Oracle® Communications

Cloud Native Configuration Console User Guide

Release 25.1.200
G29086-01
May 2025

Oracle Communications Cloud Native Configuration Console User Guide, Release 25.1.200

G29086-01

Contents

7 CNC Console Metrics

9 CNC Console KPIs

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
CAPIF	Common API Framework
cnDBTier	Oracle Communications Cloud Native Core, cnDBTier
CNC Console	Oracle Communications Cloud Native Configuration Console
CNE	Oracle Communications Cloud Native Core, Cloud Native Environment
CNI	Container Network Interface
CNLB	Cloud Native Load Balancer
CS	Common Service
CRUD Operations	CREATE, READ, UPDATE, DELETE
ECDSA	Elliptic Curve Digital Signature Algorithm
EIR	Equipment Identity Register
HTTPS	Hypertext Transfer Protocol Secure
GRR	Geo Replication Recovery
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)
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 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 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 Kubernetes cluster	Kubernetes cluster hosting M-CNCC
MC	Multi Cluster. In multi cluster, a single CNCC can manage NF instances that accessess different Kubernetes clusters.
MO	Mananged Objects
MOS	My Oracle Support
mTLS	Mutual Transport Layer Security
NEF	Oracle Communications Cloud Native Core, Network Exposure Function
NRF	Oracle Communications Cloud Native Core, Network Repository Function
OCI	Oracle Cloud Infrastructure
OCNADD	Oracle Communications Network Analytics Data Director
OCNWDAF	Oracle Communications Networks Data Analytics Function
OCNF	Oracle Communications Network Function
OSDC	Oracle Software Delivery Cloud
OSO	Oracle Communications Operations Services Overlay
PROVGW	Provisioning Gateway
REST API	Representational State Transfer Application Programming Interface
RBAC	Role Based Access Control
SAML	Security Assertion Markup Language
SBA	Service Based Architecture
SBI	Service Based Interface
SCP	Oracle Communications Cloud Native Core, Service Communication Proxy
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 25.1.2xx.
Release 25.1.200 - G29086-01, July 2025

- Added the following sections for the LCM Automation feature:
  - Added the LCM Automation feature description.
  - Updated the following section:
    * CNC Console Alert configuration in Prometheus
- Updated the namespace name to cncc-ns in the KPI expression for OCI in the following:
  - Memory Usage
  - CPU Usage

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 (CNCC Core)**: CNCC Core acts as Graphical User Interface (GUI) or Application Programming Interface (API) portal for NFs and Oracle Communications Cloud Native Environment (OCCNE) common services. CNCC Core module is the part of CNC Console that integrates with other cloud native core network functions.
- **CNC Console Identity and Access Management (CNCC IAM)**: CNCC IAM acts as local identity provider and as a broker for external identity provider. CNCC IAM module includes the 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 point platform for all Oracle products. Thus, deploying the CNC NFs on the OCI allows operational efficiency. In addition to this, the OCI also provides reduced Total Cost of Ownership (TCO), better 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 (CNCC Core) module is applicable for OCI deployment. CNC Console IAM is not applicable for OCI deployment. OCI IAM is used instead.

1.1 Reference

Refer to the following documents for more information:

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

- *Oracle Communications Cloud Native Core, OCI Adaptor User Guide*
- *Oracle Communications Cloud Native Core, 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 Installation, Upgrade, and Fault Recovery 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 sections 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. The OCNADD can filter, replicate, aggregate data, and route these data feeds to third party consumers who have subscribed to the feeds. OCNADD ensures data security, low latency, and redundancy while collecting and processing the data feeds. The OCNADD 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 monetizing, providing good quality service for the user, reducing downtime, ensuring easy network scalability, and minimizing losses. The OCNADD generated data feed can be beneficial for monitoring and troubleshooting during a network failure. OCNADD is a crucial function that aids in creating 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 provider's NF instances in the 5G core network.
- Allows NF instances to track the status of other NF instances.
- Provides OAuth2 based Access Token service for consumer NF authorization.
- Provides specific NF Type selection based on subscriber identity.
- Supports forwarding of messages from one NRF to another NRF.
- Supports georedundancy to ensure 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)

NSSF provides the following functions:

Network slices allow the users to select customized networks with different functionalities (such as mobility) and performance requirements (such as latency, availability, and reliability). Network slices differ in features supported and NF optimizations. In such cases, network slices may have different S-NSSAIs with different slice and service types. The user can deploy instances of multiple network slices delivering the same features but for different groups of User Equipments (UEs). These instances deliver different committed services as they are dedicated to a customer. The network slices may have different S-NSSAIs with the same slice or service type but different slice differentiators. The NSSF fulfills the requirement for determining the individual NF pertaining to a 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 Networks Data Analytics Function (NWDAF)

The NWDAF enables the operator to collect and analyze the data in the network through an analytics function. The 5G technology requires prescriptive analytics to drive closed-loop automation and self-healing networks. In a 5G network, the consumers of data are 5G NFs, Application Functions (AFs), and Operations, Administration, and Maintenance (OAM) and the data producers are NFs. The NWDAF supports the following functions:

- NWDAF collects data from Access and Mobility Management Function (AMF), Session Management Function (SMF), and Network Repository Function (NRF) in the network. The data is collected directly from the NFs or through the Network Exposure Function (OCNEF).
- The NWDAF is designed to provide analytics information to consumer such as NFs, AFs and OAM.

A 5G network contains a vast number of devices and sensors generating an enormous amount of data. The NWDAF function allows the Communications Service Providers (CSPs) to efficiently monitor, manage, automate, and optimize their network operations by the data collected and analytics generated across the network. The NWDAF also helps the CSPs in achieving the operational efficiency and provides an enhanced service experience.

The analytics information provided by the NWDAF is either statistical information based on past events or predictive information. This analytics information is used to balance the resources on the network. The NWDAF can predict the User Equipment (UE) location and also detect if the UE is in an abnormal location. Based on the collected analytics information, the CSPs can roll out new services or modify the existing services without waiting for a maintenance window in the network. This ensures significantly fewer chances of network experiencing downtime.

An NWDAF consumer can avail analytics information for different analytic events. Alternatively, the consumers can subscribe or unsubscribe for specific analytics information as a one-time event or periodically get notified when a specifically defined event (for example, a threshold is breached) is detected.

The NRF discovers the NWDAF instances for the NF consumers in the network. The NWDAF information can also be locally configured on the NF consumers. The NWDAF selection function in the consumer NF selects an NWDAF instance among available NWDAF instances. Different NWDAF instances present in the 5G network can be configured to provide a specific type of analytics information. This information about the NWDAF instance is described in the

NWDAF profile stored in the NRF. The consumer NFs that need specific analytics types query the NRF and include the Analytics ID based on the required data.

Oracle Communications Cloud Native Core, Converged Policy (Policy)

Policy is an NF for policy control decision 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 Equipement (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 Gateway goes down, MTAS can uses 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 solution for managing the certificates needed for Oracle 5G Network Functions (NFs). OCCM constantly monitors and renews the certificates based on their validity or expiry period.

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

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.*

Oracle Communications Cloud Native Core, Network Exposure Function (NEF)

Oracle Communications Cloud Native Core, Network Exposure Function (NEF) is a key component of the 5G Service Based Architecture. It provides a platform to securely expose the network services and capabilities offered by the 5G Network Functions (NFs) to either third-party applications or the internal Application Functions (AFs). Located between the 5G core network and third-party applications or AFs, NEF enables the external application administrators to customize the network for providing innovative services to their end-users. The applications communicate through NEF to access the internal data of the 5G core network.

NEF performs the following functions:

- Facilitates robust and secure exposure of network services, such as voice, data connectivity, charging, subscriber data, IoT, and so on to trusted third-party applications or AFs.
- Provides programmable environment access of 5G network to both internal and external application administrators through a set of northbound RESTful APIs.
- Enables AF to securely provide information to 3GPP network to authenticate, authorize, and assist in throttling the AF.
- Translates the information received from the AF to the internal 3GPP NFs, and vice versa.
- Provides support to expose information collected from other 3GPP NFs to the AF.
- Monitors User Equipment (UEs) related events present in the 5G system and makes the event information available for external exposure. For example, monitoring of user location and services.

CNC Console provides an interface to configure different services in NEF. For more information, see *Oracle Communications Cloud Native Core, Network Exposure Function User Guide*.

Oracle Communications Common API Framework (CAPIF)

Oracle Communications Common API Framework (CAPIF) is the service interface between NEF and the external third-party applications or internal Application Functions (AFs). CAPIF is a 3GPP defined secured framework to expose network service interfaces. It enables the API invokers (external applications) to discover and communicate with service APIs of the API

provider (NEF). This framework manages API security, logging of events, auditing capability, multiple service exposure, policy based routing, dynamic routing of information, and so on.

CAPIF performs the following services:

- API Manager: Responsible for managing the registration and publish functionality of NEF. The API Provider Domain (APD) manager service handles all the transactions related to NEF using the package services of the CAPIF that are useful for the core functionality of NEF.
- AF Manager: Responsible for the secured interactions between API Invokers (external applications) and API Provider (NEF). This service facilitates the following tasks:
  - API Invoker onboarding and offboarding by establishing a communication between API Invoker and CAPIF.
  - Security Context creation. The API invoker negotiates and obtains the information about service API security method from the AF manager service.
- Event Manager: Manages the subscription, unsubscription, and notification for all the events supported by CAPIF. This service facilitates AFs and other NEF services to subscribe to CAPIF specific event notifications, receive notifications about the subscribed events, and unsubscribe from the notifications.
- External Ingress Gateway: Acts as a gateway for all the HTTP requests towards CAPIF from external applications. This service provides security and load balancing functionality to manage and control the incoming traffic.
- External Egress Gateway: Acts as a gateway for all the HTTP requests from CAPIF to external applications. This service provides security and load balancing functionality to manage and control the outgoing traffic.
- Network Ingress Gateway: Acts as a gateway for all the HTTP requests towards CAPIF from NEF.
- Network Egress Gateway: Acts as a gateway for all the HTTP requests from CAPIF to NEF.

CNC Console provides an interface to configure different services in CAPIF. For more information, see *Oracle Communications Cloud Native Core, Network Exposure Function 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.
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 leverages 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 is deployed 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

This feature can be configured 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 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.

2.3 Multiple Admin support for CNCC 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 to 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). For all the admin accounts managed in CNC Console, all the existing measurements, alarms, logging mechanisms, etc. would be applicable to all admin users similar to existing Core users.

There are default (master) password policies defined for CNCC IAM users in the default realm. The option is provided to enable the policy during the installation.

Managing Multiple Admin support for CNCC IAM deployments

Enable and Configure
During the deployment, a default admin user is created. To create multiple CNCC IAM admin user refer 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 cnDBTier APIs are read only and can be viewed in CNC Console:

- Backup List: This API displays the details of stored backups, such as the ID and size of the backup.
- cnDBTier version: This API displays the cnDBTier version.
- Database Statistics Report: This API displays the number of available database.
- Georeplication Status:
  - Real-Time Overall Replication Status: This API 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: This API displays the site-specific replication status.
- Georeplication Recovery:
  - Update Cluster as Failed: This API is used to mark the the disruped cluster as failed.
  - Start Georeplication Recovery: This API is used to initiate georeplication recovery.
  - Georeplication Recovery Status: This API is used to monitor the georeplication recovery status.
- HeartBeat Status: This API displays the connectivity status between the local site and the remote site name to which CNC Console is connected.
- Local Cluster Status: This API displays the status of the local cluster.
- On-Demand Backup: This API displays the status of initiated on-demand backups.

- cnDBTier Health: This API displays the health status of the following services:
  - Replication Health Status: This API displays the health status of the replication service. It checks the following:
    * if the replication service is up or not
    * if the replication service can connect to database or not
  - Monitor Health Status: This API displays the health status of the monitor service. It checks the following:
    * if the monitor service is up or not
    * if the service can connect to database or not
    * if the metrics are fetched or not (the metrics are fetched when the service is up and vice versa)
  - NDB Health Status: This API 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 or not
    * if the pods status is up or not

    > **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: This API displays the health status of the backup manager service. It checks the following:
    * if the backup manager service is up or not
    * if the service can connect to database or not

Managing cnDBTier APIs at CNC Console

Enable

This feature is enabled automatically when cnDBTier is configured as an instance during the CNC Console deployment. 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.

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.

OSO Common Services

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

Managing Common Service Support (OSO Cards and CNE Cards)

Enable and Configure

Prometheus and Alertmanager GUIs can be accessed using the CNC Console. For more information on accessing Prometheus and Alertmanager GUIs 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:

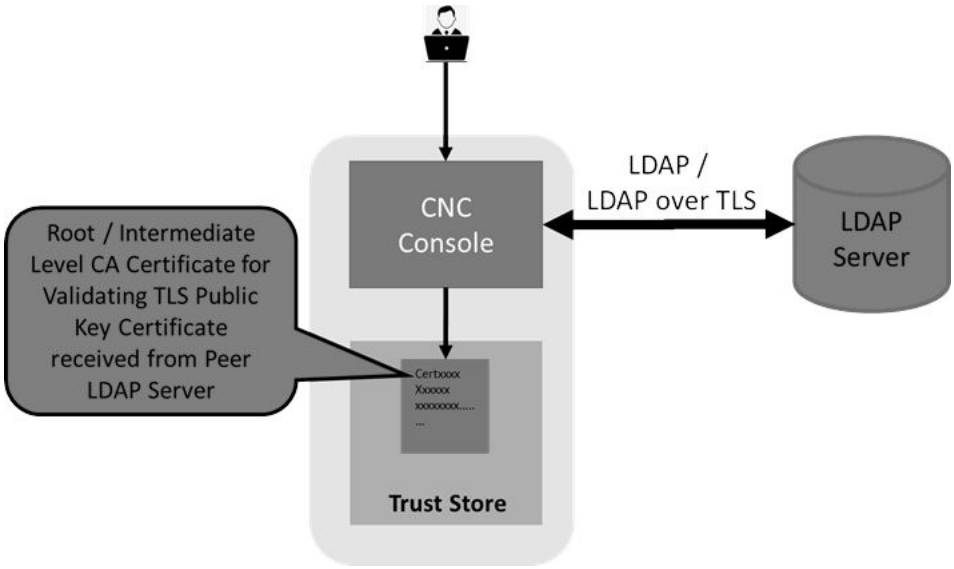
1. **Collect the logs:** For more information on how to collect logs, see CNC Console Logs.
2. **Raise a service request:** See My Oracle Support for more information on how to raise a service request.

