	Single Route_path: oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/policyapi/.*"} Multiple Route_path: oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/policyapi/.*|.*/oc-cnpolicy-configuration/.*|.*/pcf/.*"} **For OCI**: oc_ingressgateway_http_requests_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",ResourcePath=~"*policyapi*|*oc-cnpolicy-configuration*|*pcf*"}.sum()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.2.23 M-CNCC Core POLICY Responses

Table 7-37 M-CNCC Core POLICY Responses

Field	Description
Metric Details	Total number of responses sent by M-CNCC Core for POLICY
Metric Filter	Single ResourcePath: oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status="200 OK",Method="GET",ResourcePath=~".*/policyapi/.*"} Multiple ResourcePath: oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status="200 OK",Method="GET",ResourcePath=~".*/policyapi/.*|.*/oc-cnpolicy-configuration/.*|.*/pcf/.*"} **For OCI**: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Status=~"2*",ResourcePath=~"*policyapi*|*oc-cnpolicy-configuration*|*pcf*",k8Namespace="cncc-ns"}.sum()

Table 7-37 (Cont.) M-CNCC Core POLICY Responses

Field	Description
Dimensions	• Host • HttpVersion • InstanceIdentifier • Method • Route_path • Scheme • Status • ResourcePath • ResourceType • UserId • UserName • AuthenticationType
Metric Type	Counter

7.2.24 M-CNCC Core SCP Responses

Table 7-38 M-CNCC Core SCP Responses

Field	Description
Metric Details	Total number of responses sent by M-CNCC Core for SCP
Metric Filter	Single Route_path: oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status="200 OK",Method="GET",ResourcePath=~".*/ocscp/.*"} **For OCI**: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Status=~"2*",ResourcePath=~"*ocscp*",k8Namespace="cncc-ns"}.sum()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.2.25 M-CNCC Core SCP Requests

Table 7-39 M-CNCC Core SCP Requests

Field	Description
Metric Details	Total number of requests received by M-CNCC Core for SCP

Table 7-39 (Cont.) M-CNCC Core SCP Requests

Field	Description
Metric Filter	oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/ocscp/.*"} **For OCI**: oc_ingressgateway_http_requests_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",ResourcePath=~"*ocscp*"}.sum()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.2.26 M-CNCC Core SEPP Requests

Table 7-40 M-CNCC Core SEPP Requests

Field	Description
Metric Details	Total number of requests received by M-CNCC Core for SEPP
Metric Filter	oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/sepp-configuration/.*"} Multiple Route_path: oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/sepp-configuration/.*|.*/sepp/.*"} **For OCI**: oc_ingressgateway_http_requests_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Route_path=~"*sepp-configuration*|*sepp*"}.sum()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.2.27 M-CNCC Core SEPP Responses

Table 7-41 M-CNCC Core SEPP Responses

Field	Description
Metric Details	Total number of responses sent by M-CNCC Core for SEPP
Metric Filter	oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status="200 OK",Method="GET",ResourcePath=~".*/sepp-configuration/.*"} Multiple Route_path: oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status="200 OK",Method="GET",ResourcePath=~".*/sepp-configuration/.*|.*/sepp/.*"} **For OCI**: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Status=~"2*",Route_path=~"*sepp-configuration*|*sepp*",k8Namespace="cncc-ns"}.sum()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.2.28 M-CNCC Core UDR Requests

Table 7-42 M-CNCC Core UDR Requests

Field	Description
Metric Details	Total number of requests received by M-CNCC Core for UDR
Metric Filter	Single Route_path: oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/nudr-config/.*"} Multiple Route_path: oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/nudr-dr-prov/.*|.*/nudr-dr-mgm/.*|.*/nudr-group-id-map-prov/.*|.*/slf-group-prov/.*|.*/nudr-config/.*|.*/udr/nf-common-component/.*|.*/n5g-eir-prov/.*"} **For OCI**: oc_ingressgateway_http_requests_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Route_path=~"*nudr-dr-prov*|*nudr-dr-mgm*|*nudr-group-id-map-prov*|*slf-group-prov*|*n5g-eir-prov*|*nudr-config*|*udr/nf-common-component*"}.sum()

Table 7-42 (Cont.) M-CNCC Core UDR Requests

Field	Description
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.2.29 M-CNCC Core UDR Responses

Table 7-43 M-CNCC Core UDR Responses

<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td>Metric Details</td><td>Total number of responses sent by M-CNCC Core for UDR</td></tr>
<tr><td>Metric Filter</td><td>Single ResourcePath:
oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status="200 OK",Method="GET",ResourcePath=~".*/nudr-dr-prov/.*"}
Multiple ResourcePath:
oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status="200 OK",Method="GET",ResourcePath=~".*/nudr-dr-prov/.*|.*/nudr-dr-mgm/.*|.*/nudr-group-id-map-prov/.*|.*/slf-group-prov/.*|.*/nudr-config/.*|.*/udr/nf-common-component/.*|.*/n5g-eir-prov/.*"}
For OCI:
oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Status=~"2*",Route_path=~"*nudr-dr-prov*|*nudr-dr-mgm*|*nudr-group-id-map-prov*|*slf-group-prov*|*n5g-eir-prov*|*nudr-config*|*udr/nf-common-component*",k8Namespace="cncc-ns"}.sum()</td></tr>
<tr><td>Dimensions</td><td>• Host
• Method
• Route_Path
• InstanceIdentifier
• InstanceId
• ResourcePath
• ResourceType
• UserId
• UserName</td></tr>
<tr><td>Metric Type</td><td>Counter</td></tr>
</table>

7.2.30 M-CNCC Core cnDBTier Requests

Table 7-44 M-CNCC Core cnDBTier Requests

Field	Description
Metric Details	Total number of requests received by M-CNCC Core for cnDBTier
Metric Filter	oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~\".*ocdbtier.*\"} **For OCI**: oc_ingressgateway_http_requests_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",ResourcePath=~"*ocdbtier*"}.sum()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.2.31 M-CNCC Core cnDBTier Responses

Table 7-45 M-CNCC Core cnDBTier Responses

Field	Description
Metric Details	Total number of responses sent by M-CNCC Core for cnDBTier
Metric Filter	oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status="200 OK",Method="GET",ResourcePath=~".*ocdbtier.*"} **For OCI**: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",ResourcePath=~"*ocdbtier*"}.sum()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.2.32 M-CNCC Core OCCM Requests

Table 7-46 M-CNCC Core OCCM Requests

Field	Description
Metric Details	Total number of requests received by M-CNCC Core for OCCM
Metric Filter	oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*occm-config.*"} **For OCI**: oc_ingressgateway_http_requests_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Route_path=~"*occm-config*"}.sum()
Dimensions	- Host - Method - Route_Path - InstanceIdentifier - InstanceId - ResourcePath - ResourceType - UserId - UserName
Metric Type	Counter

7.2.33 M-CNCC Core OCCM Responses

Table 7-47 M-CNCC Core OCCM Responses

Field	Description
Metric Details	Total number of responses sent by M-CNCC Core for OCCM
Metric Filter	c_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status="200 OK",Method="GET",ResourcePath=~".*occm-config.*"} **For OCI**: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Status=~"2*",Route_path=~"*occm-config*",k8Namespace="cncc-ns"}.sum()
Dimensions	- Host - Method - Route_Path - InstanceIdentifier - InstanceId - ResourcePath - ResourceType - UserId - UserName
Metric Type	Counter

7.2.34 M-CNCC Core Ingress Gateway incoming TLS connections

Table 7-48 M-CNCC Core Ingress Gateway incoming TLS connections

Field	Description
Metric Details	Number of TLS connections received on the M-CNCC Core Ingress Gateway and their negotiated versions. The version can be TLS 1.2 or TLS 1.3.
Metric Filter	oc_ingressgateway_incoming_tls_connections{InstanceIdentifier=~".*mcore_ingressgateway"}
Dimensions	• NegotiatedTLSVersion • Host • Direction • InstanceIdentifier
Metric Type	Gauge

7.2.35 M-CNCC Core Certificate Expiration Seconds

Table 7-49 M-CNCC Core Ingress Gateway Certificate Expiration Seconds

Field	Description
Metric Details	Time to certificate expiry in epoch seconds.
Metric Filter	security_cert_x509_expiration_seconds{InstanceIdentifier=~".*mcore_ingressgateway"}
Dimensions	• app • chart • endpoint • container • namespace • pod • serialNumber • subject • CN (CommonName) • O (Organization) • L (Locality) • ST (State or ProvinceName) • C (CountryName)
Metric Type	Histogram

7.3 A-CNCC Core Metrics

This section provides the information about the A-CNCC Core Metrics:

7.3.1 A-CNCC Core Requests

Table 7-50 A-CNCC Core Requests

Field	Description
Metric Details	Total number of requests received by A-CNCC Core
Metric Filter	oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*acore_ingressgateway"} **For OCI:** oc_ingressgateway_http_requests_total[10m]{InstanceIdentifier=~"*acore_ingressgateway"}.sum()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.3.2 A-CNCC Core Responses

Table 7-51 A-CNCC Core Responses

Field	Description
Metric Details	Total number of responses for A-CNCC Core requests
Metric Filter	oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway"} **For OCI:** oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway"}.sum()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.3.3 A-CNCC Core Success Responses

Table 7-52 A-CNCC Core Success Responses

Field	Description
Metric Details	Total number of success responses (2xx) for A-CNCC Core requests
Metric Filter	oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*"} **For OCI:** oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Status=~"2*",ResourcePath=~"*",k8Namespace="cncc-ns"}.sum()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.3.4 A-CNCC Core 5xx Responses

Table 7-53 A-CNCC Core 5xx Responses

Field	Description
Metric Details	Total number of error responses (5xx) for A-CNCC Core requests
Metric Filter	oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"5.*"} **For OCI:** oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Status=~"5*",ResourcePath=~"*",k8Namespace="cncc-ns"}.sum()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.3.5 A-CNCC Core 4xx Responses

Table 7-54 A-CNCC Core 4xx Responses

Field	Description
Metric Details	Total number of error responses (4xx) for A-CNCC Core requests
Metric Filter	oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"4.*"} **For OCI:** oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Status=~"4*",ResourcePath=~"*",k8Namespace="cncc-ns"}.sum()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.3.6 A-CNCC Core Error Responses

Table 7-55 A-CNCC Core Error Responses

Field	Description
Metric Details	Total number of error responses sent for A-CNCC Core requests
Metric Filter	oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"4.*|5.*"} **For OCI:** oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Status=~"4*|5*",ResourcePath=~"*",k8Namespace="cncc-ns"}.sum()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.3.7 A-CNCC Core Access Token Request

Table 7-56 A-CNCC Core Access Token Request

Field	Description
Metric Details	Total number of access token requests received for A-CNCC Core
Metric Filter	oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*iam_ingressgateway",ResourcePath="/cncc/auth/realms/cncc/protocol/openid-connect/token"} *Group by user*: (CNE without Prometheus Operator) sum by(Status,Method,kubernetes_namespace,ResourceType,UserId,UserName,kubernetes_pod_name) (oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*iam_ingressgateway",ResourcePath="/cncc/auth/realms/cncc/protocol/openid-connect/token"}) *Group by user*: (CNE with Prometheus HA Operator) sum by(Status,Method,namespace,ResourceType,UserId,UserName,pod) (oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*iam_ingressgateway",ResourcePath="/cncc/auth/realms/cncc/protocol/openid-connect/token"}) **For OCI:** *Not Applicable*
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.3.8 A-CNCC Core Access Token Granted Responses

Table 7-57 A-CNCC Core Access Token Granted Responses

Field	Description
Metric Details	Total number of access token granted for A-CNCC Core

Table 7-57 (Cont.) A-CNCC Core Access Token Granted Responses

Field	Description
Metric Filter	oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",ResourcePath="/cncc/auth/realms/cncc/protocol/openid-connect/token",Status="200 OK"} *Group by user*: (CNE without Prometheus Operator) sum by(Status,Method,kubernetes_namespace,ResourceType,UserId,UserName,kubernetes_pod_name) (oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",ResourcePath="/cncc/auth/realms/cncc/protocol/openid-connect/token",Status="200 OK"}) *Group by user*: (CNE with Prometheus HA Operator) sum by(Status,Method,namespace,ResourceType,UserId,UserName,pod) (oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",ResourcePath="/cncc/auth/realms/cncc/protocol/openid-connect/token",Status="200 OK"}) **For OCI:** *Not Applicable*
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.3.9 A-CNCC Core Access Token Not Granted Responses

Table 7-58 A-CNCC Core Access Token Not Granted Responses

Field	Description
Metric Details	Total number of access token not granted for A-CNCC Core

Table 7-58 (Cont.) A-CNCC Core Access Token Not Granted Responses

Field	Description
Metric Filter	oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",ResourcePath="/cncc/auth/realms/cncc/protocol/openid-connect/token",Status=~"4.*|5.*"} *Group by user*: (CNE without Prometheus Operator) sum by(Status,Method,kubernetes_namespace,ResourceType,UserId,UserName,kubernetes_pod_name) (oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",ResourcePath="/cncc/auth/realms/cncc/protocol/openid-connect/token",Status=~"4.*|5.*"}) *Group by user*: (CNE with Prometheus HA Operator) sum by(Status,Method,namespace,ResourceType,UserId,UserName,pod) (oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",ResourcePath="/cncc/auth/realms/cncc/protocol/openid-connect/token",Status=~"4.*|5.*"}) **For OCI:** *Not Applicable*
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.3.10 A-CNCC Core User Authorization Failure Responses

Table 7-59 A-CNCC Core User Authorization Failure Responses

Field	Description
Metric Details	Total number of authorization failure responses while accessing NF services at A-CNCC Core

Table 7-59 (Cont.) A-CNCC Core User Authorization Failure Responses

Field	Description
Metric Filter	oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status="403 FORBIDDEN", ResourceType!="UNKNOWN"} *Group by user*: (CNE without Prometheus Operator) sum by(Status,Method,kubernetes_namespace,ResourceType,UserId,UserName,kubernetes_pod_name) (oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status="403 FORBIDDEN", ResourceType!="UNKNOWN"}) *Group by user*: (CNE with Prometheus HA Operator) sum by(Status,Method,namespace,ResourceType,UserId,UserName,pod) (oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status="403 FORBIDDEN", ResourceType!="UNKNOWN"}) **For OCI:** oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Status="403 FORBIDDEN",ResourceType!="UNKNOWN"}.groupBy(Status,Method,namespace,ResourceType,UserId,UserName,nodeName).sum()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.3.11 A-CNCC Core SCP Requests

Table 7-60 A-CNCC Core SCP Requests

Field	Description
Metric Details	Total number of requests received by A-CNCC Core for SCP
Metric Filter	oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*/ocscp/.*"} **For OCI:** oc_ingressgateway_http_requests_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",ResourcePath=~"*ocscp*"}.sum()

Table 7-60 (Cont.) A-CNCC Core SCP Requests

Field	Description
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.3.12 A-CNCC Core SCP Responses

Table 7-61 A-CNCC Core SCP Responses

Field	Description
Metric Details	Total number of responses sent by A-CNCC Core for SCP
Metric Filter	Single Route_path: oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status="200 OK",Method="GET",Route_path=~".*/ocscp/.*"} **For OCI:** oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Status=~"2*",ResourcePath=~"*ocscp*",k8Namespace="cncc-ns"}.sum()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.3.13 A-CNCC Core NRF Requests

Table 7-62 A-CNCC Core NRF Requests

Field	Description
Metric Details	Total number of requests received by A-CNCC Core for NRF

Table 7-62 (Cont.) A-CNCC Core NRF Requests

<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td>Metric Filter</td><td>oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*/nrf-configuration/v1/.*"}
Multiple Route_path:
oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*/nrf-configuration/v1/.*|.*/nrf-state-data/.*|.*/ocnrf-swagger/.*|.*/nrf-status-data/.*|.*/nrf/nf-common-component/.*"}
For OCI:
oc_ingressgateway_http_requests_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Route_path=~"*nrf-configuration/v1*|*nrf-state-data*|*ocnrf-swagger*|*nrf-status-data*|*nrf/nf-common-component*"}.sum()</td></tr>
<tr><td>Dimensions</td><td>• Host
• Method
• Route_Path
• InstanceIdentifier
• InstanceId
• ResourcePath
• ResourceType
• UserId
• UserName</td></tr>
<tr><td>Metric Type</td><td>Counter</td></tr>
</table>

7.3.14 A-CNCC Core NRF Responses

Table 7-63 A-CNCC Core NRF Responses

<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td>Metric Details</td><td>Total number of responses sent by A-CNCC Core for NRF</td></tr>
<tr><td>Metric Filter</td><td>Single Route_path:
oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status="200 OK",Method="GET",Route_path=~".*/nrf-configuration/v1/.*"}
Multiple Route_path:
oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status="200 OK",Method="GET",Route_path=~".*/nrf-configuration/.*|.*/nrf-state-data/.*|.*/ocnrf-swagger/.*|.*/nrf-status-data/.*|.*/nrf/nf-common-component/.*|.*/nrf-configuration/v1/.*"}
For OCI:
oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Status=~"2*",Route_path=~"*nrf-configuration/v1*|*nrf-state-data*|*ocnrf-swagger*|*nrf-status-data*|*nrf/nf-common-component*",k8Namespace="cncc-ns"}.sum()</td></tr>
</table>

Table 7-63 (Cont.) A-CNCC Core NRF Responses

Field	Description
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.3.15 A-CNCC Core UDR Requests

Table 7-64 A-CNCC Core UDR Requests

Field	Description
Metric Details	Total number of requests received by A-CNCC Core for UDR
Metric Filter	Single Route_path: oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*/nudr-config/.*"} Multiple Route_path: oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*/nudr-dr-prov/.*|.*/nudr-dr-mgm/.*|.*/nudr-group-id-map-prov/.*|.*/slf-group-prov/.*|.*/nudr-config/.*|.*/udr/nf-common-component/.*|.*/n5g-eir-prov/.*"} **For OCI:** oc_ingressgateway_http_requests_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Route_path=~"*nudr-dr-prov*|*nudr-dr-mgm*|*nudr-group-id-map-prov*|*slf-group-prov*|*n5g-eir-prov*|*nudr-config*|*udr/nf-common-component*"}.sum()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.3.16 A-CNCC Core UDR Responses

Table 7-65 A-CNCC Core UDR Responses

Field	Description
Metric Details	Total number of responses sent by A-CNCC Core for UDR
Metric Filter	Single Route_path: oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status="200 OK",Method="GET",Route_path=~".*/nudr-dr-prov/.*"} Multiple Route_path: oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status="200 OK",Method="GET",Route_path=~".*/nudr-dr-prov/.*|.*/nudr-dr-mgm/.*|.*/nudr-group-id-map-prov/.*|.*/slf-group-prov/.*|.*/nudr-config/.*|.*/udr/nf-common-component/.*|.*/n5g-eir-prov/.*"} **For OCI:** oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Status=~"2*",Route_path=~"*nudr-dr-prov*|*nudr-dr-mgm*|*nudr-group-id-map-prov*|*slf-group-prov*|*n5g-eir-prov*|*nudr-config*|*udr/nf-common-component*",k8Namespace="cncc-ns"}.sum()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.3.17 A-CNCC Core POLICY Requests

Table 7-66 A-CNCC Core POLICY Requests

Field	Description
Metric Details	Total number of requests received by A-CNCC Core for POLICY
Metric Filter	Single Route_path: oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*acore_ingressgateway",ResourcePath=~".*/policyapi/.*"} Multiple Route_path: oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*acore_ingressgateway",ResourcePath=~".*/policyapi/.*|.*/oc-cnpolicy-configuration/.*|.*/pcf/.*"} **For OCI:** oc_ingressgateway_http_requests_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",ResourcePath=~"*policyapi*|*oc-cnpolicy-configuration*|*pcf*"}.sum()

Table 7-66 (Cont.) A-CNCC Core POLICY Requests

Field	Description
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.3.18 A-CNCC Core POLICY Responses

Table 7-67 A-CNCC Core POLICY Responses

Field	Description
Metric Details	Total number of responses sent by A-CNCC Core for POLICY
Metric Filter	Single Route_path: oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status="200 OK",Method="GET",ResourcePath=~".*/policyapi/.*"} Multiple Route_path: oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status="200 OK",Method="GET",ResourcePath=~".*/policyapi/.*|.*/oc-cnpolicy-configuration/.*|.*/pcf/.*"} **For OCI:** oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Status=~"2*",ResourcePath=~"*policyapi*|*oc-cnpolicy-configuration*|*pcf*",k8Namespace="cncc-ns"}.sum()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.3.19 A-CNCC Core BSF Requests

Table 7-68 A-CNCC Core BSF Requests

Field	Description
Metric Details	Total number of requests received by A-CNCC Core for BSF
Metric Filter	oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*acore_ingressgateway",ResourcePath=~".*/bsfapi/.*"} Multiple Route_path: oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*acore_ingressgateway",ResourcePath=~".*/bsfapi/.*|.*/oc-bsf-configuration/.*|.*/bsf/.*"} **For OCI:** oc_ingressgateway_http_requests_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",ResourcePath=~"*bsfapi*|*oc-bsf-configuration*|*bsf*"}.sum()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.3.20 A-CNCC Core BSF Responses

Table 7-69 A-CNCC Core BSF Responses

Field	Description
Metric Details	Total number of responses sent by A-CNCC Core for BSF
Metric Filter	oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status="200 OK",Method="GET",ResourcePath=~".*/bsfapi/.*"} Multiple Route_path: oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status="200 OK",Method="GET",ResourcePath=~".*/bsfapi/.*|.*/oc-bsf-configuration/.*|.*/bsf/.*"} **For OCI:** oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Status=~"2*",ResourcePath=~"*bsfapi*|*oc-bsf-configuration*|*bsf*",k8Namespace="cncc-ns"}.sum()

Table 7-69 (Cont.) A-CNCC Core BSF Responses

Field	Description
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.3.21 A-CNCC Core SEPP Requests

Table 7-70 A-CNCC Core SEPP Requests

Field	Description
Metric Details	Total number of requests received by A-CNCC Core for SEPP
Metric Filter	oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status="200 OK",Method="GET",Route_path=~".*/sepp-configuration/.*"} Multiple Route_path: oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status="200 OK",Method="GET",Route_path=~".*/sepp-configuration/.*|.*/sepp/.*"} **For OCI:** oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Status=~"2*",Route_path=~"*sepp-configuration*|*sepp*",k8Namespace="cncc-ns"}.sum()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.3.22 A-CNCC Core SEPP Responses

Table 7-71 A-CNCC Core SEPP Responses

Field	Description
Metric Details	Total number of responses sent by A-CNCC Core for SEPP