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 2-1 LDAP



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

For enabling and configuring LDAP integration feature, see Integrating CNC Console LDAP Server with CNC Console IAM

Observe

For information on Metrics and KPIs, seeCNC 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.

2.7 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

For enabling and configuring 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.
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 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.
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.

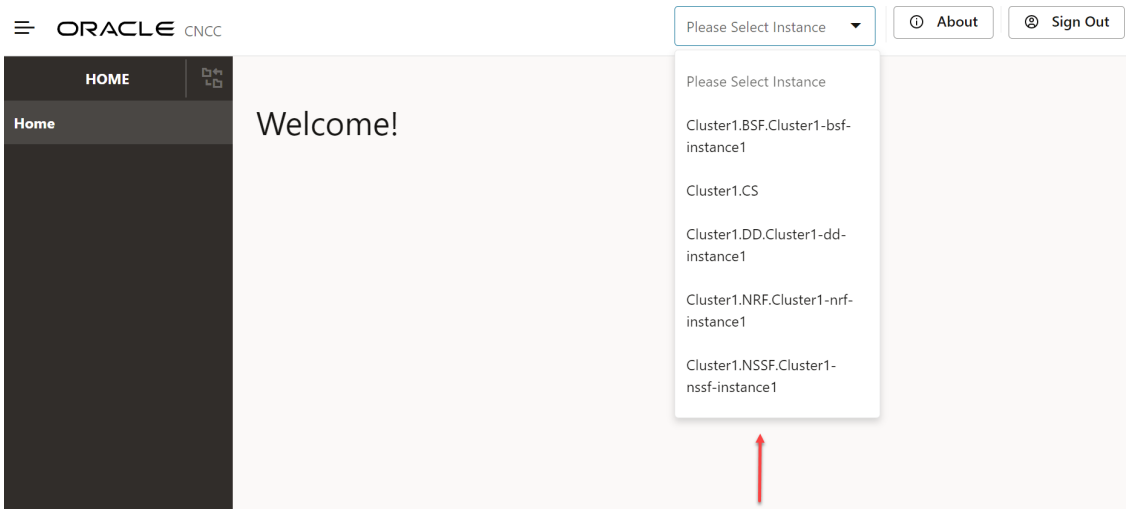
> **Note**
>
> The user must be assigned the cluster role to access multiple clusters. For more information, see Types of Roles in CNC Console.

Selecting the Instance

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

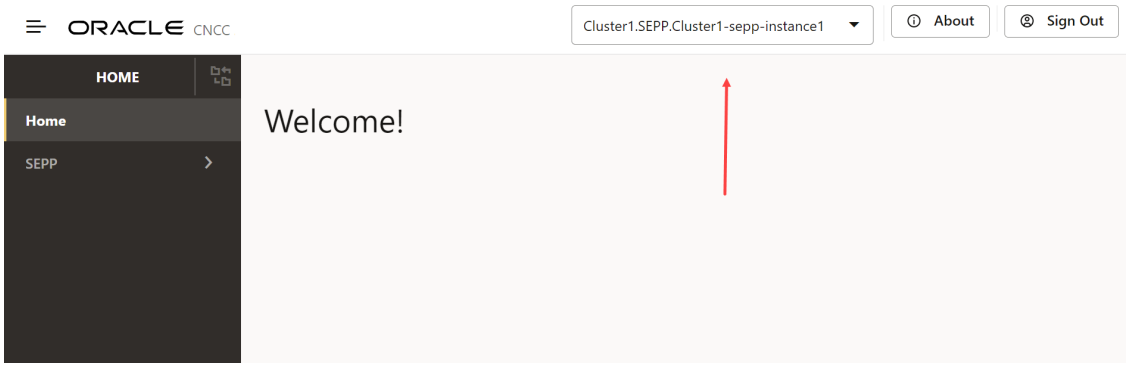
Values configured in M-CNCC Core instances section get displayed in the drop down. The naming convention used for instance drop-down display is <owner>.<type>.<instance id>

Figure 2-2 Selecting the Instance



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

Figure 2-3 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:

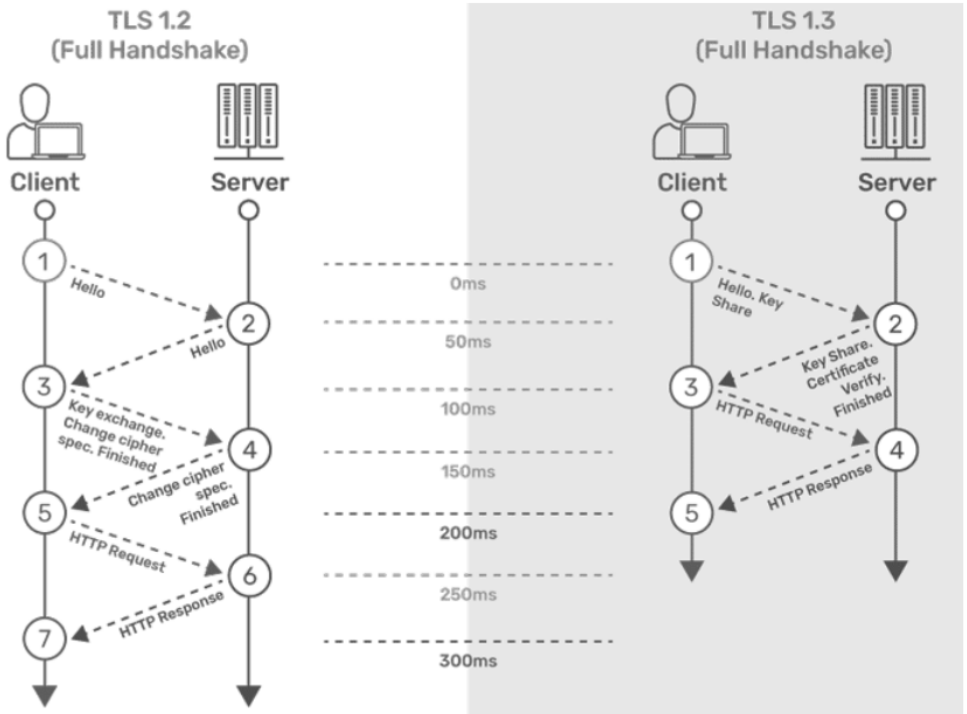
- **Handshake Protocol**: Exchanges the security parameters of a connection. Handshake messages are supplied to the TLS record layer.

- **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 2-1 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 2-2 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.

Table 2-2 (Cont.) Comparison of TLSv1.2 and TLSv1.3

Feature	TLS v1.2	TLS v1.3
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 2-3 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 "Customizong CNC Console section in the *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 mterics 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.

2.12 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.

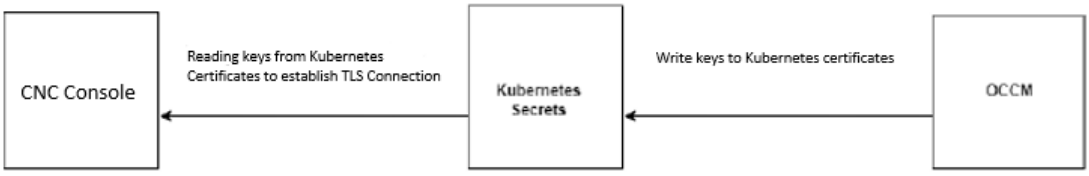
Support for OCCM

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 2-4 Establishing Connection Between CNC Console and TLS Using OCCM Key



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 privilege 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 Customising 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 2-4 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 time out 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 stop users from accessing the instances that they are not entitled to. 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 controled 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 acces, 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 allow 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 applied.

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.

The feature supports the configuration of separate networks, Network Attachment Definitions (NADs), and the Cloud Native Load Balancer (CNLB). These configurations are crucial for enabling cloud native load balancing, facilitating ingress-egress traffic separation, and optimizing load distribution within 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 alternate to the existing LBVM, lb-controller, and egress-controller solutions. You can enable or disable this feature only during a fresh CNE installation. When this feature is enabled, CNE automatically uses CNLB to control ingress traffic. To manage the egress traffic, you must preconfigure the egress network details in the `cnlb.ini` file before installing CNE.

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

Network Attachment Definitions for CNLB

A Network Attachment Definition (NAD) is a resource used to set up a network attachment, in this case, a secondary network interface to a pod. 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

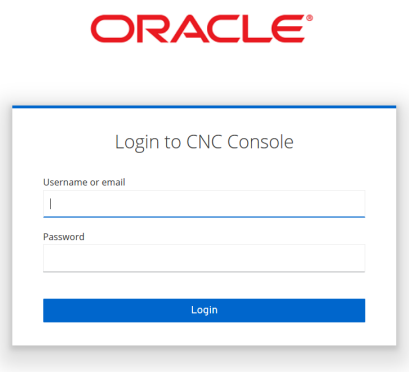
Logging into CNC Console GUI for Non-OCI Deployment

The following is the procedure to log in to CNC Console:

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

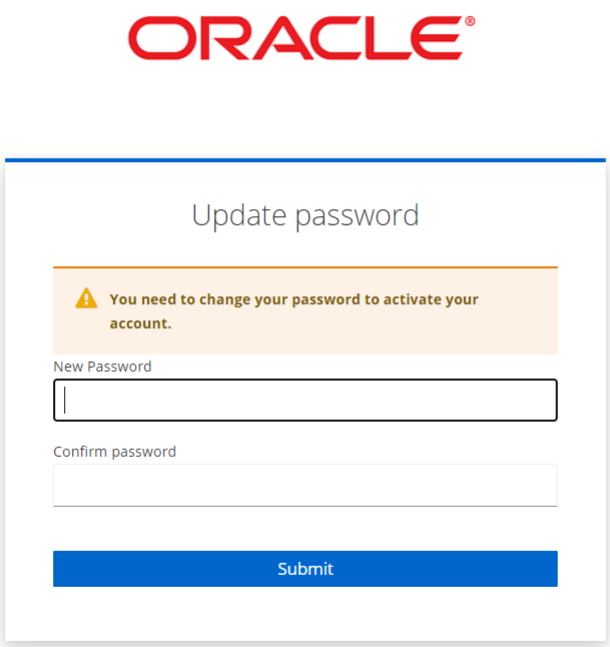
   The following screen appears:

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 password.

Figure 3-2 Update Password



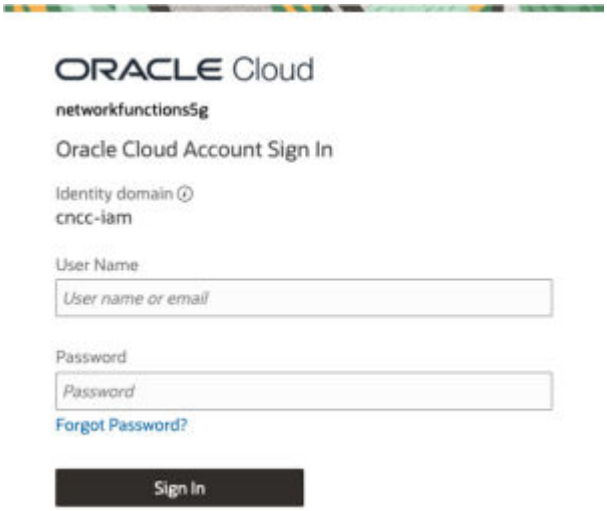
> **Note**
>
> To configure CNC Console IAM, see Configuring CNC Console IAM section.

Logging into CNC Console GUI in OCI deployment

The following is the procedure to log in to CNC Console:

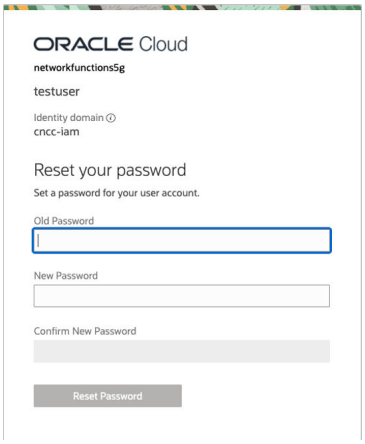
1. Open any browser.
2. Enter the M-CNCC Core URL ***http://<host name>:<port number>***.
3. Enter valid credentials.

Figure 3-3 Login page



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

Figure 3-4 Reset password



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

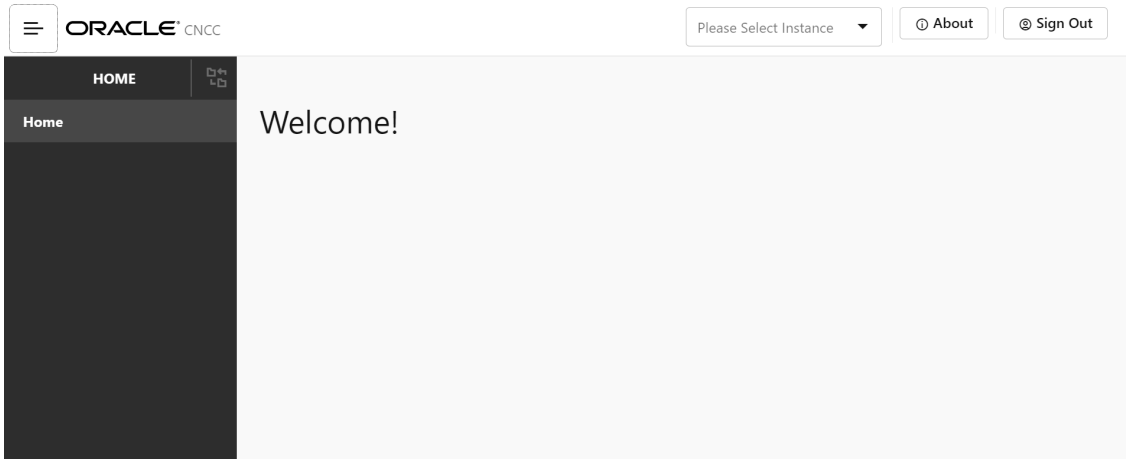
> **Note**
>
> If any issues are identified while accessing the CNC Console GUI, see the *Oracle Communications Cloud Native Configuration Console Troubleshooting Guide.*

3.1 CNC Console Dashboard

This section provides an overview of the CNC Console Dashboard.

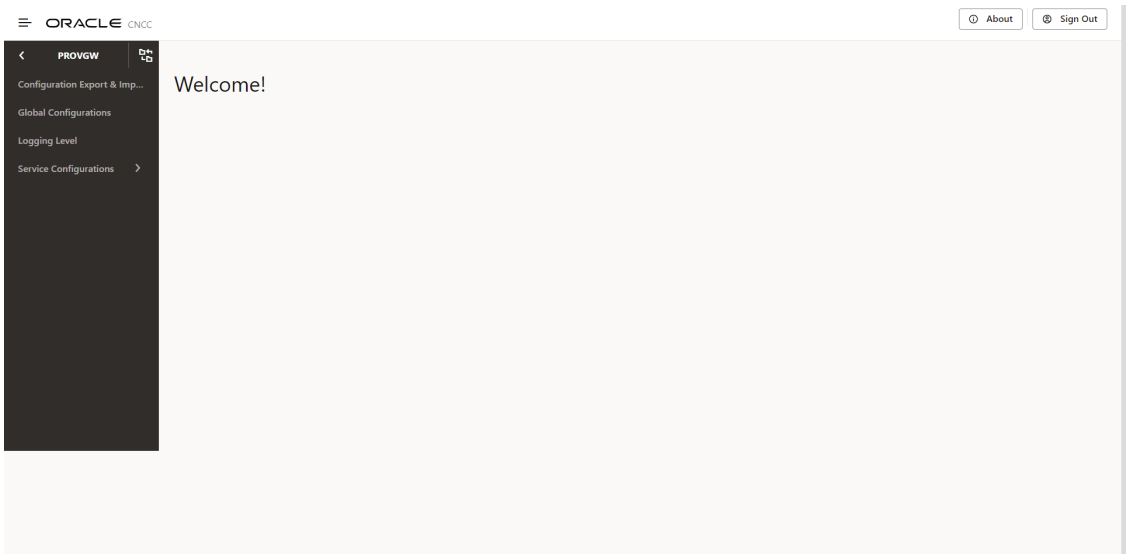
After the user logs in, the following welcome page of CNC Console appears:

Figure 3-5 CNC Console Welcome Page



- **Top Ribbon**-The top ribbon contains the following:
  - **Instance selection drop-down list**- To select the NF instance from the list.
  - **About**- Conveys the product name and the version of the Interface.
  - **Sign Out**- To sign out from the Console.
- **Left Pane - NFs and Configurations:**
  The left pane displays the selected Network Function and the configurations.
- **Right Pane - Details View:**
  The right pane displays the configurable parameters that can be updated in the selected NFs.
- **Other dashboard options:**

Figure 3-6 Dashboard Options



- The **Menu Hierarchy** button shows the navigation path from the home screen to the current menu item.
- The **Application Navigation** button allows the user to collapse the left pane and display full screen.
- The **Back Icon** allows the user to navigate to the Home menu. The screen does not get refreshed automatically. User must click Home or 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 the **Backup List** to view the list of completed backups along with 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 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

Perform 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

Perform the following procedure to view the available database.

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 database.

Table 3-3 Database Statistics Report

Fields	Description
Database Count	This field displays the number of available database.
Database Tables Count	This field displays the available database names and their table count.
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 on **View** icon 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

Perform 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 the **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 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

Perform the following procedure to mark sites as failed, perform georeplication recovery, and monitor its 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.
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

Perform the following procedure to view the available database.

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	The field displays the status of remote transfer.
Initiate Backup	The field displays whether the backup is initiated or not.

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

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

b. Enable the Initiate Backup option click **Save**.
A confirmation message "Save successfully" appears.

c. Click the**Cancel** to navigate back to the On Demand Backup screen.

d. Click the **Refresh** to reload the On Demand Backup screen.

cnDBTier Health

Perform 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 like 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 micorservices. 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

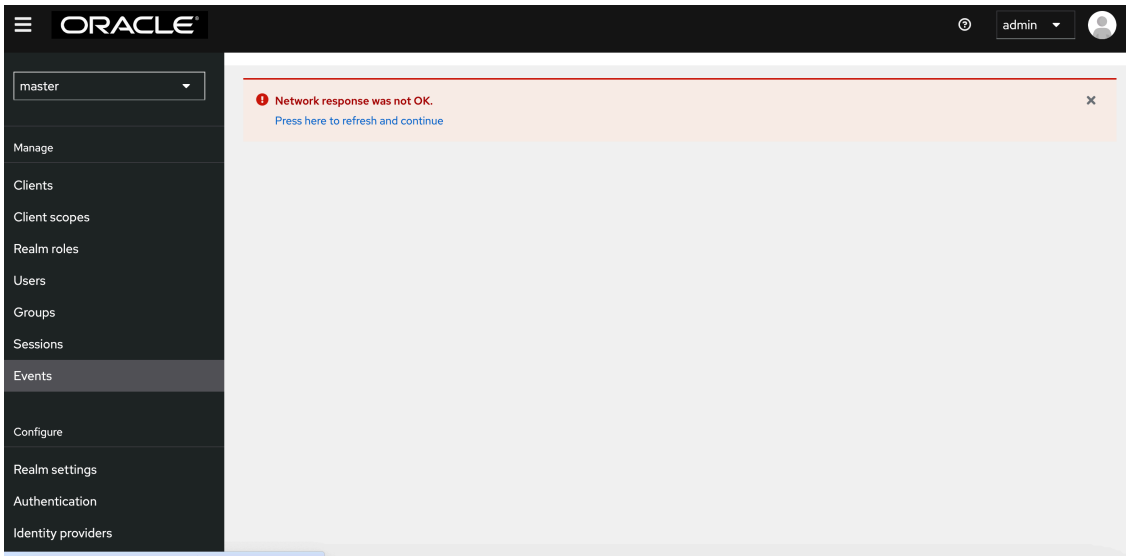
> **Note**
>
> Not applicable for OCI deployment.

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 try to perform a restricted action:

Figure 4-1 Restrict Access



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

4.1 Role Based Acess 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 to the 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 controlled by Identity and Access Management (IAM) functionality provided by CNC Console IAM.

The following roles are predefined in the CNC Console IAM:

Roles for CNCC IAM Admin Users

Admin Role

In CNCC 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.

Roles for CNCC Core Users

Read and Write Roles:

- **NF Roles**
  The user assigned with 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 user has POLICY_READ role, then the user can only read configurations of any MOs configurations 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 user has POLICY_WRITE then the user can read or write or 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 with 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 or write or 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 CNC Console application.

The user can create, read, updatem and delete MOs configurations for all NFs and CSa supported by CNC Console. For example, if the user has ADMIN then they can read, create, updare, 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 which corresponds to the cluster in which they are accessing a particular NF or CS. the nam eof 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 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 single or multiple instance roles. Instance Role name must match with 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 acess 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 access only 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 Cluster level role in case it is a multicluster deployment. This role is automatically assigned to all local users during the first upgrade when this feature is enabled. If 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 NFs 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 Cluster level role in case it is 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

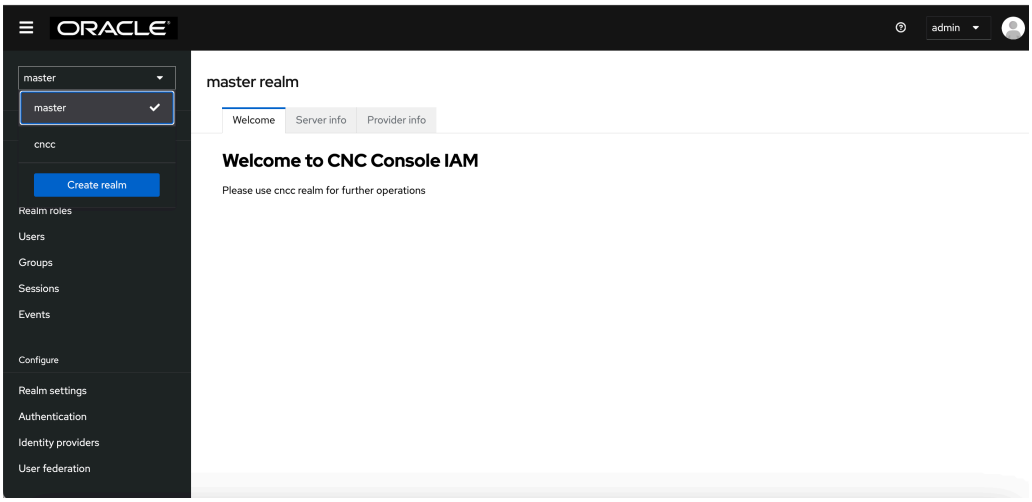
> **Note**
>
> For more information on how to assign roles to a user, see *Oracle Communications Cloud Native Configuration Console Installation, Upgrade, and Fault Recovery Guide.*

4.1.2 Accessing Roles in CNC Console Applications

Viewing the Roles

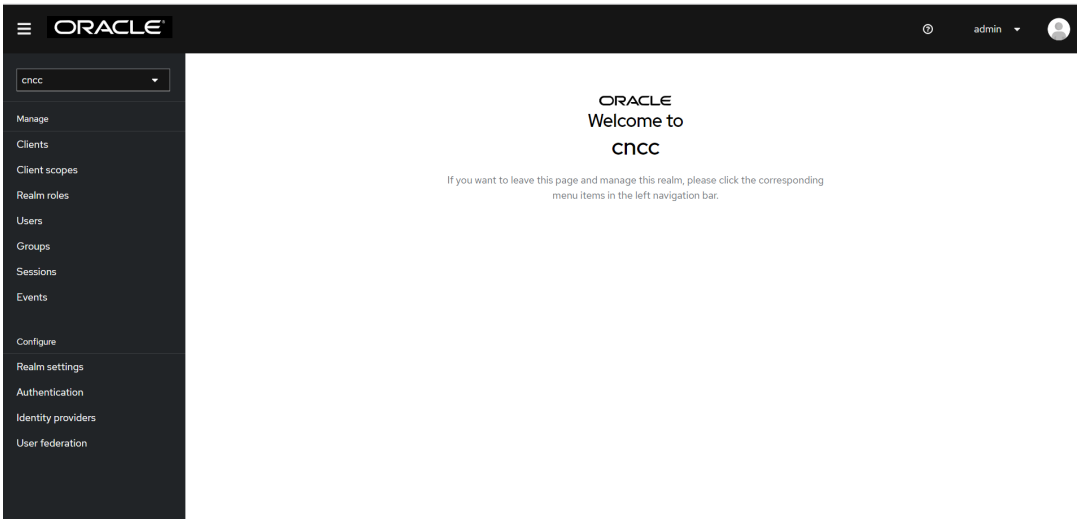
1. Log into CNC Console IAM using Admin credentials.
   Select the appropriate realm based on the users whose roles you want to view.
   a. To view roles in the default realm, select realm as shown below:

Figure 4-2 Viewing Realm



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

Figure 4-3 viewing cncc realm

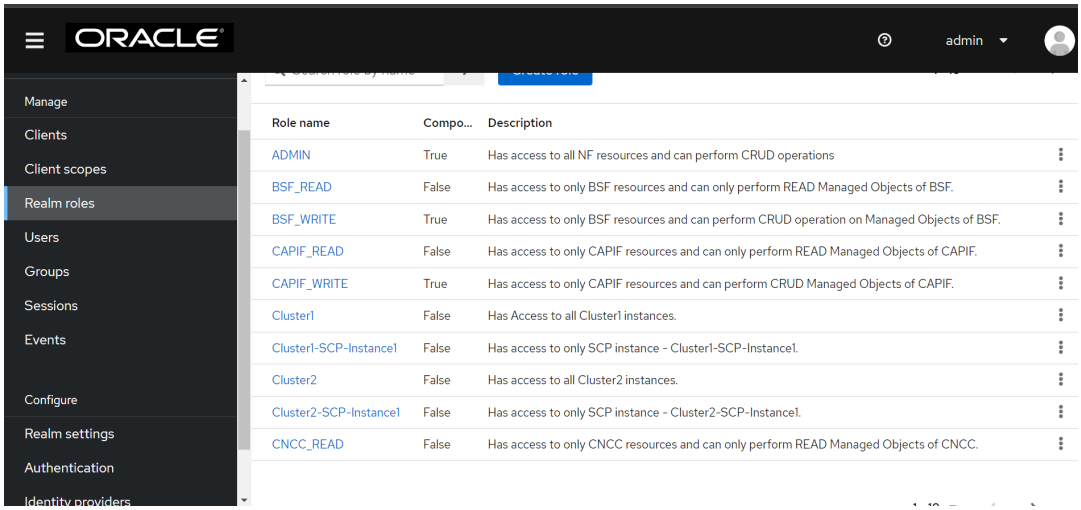


In the example below, the **cncc** realm is selected to view the available CNCC realm roles.

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 available on the right pane. (Here, the **cncc realm** is selected).