Table 7-71 (Cont.) A-CNCC Core SEPP Responses

Field	Description

Metric Filter	oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status="200 OK",Method="GET",Route_path=~".*/sepp-configuration/.*"}
	Multiple Route_path:
	oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status="200 OK",Method="GET",Route_path=~".*/sepp-configuration/.*|.*/sepp/.*"}

Dimensions	• Host • HttpVersion • InstanceIdentifier • Method • Route_path • Scheme • Status • ResourcePath • ResourceType • UserId • UserName • AuthenticationType
Metric Type	Counter

7.3.23 A-CNCC Core NSSF Requests

Table 7-72 A-CNCC Core NSSF Requests

Field	Description
Metric Details	Total number of requests received by CNCC Core A-for NSSF
Metric Filter	oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*/nnssf-configuration/.*|.*/nssf/.*"} **For OCI:** oc_ingressgateway_http_requests_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Route_path=~"*nnssf-configuration*|*nssf*"}.sum()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.3.24 A-CNCC Core NSSF Responses

Table 7-73 A-CNCC Core NSSF Responses

Field	Description
Metric Details	Total number of responses sent by A-CNCC Core for NSSF
Metric Filter	oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status="200 OK",Method="GET",Route_path=~".*/nnssf-configuration/.*|.*/nssf/.*"} **For OCI:** oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Status=~"2*",Route_path=~"*nnssf-configuration*|*nssf*",k8Namespace="cncc-ns"}.sum()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.3.25 A-CNCC Core DD Requests

Table 7-74 A-CNCC Core DD Requests

Field	Description
Metric Details	Total number of requests received by A-CNCC Core for DD
Metric Filter	oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*/ocnadd/.*|.*/ocnaddapi/.*"} **For OCI:** oc_ingressgateway_http_requests_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Route_path=~"*ocnadd*|*ocnaddapi*"}.sum()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.3.26 A-CNCC Core DD Responses

Table 7-75 A-CNCC Core DD Responses

Field	Description
Metric Details	Total number of responses sent by A-CNCC Core for DD
Metric Filter	oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status="200 OK",Method="GET",Route_path=~".*/ocnadd/.*|.*/ocnaddapi/.*"} **For OCI:** oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Status=~"2*",Route_path=~"*ocnadd*|*ocnaddapi*",k8Namespace="cncc-ns"}.sum()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.3.27 A-CNCC Core PROVGW Requests

Table 7-76 A-CNCC Core PROVGW Requests

Field	Description
Metric Details	Total number of requests received by A-CNCC Core for PROVGW
Metric Filter	oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~\".*provgw-config.*|.*provgw.*\"} **For OCI:** oc_ingressgateway_http_requests_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Route_path=~"*provgw-config*|*provgw*"}.sum()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.3.28 A-CNCC Core PROVGW Responses

Table 7-77 A-CNCC Core PROVGW Responses

Field	Description
Metric Details	Total number of responses sent by A-CNCC Core for PROVGW
Metric Filter	oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status="200 OK",Method="GET",Route_path=~".*provgw-config.*|.*provgw.*"} **For OCI:** oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Status=~"2*",Route_path=~"*provgw-config*|*provgw*",k8Namespace="cncc-ns"}.sum()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.3.29 A-CNCC Core cnDBTier Requests

Table 7-78 A-CNCC Core cnDBTier Requests

Field	Description
Metric Details	Total number of requests received by A-CNCC Core for cnDBTier
Metric Filter	oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~\".*ocdbtier.*\"} **For OCI:** oc_ingressgateway_http_requests_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Route_path=~"ocdbtier"}.sum()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.3.30 A-CNCC Core cnDBTier Responses

Table 7-79 A-CNCC Core cnDBTier Responses

Field	Description
Metric Details	Total number of responses sent by A-CNCC Core for cnDBTier
Metric Filter	oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status="200 OK",Method="GET",Route_path=~".*ocdbtier.*"} **For OCI:** oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Route_path=~"ocdbtier"}.sum()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.3.31 A-CNCC Core OCCM Requests

Table 7-80 A-CNCC Core OCCM Requests

Field	Description
Metric Details	Total number of requests received by A-CNCC Core for OCCM
Metric Filter	oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*occm-config.*"} **For OCI:** oc_ingressgateway_http_requests_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Route_path=~"*occm-config*"}.sum()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.3.32 A-CNCC Core OCCM Responses

Table 7-81 A-CNCC Core OCCM Responses

Field	Description
Metric Details	Total number of responses sent by A-CNCC Core for OCCM
Metric Filter	oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status="200 OK",Method="GET",Route_path=~".*occm-config.*"} **For OCI:** oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Status=~"2*",Route_path=~"*occm-config*",k8Namespace="cncc-ns"}.sum()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.3.33 A-CNCC Core Ingress Gateway incoming TLS connections

Table 7-82 A-CNCC Core Ingress Gateway incoming TLS connections

Field	Description
Metric Details	Number of TLS connections received on the A-CNCC Core Ingress Gateway and their negotiated versions. The version can be TLS 1.2 or TLS 1.3.
Metric Filter	oc_ingressgateway_incoming_tls_connections{InstanceIdentifier=~".*acore_ingressgateway"}
Dimensions	• NegotiatedTLSVersion • Host • Direction • InstanceIdentifier
Metric Type	Gauge

7.3.34 A-CNCC Core Ingress Gateway Certificate Expiration Seconds

Table 7-83 A-CNCC Core Ingress Gateway Certificate Expiration Seconds

Field	Description
Metric Details	Time to certificate expiry in epoch seconds.
Metric Filter	security_cert_x509_expiration_seconds{InstanceIdentifier=~".*acore_ingressgateway"}

Table 7-83 (Cont.) A-CNCC Core Ingress Gateway Certificate Expiration Seconds

Field	Description
Dimensions	• app • chart • endpoint • container • namespace • pod • serialNumber • subject • CN (CommonName) • O (Organization) • L (Locality) • ST (State or ProvinceName) • C (CountryName)
Metric Type	Histogram

7.4 CNC Console Metric Dashboards on OCI

CNC Console provides Dashboard Files for all the NFs, bundled in the CNC Console package. These files can be imported in OCI Console to view the Metrics and related Plots.

> **Note**
>
> You must ensure the user has the required Roles or permissions to view/modify the dashboards. For more information, see the Creating OCI User Management section in *Oracle Communications Cloud Native Core OCI Adaptor, NF Deployment on OCI Guide*

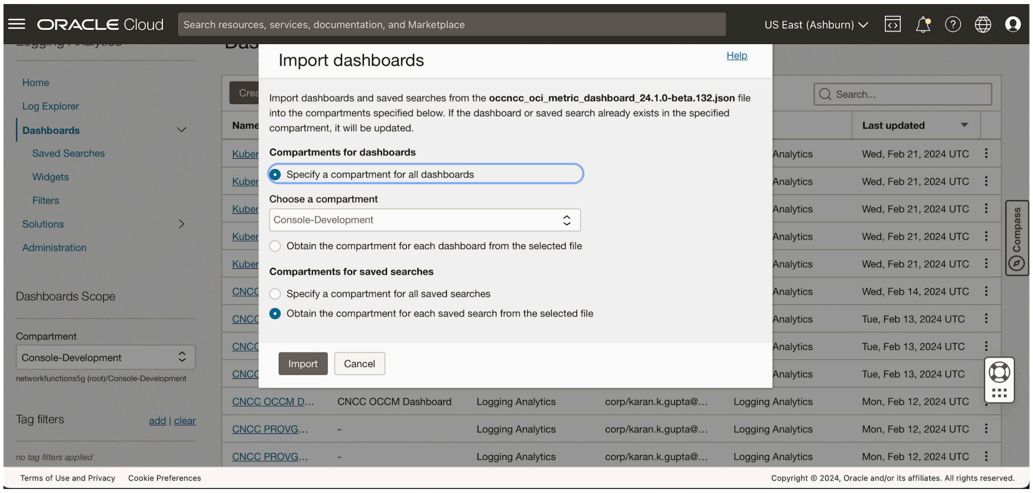
Viewing Dashboards on OCI Console

1. CNC Console CSAR package *occncc_csar_<version>.zip includes dashboard files specific to OCI deployment.* These files are zipped as *occncc_oci_metric_dashboard_<version>.zip and* placed in the Scripts directory of CSAR package.
2. Unzip the *occncc_oci_metric_dashboard_<version>.zip* file to get /occncc_oci_metric_dashboard folder containing a list of Dashboard files
3. The zip file contains one CNC Console dashboard file (occncc_oci_metric_dashboard_<version>.json) along with NF-wise dashboard files.

```
occncc_oci_bsf_metric_dashboard_<version>.json
occncc_oci_dd_metric_dashboard_<version>.json
occncc_oci_metric_dashboard_<version>.json
occncc_oci_nrf_metric_dashboard_<version>.json
occncc_oci_nssf_metric_dashboard_<version>.json
occncc_oci_occm_metric_dashboard_<version>.json
occncc_oci_policy_metric_dashboard_<version>.json
occncc_oci_provgw_metric_dashboard_<version>.json
```

```
occncc_oci_scp_metric_dashboard_<version>.json
occncc_oci_sepp_metric_dashboard_<version>.json
occncc_oci_udr_metric_dashboard_<version>.json
```

4. You must ensure to update the **k8Namespace** field in the provided dashboard files, with the actual namespace which is used for CNC Console deployment. The default value is 'cncc-ns'
5. You must update the metric namespace where the metrics will be populated. Use the same namespace that was created for metrics during the adapter installation. The default value is 'cncc_metrics'
6. Perform the following steps after logging in to OCI Console -
   a. Open the navigation menu and click **Observability and Management**.
   b. Scroll down to find **Dashboards** under **Logging Analytics**
   c. **Figure 7-1 Import Dashboards**



Import dashboards

Help

Import dashboards and saved searches from the **occncc_oci_metric_dashboard_24.1.0-beta.132.json** file into the compartments specified below. If the dashboard or saved search already exists in the specified compartment, it will be updated.

Compartments for dashboards

● Specify a compartment for all dashboards

Choose a compartment

Console-Development

○ Obtain the compartment for each dashboard from the selected file

Compartments for saved searches

○ Specify a compartment for all saved searches

● Obtain the compartment for each saved search from the selected file

Import   Cancel

   Under Dashboards section, click on **Import Dashboard** and upload the Dashboard files (as mentioned in below screenshot) provided in Console package

   d. The user needs to upload the CNC Console dashboard file (occncc_oci_metric_dashboard_<version>.json) to view the CNC Console deployment metric dashboard
   e. For viewing CNC Console NF Dashboards, respective CNC Console NF dashboards files can be uploaded in a similar way

8

CNC Console Alerts

This section provides information about CNC Console Alerts.

> **Note**
>
> - The user must use updated `occncc_agent_alertrules_<version>.yaml` file for agent cluster, in case of multicluster deployment.
> - Use `occncc_manager_alertrules_<version>.yaml` file for single cluster deployment and in the manager cluster, in case of multi cluster deployment.

Table 8-1 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.

8.1 CNC Console IAM Alerts

This section provides information about CNC Console IAM Alerts.

8.1.1 CnccIamTotalIngressTrafficRateAboveMinorThreshold

Table 8-2 CnccIamTotalIngressTrafficRateAboveMinorThreshold

Field	Details
Description	This alert notifies that CNCC IAM Ingress Message rate has crossed the configured minor threshold of 700 to 800 TPS.
Summary	namespace: {{$labels.namespace}}, podname: {{$labels.pod}}, timestamp: {{ with query "time()" }}{{ . | first | value | humanizeTimestamp }}{{ end }}: Traffic Rate is above 70 Percent of Max requests per second(1000)
Severity	minor

Table 8-2 (Cont.) CncclamTotalIngressTrafficRateAboveMinorThreshold

Field	Details
Condition	sum by(namespace,pod) (rate(oc_ingressgateway_http_requests_total{InstanceIdentifer=~".*iam_ingressgateway",namespace="cncc-ns"}[2m])) >= 700 < 800
OID	`1.3.6.1.4.1.323.5.3.51.1.2.7001`
Metric Used	`oc_ingressgateway_http_requests_total`
Recommended Actions	**Cause:** • This alert triggers when the CNC Console IAM Ingress receives more traffic than expected. • The traffic is primarily authentication, authorization, or user management requests. • For example, an Integrated NF or a common service may be sending an unusually high volume of such requests. **Diagnostic Information:** • Monitor ingress traffic to the pod using the KPI Dashboard. • Review `iam-ingress` pod logs for any irregularities or anomalies, especially spikes in authentication, authorization, or user management operations. **Recovery:** The alert is cleared automatically when ingress traffic drops below the minor threshold or exceeds the major threshold. If the alert does not clear: • Check if an Integrated NF or a common service is generating unexpectedly high volumes of authentication, authorization, or user management requests. • Analyze logs and metrics for unusual traffic patterns. • Take action to block or limit any unauthorized or unexpected traffic if necessary. For any assistance, contact My Oracle Support. Make sure to capture `iam-ingress` pod logs and relevant metrics to help Support analyze the issue.

8.1.2 CncclamTotalIngressTrafficRateAboveMajorThreshold

Table 8-3 CncclamTotalIngressTrafficRateAboveMajorThreshold

Field	Details
Description	This alert notifies that the CNCC IAM Ingress message rate has crossed the configured major threshold of 800 to 900 TPS.

Table 8-3 (Cont.) CncclamTotalIngressTrafficRateAboveMajorThreshold

Field	Details
Summary	namespace: {{$labels.namespace}}, podname: {{$labels.pod}}, timestamp: {{ with query "time()" }}{{ . | first | value | humanizeTimestamp }}{{ end }}: Traffic Rate is above 80 Percent of Max requests per second(1000)
Severity	major
Condition	sum by(namespace,pod) (rate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*iam_ingressgateway",namespace="cncc-ns"}[2m])) >= 800 < 900
OID	`1.3.6.1.4.1.323.5.3.51.1.2.7001`
Metric Used	`oc_ingressgateway_http_requests_total`
Recommended Action	**Cause:** • This alert triggers when the CNC Console IAM Ingress receives more traffic than expected. • The traffic primarily consists of authentication, authorization, or user management requests. • For example, an integrated NF or a common service may be sending an unusually high volume of such requests. **Diagnostic Information:** • Monitor ingress traffic to the pod using the KPI Dashboard. • Review the `iam-ingress` pod logs for any irregularities or anomalies, especially spikes in authentication, authorization, or user management operations. **Recovery:**The alert clears automatically when ingress traffic drops below the major threshold or exceeds the critical threshold. If the alert does not clear: • Check if an integrated NF or a common service is generating unexpectedly high volumes of authentication, authorization, or user management requests. • Analyze logs and metrics for unusual traffic patterns. • Take action to block or limit any unauthorized or unexpected traffic, if necessary. For any assistance, contact My Oracle Support. Make sure to capture the `iam-ingress` pod logs and relevant metrics to help Support analyze the issue.

8.1.3 CnccIamTotalIngressTrafficRateAboveCriticalThreshold

Table 8-4 CnccIamTotalIngressTrafficRateAboveCriticalThreshold

Field	Details
Description	This alert notifies that the CNCC IAM Ingress message rate has crossed the configured critical threshold of 900 TPS.
Summary	namespace: {{$labels.namespace}}, podname: {{$labels.pod}}, timestamp: {{ with query "time()" }}{{ . | first | value | humanizeTimestamp }}{{ end }}: Traffic Rate is above 90 Percent of Max requests per second(1000)
Severity	critical
Condition	sum by(namespace,pod) (rate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*iam_ingressgateway",namespace="cncc-ns"}[2m])) >= 900
OID	`1.3.6.1.4.1.323.5.3.51.1.2.7001`
Metric Used	`oc_ingressgateway_http_requests_total`
Recommended Action	**Cause:** • This alert triggers when the CNC Console IAM Ingress receives more traffic than expected. • The traffic primarily consists of authentication, authorization, or user management requests. • For example, an integrated NF or a common service may be sending an unusually high volume of such requests. **Diagnostic Information:** • Monitor ingress traffic to the pod using the KPI Dashboard. • Review the `iam-ingress` pod logs for any irregularities or anomalies, especially spikes in authentication, authorization, or user management operations. **Recovery:**The alert clears automatically when ingress traffic drops below the critical threshold. If the alert does not clear: • Check if an integrated NF or a common service is generating unexpectedly high volumes of authentication, authorization, or user management requests. • Analyze logs and metrics for unusual traffic patterns. • Take action to block or limit any unauthorized or unexpected traffic, if necessary. For any assistance, contact My Oracle Support. Make sure to capture the `iam-ingress` pod logs and relevant metrics to help Support analyze the issue.

8.1.4 CnccIamMemoryUsageCrossedMinorThreshold

Table 8-5 CnccIamMemoryUsageCrossedMinorThreshold

Field	Details
Description	This alert notifies that the CNCC IAM Ingress pod has reached the configured minor threshold (70%) of its memory resource limits.
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	sum by(namespace,pod) (container_memory_usage_bytes{container!="", namespace="cncc-ns", pod=~".*iam-ingress-gateway.*|.*iam-kc.*"}) / sum by(namespace, pod) (kube_pod_container_resource_limits{namespace="cncc-ns",pod=~".*iam-ingress-gateway.*|.*iam-kc.*",resource="memory"}) * 100 >= 70 < 80
OID	`1.3.6.1.4.1.323.5.3.51.1.2.7002`

Metric Used	`container_memory_usage_bytes`
	`kube_pod_container_resource_limits`
	Note: This is a Kubernetes metric used for instance availability monitoring. If the metric is not available, use a similar metric as exposed by the monitoring system.

Table 8-5 (Cont.) CncclamMemoryUsageCrossedMinorThreshold

Field	Details
Recommended Action	**Cause:** - This alert triggers when the CNC Console IAM Ingress pod's memory usage reaches the configured minor threshold (70%) of its resource limits. - Higher memory consumption can result from increased authentication, authorization, or user management requests. For example, if an integrated NF or a common service is generating more traffic than expected. **Diagnostic Information:** - Monitor memory usage metrics (`container_memory_usage_bytes` and `kube_pod_container_resource_limits`) using the KPI Dashboard. - Review the `iam-ingress` pod logs for any irregularities, spikes in memory usage, or high volumes of authentication, authorization, or user management activity. **Recovery:**The alert clears automatically when memory utilization drops below the minor threshold or exceeds the major threshold.If the alert does not clear: - Check if an integrated NF or a common service is causing consistently high levels of authentication, authorization, or user management requests, leading to increased memory usage. - Analyze logs and metrics to identify and address the source. - Take steps to optimize usage or resolve issues as needed. - Also, double-check the resource limits and requests configuration to ensure it is aligned with CNC Console recommendations. For any assistance, contact My Oracle Support. Make sure to capture the `iam-ingress` pod logs and relevant metrics to help Support analyze the issue.

8.1.5 CncclamMemoryUsageCrossedMajorThreshold

Table 8-6 CncclamMemoryUsageCrossedMajorThreshold

Field	Details
Description	This alert notifies that the CNC Console IAM Ingress pod has reached the configured major threshold (80%) of its memory resource limits.
Summary	namespace: {{$labels.namespace}}, podname: {{$labels.pod}}, timestamp: {{ with query "time()" }}{{ . | first | value | humanizeTimestamp }}{{ end }} : Memory Usage of pod exceeded 80% of its limit.

Table 8-6 (Cont.) CncclamMemoryUsageCrossedMajorThreshold

Field	Details
Severity	major
Condition	sum by(namespace,pod) (container_memory_usage_bytes{container!="", namespace="cncc-ns", pod=~".*iam-ingress-gateway.*|.*iam-kc.*"}) / sum by(namespace, pod) (kube_pod_container_resource_limits{namespace ="cncc-ns",pod=~".*iam-ingress-gateway.*|.*iam-kc.*",resource="memory"}) * 100 >= 80 < 90
OID	`1.3.6.1.4.1.323.5.3.51.1.2.7002`

Metric Used	`container_memory_usage_bytes`
	`kube_pod_container_resource_limits`
	Note: This is a Kubernetes metric used for instance availability monitoring. If the metric is not available, use a similar metric as exposed by the monitoring system.

Recommended Action	**Cause:** • This alert triggers when the CNC Console IAM Ingress pod's memory usage reaches the configured major threshold (80%) of its resource limits. • Higher memory consumption can result from increased authentication, authorization, or user management requests. For example, if an integrated NF or a common service is generating more traffic than expected. **Diagnostic Information:** • Monitor memory usage metrics (`container_memory_usage_bytes` and `kube_pod_container_resource_limits`) using the KPI Dashboard. • Review the `iam-ingress` pod logs for any irregularities, spikes in memory usage, or high volumes of authentication, authorization, or user management activity. **Recovery:**The alert clears automatically when memory utilization drops below the major threshold or exceeds the critical threshold. If the alert does not clear: • Check if an integrated NF or a common service is causing consistently high levels of authentication, authorization, or user management requests, leading to increased memory usage. • Analyze logs and metrics to identify and address the source. • Take steps to optimize usage or resolve issues as needed. • Also, double-check the resource limits and requests configuration to ensure it is aligned with CNC Console recommendations. For any assistance, contact My Oracle Support. Make sure to capture the `iam-ingress` pod logs and relevant metrics to help Support analyze the issue.

8.1.6 CnccIamMemoryUsageCrossedCriticalThreshold

Table 8-7 CnccIamMemoryUsageCrossedCriticalThreshold

Field	Details
Description	This alert notifies that the CNC Console IAM Ingress pod has reached the configured critical threshold (90%) of its memory resource limits.
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	sum by(namespace,pod) (container_memory_usage_bytes{container!="", namespace="cncc-ns", pod=~".*iam-ingress-gateway.*|.*iam-kc.*"}) / sum by(namespace, pod) (kube_pod_container_resource_limits{namespace="cncc-ns",pod=~".*iam-ingress-gateway.*|.*iam-kc.*",resource="memory"}) * 100 >= 90
OID	1.3.6.1.4.1.323.5.3.51.1.2.7002

Metric Used	`container_memory_usage_bytes`
	`kube_pod_container_resource_limits`
	Note: This is a Kubernetes metric used for instance availability monitoring. If the metric is not available, use a similar metric as exposed by the monitoring system.