Figure 4-4 Realm Role



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

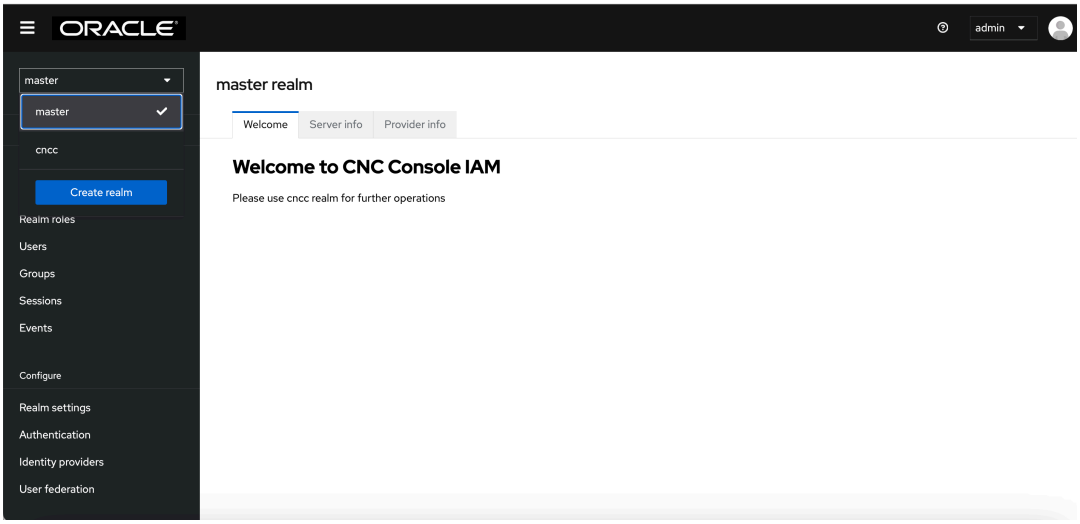
4.1.3 Creating or Updating Admin User Password in CNC Console IAM

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:

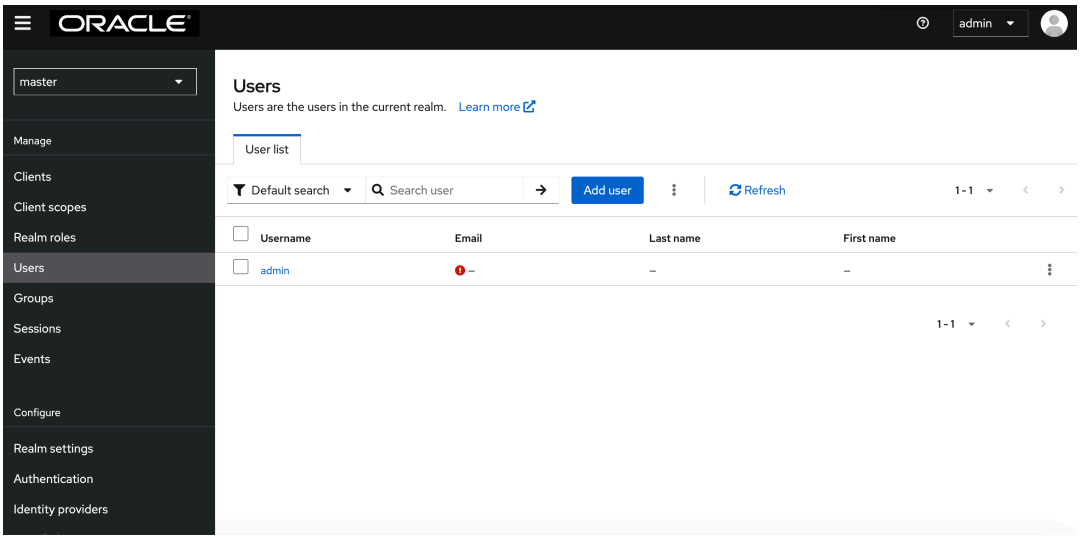
1. Login to CNC Console IAM and select the default realm.

Figure 4-5 Log in to CNC Console IAM



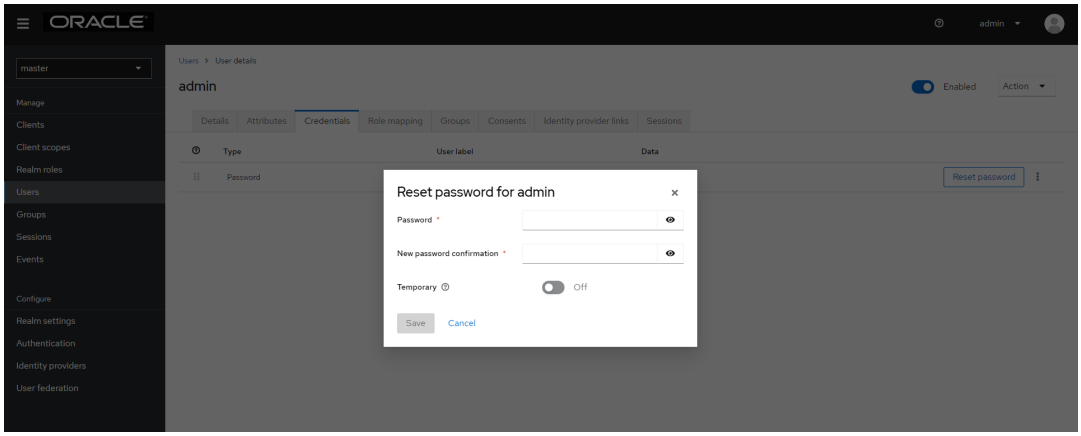
2. Select **Users** tab to see all the users in the realm. Click the user for which you want to change password.

Figure 4-6 Users



3. Under the **Credentials** tab, click **Reset Password** and set **Temporary** to Off and enter the existing **Password** and the new password. Click **Save**.

Figure 4-7 Credentials



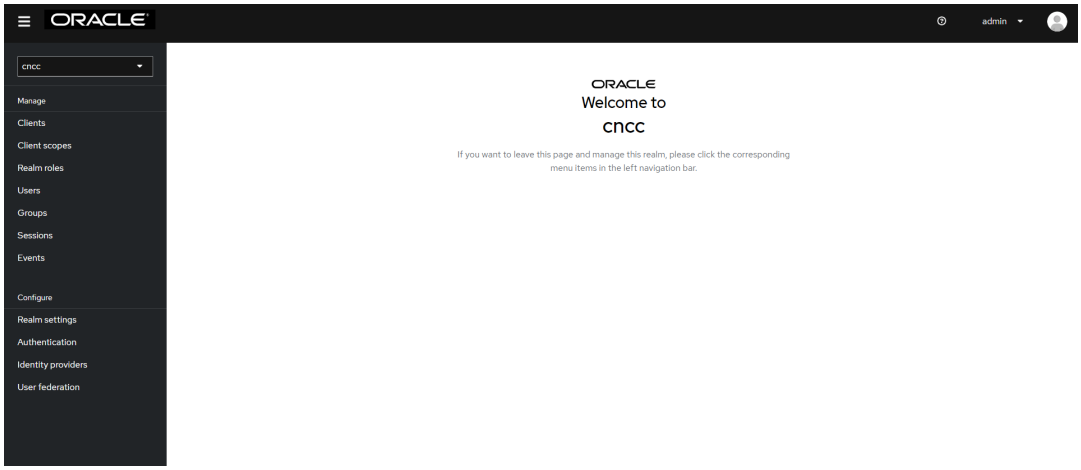
4.1.4 Creating or Updating CNC Core User Password in CNCC IAM

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:

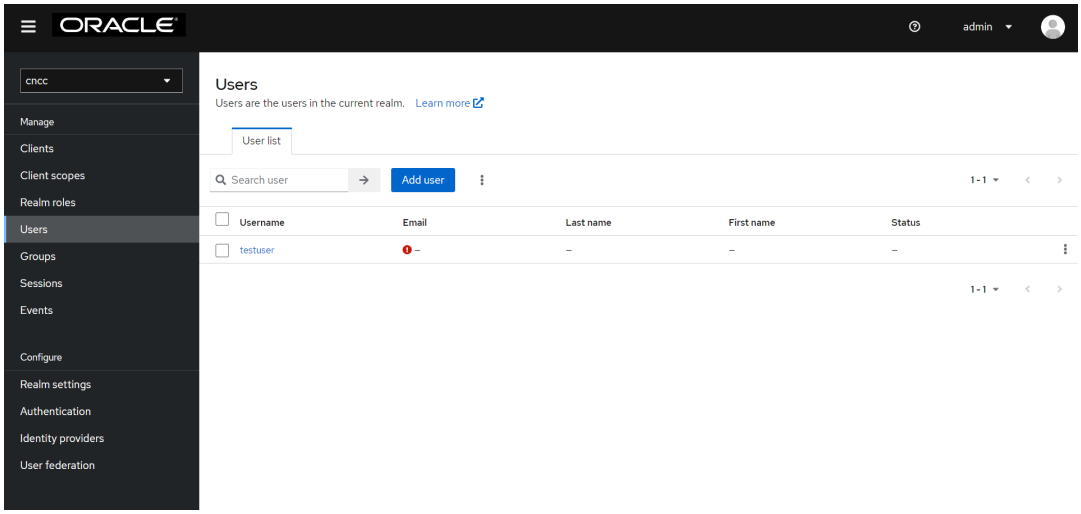
1. Login to CNC Console and select the **cncc** realm

Figure 4-8 Realm Settings



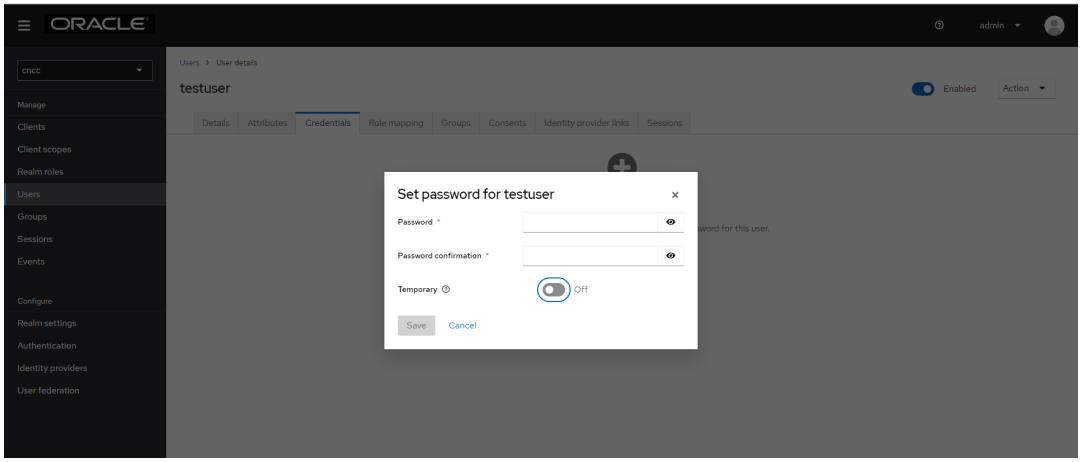
2. Click **Users** on the left pane to view all users. Click **Edit** button for that user to update the credentials.

Figure 4-9 Users



3. Under the **Credentials** tab, click **Set Password**, set **Temporary** to off, and update the **Password**. Click **Save**.

Figure 4-10 Credentials



4.1.5 Password Policies for CNCC Users

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

These password policies are disabled for CNCC IAM users by default and 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

Table 4-2 (Cont.) Password Policies for CNC Console Users

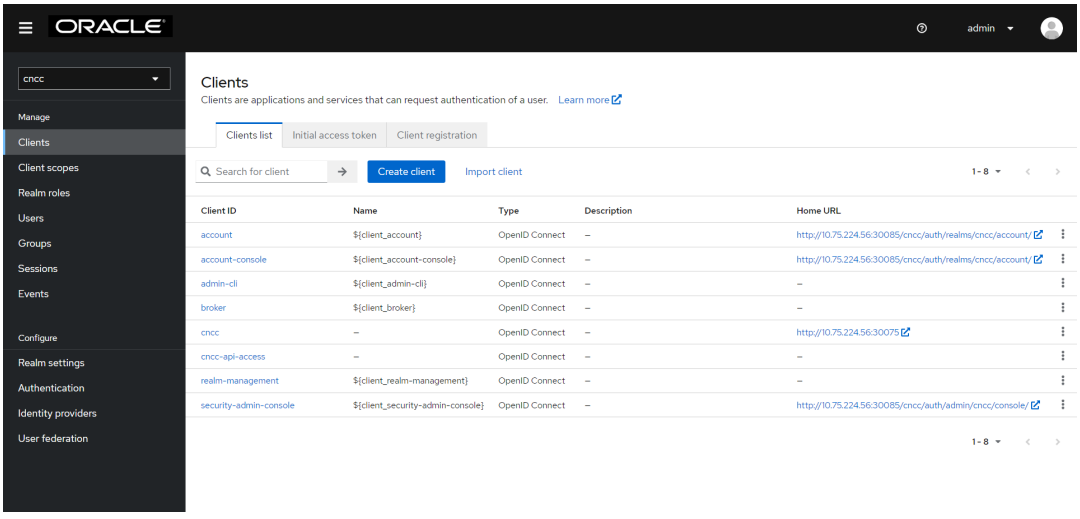
Policy	Description	Value
Special Characters	The minimum number of special characters required in the password string.	1
Uppercase Characters	The minimum number of uppercase characters required in the password string.	1
Lowercase Characters	The minimum number of lowercase characters required in the password 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 into CNC Console IAM using admin credentials provided during installation.
2. On the left pane, select **Clients** and on the right pane select the **cncc** Client ID.

Figure 4-11 Clients Screen

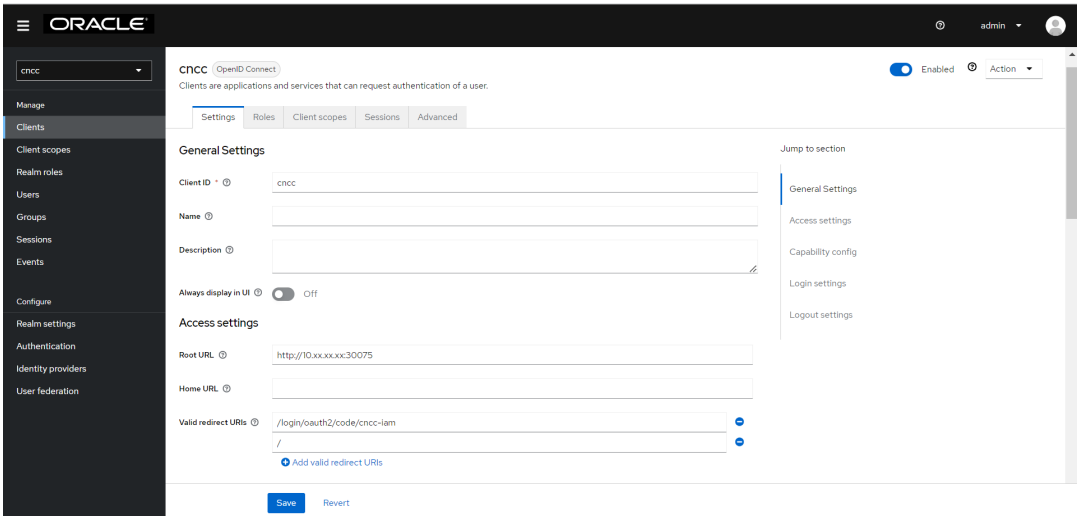


3. Enter CNC Console Core Ingress URI in the **Root URL** field and **Save.**

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

> **Note**
>
> **Valid Redirect URIs** is prepopulated, only root URL needs to be configured as part of the post installation procedure.

Figure 4-12 Redirection URL



4.3 Users in CNC Console IAM

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

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

A user created in the CNCC realm can log in to the CNCC 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 Core User Password in CNCC IAM.

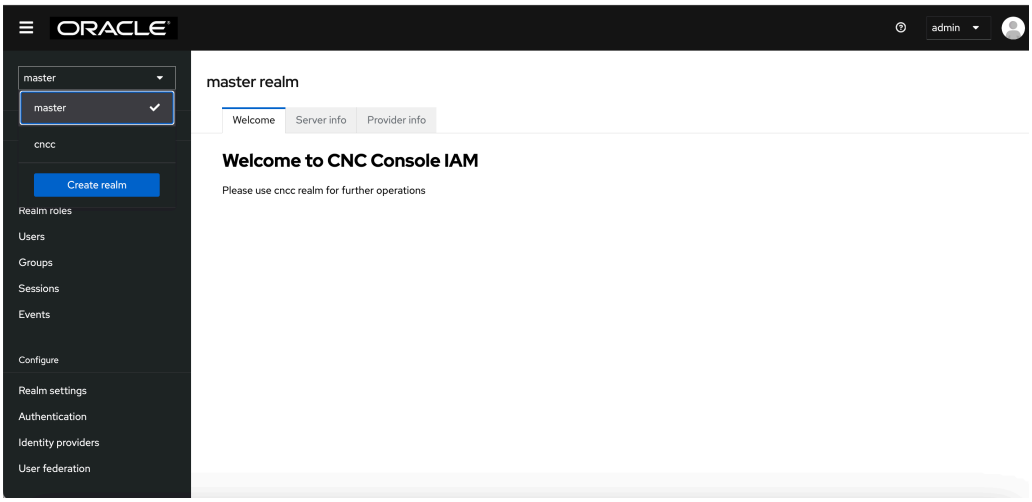
> **Note**
>
> In CNCC IAM, the default realm refers to **master** realm. Users created in default (**master**) realm refers to CNCC IAM admin users and users created in **cncc** realm refers to CNCC users.

4.3.1 Creating the Users

Perform the following procedure to create users:

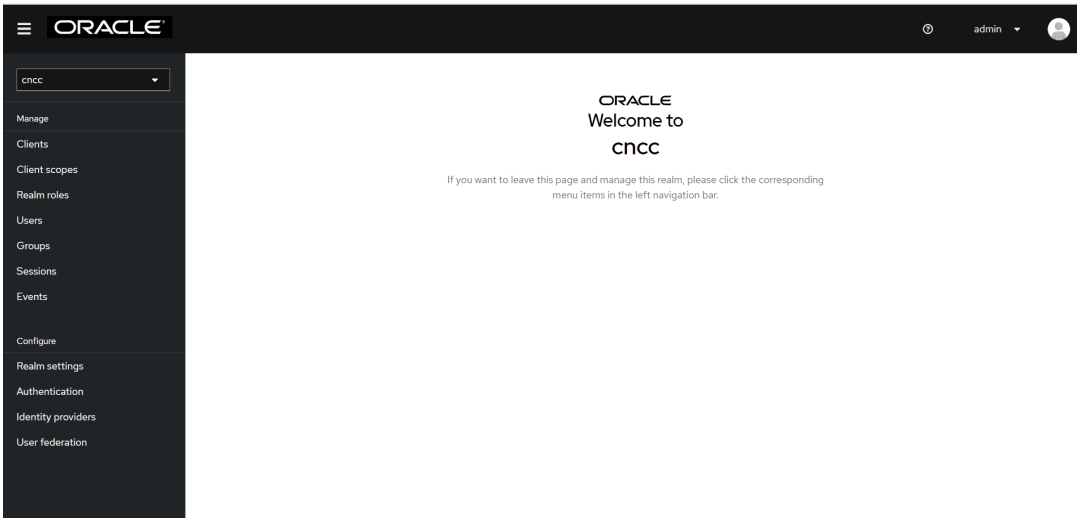
1. Login to CNC Console IAM and select the appropriate realm based on where you want to create users.
   a. To create users in default realm, select the default realm. The following screen appears:

Figure 4-13 default realm



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

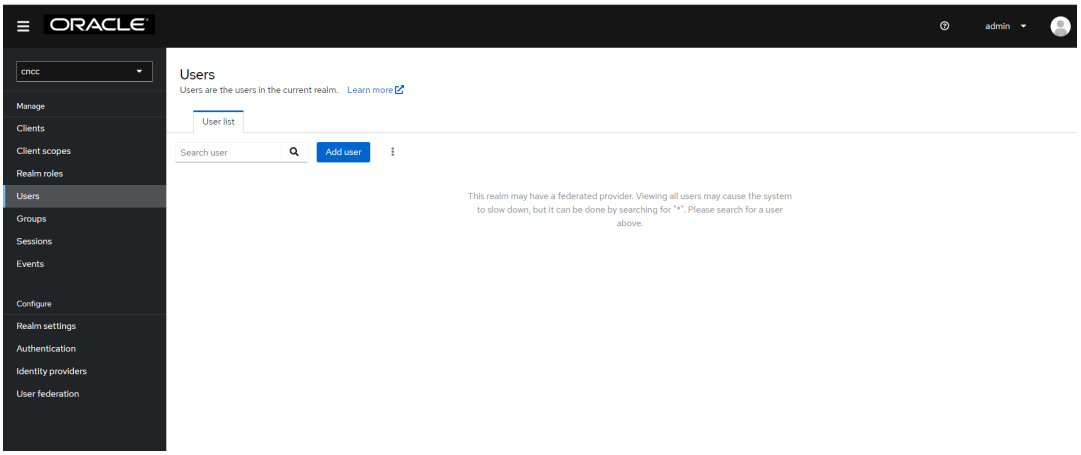
Figure 4-14 cncc realm



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

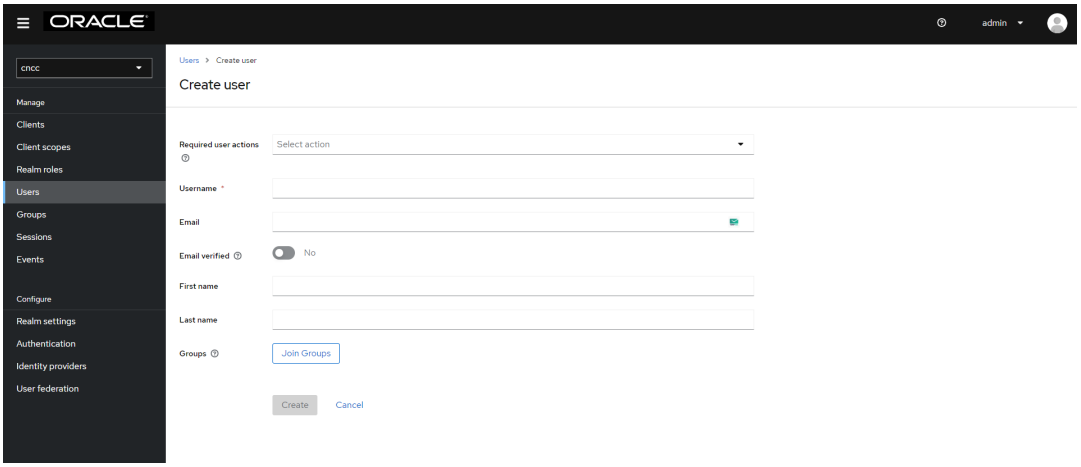
2. Click **Users** under **Manage** on the left pane and click **Add user** on the right pane.

Figure 4-15 Add User



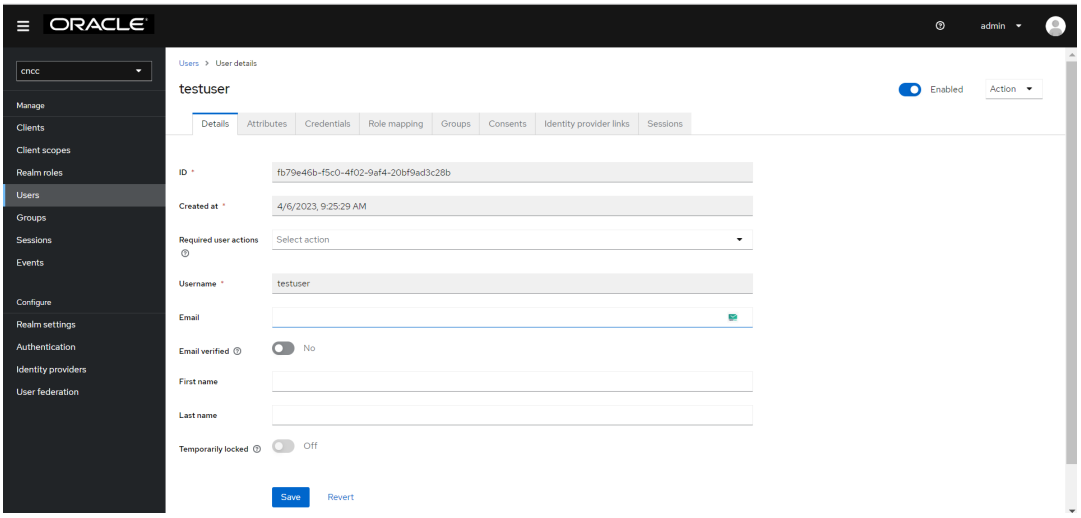
3. The **Add user** screen appears. Add the user details and click **Create**.

Figure 4-16 User Details



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

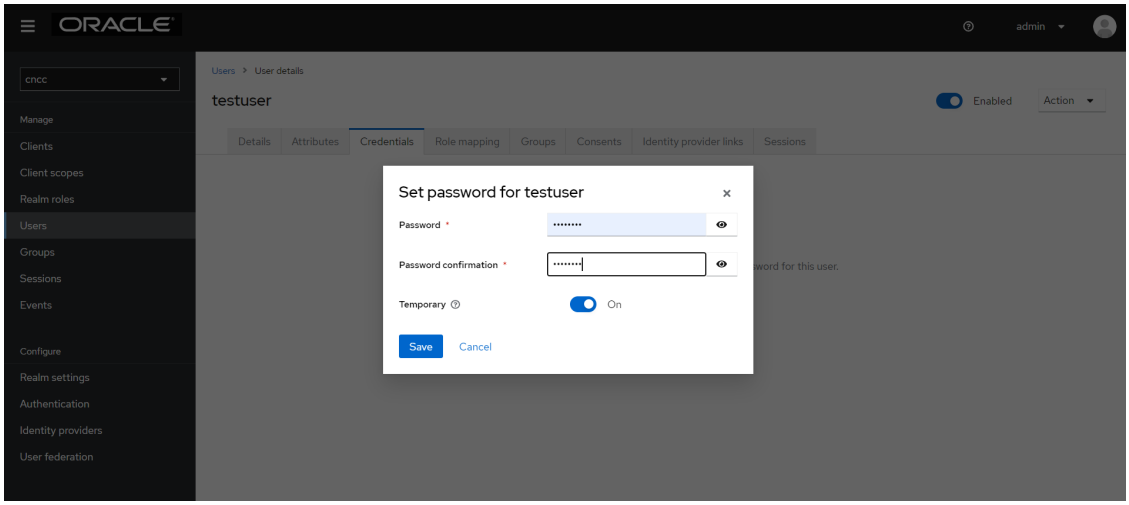
Figure 4-17 New User Created



5. Go to the **Credentials** tab and click **Set Password** to set the password for that user. Enable the **Temporary** flag to prompt the user to change their password when they login for the first time to CNC Console GUI.

> **Note**
>
> You are recommended to enable the **Temporary** flag for security.