Table 8-7 (Cont.) CncclamMemoryUsageCrossedCriticalThreshold

Field	Details
Recommended Action	**Cause:** • This alert triggers when the CNC Console IAM Ingress pod's memory usage reaches the configured critical threshold (90%) of its resource limits. • Higher memory consumption can result from increased authentication, authorization, or user management requests. For example, if an integrated NF or a common service is generating more traffic than expected. **Diagnostic Information:** • Monitor memory usage metrics (`container_memory_usage_bytes` and `kube_pod_container_resource_limits`) using the KPI Dashboard. • Review the `iam-ingress` pod logs for any irregularities, spikes in memory usage, or high volumes of authentication, authorization, or user management activity. **Recovery:**The alert clears automatically when memory utilization drops below the critical threshold. If the alert does not clear: • Check if an integrated NF or a common service is causing consistently high levels of authentication, authorization, or user management requests, leading to increased memory usage. • Analyze logs and metrics to identify and address the source. • Take steps to optimize usage or resolve issues as needed. • Also, double-check the resource limits and requests configuration to ensure it is aligned with CNC Console recommendations. For any assistance, contact My Oracle Support. Make sure to capture the `iam-ingress` pod logs and relevant metrics to help Support analyze the issue.

8.1.7 CncclamTransactionErrorRateAbove0.1Percent

Table 8-8 CncclamTransactionErrorRateAbove0.1Percent

Field	Details
Description	This alert notifies that the number of CNC Console IAM failed transactions is above 0.1 percent of the total transactions.
Summary	CNC Console IAM transaction Error Rate detected above 0.1 Percent of Total Transactions
Severity	warning

Table 8-8 (Cont.) CncclamTransactionErrorRateAbove0.1Percent

Field	Details
Condition	(sum by(namespace,pod) (rate(oc_ingressgateway_http_responses_total{Status=~"5.*",InstanceIdentifier=~".*iam_ingressgateway",namespace="cncc-ns"}[5m]) or (up * 0)))/ (sum by(namespace,pod) (rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",namespace="cncc-ns"}[5m]))) * 100 >= 0.1 < 1
OID	1.3.6.1.4.1.323.5.3.51.1.2.7003
Metric Used	`oc_ingressgateway_http_responses_total`
Recommended Action	**Cause:**This alert triggers when CNC Console IAM failed transactions exceed 0.1% of total transactions.5xx errors typically indicate server-side issues, such as: • The IAM service or its dependencies (like databases or external services) are down or unreachable. • Internal errors due to misconfiguration in the CNC Console custom_values.yaml file. • Backend processing errors, resource exhaustion (CPU or memory), or temporary overload. • Issues during authentication, authorization, or user management requests. **Diagnostic Information:** • Check the health and status of all IAM pods and their dependencies (e.g., databases, storage, or external services). • Review the `iam-ingress` pod logs for error details and stack traces, especially around the time of the alert. • Monitor service-specific metrics to isolate which operation or endpoint is generating errors. • Review recent configuration changes that may have caused backend errors or instability. **Recovery:**The alert clears automatically when the CNC Console IAM 5xx error rate drops below 0.1% or exceeds the 1% threshold. If the alert does not clear: • Investigate and resolve any backend, database, or resource issues causing 5xx errors. • Roll back recent configuration changes if a misconfiguration is suspected. • If this level of error rate is expected for your workload, note that the threshold is configurable and can be adjusted as per operational requirements. For any assistance, contact My Oracle Support. Make sure to capture the `iam-ingress` pod logs and relevant metrics to help Support analyze the issue.

8.1.8 CnccIamTransactionErrorRateAbove1Percent

Table 8-9 CncclamTransactionErrorRateAbove1Percent

Field	Details
Description	This alert notifies that the number of CNC Console IAM failed transactions is above 1 percent of the total transactions.
Summary	CNC Console IAM transaction Error Rate detected above 1 Percent of Total Transactions
Severity	warning
Condition	(sum by(namespace,pod) (rate(oc_ingressgateway_http_responses_total{Status=~"5.*",InstanceIdentifier=~".*iam_ingressgateway",namespace="cncc-ns"}[5m]) or (up * 0)))/ (sum by(namespace,pod) (rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",namespace="cncc-ns"}[5m]))) * 100 >= 1 < 10
OID	1.3.6.1.4.1.323.5.3.51.1.2.7003
Metric Used	`oc_ingressgateway_http_responses_total`

Table 8-9 (Cont.) CncclamTransactionErrorRateAbove1Percent

Field	Details
Recommended Action	**Cause:**This alert triggers when CNC Console IAM failed transactions exceed 1% of total transactions.5xx errors typically indicate server-side issues, such as: • The IAM service or its dependencies (like databases or external services) are down or unreachable. • Internal errors due to misconfiguration in the CNC Console custom_values.yaml file. • Backend processing errors, resource exhaustion (CPU or memory), or temporary overload. • Issues during authentication, authorization, or user management requests. **Diagnostic Information:** • Check the health and status of all IAM pods and their dependencies (e.g, databases, storage, or external services). • Review the `iam-ingress` pod logs for error details and stack traces, especially around the time of the alert. • Monitor service-specific metrics to isolate which operation or endpoint is generating errors. • Review recent configuration changes that may have caused backend errors or instability. **Recovery:**The alert clears automatically when the CNC Console IAM 5xx error rate drops below 1% or exceeds the 10% threshold. If the alert does not clear: • Investigate and resolve any backend, database, or resource issues causing 5xx errors. • Roll back recent configuration changes if a misconfiguration is suspected. • If this level of error rate is expected for your workload, note that the threshold is configurable and can be adjusted as per operational requirements. For any assistance, contact My Oracle Support. Make sure to capture the `iam-ingress` pod logs and relevant metrics to help Support analyze the issue.

8.1.9 CncclamTransactionErrorRateAbove10Percent

Table 8-10 CncclamTransactionErrorRateAbove10Percent

Field	Details
Description	This alert notifies that the number of CNC Console IAM failed transactions is above 10 percent of the total transactions.

Table 8-10 (Cont.) CncclamTransactionErrorRateAbove10Percent

Field	Details
Summary	CNC Console IAM transaction error rate detected above 10 percent of total transactions
Severity	minor
Condition	(sum by(namespace,pod) (rate(oc_ingressgateway_http_responses_total{Status=~"5.*",InstanceIdentifier=~".*iam_ingressgateway",namespace="cncc-ns"}[5m]) or (up * 0)))/ (sum by(namespace,pod) (rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",namespace="cncc-ns"}[5m]))) * 100 >= 10 < 25
OID	1.3.6.1.4.1.323.5.3.51.1.2.7003
Metric Used	`oc_ingressgateway_http_responses_total`

Table 8-10 (Cont.) CncclamTransactionErrorRateAbove10Percent

Field	Details
Recommended Action	**Cause:**This alert triggers when CNC Console IAM failed transactions exceed 10% of total transactions.5xx errors typically indicate server-side issues, such as: • The IAM service or its dependencies (like databases or external services) are down or unreachable. • Internal errors due to misconfiguration in the CNC Console custom_values.yaml file. • Backend processing errors, resource exhaustion (CPU or memory), or temporary overload. • Issues during authentication, authorization, or user management requests. **Diagnostic Information:** • Check the health and status of all IAM pods and their dependencies (e.g, databases, storage, or external services). • Review the `iam-ingress` pod logs for error details and stack traces, especially around the time of the alert. • Monitor service-specific metrics to isolate which operation or endpoint is generating errors. • Review recent configuration changes that may have caused backend errors or instability. **Recovery:**The alert clears automatically when the CNC Console IAM 5xx error rate drops below 10% or exceeds the 25% threshold. If the alert does not clear: • Investigate and resolve any backend, database, or resource issues causing 5xx errors. • Roll back recent configuration changes if a misconfiguration is suspected. • If this level of error rate is expected for your workload, note that the threshold is configurable and can be adjusted as per operational requirements. For any assistance, contact My Oracle Support. Make sure to capture the `iam-ingress` pod logs and relevant metrics to help Support analyze the issue.

8.1.10 CncclamTransactionErrorRateAbove25Percent

Table 8-11 CncclamTransactionErrorRateAbove25Percent

Field	Details
Description	This alert notifies that the number of CNCC IAM failed transactions is above 25 percent of the total transactions.

Table 8-11 (Cont.) CncclamTransactionErrorRateAbove25Percent

Field	Details
Summary	CNCC IAM transaction Error Rate detected above 25 Percent of Total Transactions
Severity	major
Condition	(sum by(namespace,pod) (rate(oc_ingressgateway_http_responses_total{Status=~"5.*",InstanceIdentifier=~".*iam_ingressgateway",namespace="cncc-ns"}[5m]) or (up * 0)))/ (sum by(namespace,pod) (rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",namespace="cncc-ns"}[5m]))) * 100 >= 25 < 50
OID	1.3.6.1.4.1.323.5.3.51.1.2.7003
Metric Used	`oc_ingressgateway_http_responses_total`

Table 8-11 (Cont.) CncclamTransactionErrorRateAbove25Percent

Field	Details
Recommended Action	**Cause:**This alert triggers when CNCC IAM failed transactions exceed 25% of total transactions.5xx errors typically indicate server-side issues, such as: • The IAM service or its dependencies (like databases or external services) are down or unreachable. • Internal errors due to misconfiguration in the CNC Console custom_values.yaml file. • Backend processing errors, resource exhaustion (CPU or memory), or temporary overload. • Issues during authentication, authorization, or user management requests. **Diagnostic Information:** • Check the health and status of all IAM pods and their dependencies (e.g, databases, storage, or external services). • Review the `iam-ingress` pod logs for error details and stack traces, especially around the time of the alert. • Monitor service-specific metrics to isolate which operation or endpoint is generating errors. • Review recent configuration changes that may have caused backend errors or instability. **Recovery:**The alert clears automatically when the CNCC IAM 5xx error rate drops below 25% or exceeds the 50% threshold. If the alert does not clear: • Investigate and resolve any backend, database, or resource issues causing 5xx errors. • Roll back recent configuration changes if a misconfiguration is suspected. • If this level of error rate is expected for your workload, note that the threshold is configurable and can be adjusted as per operational requirements. For any assistance, contact My Oracle Support. Make sure to capture the `iam-ingress` pod logs and relevant metrics to help Support analyze the issue.

8.1.11 CncclamTransactionErrorRateAbove50Percent

Table 8-12 CncclamTransactionErrorRateAbove50Percent

Field	Details
Description	This alert notifies that the number of CNCC IAM failed transactions is above 50 percent of the total transactions.

Table 8-12   (Cont.) CncclamTransactionErrorRateAbove50Percent

Field	Details
Summary	CNCC IAM transaction Error Rate detected above 50 Percent of Total Transactions
Severity	critical
Condition	(sum by(namespace,pod) (rate(oc_ingressgateway_http_responses_total{Status=~"5.*",InstanceIdentifier=~".*iam_ingressgateway",namespace="cncc-ns"}[5m]) or (up * 0)))/ (sum by(namespace,pod) (rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",namespace="cncc-ns"}[5m]))) * 100 >= 50
OID	1.3.6.1.4.1.323.5.3.51.1.2.7003
Metric Used	`oc_ingressgateway_http_responses_total`

Table 8-12 (Cont.) CncclamTransactionErrorRateAbove50Percent

Field	Details
Recommended Action	**Cause:**This alert triggers when CNCC IAM failed transactions exceed 50% of total transactions.5xx errors typically indicate server-side issues, such as: • The IAM service or its dependencies (like databases or external services) are down or unreachable. • Internal errors due to misconfiguration in the CNC Console custom_values.yaml file. • Backend processing errors, resource exhaustion (CPU or memory), or temporary overload. • Issues during authentication, authorization, or user management requests. **Diagnostic Information:** • Check the health and status of all IAM pods and their dependencies (e.g, databases, storage, or external services). • Review the `iam-ingress` pod logs for error details and stack traces, especially around the time of the alert. • Monitor service-specific metrics to isolate which operation or endpoint is generating errors. • Review recent configuration changes that may have caused backend errors or instability. **Recovery:**The alert clears automatically when the CNCC IAM 5xx error rate drops below 50%.If the alert does not clear: • Investigate and resolve any backend, database, or resource issues causing 5xx errors. • Roll back recent configuration changes if a misconfiguration is suspected. • If this level of error rate is expected for your workload, note that the threshold is configurable and can be adjusted as per operational requirements. For any assistance, contact My Oracle Support. Make sure to capture the `iam-ingress` pod logs and relevant metrics to help Support analyze the issue.

8.1.12 CncclamIngressGatewayServiceDown

Table 8-13 CncclamIngressGatewayServiceDown

Field	Details
Description	This alert notifies that the CNCC IAM Ingress Gateway pod is down.

Table 8-13 (Cont.) CncclamIngressGatewayServiceDown

Field	Details
Summary	namespace: {{$labels.namespace}}, timestamp: {{ with query "time()" }}{{ . | first | value | humanizeTimestamp }}{{ end }} : cncc-iam-ingress-gateway service down
Severity	critical
Condition	absent(up{pod=~".*iam-ingress-gateway.*", namespace="cncc-ns"}) or (up{pod=~".*iam-ingress-gateway.*", namespace="cncc-ns"}) == 0
OID	1.3.6.1.4.1.323.5.3.51.1.2.7004
Metric Used	`up` **Note**: This is a Prometheus metric used for instance availability monitoring. If this metric is not available, use a similar metric as exposed by the monitoring system.
Recommended Action	**Cause:**This alert triggers when the CNCC IAM Ingress Gateway pod or service is down. **Diagnostic Information:** • Check the orchestration platform (e.g, Kubernetes) logs for the `cncc-iam-ingress-gateway` pod to identify liveness or readiness probe failures. • Review application logs for the `cncc-iam-ingress-gateway` service, filtering for error or warning messages, or recent crash loops. • Verify recent configuration or deployment changes that might have impacted pod availability. • Check for resource issues (CPU, memory, disk) or dependency/service failures. **Recovery:**The alert clears automatically when the `cncc-iam-ingress-gateway` service becomes available again. If the alert does not clear: • Continue to review logs, resource allocations, and configuration for possible causes of downtime. • Address any identified issues to restore service availability. • If this downtime is expected (e.g, for planned maintenance), you may adjust the alerting threshold as needed. For any assistance, contact My Oracle Support. Make sure to capture pod logs, orchestration event logs, and relevant metrics to help Support analyze the issue.

8.1.13 CnccIamFailedLogin

Table 8-14 CnccIamFailedLogin

Field	Details
Description	This alert notifies you if there are more than 3 failed login attempts in CNCC IAM for a user within 5 minutes.
Summary	namespace: {{$labels.namespace}}, podname: {{$labels.pod}}, timestamp: {{ with query "time()" }}{{ . | first | value | humanizeTimestamp }}{{ end }}: failed login attempts are more than the configured threshold value
Severity	warning
Condition	sum by(Status,namespace,ResourcePath,Method,UserName,pod) (oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",namespace="cncc-ns",ResourcePath="/cncc/auth/realms/master/login-actions/authenticate",Method="POST",Status="200 OK"})- sum by(Status,namespace,ResourcePath,Method,UserName,pod) (oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",namespace="cncc-ns",ResourcePath="/cncc/auth/realms/master/login-actions/authenticate",Method="POST",Status="200 OK"} offset 5m) > 3
OID	1.3.6.1.4.1.323.5.3.51.1.2.7005
Metric Used	`oc_ingressgateway_http_responses_total`

Table 8-14 (Cont.) CncclamFailedLogin

Field	Details
Recommended Action	**Cause:**This alert triggers when there are more than 3 failed login attempts in CNCC IAM for a user within 5 minutes. This may be due to users entering incorrect credentials, potential automated attacks (such as brute force attempts), or issues with the login process. **Diagnostic Information:** • Verify if the affected user(s) are entering the correct username and password. • Review login logs, the `iam-kc` container, and iam-ingress pod logs for patterns, such as repeated failures from the same source. • Check for potential account lockouts or signs of automated login attempts. • Confirm there have not been any recent configuration changes that could impact the authentication process. **Recovery:**The alert is cleared automatically when the number of failed login attempts for a user falls below the configured threshold (default is 3) within the last 5 minutes. If the alert does not clear: • Investigate for possible brute-force activity, misconfigurations, or issues causing repeated login failures. • Verify the username being used and, if an incorrect password is suspected, reset the password and attempt to log in again. • If this level of failed attempts is expected for your use case, note that the threshold is configurable and can be adjusted as needed. For any assistance, contact My Oracle Support. Make sure to capture the `iam-kc` container, `iam-ingress` pod logs, and relevant metrics to help Support analyze the issue.

8.1.14 AdminUserCreation

Table 8-15 AdminUserCreation

Field	Details
Description	This alert notifies you when a new admin user is created in CNCC IAM within the last 5 minutes.
Summary	namespace: {{$labels.namespace}}, podname: {{$labels.pod}}, user: {{$labels.UserName}}, timestamp: {{ with query "time()" }}{{ . | first | value | humanizeTimestamp }}{{ end }}: Admin users have been created
Severity	warning

Table 8-15 (Cont.) AdminUserCreation

Field	Details
Condition	sum by(namespace,ResourcePath,Method,UserName,pod) (oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*iam_ingressgateway",namespace="cncc-ns",ResourcePath="/cncc/auth/admin/realms/master/users",Method="POST"}) - sum by(namespace,ResourcePath,Method,UserName,pod) (oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*iam_ingressgateway",namespace="cncc-ns",ResourcePath="/cncc/auth/admin/realms/master/users",Method="POST"} offset 5m) > 0
OID	1.3.6.1.4.1.323.5.3.51.1.2.7006
Metric Used	`oc_ingressgateway_http_requests_total`
Recommended Action	**Cause:**This alert triggers when a new admin user is created in CNCC IAM within the last 5 minutes. This may be the result of a legitimate administrative action or an unauthorized attempt to gain privileged access. **Diagnostic Information:** • Verify whether the creation of the new admin user was authorized and expected. • Review the `iam-kc` container, iam-ingress pod logs, and audit logs for information about who initiated the action, including source IP and request details. • Monitor for any unusual or suspicious activity related to admin user creation. **Recovery:**The alert is cleared automatically after the admin user creation event is processed, or when subsequent checks do not detect any new admin user creation within the last 5 minutes. • Log in to the admin GUI and review the details of the newly created admin user. • If the user is legitimate, no further action is required. If the creation was unauthorized, take appropriate steps to delete or disable the admin account, and investigate further for potential security issues. • Update policies or controls, if necessary, to prevent unauthorized admin user creation in the future. For any assistance, contact My Oracle Support. Make sure to capture the `iam-kc` container, `iam-ingress` pod logs, and relevant audit details to help Support analyze the issue.

8.1.15 CncclamAccessTokenFailure

Table 8-16 CncclamAccessTokenFailure

Field	Details
Description	This alert notifies you if there are more than 3 failed access token requests in CNCC IAM within 5 minutes.
Summary	namespace: {{$labels.namespace}}, podname: {{$labels.pod}}, timestamp: {{ with query "time()" }}{{ . | first | value | humanizeTimestamp }}{{ end }}: Access Token Failure count is above the configured threshold value
Severity	warning
Condition	sum by(Status,namespace,ResourcePath,Method,UserName,UserId,pod) (oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",namespace="cncc-ns",ResourcePath="/cncc/auth/realms/master/protocol/openid-connect/token",Method="POST",Status=~"4.*|5.*"}) - sum by(Status,namespace,ResourcePath,Method,UserName,pod) (oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",namespace="cncc-ns",ResourcePath="/cncc/auth/realms/master/protocol/openid-connect/token",Method="POST",Status=~"4.*|5.*"} offset 5m) > 3
OID	1.3.6.1.4.1.323.5.3.51.1.2.7007
Metric Used	`oc_ingressgateway_http_responses_total`

Table 8-16   (Cont.) CncclamAccessTokenFailure

Field	Details
Recommended Action	**Cause:**This alert triggers when there are more than 3 failed access token requests in CNCC IAM within 5 minutes. This may be due to incorrect user credentials, misconfigured client applications, or backend errors affecting the token generation process. **Diagnostic Information:** • Check if users or client applications are sending correct credentials and valid request parameters. • Review the iam-kc container and iam-ingress pod logs for error messages related to access token requests. • Verify if there are recent changes or configuration issues impacting the token endpoint. • Look for patterns such as repeated failures from a specific user or application. **Recovery:**The alert clears automatically when the number of failed access token requests for a user drops below the configured threshold (default: 3) within the last 5 minutes. If the alert does not clear: • Verify the credentials and parameters being used for the access token request. • If the failure is caused by expired or invalid credentials, advise users to refresh or reset their credentials and attempt to obtain a new token. • If this level of failed requests is expected, note that the threshold is configurable and can be adjusted as needed. For any assistance, contact My Oracle Support. Make sure to capture the `iam-kc` container, `iam-ingress` pod logs, and relevant metrics to help Support analyze the issue.

8.2 CNC Console Core Alerts

This section provides the information about CNC Console Core Alerts.

8.2.1 CnccCoreTotalIngressTrafficRateAboveMinorThreshold

Table 8-17   CnccCoreTotalIngressTrafficRateAboveMinorThreshold

Field	Details
Description	This alert notifies that the CNCC Core Ingress message rate has crossed the configured minor threshold of 700 to 800 TPS.

Table 8-17 (Cont.) CnccCoreTotalIngressTrafficRateAboveMinorThreshold

Field	Details
Summary	namespace: {{$labels.namespace}}, podname: {{$labels.pod}}, timestamp: {{ with query "time()" }}{{ . | first | value | humanizeTimestamp }}{{ end }}: Traffic Rate is above 70 Percent of Max requests per second(1000)
Severity	minor
Condition	sum by(namespace,pod) (rate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*core_ingressgateway",namespace="cncc-ns"}[2m])) >= 700 < 800
OID	1.3.6.1.4.1.323.5.3.51.1.2.8001
Metric Used	`oc_ingressgateway_http_requests_total`
Recommended Action	**Cause:** • This alert triggers when the CNCC Core Ingress receives more traffic than expected, primarily consisting of NF resource requests or common service requests. • For example, an integrated NF or a common service may be sending an unusually high volume of requests to the CNCC Core. **Diagnostic Information:** • Monitor ingress traffic to the pod using the KPI Dashboard. • Review the core-ingress pod logs for any irregularities or anomalies, especially spikes in NF resource or common service operations. **Recovery:** The alert clears automatically when ingress traffic drops below the minor threshold or exceeds the major threshold. If the alert does not clear: • Check if an integrated NF or a common service is generating unexpectedly high volumes of traffic. • Analyze logs and metrics for unusual patterns or possible misconfigurations leading to increased traffic. • Take action, as needed, to block or limit any unauthorized or unexpected traffic. For any assistance, contact My Oracle Support. Make sure to capture the `core-ingress` pod logs and relevant metrics to help Support analyze the issue.