Figure 4-18 Set Password



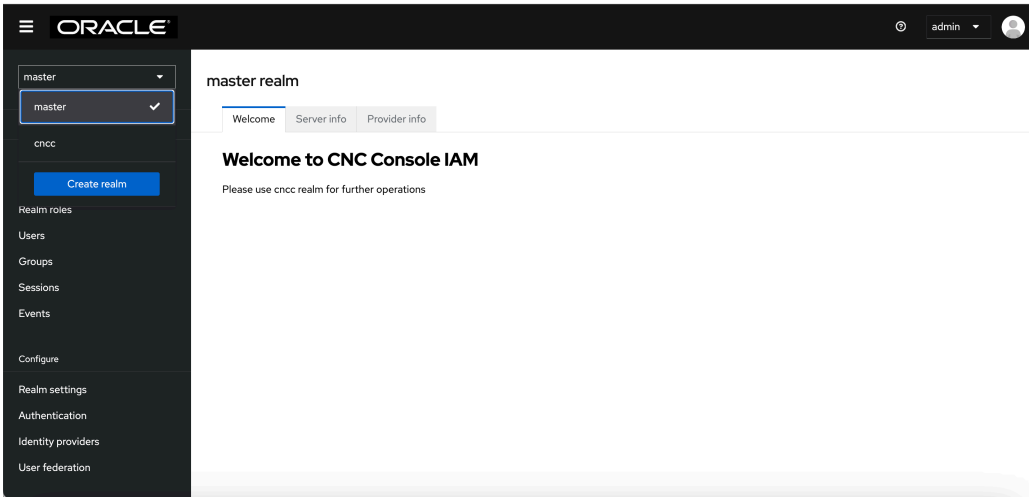
> **Note**
>
> Setting the **Temporary** flag **ON** prompts the user change the password when logging in to the CNC Console for the first time.

4.3.2 Viewing the Users

Perform the following procedure to view users:

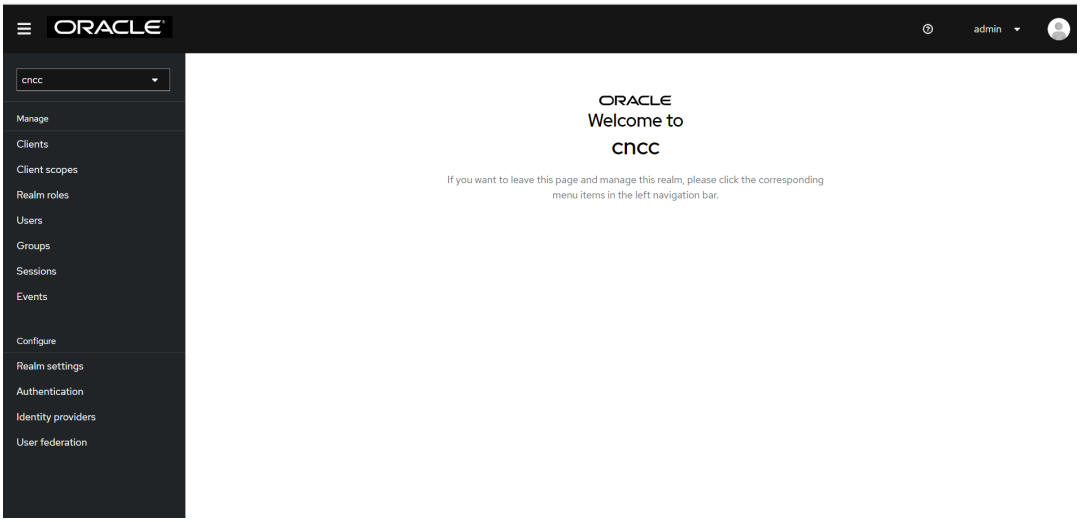
1. Login to CNC Console IAM and select the appropriate realm based on how users want to view.
   a. To view the users with administrative privileges, select the default realm.

Figure 4-19 default realm



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

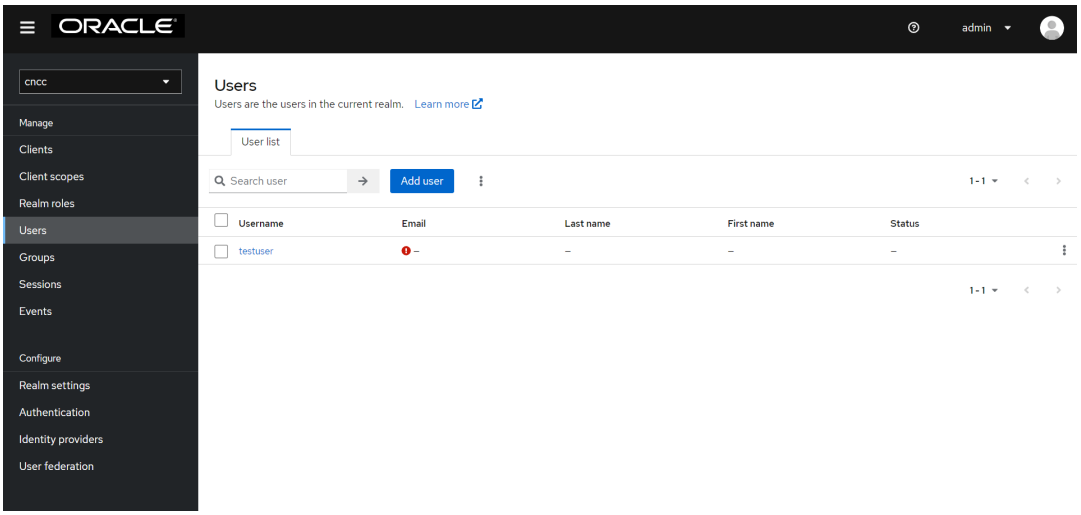
Figure 4-20 cncc realm



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 and click the button **View all users** on the page that appears.

Figure 4-21 View All Users

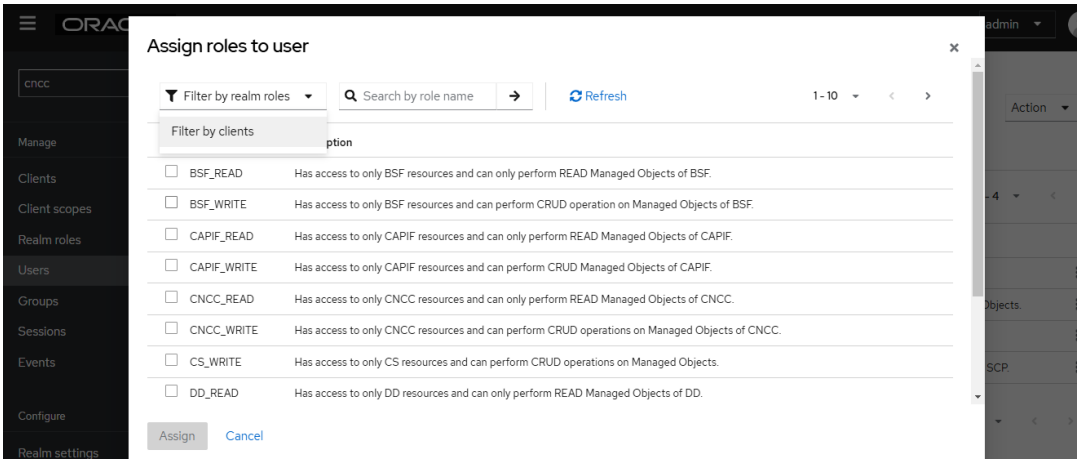


4.3.3 Assigning Roles to the User

Perform the following procedure to assign roles to the user:

1. Select a user. Navigate to the **Role Mappings** tab and click **Assign Role** to assign the user role. From the drop-down menu on the top left,select **Filter by realm roles**.

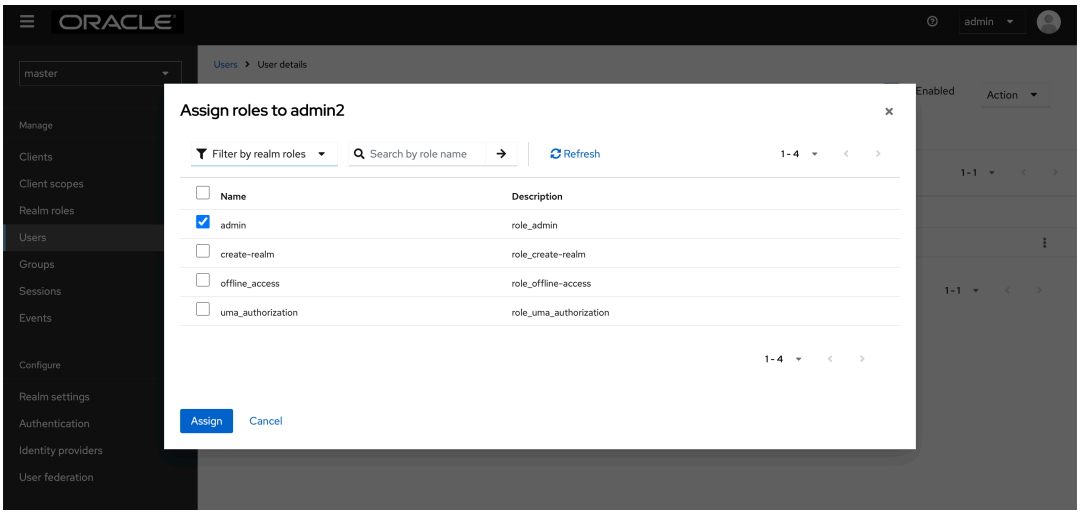
Figure 4-22 Assign Roles



> **Note**
>
> You must change number of entries displayed per page from the pagination drop-down to **100 per page** to view all the 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 4-23 Assign Roles



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 userdoes not need to login 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.

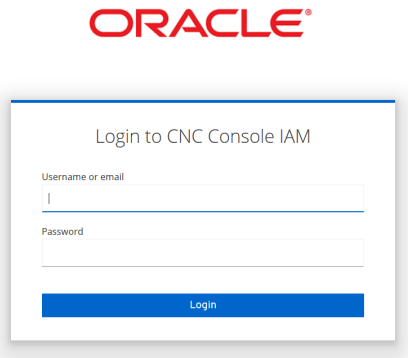
Configuring SAML Identity Provider in CNC Console IAM