8.2.2 CnccCoreTotalIngressTrafficRateAboveMajorThreshold

Table 8-18 CnccCoreTotalIngressTrafficRateAboveMajorThreshold

Field	Details
Description	This alert notifies that the CNCC Core Ingress message rate has crossed the configured major threshold of 800 to 900 TPS.
Summary	namespace: {{$labels.namespace}}, podname: {{$labels.pod}}, timestamp: {{ with query "time()" }}{{ . | first | value | humanizeTimestamp }}{{ end }}: Traffic Rate is above 80 Percent of Max requests per second(1000)
Severity	major
Condition	sum by(namespace,pod) (rate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*core_ingressgateway",namespace="cncc-ns"}[2m])) >= 800 < 900
OID	1.3.6.1.4.1.323.5.3.51.1.2.8001
Metric Used	`oc_ingressgateway_http_requests_total`
Recommended Action	**Cause:** • This alert triggers when the CNCC Core Ingress receives more traffic than expected, primarily consisting of NF resource requests or common service requests. • For example, an integrated NF or a common service may be sending an unusually high volume of requests to the CNCC Core. **Diagnostic Information:** • Monitor ingress traffic to the pod using the KPI Dashboard. • Review the core-ingress pod logs for any irregularities or anomalies, especially spikes in NF resource or common service operations. **Recovery:** The alert clears automatically when ingress traffic drops below the major threshold or exceeds the critical threshold. If the alert does not clear: • Check if an integrated NF or a common service is generating unexpectedly high volumes of traffic. • Analyze logs and metrics for unusual patterns or possible misconfigurations leading to increased traffic. • Take action, as needed, to block or limit any unauthorized or unexpected traffic. For any assistance, contact My Oracle Support. Make sure to capture the `core-ingress` pod logs and relevant metrics to help Support analyze the issue.

8.2.3 CnccCoreTotalIngressTrafficRateAboveCriticalThreshold

Table 8-19 CnccCoreTotalIngressTrafficRateAboveCriticalThreshold

Field	Details
Description	This alert notifies that the CNCC Core Ingress message rate has crossed the configured critical threshold of 900 TPS.
Summary	namespace: {{$labels.namespace}}, podname: {{$labels.pod}}, timestamp: {{ with query "time()" }}{{ . | first | value | humanizeTimestamp }}{{ end }}: Traffic Rate is above 90 Percent of Max requests per second(1000)
Severity	critical
Condition	sum by(namespace,pod)(rate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*core_ingressgateway",namespace="cncc-ns"}[2m])) >= 900
OID	1.3.6.1.4.1.323.5.3.51.1.2.8001
Metric Used	`oc_ingressgateway_http_requests_total`
Recommended Action	**Cause:** - This alert triggers when the CNCC Core Ingress receives more traffic than expected, primarily consisting of NF resource requests or common service requests. - For example, an integrated NF or a common service may be sending an unusually high volume of requests to the CNCC Core. **Diagnostic Information:** - Monitor ingress traffic to the pod using the KPI Dashboard. - Review the core-ingress pod logs for any irregularities or anomalies, especially spikes in NF resource or common service operations. **Recovery:** The alert clears automatically when ingress traffic drops below the critical threshold. If the alert does not clear: - Check if an integrated NF or a common service is generating unexpectedly high volumes of traffic. - Analyze logs and metrics for unusual patterns or possible misconfigurations leading to increased traffic. - Take action, as needed, to block or limit any unauthorized or unexpected traffic. For any assistance, contact My Oracle Support. Make sure to capture the core-ingress pod logs and relevant metrics to help Support analyze the issue.

8.2.4 CnccCoreMemoryUsageCrossedMinorThreshold

Table 8-20 CnccCoreMemoryUsageCrossedMinorThreshold

Field	Details
Description	This alert notifies that the CNCC Core Ingress pod has reached the configured minor threshold (70%) of its memory resource limits.
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	sum by(namespace,pod) (container_memory_usage_bytes{container!="", namespace="cncc-ns", pod=~".*core-cmservice.*|.*core-ingress-gateway.*"}) / sum by(namespace, pod) (kube_pod_container_resource_limits{namespace="cncc-ns",pod=~".*core-cmservice.*|.*core-ingress-gateway.*",resource="memory"}) * 100 >= 70 < 80
OID	1.3.6.1.4.1.323.5.3.51.1.2.8002

Metric Used	`container_memory_usage_bytes`
	`kube_pod_container_resource_limits`
	Note: This is a Kubernetes metric used for instance availability monitoring. If the metric is not available, use a similar metric as exposed by the monitoring system.

Table 8-20 (Cont.) CnccCoreMemoryUsageCrossedMinorThreshold

Field	Details
Recommended Action	**Cause:** - This alert triggers when the CNCC Core Ingress pod's memory usage reaches the configured minor threshold (70%) of its resource limits. - Higher memory consumption can result from increased NF resource or common service requests. For example, if a network function or common service is generating more traffic than expected. **Diagnostic Information:** - Monitor memory usage metrics (`container_memory_usage_bytes` and `kube_pod_container_resource_limits`) using the KPI Dashboard. - Review the `core-ingress` pod logs for any irregularities, spikes in memory usage, or high volumes of NF resource or common service activity. **Recovery:** The alert clears automatically when memory utilization drops below the minor threshold or exceeds the major threshold. If the alert does not clear: - Check if any network function or common service is causing consistently high resource requests, leading to increased memory usage. - Analyze logs and metrics to identify and address the source of high usage. - Take steps to optimize memory usage or resolve any configuration issues. - Also, double-check the resource limits and requests configuration to ensure it is aligned with CNCC Console recommendations. For any assistance, contact My Oracle Support. Make sure to capture the core-ingress pod logs and relevant metrics to help Support analyze the issue.

8.2.5 CnccCoreMemoryUsageCrossedMajorThreshold

Table 8-21 CnccCoreMemoryUsageCrossedMajorThreshold

Field	Details
Description	This alert notifies that the CNCC Core Ingress pod has reached the configured major threshold (80%) of its memory resource limits.
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

Table 8-21 (Cont.) CnccCoreMemoryUsageCrossedMajorThreshold

Field	Details
Condition	sum by(namespace,pod) (container_memory_usage_bytes{container!="", namespace="cncc-ns", pod=~".*core-cmservice.*|.*core-ingress-gateway.*"}) / sum by(namespace, pod) (kube_pod_container_resource_limits{namespace ="cncc-ns",pod=~".*core-cmservice.*|.*core-ingress-gateway.*",resource="memory"}) * 100 >= 80 < 90
OID	1.3.6.1.4.1.323.5.3.51.1.2.8002

Metric Used	`container_memory_usage_bytes`
	`kube_pod_container_resource_limits`
	Note: This is a Kubernetes metric used for instance availability monitoring. If the metric is not available, use a similar metric as exposed by the monitoring system.

Recommended Action	**Cause:** • This alert triggers when the CNCC Core Ingress pod's memory usage reaches the configured major threshold (80%) of its resource limits. • Higher memory consumption can result from increased NF resource or common service requests. For example, if a network function or common service is generating more traffic than expected. **Diagnostic Information:** • Monitor memory usage metrics (`container_memory_usage_bytes` and `kube_pod_container_resource_limits`) using the KPI Dashboard. • Review the core-ingress pod logs for any irregularities, spikes in memory usage, or high volumes of NF resource or common service activity. **Recovery:**The alert clears automatically when memory utilization drops below the major threshold or exceeds the critical threshold. If the alert does not clear: • Check if any network function or common service is causing consistently high resource requests, leading to increased memory usage. • Analyze logs and metrics to identify and address the source of high usage. • Take steps to optimize memory usage or resolve any configuration issues. • Also, double-check the resource limits and requests configuration to ensure it is aligned with CNCC Console recommendations. For any assistance, contact My Oracle Support. Make sure to capture the `core-ingress` pod logs and relevant metrics to help Support analyze the issue.

8.2.6 CnccCoreMemoryUsageCrossedCriticalThreshold

Table 8-22 CnccCoreMemoryUsageCrossedCriticalThreshold

Field	**Details**
Description	This alert notifies that the CNCC Core Ingress pod has reached the configured critical threshold (90%) of its memory resource limits.
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	sum by(namespace,pod) (container_memory_usage_bytes{container!="", namespace="cncc-ns", pod=~".*core-cmservice.*|.*core-ingress-gateway.*"}) / sum by(namespace, pod) (kube_pod_container_resource_limits{namespace="cncc-ns",pod=~".*core-cmservice.*|.*core-ingress-gateway.*",resource="memory"}) * 100 >= 90
OID	1.3.6.1.4.1.323.5.3.51.1.2.8002

Metric Used	`container_memory_usage_bytes`
	`kube_pod_container_resource_limits`
	Note: This is a Kubernetes metric used for instance availability monitoring. If the metric is not available, use a similar metric as exposed by the monitoring system.

Table 8-22 (Cont.) CnccCoreMemoryUsageCrossedCriticalThreshold

Field	Details
Recommended Action	**Cause:** - This alert triggers when the CNCC Core Ingress pod's memory usage reaches the configured critical threshold (90%) of its resource limits. - Higher memory consumption can result from increased NF resource or common service requests. For example, if a network function or common service is generating more traffic than expected. **Diagnostic Information:** - Monitor memory usage metrics (`container_memory_usage_bytes` and `kube_pod_container_resource_limits`) using the KPI Dashboard. - Review the core-ingress pod logs for any irregularities, spikes in memory usage, or high volumes of NF resource or common service activity. **Recovery:** The alert clears automatically when memory utilization drops below the critical threshold. If the alert does not clear: - Check if any network function or common service is causing consistently high resource requests, leading to increased memory usage. - Analyze logs and metrics to identify and address the source of high usage. - Take steps to optimize memory usage or resolve any configuration issues. - Also, double-check the resource limits and requests configuration to ensure it is aligned with CNC Console recommendations. For any assistance, contact My Oracle Support. Make sure to capture the `core-ingress` pod logs and relevant metrics to help Support analyze the issue.

8.2.7 CnccCoreTransactionErrorRateAbove0.1Percent

Table 8-23 CnccCoreTransactionErrorRateAbove0.1Percent

Field	Details
Description	This alert notifies that the number of CNCC Core failed transactions is above 0.1 percent of the total transactions.

Table 8-23 (Cont.) CnccCoreTransactionErrorRateAbove0.1Percent

Field	Details
Summary	CNCC Core transaction Error Rate detected above 0.1 Percent of Total Transactions
Severity	warning
Condition	(sum by(namespace,pod) (rate(oc_ingressgateway_http_responses_total{Status=~"5.*",InstanceIdentifier=~".*core_ingressgateway",namespace="cncc-ns"}[5m]) or (up * 0)))/ (sum by(namespace,pod) (rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*core_ingressgateway",namespace="cncc-ns"}[5m]))) *100 >= 0.1 < 1
OID	1.3.6.1.4.1.323.5.3.51.1.2.8003
Metric Used	`oc_ingressgateway_http_responses_total`

Table 8-23 (Cont.) CnccCoreTransactionErrorRateAbove0.1Percent

Field	Details
Recommended Action	**Cause:** • This alert triggers when CNCC Core failed transactions exceed 0.1% of total transactions. • 5xx errors typically indicate server-side issues, such as the Core service or its dependencies (e.g, NFs, databases, or common services) being down, unreachable, overloaded, or misconfigured. • Unexpected spikes may also result from backend processing errors, resource exhaustion, or recent configuration changes. • Misconfiguration of any network function (NF) instance in the CNCC custom_values.yaml file can also lead to increased failure rates. **Diagnostic Information:** • Monitor the health and status of all Core pods and their dependencies (such as databases, network functions, and external services). • Review the core-ingress pod logs for error messages and stack traces, especially around the time of the alert. • Examine service-specific and application metrics to pinpoint which operations or endpoints are failing. • Check for recent configuration changes or deployment updates that could impact backend stability. • Review the CNCC custom_values.yaml for any misconfiguration in network function (NF) instances. **Recovery:** The alert is cleared automatically when the CNCC Core 5xx error rate drops below 0.1% or exceeds the 1% threshold. If the alert does not clear: • Investigate and address any backend, resource, or configuration issues causing errors. • Review CNCC custom_values.yaml for any misconfiguration in NF instances and correct them if necessary. • Coordinate with relevant teams to resolve dependency or service interruptions. • If this error level is expected for your workload, note that the threshold is configurable and can be adjusted as needed. For any assistance, contact My Oracle Support. Make sure to capture the core-ingress pod logs and relevant metrics to help Support analyze the issue.

8.2.8 CnccCoreTransactionErrorRateAbove1Percent

Table 8-24 CnccCoreTransactionErrorRateAbove1Percent

Field	Details
Description	This alert notifies that the number of CNCC Core failed transactions is above 1 percent of the total transactions.
Summary	CNCC Core transaction Error Rate detected above 1 Percent of Total Transactions
Severity	warning
Condition	(sum by(namespace,pod) (rate(oc_ingressgateway_http_responses_total{Status=~"5.*",InstanceIdentifier=~".*core_ingressgateway",namespace="cncc-ns"}[5m]) or (up * 0)))/ (sum by(namespace,pod) (rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*core_ingressgateway",namespace="cncc-ns"}[5m]))) *100 >= 1 < 10
OID	1.3.6.1.4.1.323.5.3.51.1.2.8003
Metric Used	`oc_ingressgateway_http_responses_total`

Table 8-24 (Cont.) CnccCoreTransactionErrorRateAbove1Percent

Field	Details
Recommended Action	**Cause:** • This alert triggers when CNCC Core failed transactions exceed 1% of total transactions. • 5xx errors typically indicate server-side issues, such as the Core service or its dependencies (e.g, NFs, databases, or common services) being down, unreachable, overloaded, or misconfigured. • Unexpected spikes may also result from backend processing errors, resource exhaustion, or recent configuration changes. • Misconfiguration of any network function (NF) instance in the CNCC `custom_values.yaml` file can also lead to increased failure rates. **Diagnostic Information:** • Monitor the health and status of all Core pods and their dependencies (such as databases, network functions, and external services). • Review the core-ingress pod logs for error messages and stack traces, especially around the time of the alert. • Examine service-specific and application metrics to pinpoint which operations or endpoints are failing. • Check for recent configuration changes or deployment updates that could impact backend stability. • Review the CNCC custom_values.yaml file for any misconfiguration in network function (NF) instances. **Recovery:** The alert is cleared automatically when the CNCC Core 5xx error rate drops below 1% or exceeds the 10% threshold. If the alert does not clear: • Investigate and address any backend, resource, or configuration issues causing errors. • Review the CNCC custom_values.yaml file for any misconfiguration in NF instances and correct if necessary. • Coordinate with relevant teams to resolve dependency or service interruptions. • If this error level is expected for your workload, note that the threshold is configurable and can be adjusted as needed. For any assistance, contact My Oracle Support. Make sure to capture the `core-ingress` pod logs and relevant metrics to help Support analyze the issue.

8.2.9 CnccCoreTransactionErrorRateAbove10Percent

Table 8-25   CnccCoreTransactionErrorRateAbove10Percent

Field	Details
Description	This alert notifies that the number of CNCC Core failed transactions is above 10 percent of the total transactions.
Summary	CNCC Core transaction Error Rate detected above 10 Percent of Total Transactions
Severity	minor
Condition	(sum by(namespace,pod) (rate(oc_ingressgateway_http_responses_total{Status=~"5.*",InstanceIdentifier=~".*core_ingressgateway",namespace="cncc-ns"}[5m]) or (up * 0)))/ (sum by(namespace,pod) (rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*core_ingressgateway",namespace="cncc-ns"}[5m]))) *100 >= 10 < 25
OID	1.3.6.1.4.1.323.5.3.51.1.2.8003
Metric Used	`oc_ingressgateway_http_responses_total`

Table 8-25 (Cont.) CnccCoreTransactionErrorRateAbove10Percent

Field	Details
Recommended Action	**Cause:** - This alert triggers when CNCC Core failed transactions exceed 10% of total transactions. - 5xx errors typically indicate server-side issues, such as the Core service or its dependencies (e.g, NFs, databases, or common services) being down, unreachable, overloaded, or misconfigured. - Unexpected spikes may also result from backend processing errors, resource exhaustion, or recent configuration changes. - Misconfiguration of any network function (NF) instance in the CNCC `custom_values.yaml` file can also lead to increased failure rates. **Diagnostic Information:** - Monitor the health and status of all Core pods and their dependencies (such as databases, network functions, and external services). - Review the core-ingress pod logs for error messages and stack traces, especially around the time of the alert. - Examine service-specific and application metrics to pinpoint which operations or endpoints are failing. - Check for recent configuration changes or deployment updates that could impact backend stability. - Review the CNCC `custom_values.yaml` file for any misconfiguration in network function (NF) instances. **Recovery:** The alert is cleared automatically when the CNCC Core 5xx error rate drops below 10% or exceeds the 25% threshold. If the alert does not clear: - Investigate and address any backend, resource, or configuration issues causing errors. - Review the CNCC custom_values.yaml file for any misconfiguration in NF instances and correct if necessary. - Coordinate with relevant teams to resolve dependency or service interruptions. - If this error level is expected for your workload, note that the threshold is configurable and can be adjusted as needed. For any assistance, contact My Oracle Support. Make sure to capture the `core-ingress` pod logs and relevant metrics to help Support analyze the issue.

8.2.10 CnccCoreTransactionErrorRateAbove25Percent

Table 8-26 CnccCoreTransactionErrorRateAbove25Percent

Field	Details
Description	This alert notifies that the number of CNCC Core failed transactions is above 25 percent of the total transactions.
Summary	CNCC Core transaction Error Rate detected above 25 Percent of Total Transactions
Severity	major
Condition	(sum by(namespace,pod) (rate(oc_ingressgateway_http_responses_total{Status=~"5.*",InstanceIdentifier=~".*core_ingressgateway",namespace="cncc-ns"}[5m]) or (up * 0)))/ (sum by(namespace,pod) (rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*core_ingressgateway",namespace="cncc-ns"}[5m]))) *100 >= 25 < 50
OID	1.3.6.1.4.1.323.5.3.51.1.2.8003
Metric Used	`oc_ingressgateway_http_responses_total`

Table 8-26 (Cont.) CnccCoreTransactionErrorRateAbove25Percent

Field	Details
Recommended Action	**Cause:** • This alert triggers when CNCC Core failed transactions exceed 25% of total transactions. • 5xx errors typically indicate server-side issues, such as the Core service or its dependencies (e.g, NFs, databases, or common services) being down, unreachable, overloaded, or misconfigured. • Unexpected spikes may also result from backend processing errors, resource exhaustion, or recent configuration changes. • Misconfiguration of any network function (NF) instance in the CNCC `custom_values.yaml` file can also lead to increased failure rates. **Diagnostic Information:** • Monitor the health and status of all Core pods and their dependencies (such as databases, network functions, and external services). • Review the core-ingress pod logs for error messages and stack traces, especially around the time of the alert. • Examine service-specific and application metrics to pinpoint which operations or endpoints are failing. • Check for recent configuration changes or deployment updates that could impact backend stability. • Review the CNCC `custom_values.yaml` file for any misconfiguration in network function (NF) instances. **Recovery:** The alert is cleared automatically when the CNCC Core 5xx error rate drops below 25% or exceeds the 50% threshold.If the alert does not clear: • Investigate and address any backend, resource, or configuration issues causing errors. • Review the CNCC `custom_values.yaml` file for any misconfiguration in NF instances and correct if necessary. • Coordinate with relevant teams to resolve dependency or service interruptions. • If this error level is expected for your workload, note that the threshold is configurable and can be adjusted as needed. For any assistance, contact My Oracle Support. Make sure to capture the `core-ingress` pod logs and relevant metrics to help Support analyze the issue.

8.2.11 CnccCoreTransactionErrorRateAbove50Percent

Table 8-27 CnccCoreTransactionErrorRateAbove50Percent

Field	Details
Description	This alert notifies that the number of CNCC Core failed transactions is above 50 percent of the total transactions.
Summary	CNCC Core transaction error rate detected above 50 percent of total transactions
Severity	critical
Condition	(sum by(namespace,pod) (rate(oc_ingressgateway_http_responses_total{Status=~"5.*",InstanceIdentifier=~".*core_ingressgateway",namespace="cncc-ns"}[5m]) or (up * 0)))/ (sum by(namespace,pod) (rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*core_ingressgateway",namespace="cncc-ns"}[5m]))) *100 >= 50
OID	1.3.6.1.4.1.323.5.3.51.1.2.8003
Metric Used	`oc_ingressgateway_http_responses_total`

Table 8-27 (Cont.) CnccCoreTransactionErrorRateAbove50Percent

Field	Details
Recommended Action	**Cause:** • This alert triggers when CNCC Core failed transactions exceed 50% of total transactions. • 5xx errors typically indicate server-side issues, such as the Core service or its dependencies (e.g, NFs, databases, or common services) being down, unreachable, overloaded, or misconfigured. • Unexpected spikes may also result from backend processing errors, resource exhaustion, or recent configuration changes. • Misconfiguration of any network function (NF) instance in the CNCC `custom_values.yaml` file can also lead to increased failure rates. **Diagnostic Information:** • Monitor the health and status of all Core pods and their dependencies (such as databases, network functions, and external services). • Review the core-ingress pod logs for error messages and stack traces, especially around the time of the alert. • Examine service-specific and application metrics to pinpoint which operations or endpoints are failing. • Check for recent configuration changes or deployment updates that could impact backend stability. • Review the CNCC `custom_values.yaml` file for any misconfiguration in network function (NF) instances. **Recovery:** The alert is cleared automatically when the CNCC Core 5xx error rate drops below the 50% threshold. If the alert does not clear: • Investigate and address any backend, resource, or configuration issues causing errors. • Review the CNCC `custom_values.yaml` file for any misconfiguration in NF instances and correct if necessary. • Coordinate with relevant teams to resolve dependency or service interruptions. • If this error level is expected for your workload, note that the threshold is configurable and can be adjusted as needed. For any assistance, contact My Oracle Support. Make sure to capture the `core-ingress` pod logs and relevant metrics to help Support analyze the issue.

8.2.12 CnccCoreIngressGatewayServiceDown

Table 8-28 CnccCoreIngressGatewayServiceDown

Field	Details
Description	This alert notifies that the CNCC Core Ingress Gateway pod is down.
Summary	namespace: {{$labels.namespace}}, timestamp: {{ with query "time()" }}{{ . | first | value | humanizeTimestamp }}{{ end }} : cncc-core-ingress-gateway service down
Severity	critical
Condition	absent(up{pod=~".*core-ingress-gateway.*", namespace="cncc-ns"}) or (up{pod=~".*core-ingress-gateway.*", namespace="cncc-ns"}) == 0
OID	1.3.6.1.4.1.323.5.3.51.1.2.8004
Metric Used	`up` **Note:** This is a Prometheus metric used for instance availability monitoring. If this metric is not available, use a similar metric as exposed by the monitoring system.
Recommended Action	**Cause:** This alert triggers when the CNCC Core Ingress Gateway pod or service is down. **Diagnostic Information:** • Check the orchestration platform (e.g., Kubernetes) logs for the `cncc-core-ingress-gateway` pod to identify liveness or readiness probe failures. • Review application logs for the `cncc-core-ingress-gateway` service, filtering for error or warning messages, or recent crash loops. • Verify recent configuration or deployment changes that might have impacted pod availability. • Check for resource issues (CPU, memory, disk) or dependency/service failures. **Recovery:** The alert clears automatically when the `cncc-core-ingress-gateway` service becomes available again. If the alert does not clear: • Continue to review logs, resource allocations, and configuration for possible causes of downtime. • Address any identified issues to restore service availability. • If this downtime is expected (e.g, for planned maintenance), you may adjust the alerting threshold as needed. For any assistance, contact My Oracle Support. Make sure to capture pod logs, orchestration event logs, and relevant metrics to help Support analyze the issue.

8.2.13 CnccCoreFailedLogin

Table 8-29 CnccCoreFailedLogin

Field	Details
Description	This alert notifies you if there are more than 3 failed login attempts in CNCC Core for a user within 5 minutes.
Summary	namespace: {{$labels.namespace}}, podname: {{$labels.pod}}, timestamp: {{ with query "time()" }}{{ . | first | value | humanizeTimestamp }}{{ end }}: failed login attempts are more than the configured threshold value
Severity	warning
Condition	sum by(Status,namespace,ResourcePath,Method,UserName,pod) (oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",namespace="cncc-ns",ResourcePath="/cncc/auth/realms/cncc/login-actions/authenticate",Method="POST",Status="200 OK"}) - sum by(Status,namespace,ResourcePath,Method,UserName,pod) (oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",namespace="cncc-ns",ResourcePath="/cncc/auth/realms/cncc/login-actions/authenticate",Method="POST",Status="200 OK"} offset 5m) > 3
OID	1.3.6.1.4.1.323.5.3.51.1.2.8005
Metric Used	`oc_ingressgateway_http_responses_total`

Table 8-29   (Cont.) CnccCoreFailedLogin

Field	Details
Recommended Action	**Cause:** - This alert triggers when there are more than 3 failed login attempts for a user in CNCC Core within 5 minutes. - This may be caused by users entering incorrect credentials, automated login attempts (such as brute-force attacks), or issues with the authentication process. **Diagnostic Information:** - Check if the affected user(s) are entering the correct username and password. - Review authentication and `core-ingress` pod logs for patterns such as repeated failures from the same source. - Investigate any potential account lockouts, recent configuration changes, or signs of automated or scripted login attempts. - Ensure the authentication service and related configurations are working as expected. **Recovery:** The alert is cleared automatically when the number of failed login attempts for a user drops below the configured threshold (default: 3) within the last 5 minutes. If the alert does not clear: - Investigate for possible brute-force activity, misconfigurations, or issues causing repeated login failures. - Verify the username being used. If the password is suspected to be incorrect, reset the password and attempt to log in again. - If this level of failed attempts is expected for your use case, note that the threshold is configurable and can be adjusted as needed. For any assistance, contact My Oracle Support. Make sure to capture the `core-ingress` pod logs and relevant metrics to help Support analyze the issue.

8.2.14 CnccCoreUnauthorizedAccess

Table 8-30   CnccCoreUnauthorizedAccess

Field	Details
Description	This alert notifies you if there are more than 3 unauthorized access (403 Forbidden) attempts in CNCC Core within 5 minutes.
Summary	namespace: {{$labels.namespace}}, podname: {{$labels.pod}}, timestamp: {{ with query "time()" }}{{ . | first | value | humanizeTimestamp }}{{ end }}: Unauthorized Access for CNCC-Core are more than threshold value
Severity	warning

Table 8-30 (Cont.) CnccCoreUnauthorizedAccess

Field	Details
Condition	sum by(Status,Method,namespace,ResourceType,UserId,UserName,pod) (oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*core_ingressgateway",namespace="cncc-ns",Status="403 FORBIDDEN", ResourceType!="UNKNOWN"}) - sum by(Status,Method,namespace,ResourceType,UserId,UserName,pod) (oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*core_ingressgateway",namespace="cncc-ns",Status="403 FORBIDDEN",ResourceType!="UNKNOWN"} offset 5m) > 3
OID	1.3.6.1.4.1.323.5.3.51.1.2.8006
Metric Used	`oc_ingressgateway_http_responses_total`
Recommended Action	**Cause:** • This alert triggers when there are more than 3 unauthorized access (403 Forbidden) attempts in CNCC Core within 5 minutes. • Unauthorized access can occur when users or services attempt to access resources without sufficient permissions. • Misconfigured access controls or recent changes in user roles or policies may also lead to repeated 403 errors. **Diagnostic Information:** • Check if the users or services encountering these errors have the necessary permissions for the requested resources. • Review `core-ingress` pod logs and access logs for repeated failed attempts from particular users, roles, services, or source IPs. • Examine recent changes in role assignments, access policies, or configurations that might have resulted in new authorization failures. • Investigate for potential automated or scripted attempts to access restricted resources. **Recovery:**This alert will clear automatically when unauthorized access attempts for a particular user or service drop below the configured threshold (default: 3) within the last 5 minutes. If the alert does not clear: • Verify that user and service permissions are configured appropriately. • Correct any misconfigured access controls, roles, or policies as needed. • If these attempts are expected for your environment, note that the threshold is configurable and can be adjusted as required. For any assistance, contact My Oracle Support. Make sure to capture the `core-ingress` pod logs and relevant metrics to help Support analyze the issue.

8.2.15 CnccCoreAccessTokenFailure

Table 8-31 CnccCoreAccessTokenFailure

Field	Details
Description	This alert notifies you if there are more than 3 failed access token requests in CNCC Core within 5 minutes.
Summary	namespace: {{$labels.namespace}}, podname: {{$labels.pod}}, timestamp: {{ with query "time()" }}{{ . | first | value | humanizeTimestamp }}{{ end }}: Access Token Failure count is above the configured threshold value
Severity	warning
Condition	sum by(Status,namespace,ResourcePath,Method,UserName,pod) (oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",namespace="cncc-ns",ResourcePath="/cncc/auth/realms/cncc/protocol/openid-connect/token",Method="POST",Status=~"4.*|5.*"}) - sum by(Status,namespace,ResourcePath,Method,UserName,pod) (oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",namespace="cncc-ns",ResourcePath="/cncc/auth/realms/cncc/protocol/openid-connect/token",Method="POST",Status=~"4.*|5.*"} offset 5m) > 3
OID	1.3.6.1.4.1.323.5.3.51.1.2.8007
Metric Used	`oc_ingressgateway_http_responses_total`

Table 8-31 (Cont.) CnccCoreAccessTokenFailure

Field	Details
Recommended Action	**Cause:** • This alert triggers when there are more than 3 failed access token requests in CNCC Core within 5 minutes. • Failures may occur due to incorrect credentials, misconfigured client applications, or backend errors affecting token generation. • Configuration issues or recent changes in the token endpoint can also cause failures. **Diagnostic Information:** • Check if users or client applications are using the correct credentials and valid request parameters. • Review the `core-ingress` pod logs for errors related to access token requests. • Look for repeated failures from specific users or applications, or patterns suggesting misconfiguration. • Verify whether there have been any recent changes to the token endpoint or related configurations. **Recovery:** The alert clears automatically when the number of failed access token requests for a user drops below the configured threshold (default: 3) within the last 5 minutes.If the alert does not clear: • Verify the credentials and parameters being used by users. • If requests are failing due to expired or incorrect credentials, reset the credentials and attempt to generate a new token. • Address any misconfigurations or backend issues found during diagnostics. • If this failure rate is expected, note that the threshold for the alert is configurable and can be adjusted as needed. For any assistance, contact My Oracle Support. Make sure to capture the `core-ingress` pod logs and relevant metrics to help Support analyze the issue.

8.3 Validating Alerts

Configure and Validate Alerts in Prometheus Server. Refer to CNCC Alert Configuration section 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 to Rules.
3. Search CNCC. CNCC Alerts list is displayed.

> **Note**
>
> If you are unable to see the alerts, it means the alert file is not loaded in a proper format which the Prometheus server accepts. Modify the file and try again

8.4 Disabling Alerts

This section explains how to disable the alerts in CNC Console.

1. Edit manager or agent alert file to remove specific alert.
2. Remove complete content of the specific alert from the manager or agent alert file
   For example, if you want to remove `CnccIamTotalIngressTrafficRateAboveMinorThreshold alert` from `occncc_manager_alerting_rules_promha_<version>.yaml` file, remove the complete content like below:

   cncc_alert_rules_<version>.yaml

   ```
   ## ALERT SAMPLE START##
   - alert: CnccIamTotalIngressTrafficRateAboveMinorThreshold
    annotations:
    description: 'CNCC IAM Ingress traffic Rate is above the configured minor
   threshold i.e. 700 requests per second (current value is: {{ $value }})'
    summary: 'namespace: {{$labels.kubernetes_namespace}}, podname:
   {{$labels.kubernetes_pod_name}}, timestamp: {{ with query "time()" }}{{ .
   | first | value | humanizeTimestamp }}{{ end }}: Traffic Rate is above 70
   Percent of Max requests per second(1000)'
    expr:
   sum(rate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*cncc-
   iam_ingressgateway",kubernetes_namespace="cncc"}[2m])) > 0
    labels:
    severity: minor
    oid: "1.3.6.1.4.1.323.5.3.51.1.2.7001"
    namespace: ' {{ $labels.kubernetes_namespace }} '
    podname: ' {{$labels.kubernetes_pod_name}} '
   ## ALERT SAMPLE END##
   ```