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/
   ```

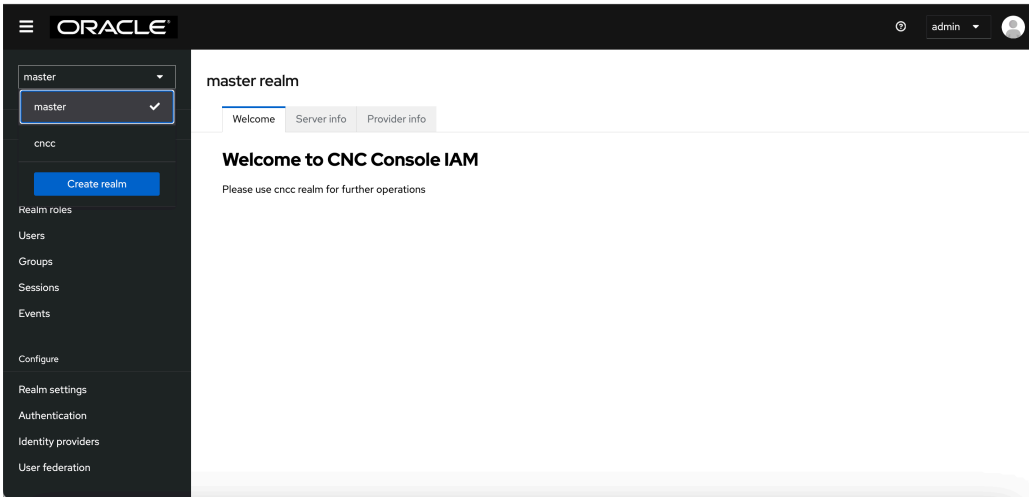
Figure 4-24 Login screen



2. Select the appropriate realm based on where you want to enable authentication through an identity provider:

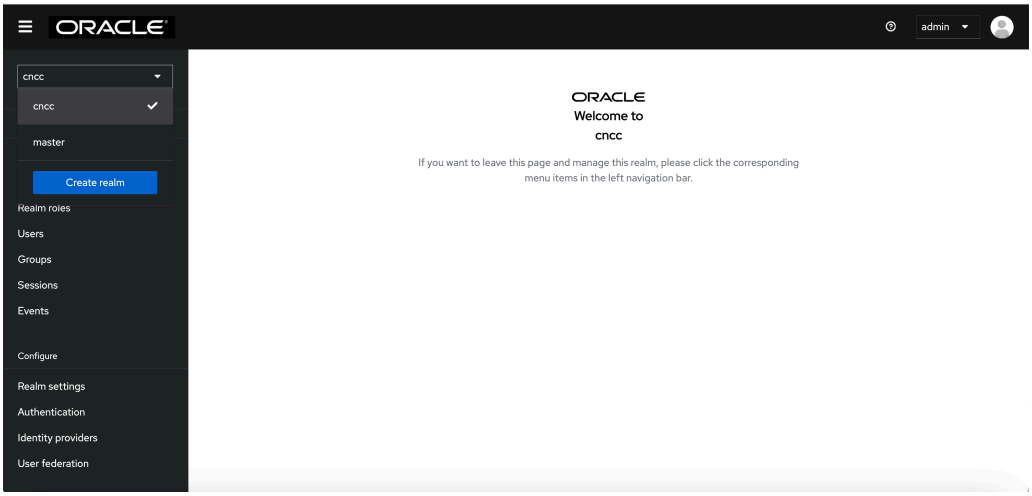
   a. To enable authentication for CNCC IAM admin users, choose the default realm.

Figure 4-25 default realm



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

Figure 4-26 cncc realm

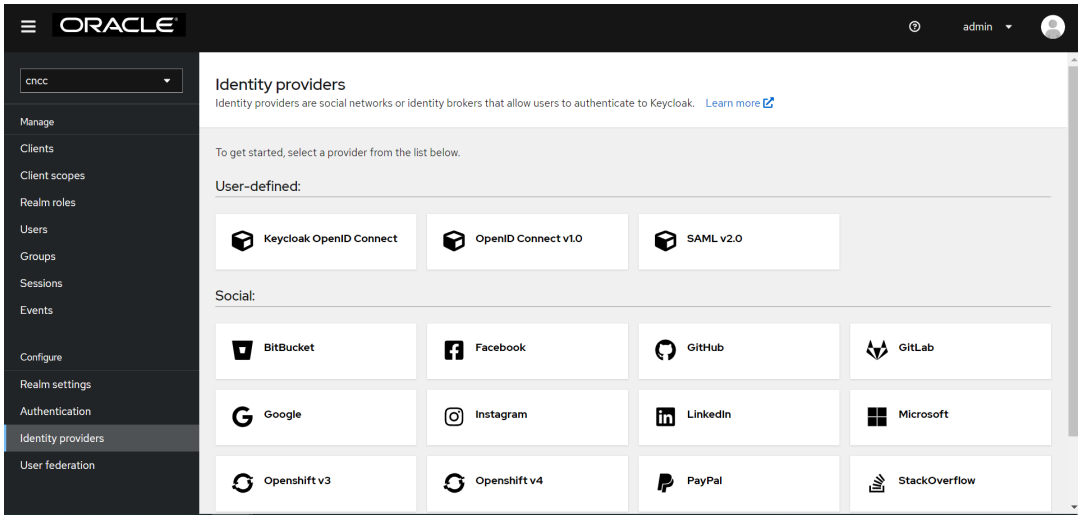


In the following example, the **cncc Realm** is selected to enable authentication through an identity provider to CNCC.

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

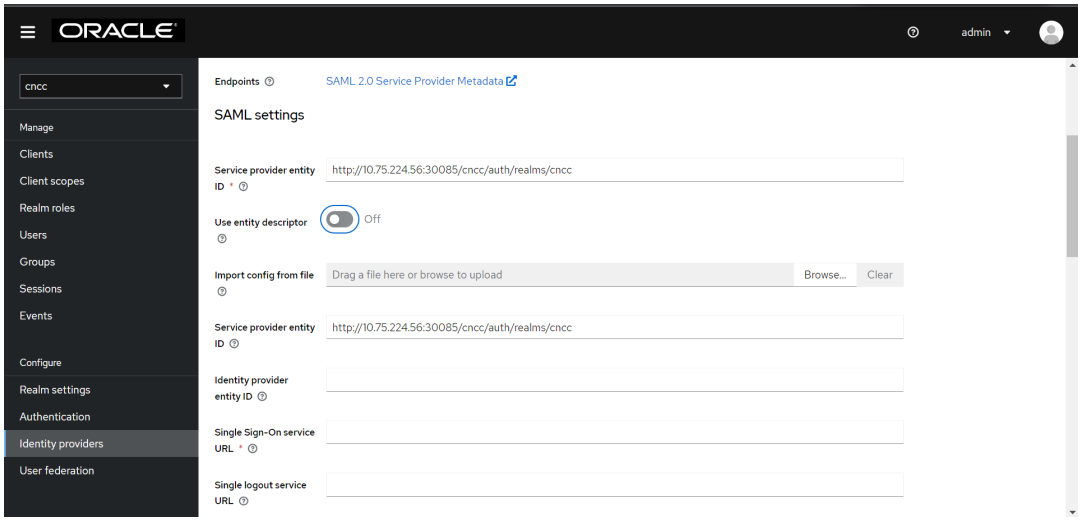
3. Click the **cncc** realm and click **Identity providers** tab on the left pane. **Identity providers** screen appears on the right pane.

Figure 4-27 Identity Provider Screen



4. Click the **SAML v2.0** button under **User-defined**. The **Add SAML Provider** screen appears.

Figure 4-28 SAML Settings



> **Note**
>
> - Give an appropriate name for the **Display Name** field.
> - To import the metadata file exported from SAML client in the IdP, disable the **Use Entity descriptor** flag, and upload the file from the **Browse** button of "Import from config file".

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.**

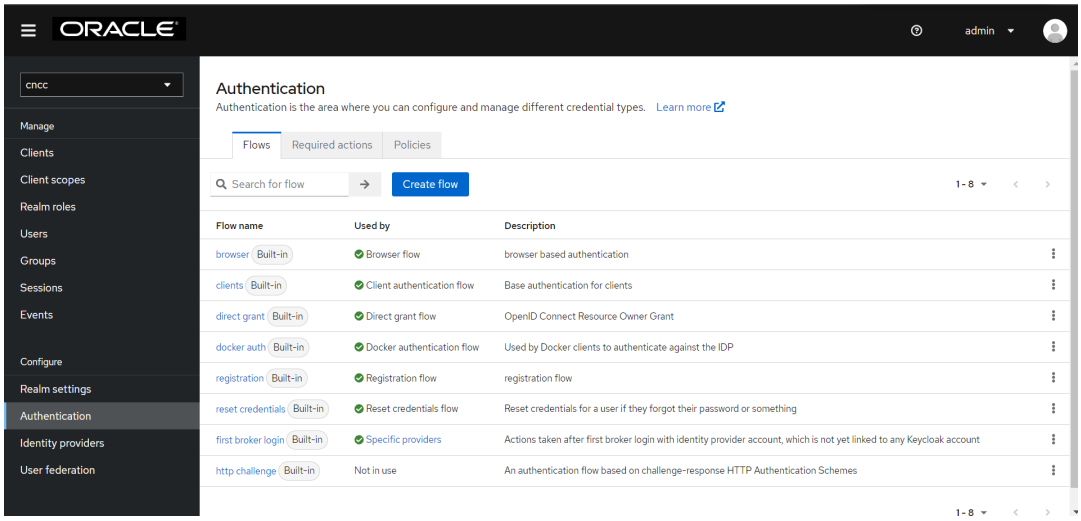
g. Provide value for **Validating X509 Certificates** (If you are using Keycloak as an IdP, use the certificate from **master** realm -> Realm Settings -> Keys).

h. Click **Add**.

IdP is now configured manually.

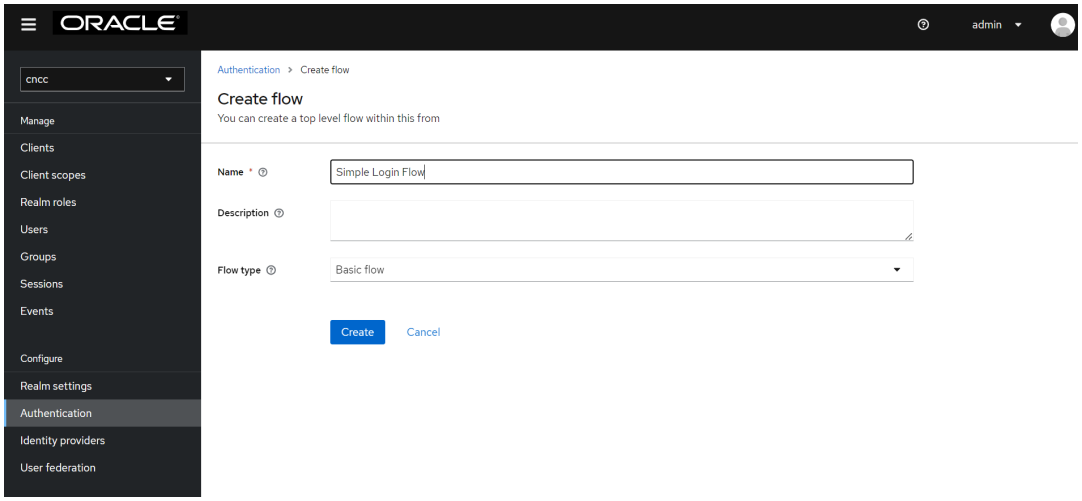
5. To create custom **First Login Flow**, click **Authentication** tab on the left pane. The **Authentication** screen appears.

Figure 4-29 Authentication



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

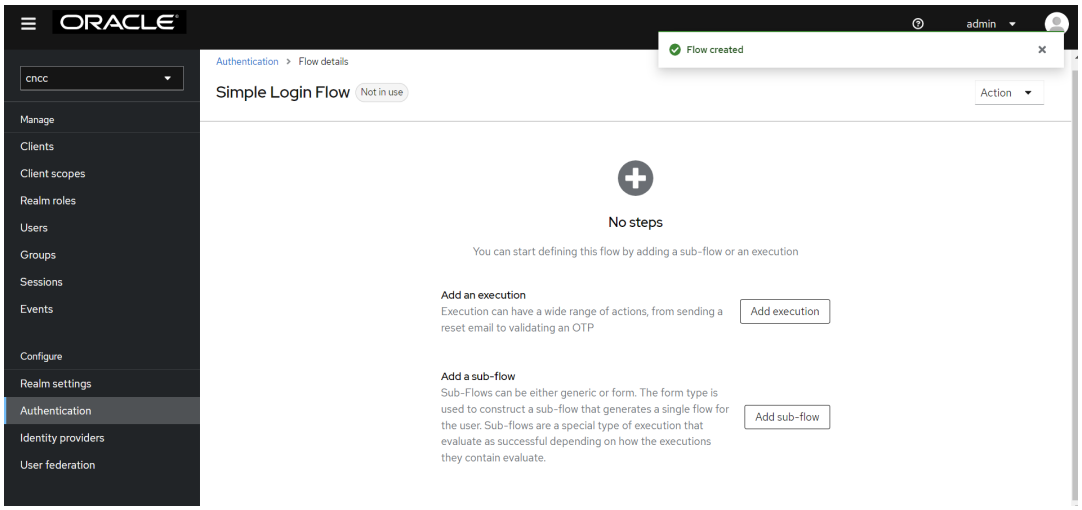
Figure 4-30 Create Flow



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

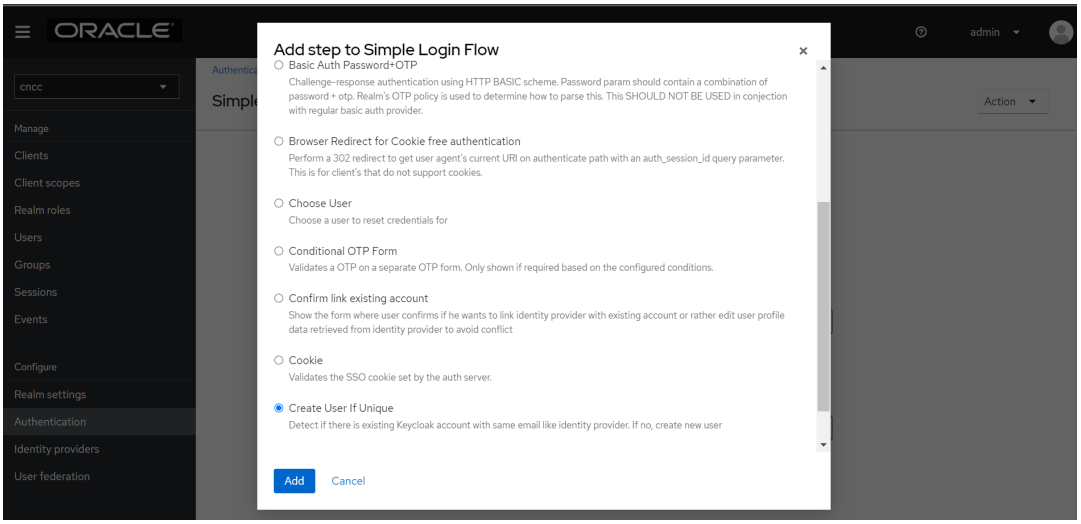
7. The **Simple Login Flow** screen appears. Click **Add execution** on the right pane.

Figure 4-31 Simple Login Flow



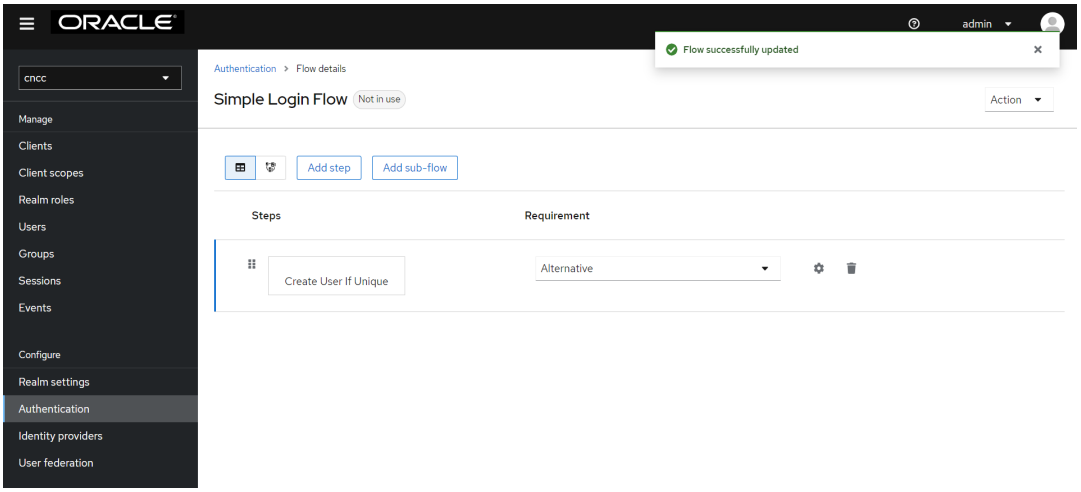
8. Select **Create User If Unique**, and click **Add**.

Figure 4-32 Add Step to Simple Login Flow



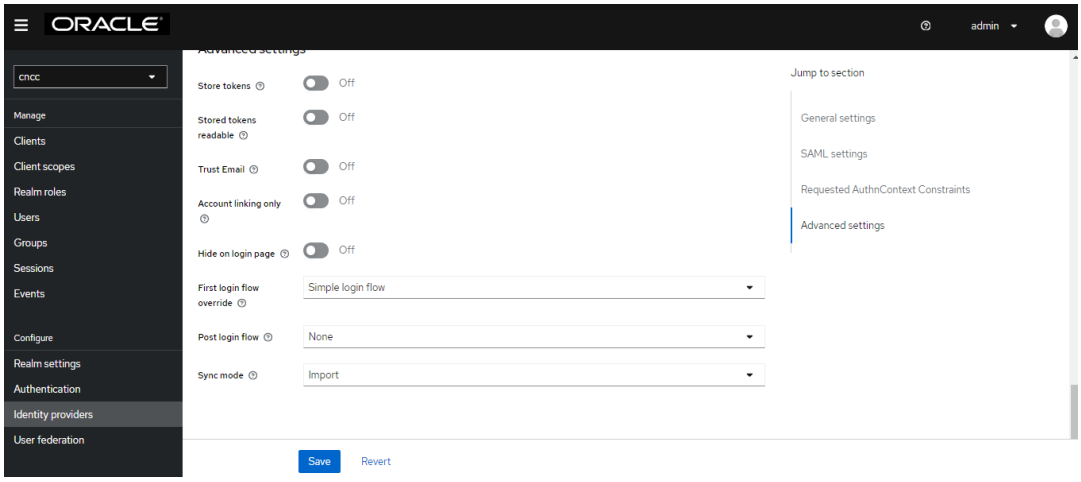
9. You will be redirected to **Authentication** page. From **Requirement** section, select **Alternative**.

Figure 4-33 Authentication



10. Click **Identity providers** in the left pane. Click the name of the Identity provide created in the previous steps, and scroll down to **Advanced Settings**. Select the custom flow from **First Login Flow** drop-down list.