3. Perform Alert configuration. See CNCC Alert Configuration section for details.

8.5 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
```

8.6 CNC Console MIB Files

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.

occncc_mib_tc_<version>.mib
This file is considered as CNCC top level mib file, where the Objects and their data types are defined.

occncc_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.

9 CNC Console KPIs

This section provides the information about CNC Console KPIs:

9.1 CNC Console IAM KPIs

> ⓘ **Note**
>
> Not applicable for OCI deployment.

This section provides the information about CNC Console IAM KPIs:

9.1.1 M-CNCC IAM Requests

Table 9-1   M-CNCC IAM Requests

Description	CNCC IAM Requests
Expression	1. **For CNE without Prometheus Operator**: all: sum(irate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*iam_ingressgateway",kubernetes_namespace="$namespace"}[2m])) **For CNE with Prometheus HA Operator:** all:sum(irate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*iam_ingressgateway",namespace="$namespace"}[2m])) **For OCI:** *Not Applicable*

9.1.2 M-CNCC IAM Responses

Table 9-2   M-CNCC IAM Responses

Description	M-CNCC IAM Responses

Table 9-2 (Cont.) M-CNCC IAM Responses

Expression	**For CNE without Prometheus Operator:** 1. all: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",Status=~"2.*",Route_path=~".*",kubernetes_namespace="$namespace"}[5m])) 2. all_error: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",Status=~"4.*\|5.*",Route_path=~".*",kubernetes_namespace="$namespace"}[5m])) 3. all_4xx: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",Status=~"4.*",Route_path=~".*",kubernetes_namespace="$namespace"}[5m])) 4. all_5xx: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",Status=~"5.*",Route_path=~".*",kubernetes_namespace="$namespace"}[5m])) **For CNE with Prometheus HA Operator:** 1. all: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",Status=~"2.*",Route_path=~".*",namespace="$namespace"}[5m])) 2. all_error: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",Status=~"4.*\|5.*",Route_path=~".*",namespace="$namespace"}[5m])) 3. all_4xx: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",Status=~"4.*",Route_path=~".*",namespace="$namespace"}[5m])) 4. all_5xx: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",Status=~"5.*",Route_path=~".*",namespace="$namespace"}[5m])) **For OCI:** *Not Applicable*

9.1.3 M-CNCC IAM Success Rate

Table 9-3 M-CNCC IAM Success Rate

Description	M-CNCC IAM Success Rate (2xx responses divided by total responses)

Table 9-3 (Cont.) M-CNCC IAM Success Rate

Expression	**For CNE without Prometheus Operator:** 1. all: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",Status=~"2.*",Route_path=~".*",kubernetes_namespace="$namespace"}[5m]))/sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",Route_path=~".*",kubernetes_namespace="$namespace"}[5m]))*100 **For CNE with Prometheus HA Operator:** 1. all: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",Status=~"2.*",Route_path=~".*",namespace="$namespace"}[5m]))/sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",Route_path=~".*",namespace="$namespace"}[5m]))*100 **For OCI:** *Not Applicable*

9.1.4 M-CNCC IAM Error Rate

Table 9-4 M-CNCC IAM Error Rate

Description	M-CNCC IAM Error Rate (4xx or 5xx responses divided by total responses) for all as well as specific NFs
Expression	**For CNE without Prometheus Operator:** 1. all: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",Status=~"4.*|5.*",Route_path=~".*",kubernetes_namespace="$namespace"}[5m]))/sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",Route_path=~".*",kubernetes_namespace="$namespace"}[5m]))*100 **For CNE with Prometheus HA Operator:** 1. all: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*iam_ingressgateway",Status=~"4.*|5.*",Route_path=~".*",namespace="$namespace"}[5m]))/sum(increase(oc_ingressgateway_http_responses_total{Route_path=~".*",namespace="$namespace"}[5m]))*100 **For OCI:** *Not Applicable*

9.2 M-CNCC Core KPIs

This section provides the information about M-CNCC Core KPIs:

9.2.1 M-CNCC Core Requests

Table 9-5 M-CNCC Core Requests

Description	M-CNCC Core Requests for NFs and specific NFs

Table 9-5 (Cont.) M-CNCC Core Requests

<table>
<tr><td>Expression</td><td>1. <b>For CNE without Prometheus Operator</b>: all: sum(irate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*mcore_ingressgateway",kubernetes_namespace="$namespace"}[2m]))
<b>For CNE with Prometheus HA Operator</b>: all: sum(irate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*mcore_ingressgateway",namespace="$namespace"}[2m]))
2. scp: sum(irate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/ocscp/.*"} [2m]))
3. nrf: sum(irate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/nrf-configuration/v1.*|.*/nrf-state-data/.*|.*/ocnrf-swagger/.*|.*/nrf-status-data/.*|.*/nrf/nf-common-component/.*"} [2m]))
4. udr: sum(irate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/nudr-dr-prov/.*|.*/nudr-dr-mgm/.*|.*/nudr-group-id-map-prov/.*|.*/slf-group-prov/.*|.*/nudr-config/.*|.*/udr/nf-common-component/.*|.*/n5g-eir-prov/.*"} [2m]))
5. policy: sum(irate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/policyapi/.*|.*/oc-cnpolicy-configuration/.*|.*/pcf/.*"} [2m]))
6. bsf: sum(irate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/bsfapi.*|.*/oc-bsf-configuration/.*|.*/bsf/.*"} [2m]))
7. sepp: sum(irate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/sepp-configuration/.*|.*/sepp/.*"} [2m]))
8. nssf: sum(irate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/nnssf-configuration/.*|.*/nssf/.*"} [2m]))
9. dd: sum(irate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/ocnadd/.*|.*/ocnaddapi/.*"} [2m]))
10. provgw: sum(irate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*provgw-config.*|.*provgw.*"} [2m]))
11. cndbtier: sum(irate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*ocdbtier.*"} [2m]))</td></tr>
</table>

Table 9-5 (Cont.) M-CNCC Core Requests

<table>
<tr><td></td><td>12. occm: sum(irate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*occm-config.*"} [2m]))
For OCI:
1. scp**:** oc_ingressgateway_http_requests_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",ResourcePath=~"*ocscp*"}.sum()
2. nrf: oc_ingressgateway_http_requests_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Route_path=~"*nrf-configuration/v1*|*nrf-state-data*|*ocnrf-swagger*|*nrf-status-data*|*nrf/nf-common-component*"}.sum()
3. udr: oc_ingressgateway_http_requests_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Route_path=~"*nudr-dr-prov*|*nudr-dr-mgm*|*nudr-group-id-map-prov*|*slf-group-prov*|*n5g-eir-prov*|*nudr-config*|*udr/nf-common-component*"}.sum()
4. policy: oc_ingressgateway_http_requests_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",ResourcePath=~"*policyapi*|*oc-cnpolicy-configuration*|*pcf*"}.sum()
5. bsf: oc_ingressgateway_http_requests_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",ResourcePath=~"*bsfapi*|*oc-bsf-configuration*|*bsf*"}.sum()
6. sepp: oc_ingressgateway_http_requests_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Route_path=~"*sepp-configuration*|*sepp*"}.sum()
7. nssf: oc_ingressgateway_http_requests_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Route_path=~"*nnssf-configuration*|*nssf*"}.sum()
8. dd: oc_ingressgateway_http_requests_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Route_path=~"*ocnadd*|*ocnaddapi*"}.sum()
9. provgw: oc_ingressgateway_http_requests_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Route_path=~"*provgw-config*|*provgw*"}.sum()
10. occm: oc_ingressgateway_http_requests_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Route_path=~"*occm-config*"}.sum()</td></tr>
</table>

9.2.2 M-CNCC Core Responses

Table 9-6 M-CNCC Core Responses

Description	M-CNCC Core Responses for all as well as specific NFs

Table 9-6 (Cont.) M-CNCC Core Responses

Expression	**For CNE without Prometheus Operator** **1.** all_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*",kubernetes_namespace="$namespace"}[5m])) **1.** all_error: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"4.*\|5.*",ResourcePath=~".*",kubernetes_namespace="$namespace"}[5m])) **2.** all_4xx: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"4.*",ResourcePath=~".*",kubernetes_namespace="$namespace"}[5m])) **3.** all_5xx: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"5.*",ResourcePath=~".*",kubernetes_namespace="$namespace"}[5m])) **4.** scp_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*/ocscp/.*",kubernetes_namespace="$namespace"}[5m])) **5.** nrf_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*/nrf-configuration/v1.*\|.*/nrf-state-data/.*\|.*/ocnrf-swagger/.*\|.*/nrf-status-data/.*\|.*/nrf/nf-common-component/.*",kubernetes_namespace="$namespace"}[5m])) **6.** udr_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*/nudr-dr-prov/.*|.*/nudr-dr-mgm/.*|.*/nudr-group-id-map-prov/.*|.*/slf-group-prov/.*|.*/nudr-config/.*|.*/udr/nf-common-component/.*|.*/n5g-eir-prov/.*",kubernetes_namespace="$namespace"}[5m])) **7.** policy_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*/policyapi/.*\|.*/oc-cnpolicy-configuration/.*\|.*/pcf/.*",kubernetes_namespace="$namespace"}[5m])) **8.** bsf_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*/bsfapi/.*|.*/oc-bsf-configuration/.*|.*/bsf/.*",kubernetes_namespace="$namespace"}[5m])) **9.** sepp_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",Resource

Table 9-6   (Cont.) M-CNCC Core Responses

<table>
<tr><td></td><td>Path=~".*/sepp-configuration/.*|.*/sepp/.*",kubernetes_namespace="$namespace"}[5m]))
10. nssf_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*/nnssf-configuration/.*|.*/nssf/.*",kubernetes_namespace="$namespace"}[5m]))
11. dd_success:sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath~".*/ocnadd/.*|.*/ocnaddapi/.*",kubernetes_namespace="$namespace"}[5m]))
12. provgw_success:sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath~".*provgw-config.*|.*provgw.*",kubernetes_namespace="$namespace"}[5m]))
13. cndbtier_success:sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath~".*ocdbtier.*",kubernetes_namespace="$namespace"}[5m]))
14. occm_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*occm-config.*",kubernetes_namespace="$namespace"}[5m]))
For CNE with Prometheus HA Operator:
1. all_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*",namespace="$namespace"}[5m]))
1. all_error: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"4.*|5.*",ResourcePath=~".*",namespace="$namespace"}[5m]))
2. all_4xx: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"4.*",ResourcePath=~".*",namespace="$namespace"}[5m]))
3. all_5xx: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"5.*",ResourcePath=~".*",namespace="$namespace"}[5m]))
4. scp_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*/ocscp/.*",namespace="$namespace"}[5m]))
5. nrf_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*/nrf-configuration/v1.*|.*/nrf-state-data/.*|.*/ocnrf-</td></tr>
</table>

Table 9-6 (Cont.) M-CNCC Core Responses

	swagger/.*|.*/nrf-status-data/.*|.*/nrf/nf-common-component/.*",namespace="$namespace"}[5m])) 6. udr_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*/nudr-dr-prov/.*\|.*/nudr-dr-mgm/.*\|.*/nudr-group-id-map-prov/.*\|.*/slf-group-prov/.*\|.*/nudr-config/.*\|.*/udr/nf-common-component/.*\|.*/n5g-eir-prov/.*",namespace="$namespace"}[5m])) 7. policy_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*/policyapi/.*|.*/oc-cnpolicy-configuration/.*|.*/pcf/.*",namespace="$namespace"}[5m])) 8. bsf_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*/bsfapi/.*\|.*/oc-bsf-configuration/.*\|.*/bsf/.*",namespace="$namespace"}[5m])) 9. sepp_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*/sepp-configuration/.*|.*/sepp/.*",namespace="$namespace"}[5m])) 10. nssf_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*/nnssf-configuration/.*\|.*/nssf/.*",namespace="$namespace"}[5m])) 11. dd_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*/ocnadd/.*|.*/ocnaddapi/.*",namespace="$namespace"}[5m])) 12. provgw_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*provgw-config.*\|.*provgw.*",namespace="$namespace"}[5m])) 13. cndbtier_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*ocdbtier.*",namespace="$namespace"}[5m])) 14. occm_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*occm-config.*",namespace="$namespace"}[5m])) **For OCI:**

Table 9-6 (Cont.) M-CNCC Core Responses

	1. all success: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Status=~"2*",ResourcePath=~"*",k8Namespace="cncc-ns"}.sum() 2. all error: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Status=~"4*|5*",ResourcePath=~"*",k8Namespace="cncc-ns"}.sum() 3. all 4xx: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Status=~"4*",ResourcePath=~"*",k8Namespace="cncc-ns"}.sum() 4. all 5xx: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Status=~"5*",ResourcePath=~"*",k8Namespace="cncc-ns"}.sum() 5. scp_success: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Status=~"2*",ResourcePath=~"*ocscp*",k8Namespace="cncc-ns"}.sum() 6. nrf_success: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Status=~"2*",Route_path=~"*nrf-configuration/v1*|*nrf-state-data*|*ocnrf-swagger*|*nrf-status-data*|*nrf/nf-common-component*",k8Namespace="cncc-ns"}.sum() 7. udr_success: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Status=~"2*",Route_path=~"*nudr-dr-prov*|*nudr-dr-mgm*|*nudr-group-id-map-prov*|*slf-group-prov*|*n5g-eir-prov*|*nudr-config*|*udr/nf-common-component*",k8Namespace="cncc-ns"}.sum() 8. policy_success: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Status=~"2*",ResourcePath=~"*policyapi*|*oc-cnpolicy-configuration*|*pcf*",k8Namespace="cncc-ns"}.sum() 9. bsf_success: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Status=~"2*",ResourcePath=~"*bsfapi*|*oc-bsf-configuration*|*bsf*",k8Namespace="cncc-ns"}.sum() 10. sepp_success: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Status=~"2*",Route_path=~"*sepp-configuration*|*sepp*",k8Namespace="cncc-ns"}.sum() 11. nssf_success: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Status=~"2*",Route_path=~"*nnssf-configuration*|*nssf*",k8Namespace="cncc-ns"}.sum() 12. dd_success: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Status=~"2*",Route_path=~"*ocnadd*|*ocnaddapi*",k8Namespace="cncc-ns"}.sum() 13. provgw_success: oc_ingressgateway_http_responses_total[10m]

Table 9-6 (Cont.) M-CNCC Core Responses

	{InstanceIdentifier=~"*mcore_ingressgateway",Status=~"2*",Route_path=~"*provgw-config*|*provgw*",k8Namespace="cncc-ns"}.sum() 14. occm_success: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*mcore_ingressgateway",Status=~"2*",Route_path=~"*occm-config*",k8Namespace="cncc-ns"}.sum()

9.2.3 M-CNCC Core Success Rate

Table 9-7 M-CNCC Core Success Rate

Description	M-CNCC Core Success Rate (2xx responses divided by total responses) for all as well as specific NFs

Table 9-7 (Cont.) M-CNCC Core Success Rate

<table>
<tr><th></th><th></th></tr>
<tr><td>Expression</td><td><b>For CNE without Prometheus Operator:</b>
1. all: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*",kubernetes_namespace="$namespace"}[5m]))/sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*",kubernetes_namespace="$namespace"}[5m]))*100
2. scp: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*/ocscp/.*",kubernetes_namespace="$namespace"}[5m]))/sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/ocscp/.*",kubernetes_namespace="$namespace"}[5m]))*100
3. nrf: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*/nrf-configuration/v1/.*|.*/nrf-state-data/.*|.*/ocnrf-swagger/.*|.*/nrf-status-data/.*|.*/nrf/nf-common-component/.*",kubernetes_namespace="$namespace"}[5m]))/sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/nrf-configuration/v1/.*|.*/nrf-state-data/.*|.*/ocnrf-swagger/.*|.*/nrf-status-data/.*|.*/nrf/nf-common-component/.*",kubernetes_namespace="$namespace"}[5m]))*100
4. udr: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*/nudr-dr-prov/.*|.*/nudr-dr-mgm/.*|.*/nudr-group-id-map-prov/.*|.*/slf-group-prov/.*|.*/nudr-config/.*|.*/udr/nf-common-component/.*|.*/n5g-eir-prov/.*",kubernetes_namespace="$namespace"}[5m]))/sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/nudr-dr-prov/.*|.*/nudr-dr-mgm/.*|.*/nudr-group-id-map-prov/.*|.*/slf-group-prov/.*|.*/nudr-config/.*|.*/udr/nf-common-component/.*|.*/n5g-eir-prov/.*",kubernetes_namespace="$namespace"}[5m]))*100
5. policy: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*/policyapi/.*|.*/oc-cnpolicy-configuration/.*|.*/pcf/.*",kubernetes_namespace="$namespace"}[5m]))/sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/policyapi/.*|.*/oc-cnpolicy-configuration/.*|.*/pcf/.*",kubernetes_namespace="$namespace"}[5m]))*100
6. bsf: sum(rate(oc_ingressgateway_http_responses_total{InstanceId</td></tr>
</table>

Table 9-7 (Cont.) M-CNCC Core Success Rate

	entifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*/bsfapi/.*|.*/oc-bsf-configuration/.*|.*/bsf/.*",kubernetes_namespace="$namespace"}[5m]))/sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/bsfapi.*\|.*/oc-bsf-configuration/.*\|.*/bsf/.*",kubernetes_namespace="$namespace"}[5m]))*100 7. sepp: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*/sepp-configuration/.*|.*/sepp/.*",kubernetes_namespace="$namespace"}[5m]))/sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/sepp-configuration/.*\|.*/sepp/.*",kubernetes_namespace="$namespace"}[5m]))*100 8. nssf: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*/nnssf-configuration/.*|.*/nssf/.*",kubernetes_namespace="$namespace"}[5m]))/sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/nnssf-configuration/.*\|.*/nssf/.*",kubernetes_namespace="$namespace"}[5m]))*100 9. dd: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*/ocnadd/.*|.*/ocnaddapi/.*",kubernetes_namespace="$namespace"}[5m]))/sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/ocnadd/.*\|.*/ocnaddapi/.*",kubernetes_namespace="$namespace"}[5m]))*100 10. provgw: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*provgw-config.*|.*provgw.*",kubernetes_namespace="$namespace"}[5m]))/sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*provgw-config.*\|.*provgw.*",kubernetes_namespace="$namespace"}[5m]))*100 11. cndbtier: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*ocdbtier.*",kubernetes_namespace="$namespace"}[5m]))/sum(rate(oc_ingressgateway_http_responses_total{InstanceId

Table 9-7 (Cont.) M-CNCC Core Success Rate

	entifier=~".*mcore_ingressgateway",ResourcePath=~".*ocdbtier.*",kubernetes_namespace="$namespace"}[5m]))*100 12. occm: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*occm-config.*",kubernetes_namespace="$namespace"}[5m]))/ sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*occm-config.*",kubernetes_namespace="$namespace"}[5m]))*100 **For CNE with Prometheus HA Operator:** 1. all: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*",namespace="$namespace"}[5m]))/ sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*", namespace="$namespace"}[5m]))*100 2. scp: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*/ocscp/.*",namespace="$namespace"}[5m]))/ sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/ocscp/.*",namespace="$namespace"}[5m]))*100 3. nrf: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*/nrf-configuration/v1/.*\|.*/nrf-state-data/.*\|.*/ocnrf-swagger/.*\|.*/nrf-status-data/.*\|.*/nrf/nf-common-component/.*",namespace="$namespace"}[5m]))/ sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/nrf-configuration/v1/.*|.*/nrf-state-data/.*|.*/ocnrf-swagger/.*|.*/nrf-status-data/.*|.*/nrf/nf-common-component/.*",namespace="$namespace"}[5m]))*100 4. udr: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*/nudr-dr-prov/.*\|.*/nudr-dr-mgm/.*\|.*/nudr-group-id-map-prov/.*\|.*/slf-group-prov/.*\|.*/nudr-config/.*\|.*/udr/nf-common-component/.*\|.*/n5g-eir-prov/.*",namespace="$namespace"}[5m]))/ sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/nudr-dr-prov/.*|.*/nudr-dr-mgm/.*|.*/nudr-group-id-map-prov/.*|.*/slf-group-prov/.*|.*/nudr-config/.*|.*/udr/nf-common-component/.*|.*/n5g-eir-prov/.*",namespace="$namespace"}[5m]))*100 5. policy: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*/policyapi/.*|.*/oc-cnpolicy-

Table 9-7 (Cont.) M-CNCC Core Success Rate

<table>
<tr><td></td><td>configuration/.*|.*/pcf/.*",namespace="$namespace"}[5m]))/sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/policyapi/.*|.*/oc-cnpolicy-configuration/.*|.*/pcf/.*",namespace="$namespace"}[5m]))*100
6. bsf: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*/bsfapi/.*|.*/oc-bsf-configuration/.*|.*/bsf/.*",namespace="$namespace"}[5m]))/sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/bsfapi.*|.*/oc-bsf-configuration/.*|.*/bsf/.*",namespace="$namespace"}[5m]))*100
7. sepp: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*/sepp-configuration/.*|.*/sepp/.*",namespace="$namespace"}[5m]))/sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/sepp-configuration/.*|.*/sepp/.*",namespace="$namespace"}[5m]))*100
8. nssf: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*/nnssf-configuration/.*|.*/nssf/.*",namespace="$namespace"}[5m]))/sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/nnssf-configuration/.*|.*/nssf/.*",namespace="$namespace"}[5m]))*100
9. dd: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*/ocnadd/.*|.*/ocnaddapi/.*",namespace="$namespace"}[5m]))/sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/ocnadd/.*|.*/ocnaddapi/.*",namespace="$namespace"}[5m]))*100
10. provgw: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*provgw-config.*|.*provgw.*",namespace="$namespace"}[5m]))/sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*provgw-config.*|.*provgw.*",namespace="$namespace"}[5m]))*100
11. cndbtier: sum(rate(oc_ingressgateway_http_responses_total{InstanceId</td></tr>
</table>

Table 9-7 (Cont.) M-CNCC Core Success Rate

	entifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*ocdbtier.*",namespace="$namespace"}[5m]))/ sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*ocdbtier.*",namespace="$namespace"}[5m]))*100 12. occm: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*occm-config.*",namespace="$namespace"}[5m]))/ sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*occm-config.*",namespace="$namespace"}[5m]))*100 **For OCI:** 1. all: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*mcore_ingressgateway",Status=~"2*",ResourcePath=~"*",k8Namespace="cncc-ns"}.increment().grouping().sum()/ oc_ingressgateway_http_responses_total[10m] {ResourcePath=~"*",k8Namespace="cncc-ns"}.increment().grouping().sum() * 100 2. scp: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*mcore_ingressgateway",Status=~"2*",ResourcePath=~"*ocscp*",k8Namespace="cncc-ns"}.rate().grouping().sum()/ oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*mcore_ingressgateway",ResourcePath=~"*ocscp*",k8Namespace="cncc-ns"}.rate().grouping().sum() * 100 3. nrf: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*mcore_ingressgateway",Status=~"2*",Route_path=~"*nrf-configuration/v1*|*nrf-state-data*|*ocnrf-swagger*|*nrf-status-data*|*nrf/nf-common-component*",k8Namespace="cncc-ns"}.rate().grouping().sum()/ oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*core_ingressgateway",Route_path=~"*nrf-configuration/v1*|*nrf-state-data*|*ocnrf-swagger*|*nrf-status-data*|*nrf/nf-common-component*",k8Namespace="cncc-ns"}.rate().grouping().sum() * 100 4. udr: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*more_ingressgateway",Status=~"2*",Route_path=~"*nudr-dr-prov*|*nudr-dr-mgm*|*nudr-group-id-map-prov*|*slf-group-prov*|*n5g-eir-prov*|*nudr-config*|*udr/nf-common-component*",k8Namespace="cncc-ns"}.rate().grouping().sum()/ oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*core_ingressgateway",Route_path=~"*nudr-dr-prov*|*nudr-dr-mgm*|*nudr-group-id-map-prov*|*slf-group-prov*|*n5g-eir-prov*|*nudr-config*|*udr/nf-common-component*",k8Namespace="cncc-ns"}.rate().grouping().sum() * 100

Table 9-7 (Cont.) M-CNCC Core Success Rate

	5. policy: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Status=~"2*",ResourcePath=~"*policyapi*|*oc-cnpolicy-configuration*|*pcf*",k8Namespace="cncc-ns"}.rate().grouping().sum()/oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*core_ingressgateway",ResourcePath=~"*policyapi*|*oc-cnpolicy-configuration*|*pcf*",k8Namespace="cncc-ns"}.rate().grouping().sum() * 100 6. bsf: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Status=~"2*",ResourcePath=~"*bsfapi*|*oc-bsf-configuration*|*bsf*",k8Namespace="cncc-ns"}.rate().grouping().sum()/oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",ResourcePath=~"*bsfapi*|*oc-bsf-configuration*|*bsf*",k8Namespace="cncc-ns"}.rate().grouping().sum() * 100 7. sepp: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Status=~"2*",Route_path=~"*sepp-configuration*|*sepp*",k8Namespace="cncc-ns"}.rate().grouping().sum()/oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Route_path=~"*sepp-configuration*|*sepp*",k8Namespace="cncc"}.rate().grouping().sum() * 100 8. nssf: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Status=~"2*",Route_path=~"*nnssf-configuration*|*nssf*",k8Namespace="cncc-ns"}.rate().grouping().sum()/oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Route_path=~"*nnssf-configuration*|*nssf*",k8Namespace="cncc-ns"}.rate().grouping().sum() * 100 9. dd: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Status=~"2*",Route_path=~"*ocnadd*|*ocnaddapi*",k8Namespace="cncc-ns"}.rate().grouping().sum()/oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Route_path=~"*ocnadd*|*ocnaddapi*",k8Namespace="cncc-ns"}.rate().grouping().sum() * 100 10. provgw: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Status=~"2*",Route_path=~"*provgw-config*|*provgw*",k8Namespace="cncc-ns"}.rate().grouping().sum()/oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Route_path=~"*provgw-config*|*provgw*",k8Namespace="cncc-ns"}.rate().grouping().sum() * 100 11. occm: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Status=~"2*",Route_path=~"*occm-config*",k8Namespace="cncc-ns"}.rate().grouping().sum()/oc_ingressgateway_http_responses_total[10m]

Table 9-7 (Cont.) M-CNCC Core Success Rate

	{InstanceIdentifier=~"*mcore_ingressgateway",Route_path=~"*occm-config*",k8Namespace="cncc-ns"}.rate().grouping().sum() * 100

9.2.4 M-CNCC Core Error Rate

Table 9-8 M-CNCC Core Error Rate

Description	M-CNCC Core Error Rate (4xx or 5xx responses divided by total responses) for all as well as specific NFs

Table 9-8 (Cont.) M-CNCC Core Error Rate

<table>
<tr><td>Expression</td><td><b>For CNE without Prometheus Operator:</b>
1. all: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"4.*|5.*",ResourcePath=~".*",kubernetes_namespace="$namespace"}[5m]))/ sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*",kubernetes_namespace="$namespace"}[5m]))*100
2. scp: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"4.*|5.*",ResourcePath=~".*/ocscp/.*",kubernetes_namespace="$namespace"}[5m]))/ sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/ocscp/.*",kubernetes_namespace="$namespace"}[5m]))*100
3. nrf: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"4.*|5.*",ResourcePath=~".*/nrf-configuration/v1/.*|.*/nrf-state-data/.*|/ocnrf-swagger/.*|/nrf-status-data/.*|.*/nrf/nf-common-component/.*",kubernetes_namespace="$namespace"}[5m]))/ sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/nrf-configuration/v1/.*|.*/nrf-state-data/.*|.*/ocnrf-swagger/.*|.*/nrf-status-data/.*|.*/nrf/nf-common-component/.*"kubernetes_namespace="$namespace"}[5m]))*100
4. udr: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"4.*|5.*",ResourcePath=~".*/nudr-dr-prov/.*|.*/nudr-dr-mgm/.*|.*/nudr-group-id-map-prov/.*|.*/slf-group-prov/.*|.*/nudr-config/.*|.*/udr/nf-common-component/.*|.*/n5g-eir-prov/.*",kubernetes_namespace="$namespace"}[5m]))/ sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/nudr-dr-prov/.*|.*/nudr-dr-mgm/.*|.*/nudr-group-id-map-prov/.*|.*/slf-group-prov/.*|.*/nudr-config/.*|.*/udr/nf-common-component/.*|.*/n5g-eir-prov/.*",kubernetes_namespace="$namespace"}[5m]))*100
5. policy: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"4.*|5.*",ResourcePath=~".*/policyapi/.*|.*/oc-cnpolicy-configuration/.*|.*/pcf/.*",kubernetes_namespace="$namespace"}[5m]))/ sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/policyapi/.*|.*/oc-cnpolicy-configuration/.*|.*/pcf/.*",kubernetes_namespace="$namespace"}[5m]))*100</td></tr>
</table>

Table 9-8   (Cont.) M-CNCC Core Error Rate

<table>
<tr><td></td><td>6. bsf: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"4.*|5.*",ResourcePath=~".*/bsfapi/.*|.*/oc-bsf-configuration/.*|.*/bsf/.*",kubernetes_namespace="$namespace"}[5m]))/sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/bsfapi.*|.*/oc-bsf-configuration/.*|.*/bsf/.*",kubernetes_namespace="$namespace"}[5m]))*100
7. sepp: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"4.*|5.*",ResourcePath=~".*/sepp-configuration/.*|.*/sepp/.*",kubernetes_namespace="$namespace"}[5m]))/sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/sepp-configuration/.*|.*/sepp/.*",kubernetes_namespace="$namespace"}[5m]))*100
8. nssf: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"4.*|5.*",ResourcePath=~".*/nnssf-configuration/.*|.*/nssf/.*",kubernetes_namespace="$namespace"}[5m]))/sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/nnssf-configuration/.*|.*/nssf/.*",kubernetes_namespace="$namespace"}[5m]))*100
9. dd: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"4.*|5.*",ResourcePath=~".*/ocnadd/.*|.*/ocnaddapi/.*",kubernetes_namespace="$namespace"}[5m]))/sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/ocnadd/.*|.*/ocnaddapi/.*",kubernetes_namespace="$namespace"}[5m]))*100
10. provgw: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"4.*|5.*",ResourcePath=~".*provgw-config.*|.*provgw.*",kubernetes_namespace="$namespace"}[5m]))/sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*provgw-config.*|.*provgw.*",kubernetes_namespace="$namespace"}[5m]))*100
11. cndbtier: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"4.*|5.*",ResourcePath=~".*ocdbtier.*",kubernetes_namespace="$</td></tr>
</table>

Table 9-8 (Cont.) M-CNCC Core Error Rate

<table>
<tr><td></td><td>namespace"}[5m]))/ sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*ocdbtier.*",kubernetes_namespace="$namespace"}[5m]))*100
12. occm: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"4.*|5.*",ResourcePath=~".*occm-config.*",kubernetes_namespace="$namespace"}[5m]))/ sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*occm-config.*",kubernetes_namespace="$namespace"}[5m]))*100
For CNE with Prometheus HA Operator:
1. all: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"4.*|5.*",ResourcePath=~".*",namespace="$namespace"}[5m]))/ sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*", namespace="$namespace"}[5m]))*100
2. scp: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"4.*|5.*",ResourcePath=~".*/soothsayer/v1/.*.*/ocscp/.*",namespace="$namespace"}[5m]))/ sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/ocscp/.*",namespace="$namespace"}[5m]))*100
3. nrf: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"4.*|5.*",ResourcePath=~".*/nrf-configuration/v1/.*|.*/nrf-state-data/.*|/ocnrf-swagger/.*|/nrf-status-data/.*|.*/nrf/nf-common-component/.*",namespace="$namespace"}[5m]))/ sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/nrf-configuration/v1/.*|.*/nrf-state-data/.*|.*/ocnrf-swagger/.*|.*/nrf-status-data/.*|.*/nrf/nf-common-component/.*"namespace="$namespace"}[5m]))*100
4. udr: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"4.*|5.*",ResourcePath=~".*/nudr-dr-prov/.*|.*/nudr-dr-mgm/.*|.*/nudr-group-id-map-prov/.*|.*/slf-group-prov/.*|.*/nudr-config/.*|.*/udr/nf-common-component/.*|.*/n5g-eir-prov/.*",namespace="$namespace"}[5m]))/ sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/nudr-dr-prov/.*|.*/nudr-dr-mgm/.*|.*/nudr-group-id-map-prov/.*|.*/slf-group-prov/.*|.*/nudr-config/.*|.*/udr/nf-common-component/.*|.*/n5g-eir-prov/.*",namespace="$namespace"}[5m]))*100</td></tr>
</table>

Table 9-8 (Cont.) M-CNCC Core Error Rate

<table>
<tr><th></th><th></th></tr>
<tr><td></td><td>5. policy: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"4.*|5.*",ResourcePath=~".*/policyapi/.*|.*/oc-cnpolicy-configuration/.*|.*/pcf/.*",namespace="$namespace"}[5m]))/sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/policyapi/.*|.*/oc-cnpolicy-configuration/.*|.*/pcf/.*",namespace="$namespace"}[5m]))*100
6. bsf: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"4.*|5.*",ResourcePath=~".*/bsfapi/.*|.*/oc-bsf-configuration/.*|.*/bsf/.*",namespace="$namespace"}[5m]))/sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/bsfapi.*|.*/oc-bsf-configuration/.*|.*/bsf/.*",namespace="$namespace"}[5m]))*100
7. sepp: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"4.*|5.*",ResourcePath=~".*/sepp-configuration/.*|.*/sepp/.*",namespace="$namespace"}[5m]))/sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/sepp-configuration/.*|.*/sepp/.*",namespace="$namespace"}[5m]))*100
8. nssf: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"4.*|5.*",ResourcePath=~".*/nnssf-configuration/.*|.*/nssf/.*",namespace="$namespace"}[5m]))/sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/nnssf-configuration/.*|.*/nssf/.*",namespace="$namespace"}[5m]))*100
9. dd: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"4.*|5.*",ResourcePath=~".*/ocnadd/.*|.*/ocnaddapi/.*",namespace="$namespace"}[5m]))/sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/ocnadd/.*|.*/ocnaddapi/.*"",namespace="$namespace"}[5m]))*100
10. provgw: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"4.*|5.*",ResourcePath=~".*provgw-config.*|.*provgw.*",namespace="$namespace"}[5m]))/sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Route_path=~".*pro</td></tr>
</table>

Table 9-8 (Cont.) M-CNCC Core Error Rate

<table>
<tr><th></th><th></th></tr>
<tr><td></td><td>vgw-config.*|.*provgw.*",namespace="$namespace"}[5m]))*100
11. cndbtier: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"4.*|5.*",ResourcePath=~".*ocdbtier.*",namespace="$namespace"}[5m]))/sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*ocdbtier.*",namespace="$namespace"}[5m]))*100
12. occm: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"4.*|5.*",ResourcePath=~".*occm-config.*",namespace="$namespace"}[5m]))/sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*occm-config.*",namespace="$namespace"}[5m]))*100
For OCI:
1. all: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Status=~"4*|5*",ResourcePath=~"*",k8Namespace="cncc-ns"}.increment().grouping().sum()/oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*core_ingressgateway",ResourcePath=~"*",k8Namespace="cncc-ns"}.increment().grouping().sum() * 100
2. scp: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Status=~"4*|5*",ResourcePath=~"*ocscp*",k8Namespace="cncc-ns"}.increment().grouping().sum()/oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",ResourcePath=~"*ocscp*",k8Namespace="cncc-ns"}.increment().grouping().sum() * 100
3. nrf: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Status=~"4*|5*",Route_path=~"*nrf-configuration/v1*|*nrf-state-data*|*ocnrf-swagger*|*nrf-status-data*|*nrf/nf-common-component*",k8Namespace="cncc-ns"}.increment().grouping().sum()/oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*core_ingressgateway",Route_path=~"*nrf-configuration/v1*|*nrf-state-data*|*ocnrf-swagger*|*nrf-status-data*|*nrf/nf-common-component*",k8Namespace="cncc-ns"}.increment().grouping().sum() * 100
4. udr: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Status=~"4*|5*",Route_path=~"*nudr-dr-prov*|*nudr-dr-mgm*|*nudr-group-id-map-prov*|*slf-group-prov*|*n5g-eir-prov*|*nudr-config*|*udr/nf-common-component*",k8Namespace="cncc-ns"}.increment().grouping().sum()/</td></tr>
</table>

Table 9-8 (Cont.) M-CNCC Core Error Rate

<table>
<tr><td></td><td>oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*core_ingressgateway",Route_path=~"*nudr-dr-prov*|*nudr-dr-mgm*|*nudr-group-id-map-prov*|*slf-group-prov*|*n5g-eir-prov*|*nudr-config*|*udr/nf-common-component*",k8Namespace="cncc-ns"}.increment().grouping().sum() * 100
5. policy: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Status=~"4*|5*",ResourcePath=~"*policyapi*|*oc-cnpolicy-configuration*|*pcf*",k8Namespace="cncc-ns"}.increment().grouping().sum()/oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*core_ingressgateway",ResourcePath=~"*policyapi*|*oc-cnpolicy-configuration*|*pcf*",k8Namespace="cncc-ns"}.increment().grouping().sum() * 100
6. bsf: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Status=~"4*|5*",ResourcePath=~"*bsfapi*|*oc-bsf-configuration*|*bsf*",k8Namespace="cncc-ns"}.increment().grouping().sum()/oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",ResourcePath=~"*bsfapi*|*oc-bsf-configuration*|*bsf*",k8Namespace="cncc-ns"}.increment().grouping().sum() * 100
7. sepp: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Status=~"4*|5*",Route_path=~"*sepp-configuration*|*sepp*",k8Namespace="cncc"}.increment().grouping().sum()/oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Route_path=~"*sepp-configuration*|*sepp*",k8Namespace="cncc-ns"}.increment().grouping().sum() * 100
8. nssf: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Status=~"4*|5*",Route_path=~"*nnssf-configuration*|*nssf*",k8Namespace="cncc-ns"}.increment().grouping().sum()/oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Route_path=~"*nnssf-configuration*|*nssf*",k8Namespace="cncc-ns"}.increment().grouping().sum() * 100
9. dd: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Status=~"4*|5*",Route_path=~"*ocnadd*|*ocnaddapi*",k8Namespace="cncc-ns"}.increment().grouping().sum()/oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Route_path=~"*ocnadd*|*ocnaddapi*",k8Namespace="cncc-ns"}.increment().grouping().sum() * 100
10. provgw: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*mcore_ingressgateway",Status=~"4*|5*",Route_path=~"*provgw-config*|*provgw*",k8Namespace="cncc-</td></tr>
</table>

Table 9-8 (Cont.) M-CNCC Core Error Rate

	ns"}.increment().grouping().sum()/ oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*mcore_ingressgateway",Route_path=~" *provgw-config*|*provgw*",k8Namespace="cncc-ns"}.increment().grouping().sum() * 100 11. occm: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*mcore_ingressgateway",Status=~"4*|5*",Route_path=~"*occm-config*",k8Namespace="cncc-ns"}.increment().grouping().sum()/ oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*mcore_ingressgateway",Route_path=~" *occm-config*",k8Namespace="cncc-ns"}.increment().grouping().sum() * 100

9.3 A-CNCC Core KPIs

This section provides the information about A-CNCC Core KPIs:

9.3.1 A-CNCC Core Requests

Table 9-9 A-CNCC Core Requests

Description	A-CNCC Core Requests for all as well as specific NFs

Table 9-9 (Cont.) A-CNCC Core Requests

<table>
<tr><td>Expression</td><td>1. **For CNE without Prometheus Operator**: all: sum(irate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*acore_ingressgateway",kubernetes_namespace="$namespace"}[2m]))
For CNE with Prometheus HA Operator: all: sum(irate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*acore_ingressgateway",namespace="$namespace"}[2m]))
2. scp: sum(irate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*/ocscp/.*"} [2m]))
3. nrf: sum(irate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*/nrf-configuration/v1.*|.*/nrf-state-data/.*|.*/ocnrf-swagger/.*|.*/nrf-status-data/.*|.*/nrf/nf-common-component/.*"} [2m]))
4. udr: sum(irate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*/nudr-dr-prov/.*|.*/nudr-dr-mgm/.*|.*/nudr-group-id-map-prov/.*|.*/slf-group-prov/.*|.*/nudr-config/.*|.*/udr/nf-common-component/.*|.*/n5g-eir-prov/.*"} [2m]))
5. policy: sum(irate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*acore_ingressgateway",ResourcePath=~".*/policyapi/.*|.*/oc-cnpolicy-configuration/.*|.*/pcf/.*"} [2m]))
6. bsf: sum(irate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*acore_ingressgateway",ResourcePath=~".*/bsfapi.*|.*/oc-bsf-configuration/.*|.*/bsf/.*"} [2m]))
7. sepp: sum(irate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*/sepp-configuration/.*|.*/sepp/.*"} [2m]))
8. nssf: sum(irate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*/nnssf-configuration/.*|.*/nssf/.*"} [2m]))
9. dd: sum(irate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*/ocnadd/.*|.*/ocnaddapi/.*"} [2m]))
10. provgw: sum(irate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*provgw-config.*|.*provgw.*"} [2m]))
11. cndbtier: sum(irate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*ocdbtier.*"} [2m]))</td></tr>
</table>

Table 9-9 (Cont.) A-CNCC Core Requests

<table>
<tr><td></td><td>12. occm: sum(irate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*occm-config.*"} [2m]))
For OCI:
1. scp: oc_ingressgateway_http_requests_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",ResourcePath=~"*ocscp*"}.sum()
2. nrf: oc_ingressgateway_http_requests_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Route_path=~"*nrf-configuration/v1*|*nrf-state-data*|*ocnrf-swagger*|*nrf-status-data*|*nrf/nf-common-component*"}.sum()
3. udr: oc_ingressgateway_http_requests_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Route_path=~"*nudr-dr-prov*|*nudr-dr-mgm*|*nudr-group-id-map-prov*|*slf-group-prov*|*n5g-eir-prov*|*nudr-config*|*udr/nf-common-component*"}.sum()
4. policy: oc_ingressgateway_http_requests_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",ResourcePath=~"*policyapi*|*oc-cnpolicy-configuration*|*pcf*"}.sum()
5. bsf: oc_ingressgateway_http_requests_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",ResourcePath=~"*bsfapi*|*oc-bsf-configuration*|*bsf*"}.sum()
sepp: oc_ingressgateway_http_requests_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Route_path=~"*sepp-configuration*|*sepp*"}.sum()
6. nssf: oc_ingressgateway_http_requests_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Route_path=~"*nnssf-configuration*|*nssf*"}.sum()
7. dd: oc_ingressgateway_http_requests_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Route_path=~"*ocnadd*|*ocnaddapi*"}.sum()
8. provgw: oc_ingressgateway_http_requests_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Route_path=~"*provgw-config*|*provgw*"}.sum()
9. occm: oc_ingressgateway_http_requests_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Route_path=~"*occm-config*"}.sum()</td></tr>
</table>

9.3.2 A-CNCC Core Responses

Table 9-10 A-CNCC Core Responses

Description	A-CNCC Core Responses for all as well as specific NFs

Table 9-10 (Cont.) A-CNCC Core Responses

<table>
<tr><td>Expression</td><td><b>For CNE without Prometheus Operator</b>
1. all_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path=~".*",kubernetes_namespace="$namespace"}[5m]))
1. all_error: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"4.*|5.*",Route_path=~".*",kubernetes_namespace="$namespace"}[5m]))
2. all_4xx: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"4.*",Route_path=~".*",kubernetes_namespace="$namespace"}[5m]))
3. all_5xx: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"5.*",Route_path=~".*",kubernetes_namespace="$namespace"}[5m]))
4. scp_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path=~".*/ocscp/.*",kubernetes_namespace="$namespace"}[5m]))
5. nrf_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path=~".*/nrf-configuration/v1.*|.*/nrf-state-data/.*|.*/ocnrf-swagger/.*|.*/nrf-status-data/.*|.*/nrf/nf-common-component/.*",kubernetes_namespace="$namespace"}[5m]))
6. udr_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path=~".*/nudr-dr-prov/.*|.*/nudr-dr-mgm/.*|.*/nudr-group-id-map-prov/.*|.*/slf-group-prov/.*|.*/nudr-config/.*|.*/udr/nf-common-component/.*|.*/n5g-eir-prov/.*",kubernetes_namespace="$namespace"}[5m]))
7. policy_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",ResourcePath=~".*/policyapi/.*|.*/oc-cnpolicy-configuration/.*|.*/pcf/.*",kubernetes_namespace="$namespace"}[5m]))
8. bsf_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",ResourcePath=~".*/bsfapi/.*|.*/oc-bsf-configuration/.*|.*/bsf/.*",kubernetes_namespace="$namespace"}[5m]))
9. sepp_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path</td></tr>
</table>

Table 9-10 (Cont.) A-CNCC Core Responses

<table>
<tr><th></th><th></th></tr>
<tr><td></td><td>=~".*/sepp-configuration/.*|.*/sepp/.*",kubernetes_namespace="$namespace"}[5m]))
10. nssf_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path=~".*/nnssf-configuration/.*|.*/nssf/.*",kubernetes_namespace="$namespace"}[5m]))
11. dd_success:sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path~".*/ocnadd/.*|.*/ocnaddapi/.*",kubernetes_namespace="$namespace"}[5m]))
12. provgw_success:sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path~".*provgw-config.*|.*provgw.*",kubernetes_namespace="$namespace"}[5m]))
13. cndbtier_success:sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path~".*ocdbtier.*",kubernetes_namespace="$namespace"}[5m]))
14. occm_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path=~".*occm-config.*",kubernetes_namespace="$namespace"}[5m]))
For CNE with Prometheus HA Operator:
1. all_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path=~".*",namespace="$namespace"}[5m]))
1. all_error: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"4.*|5.*",Route_path=~".*",namespace="$namespace"}[5m]))
2. all_4xx: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"4.*",Route_path=~".*",namespace="$namespace"}[5m]))
3. all_5xx: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"5.*",Route_path=~".*",namespace="$namespace"}[5m]))
4. scp_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path=~".*/ocscp/.*",namespace="$namespace"}[5m]))
5. nrf_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path=~".*/nrf-configuration/v1.*|.*/nrf-state-data/.*|.*/ocnrf-</td></tr>
</table>

Table 9-10   (Cont.) A-CNCC Core Responses

<table>
<tr><td></td><td>swagger/.*|.*/nrf-status-data/.*|.*/nrf/nf-common-component/.*",namespace="$namespace"}[5m]))
6. udr_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path=~".*/nudr-dr-prov/.*|.*/nudr-dr-mgm/.*|.*/nudr-group-id-map-prov/.*|.*/slf-group-prov/.*|.*/nudr-config/.*|.*/udr/nf-common-component/.*|.*/n5g-eir-prov/.*",namespace="$namespace"}[5m]))
7. policy_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",ResourcePath=~".*/policyapi/.*|.*/oc-cnpolicy-configuration/.*|.*/pcf/.*",namespace="$namespace"}[5m]))
8. bsf_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",ResourcePath=~".*/bsfapi/.*|.*/oc-bsf-configuration/.*|.*/bsf/.*",namespace="$namespace"}[5m]))
9. sepp_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path=~".*/sepp-configuration/.*|.*/sepp/.*",namespace="$namespace"}[5m]))
10. nssf_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path=~".*/nnssf-configuration/.*|.*/nssf/.*",namespace="$namespace"}[5m]))
11. dd_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path=~".*/ocnadd/.*|.*/ocnaddapi/.*",namespace="$namespace"}[5m]))
12. provgw_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path=~".*provgw-config.*|.*provgw.*",namespace="$namespace"}[5m]))
13. cndbtier_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path=~".*ocdbtier.*",namespace="$namespace"}[5m]))
14. occm_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path=~".*occm-config.*",namespace="$namespace"}[5m]))
For OCI:</td></tr>
</table>

Table 9-10 (Cont.) A-CNCC Core Responses

	1. all success: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*acore_ingressgateway",Status=~"2*",ResourcePath=~"*",k8Namespace="cncc-ns"}.sum() 2. all error: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*acore_ingressgateway",Status=~"4*|5*",ResourcePath=~"*",k8Namespace="cncc-ns"}.sum() 3. all 4xx: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*acore_ingressgateway",Status=~"4*",ResourcePath=~"*",k8Namespace="cncc-ns"}.sum() 4. all 5xx: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*acore_ingressgateway",Status=~"5*",ResourcePath=~"*",k8Namespace="cncc-ns"}.sum() 5. scp_success: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*acore_ingressgateway",Status=~"2*",ResourcePath=~"*ocscp*",k8Namespace="cncc-ns"}.sum() 6. nrf_success: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*acore_ingressgateway",Status=~"2*",Route_path=~"*nrf-configuration/v1*|*nrf-state-data*|*ocnrf-swagger*|*nrf-status-data*|*nrf/nf-common-component*",k8Namespace="cncc-ns"}.sum() 7. udr_success: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*acore_ingressgateway",Status=~"2*",Route_path=~"*nudr-dr-prov*|*nudr-dr-mgm*|*nudr-group-id-map-prov*|*slf-group-prov*|*n5g-eir-prov*|*nudr-config*|*udr/nf-common-component*",k8Namespace="cncc-ns"}.sum() 8. policy_success: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*acore_ingressgateway",Status=~"2*",ResourcePath=~"*policyapi*|*oc-cnpolicy-configuration*|*pcf*",k8Namespace="cncc-ns"}.sum() 9. bsf_success: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*acore_ingressgateway",Status=~"2*",ResourcePath=~"*bsfapi*|*oc-bsf-configuration*|*bsf*",k8Namespace="cncc-ns"}.sum() 10. sepp_success: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*acore_ingressgateway",Status=~"2*",Route_path=~"*sepp-configuration*|*sepp*",k8Namespace="cncc-ns"}.sum() 11. nssf_success: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*acore_ingressgateway",Status=~"2*",Route_path=~"*nnssf-configuration*|*nssf*",k8Namespace="cncc-ns"}.sum() 12. dd_success: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*acore_ingressgateway",Status=~"2*",Route_path=~"*ocnadd*|*ocnaddapi*",k8Namespace="cncc-ns"}.sum() 13. provgw_success: oc_ingressgateway_http_responses_total[10m]

Table 9-10 (Cont.) A-CNCC Core Responses

	{InstanceIdentifier=~"*acore_ingressgateway",Status=~"2*",Route_path=~"*provgw-config*|*provgw*",k8Namespace="cncc-ns"}.sum() 14. occm_success: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*acore_ingressgateway",Status=~"2*",Route_path=~"*occm-config*",k8Namespace="cncc-ns"}.sum()

9.3.3 A-CNCC Core Success Rate

Table 9-11 A-CNCC Core Success Rate

Description	A-CNCC Core Success Rate (2xx responses divided by total responses) for all as well as specific NFs

Table 9-11 (Cont.) A-CNCC Core Success Rate

<table>
<tr><td>Expression</td><td><b>For CNE without Prometheus Operator:</b>
1. all: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path=~".*",kubernetes_namespace="$namespace"}[5m]))/sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*",kubernetes_namespace="$namespace"}[5m]))*100
2. scp: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path=~".*/ocscp/.*",kubernetes_namespace="$namespace"}[5m]))/sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*/ocscp/.*",kubernetes_namespace="$namespace"}[5m]))*100
3. nrf: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path=~".*/nrf-configuration/v1/.*|.*/nrf-state-data/.*|.*/ocnrf-swagger/.*|.*/nrf-status-data/.*|.*/nrf/nf-common-component/.*",kubernetes_namespace="$namespace"}[5m]))/sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*/nrf-configuration/v1/.*|.*/nrf-state-data/.*|.*/ocnrf-swagger/.*|.*/nrf-status-data/.*|.*/nrf/nf-common-component/.*",kubernetes_namespace="$namespace"}[5m]))*100
4. udr: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path=~".*/nudr-dr-prov/.*|.*/nudr-dr-mgm/.*|.*/nudr-group-id-map-prov/.*|.*/slf-group-prov/.*|.*/nudr-config/.*|.*/udr/nf-common-component/.*|.*/n5g-eir-prov/.*",kubernetes_namespace="$namespace"}[5m]))/sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*/nudr-dr-prov/.*|.*/nudr-dr-mgm/.*|.*/nudr-group-id-map-prov/.*|.*/slf-group-prov/.*|.*/nudr-config/.*|.*/udr/nf-common-component/.*|.*/n5g-eir-prov/.*",kubernetes_namespace="$namespace"}[5m]))*100
5. policy: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",ResourcePath=~".*/policyapi/.*|.*/oc-cnpolicy-configuration/.*|.*/pcf/.*",kubernetes_namespace="$namespace"}[5m]))/sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",ResourcePath=~".*/policyapi/.*|.*/oc-cnpolicy-configuration/.*|.*/pcf/.*",kubernetes_namespace="$namespace"}[5m]))*100
6. bsf: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",ResourcePat</td></tr>
</table>

Table 9-11   (Cont.) A-CNCC Core Success Rate

<table>
<tr><td></td><td>h=~".*/bsfapi/.*|.*/oc-bsf-configuration/.*|.*/bsf/.*",kubernetes_namespace="$namespace"}[5m]))/sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",ResourcePath=~".*/bsfapi.*|.*/oc-bsf-configuration/.*|.*/bsf/.*",kubernetes_namespace="$namespace"}[5m]))*100
7. sepp: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path=~".*/sepp-configuration/.*|.*/sepp/.*",kubernetes_namespace="$namespace"}[5m]))/sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*/sepp-configuration/.*|.*/sepp/.*",kubernetes_namespace="$namespace"}[5m]))*100
8. nssf: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path=~".*/nnssf-configuration/.*|.*/nssf/.*",kubernetes_namespace="$namespace"}[5m]))/sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*/nnssf-configuration/.*|.*/nssf/.*",kubernetes_namespace="$namespace"}[5m]))*100
9. dd: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path=~".*/ocnadd/.*|.*/ocnaddapi/.*",kubernetes_namespace="$namespace"}[5m]))/sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*/ocnadd/.*|.*/ocnaddapi/.*",kubernetes_namespace="$namespace"}[5m]))*100
10. provgw: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path=~".*provgw-config.*|.*provgw.*",kubernetes_namespace="$namespace"}[5m]))/sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*provgw-config.*|.*provgw.*",kubernetes_namespace="$namespace"}[5m]))*100
11. cndbtier: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path=~".*ocdbtier.*",kubernetes_namespace="$namespace"}[5m]))/sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*ocdbtier.*",kubernetes_namespace="$namespace"}[5m]))*100</td></tr>
</table>

Table 9-11 (Cont.) A-CNCC Core Success Rate

<table>
<tr><th></th><th></th></tr>
<tr><td></td><td>12. occm: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path=~".*occm-config.*",kubernetes_namespace="$namespace"}[5m]))/sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*occm-config.*",kubernetes_namespace="$namespace"}[5m]))*100
For CNE with Prometheus HA Operator:
1. all: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*core_ingressgateway",Status=~"2.*",Route_path=~".*",namespace="$namespace"}[5m]))/sum(increase(oc_ingressgateway_http_responses_total{Route_path=~".*",namespace="$namespace"}[5m]))*100
2. scp: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*core_ingressgateway",Status=~"2.*",Route_path=~".*/ocscp/.*",namespace="$namespace"}[5m]))/sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*core_ingressgateway",Route_path=~".*/ocscp/.*",namespace="$namespace"}[5m]))*100
3. nrf: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*core_ingressgateway",Status=~"2.*",Route_path=~".*/nrf-configuration/v1/.*|.*/nrf-state-data/.*|.*/ocnrf-swagger/.*|.*/nrf-status-data/.*|.*/nrf/nf-common-component/.*",namespace="$namespace"}[5m]))/sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*core_ingressgateway",Route_path=~".*/nrf-configuration/v1/.*|.*/nrf-state-data/.*|.*/ocnrf-swagger/.*|.*/nrf-status-data/.*|.*/nrf/nf-common-component/.*",namespace="$namespace"}[5m]))*100
4. udr: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*core_ingressgateway",Status=~"2.*",Route_path=~".*/nudr-dr-prov/.*|.*/nudr-dr-mgm/.*|.*/nudr-group-id-map-prov/.*|.*/slf-group-prov/.*|.*/nudr-config/.*|.*/udr/nf-common-component/.*|.*/n5g-eir-prov/.*",namespace="$namespace"}[5m]))/sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*core_ingressgateway",Route_path=~".*/nudr-dr-prov/.*|.*/nudr-dr-mgm/.*|.*/nudr-group-id-map-prov/.*|.*/slf-group-prov/.*|.*/nudr-config/.*|.*/udr/nf-common-component/.*|.*/n5g-eir-prov/.*",namespace="$namespace"}[5m]))*100
5. policy: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*core_ingressgateway",Status=~"2.*",ResourcePath=~".*/policyapi/.*|.*/oc-cnpolicy-configuration/.*|.*/pcf/.*",namespace="$namespace"}[5m]))/sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*core_ingressgateway",ResourcePath=~".*/policyapi/.*|.*/oc-cnpolicy-</td></tr>
</table>

Table 9-11 (Cont.) A-CNCC Core Success Rate

<table>
<tr><td></td><td>configuration/.*|.*/pcf/.*",namespace="$namespace"}[5m]))*100
6. bsf: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*core_ingressgateway",Status=~"2.*",ResourcePath=~".*/bsfapi/.*|.*/oc-bsf-configuration/.*|.*/bsf/.*",namespace="$namespace"}[5m]))/sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*core_ingressgateway",ResourcePath=~".*/bsfapi.*|.*/oc-bsf-configuration/.*|.*/bsf/.*",namespace="$namespace"}[5m]))*100
7. sepp: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*core_ingressgateway",Status=~"2.*",Route_path=~".*/sepp-configuration/.*|.*/sepp/.*",namespace="$namespace"}[5m]))/sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*core_ingressgateway",Route_path=~".*/sepp-configuration/.*|.*/sepp/.*",namespace="$namespace"}[5m]))*100
8. nssf: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*core_ingressgateway",Status=~"2.*",Route_path=~".*/nnssf-configuration/.*|.*/nssf/.*",namespace="$namespace"}[5m]))/sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*core_ingressgateway",Route_path=~".*/nnssf-configuration/.*|.*/nssf/.*",namespace="$namespace"}[5m]))*100
9. dd: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*core_ingressgateway",Status=~"2.*",Route_path=~".*/ocnadd/.*|.*/ocnaddapi/.*",namespace="$namespace"}[5m]))/sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*core_ingressgateway",Route_path=~".*/ocnadd/.*|.*/ocnaddapi/.*",namespace="$namespace"}[5m]))*100
10. provgw: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*core_ingressgateway",Status=~"2.*",Route_path=~".*provgw-config.*|.*provgw.*",namespace="$namespace"}[5m]))/sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*core_ingressgateway",Route_path=~".*provgw-config.*|.*provgw.*",namespace="$namespace"}[5m]))*100
11. cndbtier: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*core_ingressgateway",Status=~"2.*",Route_path=~".*ocdbtier.*",namespace="$namespace"}[5m]))/sum(rate(oc_ingressgateway_http_responses_total{InstanceId</td></tr>
</table>

Table 9-11 (Cont.) A-CNCC Core Success Rate

	entifier=~".*core_ingressgateway",Route_path=~".*ocdbtier.*", namespace="$namespace"}[5m]))*100 12. occm: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*core_ingressgateway",Status=~"2.*",Route_path=~".*occm-config.*",namespace="$namespace"}[5m]))/ sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*core_ingressgateway",Route_path=~".*occm-config.*",namespace="$namespace"}[5m]))*100 **For OCI:** 1. all: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*acore_ingressgateway",Status=~"2*",ResourcePath=~"*",k8Namespace="cncc-ns"}.increment().grouping().sum()/ oc_ingressgateway_http_responses_total[10m] {ResourcePath=~"*",k8Namespace="cncc-ns"}.increment().grouping().sum() * 100 2. scp**:** oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*acore_ingressgateway",Status=~"2*",ResourcePath=~"*ocscp*",k8Namespace="cncc-ns"}.rate().grouping().sum()/ oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*acore_ingressgateway",ResourcePath=~"*ocscp*",k8Namespace="cncc-ns"}.rate().grouping().sum() * 100 3. nrf: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*acore_ingressgateway",Status=~"2*",Route_path=~"*nrf-configuration/v1*|*nrf-state-data*|*ocnrf-swagger*|*nrf-status-data*|*nrf/nf-common-component*",k8Namespace="cncc-ns"}.rate().grouping().sum()/ oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*core_ingressgateway",Route_path=~"*nrf-configuration/v1*|*nrf-state-data*|*ocnrf-swagger*|*nrf-status-data*|*nrf/nf-common-component*",k8Namespace="cncc-ns"}.rate().grouping().sum() * 100 4. udr: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*acore_ingressgateway",Status=~"2*",Route_path=~"*nudr-dr-prov*|*nudr-dr-mgm*|*nudr-group-id-map-prov*|*slf-group-prov*|*n5g-eir-prov*|*nudr-config*|*udr/nf-common-component*",k8Namespace="cncc-ns"}.rate().grouping().sum()/ oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*core_ingressgateway",Route_path=~"*nudr-dr-prov*|*nudr-dr-mgm*|*nudr-group-id-map-prov*|*slf-group-prov*|*n5g-eir-prov*|*nudr-config*|*udr/nf-common-component*",k8Namespace="cncc-ns"}.rate().grouping().sum() * 100 5. policy: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*acore_ingressgateway",Status=~"2*",ResourcePath=~"*policyapi*|*oc-cnpolicy-configuration*|*pcf*",k8Namespace="cncc-ns"}.rate().grouping().sum()/

Table 9-11 (Cont.) A-CNCC Core Success Rate

<table>
<tr><th></th><th></th></tr>
<tr><td></td><td>oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*core_ingressgateway",ResourcePath=~"*policyapi*|*oc-cnpolicy-configuration*|*pcf*",k8Namespace="cncc-ns"}.rate().grouping().sum() * 100
6. bsf: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Status=~"2*",ResourcePath=~"*bsfapi*|*oc-bsf-configuration*|*bsf*",k8Namespace="cncc-ns"}.rate().grouping().sum()/oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",ResourcePath=~"*bsfapi*|*oc-bsf-configuration*|*bsf*",k8Namespace="cncc-ns"}.rate().grouping().sum() * 100
7. sepp: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Status=~"2*",Route_path=~"*sepp-configuration*|*sepp*",k8Namespace="cncc-ns"}.rate().grouping().sum()/oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Route_path=~"*sepp-configuration*|*sepp*",k8Namespace="cncc"}.rate().grouping().sum() * 100
8. nssf: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Status=~"2*",Route_path=~"*nnssf-configuration*|*nssf*",k8Namespace="cncc-ns"}.rate().grouping().sum()/oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Route_path=~"*nnssf-configuration*|*nssf*",k8Namespace="cncc-ns"}.rate().grouping().sum() * 100
9. dd: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Status=~"2*",Route_path=~"*ocnadd*|*ocnaddapi*",k8Namespace="cncc-ns"}.rate().grouping().sum()/oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Route_path=~"*ocnadd*|*ocnaddapi*",k8Namespace="cncc-ns"}.rate().grouping().sum() * 100
10. provgw: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Status=~"2*",Route_path=~"*provgw-config*|*provgw*",k8Namespace="cncc-ns"}.rate().grouping().sum()/oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Route_path=~"*provgw-config*|*provgw*",k8Namespace="cncc-ns"}.rate().grouping().sum() * 100
11. occm: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Status=~"2*",Route_path=~"*occm-config*",k8Namespace="cncc-ns"}.rate().grouping().sum()/oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Route_path=~"*occm-config*",k8Namespace="cncc-ns"}.rate().grouping().sum() * 100</td></tr>
</table>

9.3.4 A-CNCC Core Error Rate

Table 9-12   A-CNCC Core Error Rate

Description	A-CNCC Core Error Rate (4xx or 5xx responses divided by total responses) for all as well as specific NFs

Table 9-12 (Cont.) A-CNCC Core Error Rate

<table>
<tr><td>Expression</td><td><b>For CNE without Prometheus Operator:</b>
1. all: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"4.*|5.*",Route_path=~".*",kubernetes_namespace="$namespace"}[5m]))/sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*",kubernetes_namespace="$namespace"}[5m]))*100
2. scp: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"4.*|5.*",Route_path=~".*/ocscp/.*",kubernetes_namespace="$namespace"}[5m]))/sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*/ocscp/.*",kubernetes_namespace="$namespace"}[5m]))*100
3. nrf: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"4.*|5.*",Route_path=~".*/nrf-configuration/v1/.*|.*/nrf-state-data/.*|/ocnrf-swagger/.*|/nrf-status-data/.*|.*/nrf/nf-common-component/.*",kubernetes_namespace="$namespace"}[5m]))/sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*/nrf-configuration/v1/.*|.*/nrf-state-data/.*|.*/ocnrf-swagger/.*|.*/nrf-status-data/.*|.*/nrf/nf-common-component/.*"kubernetes_namespace="$namespace"}[5m]))*100
4. udr: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"4.*|5.*",Route_path=~".*/nudr-dr-prov/.*|.*/nudr-dr-mgm/.*|.*/nudr-group-id-map-prov/.*|.*/slf-group-prov/.*|.*/nudr-config/.*|.*/udr/nf-common-component/.*|.*/n5g-eir-prov/.*",kubernetes_namespace="$namespace"}[5m]))/sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*/nudr-dr-prov/.*|.*/nudr-dr-mgm/.*|.*/nudr-group-id-map-prov/.*|.*/slf-group-prov/.*|.*/nudr-config/.*|.*/udr/nf-common-component/.*|.*/n5g-eir-prov/.*",kubernetes_namespace="$namespace"}[5m]))*100
5. policy: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"4.*|5.*",ResourcePath=~".*/policyapi/.*|.*/oc-cnpolicy-configuration/.*|.*/pcf/.*",kubernetes_namespace="$namespace"}[5m]))/sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",InstanceIdentifier=~".*core_ingressgateway",ResourcePath=~".*/policyapi/.*|.*/oc-cnpolicy-configuration/.*|.*/pcf/.*",kubernetes_namespace="$namespace"}[5m]))*100</td></tr>
</table>

Table 9-12 (Cont.) A-CNCC Core Error Rate

<table>
<tr><td></td><td>6. bsf: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"4.*|5.*",ResourcePath=~".*/bsfapi/.*|.*/oc-bsf-configuration/.*|.*/bsf/.*",kubernetes_namespace="$namespace"}[5m]))/sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",InstanceIdentifier=~".*core_ingressgateway",ResourcePath=~".*/bsfapi.*|.*/oc-bsf-configuration/.*|.*/bsf/.*",kubernetes_namespace="$namespace"}[5m]))*100
7. sepp: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"4.*|5.*",Route_path=~".*/sepp-configuration/.*|.*/sepp/.*",kubernetes_namespace="$namespace"}[5m]))/sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",InstanceIdentifier=~".*core_ingressgateway",Route_path=~".*/sepp-configuration/.*|.*/sepp/.*",kubernetes_namespace="$namespace"}[5m]))*100
8. nssf: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"4.*|5.*",Route_path=~".*/nnssf-configuration/.*|.*/nssf/.*",kubernetes_namespace="$namespace"}[5m]))/sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",InstanceIdentifier=~".*core_ingressgateway",Route_path=~".*/nssf-configuration/.*|.*/nssf/.*",kubernetes_namespace="$namespace"}[5m]))*100
9. dd: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"4.*|5.*",Route_path=~".*/ocnadd/.*|.*/ocnaddapi/.*",kubernetes_namespace="$namespace"}[5m]))/sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",InstanceIdentifier=~".*core_ingressgateway",Route_path=~".*/ocnadd/.*|.*/ocnaddapi/.*",kubernetes_namespace="$namespace"}[5m]))*100
10. provgw: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"4.*|5.*",Route_path=~".*provgw-config.*|.*provgw.*",kubernetes_namespace="$namespace"}[5m]))/sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",InstanceIdentifier=~".*core_ingressgateway",Route_path=~".*provgw-config.*|.*provgw.*",kubernetes_namespace="$namespace"}[5m]))*100
11. cndbtier: sum(increase(oc_ingressgateway_http_responses_total{Insta</td></tr>
</table>

Table 9-12 (Cont.) A-CNCC Core Error Rate

	nceIdentifier=~".*acore_ingressgateway",Status=~"4.*|5.*",Route_path=~".*ocdbtier.*",kubernetes_namespace="$namespace"}[5m]))/sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",InstanceIdentifier=~".*core_ingressgateway",Route_path=~".*ocdbtier.*",kubernetes_namespace="$namespace"}[5m]))*100 12. occm: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"4.*|5.*",Route_path=~".*occm-config.*",kubernetes_namespace="$namespace"}[5m]))/sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",InstanceIdentifier=~".*core_ingressgateway",Route_path=~".*occm-config.*",kubernetes_namespace="$namespace"}[5m]))*100 **For CNE with Prometheus HA Operator:** 1. all: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"4.*|5.*",Route_path=~".*",namespace="$namespace"}[5m]))/sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*",namespace="$namespace"}[5m]))*100 2. scp: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"4.*|5.*",Route_path=~".*/soothsayer/v1/.*.*/ocscp/.*",namespace="$namespace"}[5m]))/sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*/ocscp/.*",namespace="$namespace"}[5m]))*100 3. nrf: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"4.*|5.*",Route_path=~".*/nrf-configuration/v1/.*|.*/nrf-state-data/.*|/ocnrf-swagger/.*|/nrf-status-data/.*|.*/nrf/nf-common-component/.*",namespace="$namespace"}[5m]))/sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*/nrf-configuration/v1/.*\|.*/nrf-state-data/.*\|.*/ocnrf-swagger/.*\|.*/nrf-status-data/.*\|.*/nrf/nf-common-component/.*"namespace="$namespace"}[5m]))*100 4. udr: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"4.*|5.*",Route_path=~".*/nudr-dr-prov/.*|.*/nudr-dr-mgm/.*|.*/nudr-group-id-map-prov/.*|.*/slf-group-prov/.*|.*/nudr-config/.*|.*/udr/nf-common-component/.*|.*/n5g-eir-prov/.*",namespace="$namespace"}[5m]))/sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*/nudr-dr-prov/.*|.*/nudr-dr-mgm/.*|.*/nudr-group-id-map-prov/.*|.*/slf-group-prov/.*|.*/nudr-config/.*|.*/udr/nf-common-

Table 9-12 (Cont.) A-CNCC Core Error Rate

<table>
<tr><th></th><th></th></tr>
<tr><td></td><td>component/.*|.*/n5g-eir-prov/.*",namespace="$namespace"}[5m]))*100
5. policy: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"4.*|5.*",ResourcePath=~".*/policyapi/.*|.*/oc-cnpolicy-configuration/.*|.*/pcf/.*",namespace="$namespace"}[5m]))/ sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",InstanceIdentifier=~".*core_ingressgateway",ResourcePath=~".*/policyapi/.*|.*/oc-cnpolicy-configuration/.*|.*/pcf/.*",namespace="$namespace"}[5m]))*100
6. bsf: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"4.*|5.*",ResourcePath=~".*/bsfapi/.*|.*/oc-bsf-configuration/.*|.*/bsf/.*",namespace="$namespace"}[5m]))/ sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",InstanceIdentifier=~".*core_ingressgateway",ResourcePath=~".*/bsfapi.*|.*/oc-bsf-configuration/.*|.*/bsf/.*",namespace="$namespace"}[5m]))*100
7. sepp: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"4.*|5.*",Route_path=~".*/sepp-configuration/.*|.*/sepp/.*",namespace="$namespace"}[5m]))/ sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",InstanceIdentifier=~".*core_ingressgateway",Route_path=~".*/sepp-configuration/.*|.*/sepp/.*",namespace="$namespace"}[5m]))*100
8. nssf: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"4.*|5.*",Route_path=~".*/nnssf-configuration/.*|.*/nssf/.*",namespace="$namespace"}[5m]))/ sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",InstanceIdentifier=~".*core_ingressgateway",Route_path=~".*/nnssf-configuration/.*|.*/nssf/.*",namespace="$namespace"}[5m]))*100
9. dd: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"4.*|5.*",Route_path=~".*/ocnadd/.*|.*/ocnaddapi/.*",namespace="$namespace"}[5m]))/ sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",InstanceIdentifier=~".*core_ingressgateway",Route_path=~".*/ocnadd/.*|.*/ocnaddapi/.*"",namespace="$namespace"}[5m]))*100
10. provgw: sum(increase(oc_ingressgateway_http_responses_total{Insta</td></tr>
</table>

Table 9-12 (Cont.) A-CNCC Core Error Rate

<table>
<tr><td></td><td>nceIdentifier=~".*acore_ingressgateway",Status=~"4.*|5.*",Route_path=~".*provgw-config.*|.*provgw.*",namespace="$namespace"}[5m]))/sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",InstanceIdentifier=~".*core_ingressgateway",Route_path=~".*provgw-config.*|.*provgw.*",namespace="$namespace"}[5m]))*100
11. cndbtier: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"4.*|5.*",Route_path=~".*ocdbtier.*",namespace="$namespace"}[5m]))/sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",InstanceIdentifier=~".*core_ingressgateway",Route_path=~".*ocdbtier.*",namespace="$namespace"}[5m]))*100
12. occm: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"4.*|5.*",Route_path=~".*occm-config.*",namespace="$namespace"}[5m]))/sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",InstanceIdentifier=~".*core_ingressgateway",Route_path=~".*occm-config.*",namespace="$namespace"}[5m]))*100
13.
For OCI:
1. all: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Status=~"4*|5*",ResourcePath=~"*",k8Namespace="cncc-ns"}.increment().grouping().sum()/oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*core_ingressgateway",ResourcePath=~"*",k8Namespace="cncc-ns"}.increment().grouping().sum() * 100
2. scp**:** oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Status=~"4*|5*",ResourcePath=~"*ocscp*",k8Namespace="cncc-ns"}.increment().grouping().sum()/oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",ResourcePath=~"*ocscp*",k8Namespace="cncc-ns"}.increment().grouping().sum() * 100
3. nrf: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Status=~"4*|5*",Route_path=~"*nrf-configuration/v1*|*nrf-state-data*|*ocnrf-swagger*|*nrf-status-data*|*nrf/nf-common-component*",k8Namespace="cncc-ns"}.increment().grouping().sum()/oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*core_ingressgateway",Route_path=~"*nrf-configuration/v1*|*nrf-state-data*|*ocnrf-swagger*|*nrf-status-data*|*nrf/nf-common-</td></tr>
</table>

Table 9-12 (Cont.) A-CNCC Core Error Rate

<table>
<tr><th></th><th></th></tr>
<tr><td></td><td>component*",k8Namespace="cncc-ns"}.increment().grouping().sum() * 100
4. udr: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Status=~"4*|5*",Route_path=~"*nudr-dr-prov*|*nudr-dr-mgm*|*nudr-group-id-map-prov*|*slf-group-prov*|*n5g-eir-prov*|*nudr-config*|*udr/nf-common-component*",k8Namespace="cncc-ns"}.increment().grouping().sum()/oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*core_ingressgateway",Route_path=~"*nudr-dr-prov*|*nudr-dr-mgm*|*nudr-group-id-map-prov*|*slf-group-prov*|*n5g-eir-prov*|*nudr-config*|*udr/nf-common-component*",k8Namespace="cncc-ns"}.increment().grouping().sum() * 100
5. policy: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Status=~"4*|5*",ResourcePath=~"*policyapi*|*oc-cnpolicy-configuration*|*pcf*",k8Namespace="cncc-ns"}.increment().grouping().sum()/oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*core_ingressgateway",ResourcePath=~"*policyapi*|*oc-cnpolicy-configuration*|*pcf*",k8Namespace="cncc-ns"}.increment().grouping().sum() * 100
6. bsf: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Status=~"4*|5*",ResourcePath=~"*bsfapi*|*oc-bsf-configuration*|*bsf*",k8Namespace="cncc-ns"}.increment().grouping().sum()/oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",ResourcePath=~"*bsfapi*|*oc-bsf-configuration*|*bsf*",k8Namespace="cncc-ns"}.increment().grouping().sum() * 100
7. sepp: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Status=~"4*|5*",Route_path=~"*sepp-configuration*|*sepp*",k8Namespace="cncc"}.increment().grouping().sum()/oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Route_path=~"*sepp-configuration*|*sepp*",k8Namespace="cncc-ns"}.increment().grouping().sum() * 100
8. nssf: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Status=~"4*|5*",Route_path=~"*nnssf-configuration*|*nssf*",k8Namespace="cncc-ns"}.increment().grouping().sum()/oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Route_path=~"*nnssf-configuration*|*nssf*",k8Namespace="cncc-ns"}.increment().grouping().sum() * 100
9. dd: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Status=~"4*|5*",Route_path=~"*ocnadd*|*ocnaddapi*",k8Namespace="cncc-ns"}.increment().grouping().sum()/</td></tr>
</table>

Table 9-12 (Cont.) A-CNCC Core Error Rate

	oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Route_path=~"*ocnadd*|*ocnaddapi*",k8Namespace="cncc-ns"}.increment().grouping().sum() * 100 10. provgw: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Status=~"4*|5*",Route_path=~"*provgw-config*|*provgw*",k8Namespace="cncc-ns"}.increment().grouping().sum()/oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Route_path=~"*provgw-config*|*provgw*",k8Namespace="cncc-ns"}.increment().grouping().sum() * 100 11. occm: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Status=~"4*|5*",Route_path=~"*occm-config*",k8Namespace="cncc-ns"}.increment().grouping().sum()/oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Route_path=~"*occm-config*",k8Namespace="cncc-ns"}.increment().grouping().sum() * 100

9.4 CNC Console Resource Usage KPIs

This section provides the information about CNC Console common KPIs:

9.4.1 CPU Usage

Table 9-13 CPU Usage

Description	CPU usage by all as well as individual microservices in CNCC

Table 9-13 (Cont.) CPU Usage

Expression	1. all: sum(rate(container_cpu_usage_seconds_total{container!="",pod=~".*",namespace="$namespace"}[2m])) 2. cncc-core-cmservice: sum(rate(container_cpu_usage_seconds_total{container!="",pod=~".*core-cmservice.*",namespace="$namespace"}[2m])) 3. cncc-mcore-ingress: sum(rate(container_cpu_usage_seconds_total{container!="",pod=~".*mcore-ingress.*",namespace="$namespace"}[2m])) 4. cncc-acore-ingress: sum(rate(container_cpu_usage_seconds_total{container!="",pod=~".*acore-ingress.*",namespace="$namespace"}[2m])) 5. cncc-iam-ingress: sum(rate(container_cpu_usage_seconds_total{container!="",pod=~".*iam-ingress.*",namespace="$namespace"}[2m])) 6. cncc-iam-kc: sum(rate(container_cpu_usage_seconds_total{container!="",pod=~".*iam-kc.*",namespace="$namespace"}[2m])) **For OCI:** 1. all: container_cpu_usage_seconds_total[10m]{container!=\"POD\",pod=~\"*\",namespace=\"cncc-ns\"}.rate().groupBy(pod,namespace).sum()

9.4.2 Memory Usage

Table 9-14 Memory Usage

Description	Memory usage by all as well as individual microservices in CNCC

Table 9-14 (Cont.) Memory Usage

Expression	1. all: sum (container_memory_usage_bytes{container!="", pod=~".*",namespace="$namespace"}) 2. cncc-core-cmservice: sum (container_memory_usage_bytes{container!="", pod=~".*core-cmservice.*",namespace="$namespace"}) 3. cncc-mcore-ingress: sum (container_memory_usage_bytes{container!="", pod=~".*mcore-ingress.*",namespace="$namespace"}) 4. cncc-acore-ingress: sum (container_memory_usage_bytes{container!="", pod=~".*acore-ingress.*",namespace="$namespace"}) 5. cncc-iam-ingress: sum (container_memory_usage_bytes{container!="", pod=~".*iam-ingress.*",namespace="$namespace"}) 6. cncc-iam-kc: sum (container_memory_usage_bytescontainer!="", {pod=~".*iam-kc.*",namespace="$namespace"}) **For OCI:** 1. all: container_memory_usage_bytes[10m]{container!=\"POD\",namespace=\"cncc-ns\", pod=~\"*\"}.max()