Figure 4-34 Advanced Settings Page



11. Click **Save**.
    The above screen appears. Now the SAML Idp roles must be mapped with CNC Console IAM API roles.

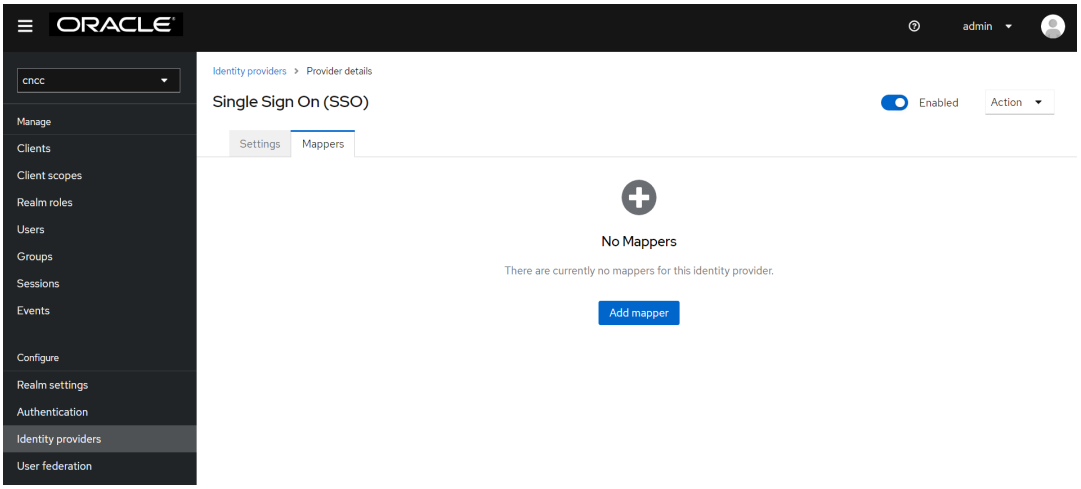
> **Note**
>
> **CNC Console IAM(SP) Configuration in IdP**
>
> In a SAML based SSO Implementation, the IdP needs to send SAML assertions towards a Service Provider (CNC Console IAM in this case) endpoint.
>
> Use the following CNC Console endpoint in the IdP:
>
> `http://<IP/FQDN>:<PORT>/cncc/auth/realms/cncc/broker/saml/endpoint`
>
> 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 4-35 Single Sign On



2. The **Add Identity Provider Mapper** screen appears.
   - Give an appropriate name for the Identity Provider Mapper in the **Name** field.
   - Select 'SAML Attribute to Role' from **Mapper Type** drop-down.
   - Enter the **Attribute Value** as the one of the roles added in SAML IdP. For example: 'NRF', 'SCP', etc.
   - Click **Select Role** to select the API roles to be enabled for this mapping.
   - Click **Assign**. Then click **Save**.

Figure 4-36 Example Values for scp role mapper for cncc realm

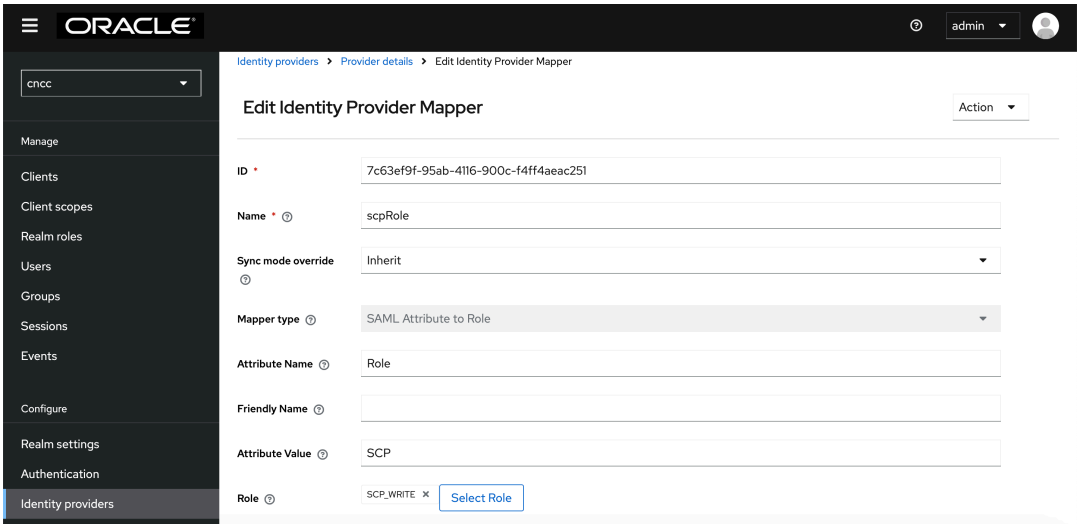


Figure 4-37 Example Values for admin role mapper for default realm

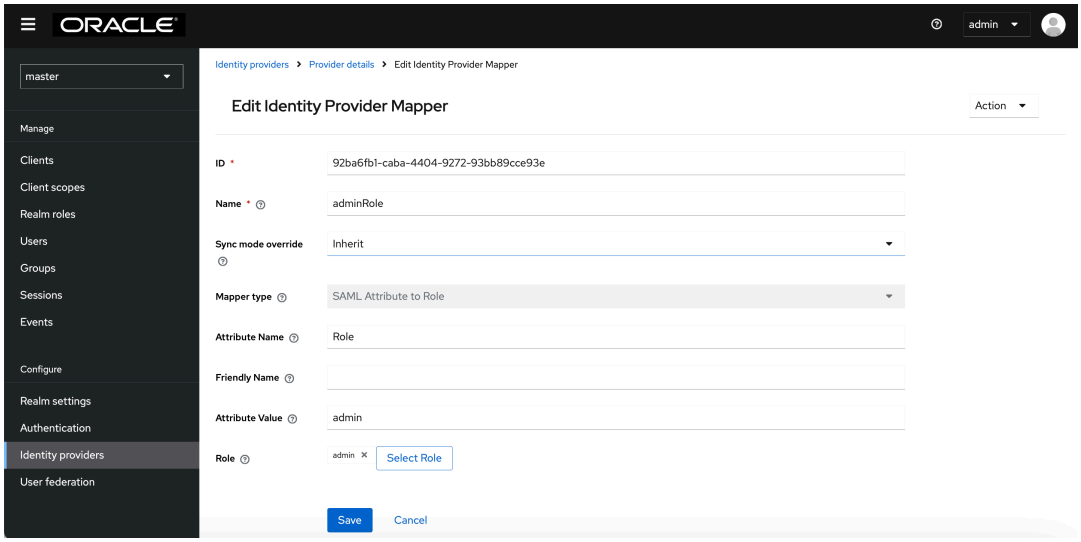
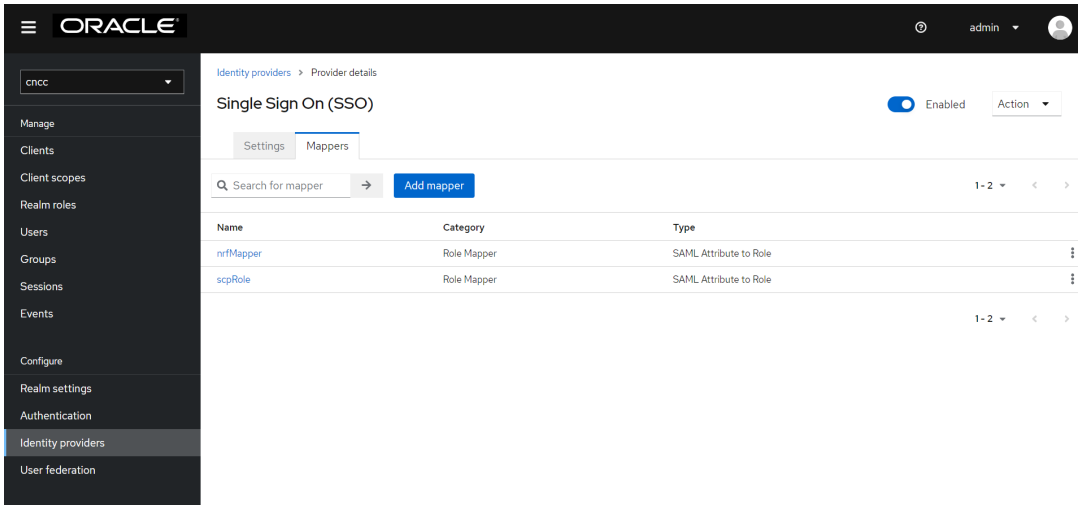


Figure 4-38 Single Sign On



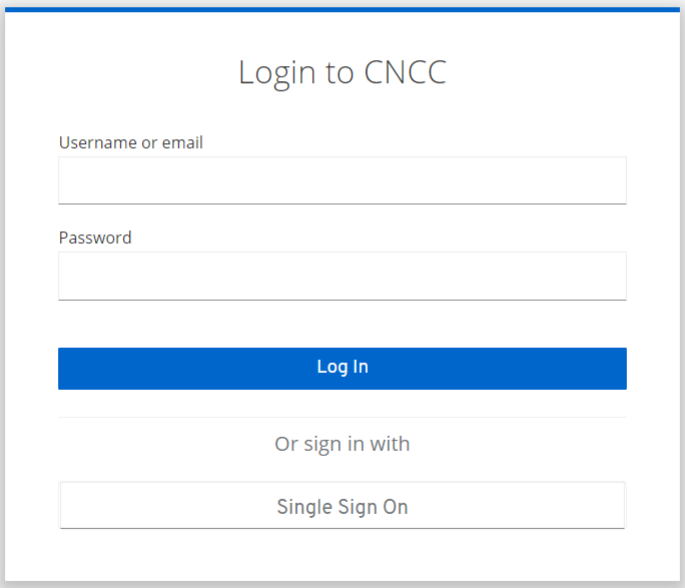
You can create as many mapping as required.

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 it into existing LDAP and Active Directory servers.

User Federation in CNC Console IAM let the user to sync users and groups from LDAP and Active Directory servers and assign roles respectively.

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

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

CNCC IAM uses SSL for communication with the LDAP server. The truststore must be properly configured on the CNCC IAM server side, otherwise CNCC 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 our LDAP with CNCC IAM to import users in **cncc** realm.

Sample LDAP ldif File
This is a sample ldap-ldif file that ldap is loaded with for importing LDAP users and groups to CNCC 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 CNCC IAM for importing users into the default realm.

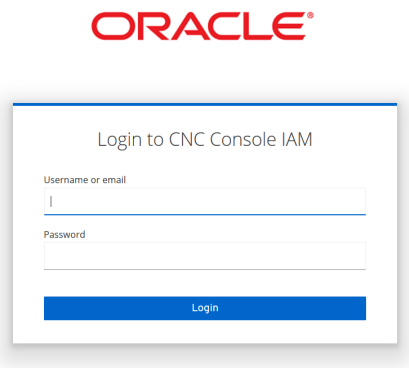
4.5.1 Configuring User Federation with CNC Console IAM

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

To configure user federation:

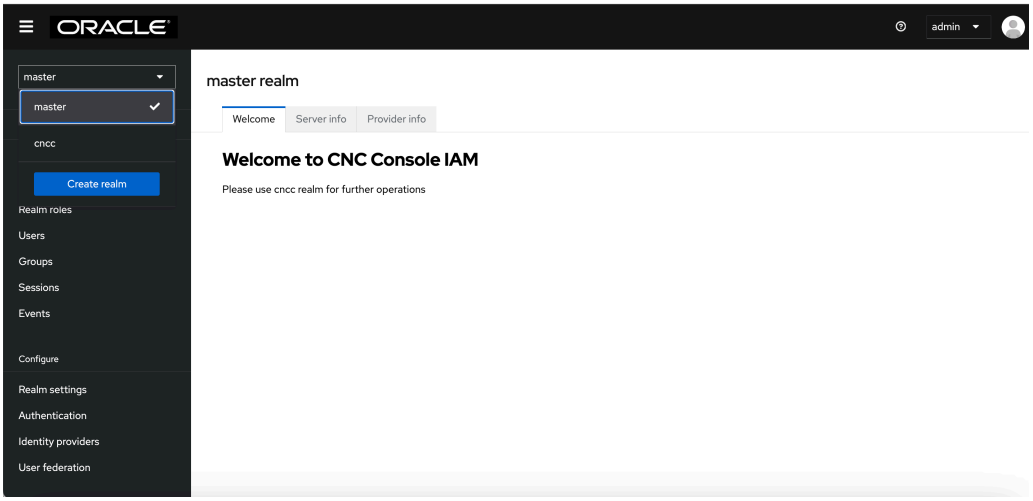
1. Login to CNC Console IAM console **http://<cncc-iam-ingress-ip>:<cncc-iam-ingress-port>** using admin credentials provided during CNC Console IAM installation.

Figure 4-39 Login Screen



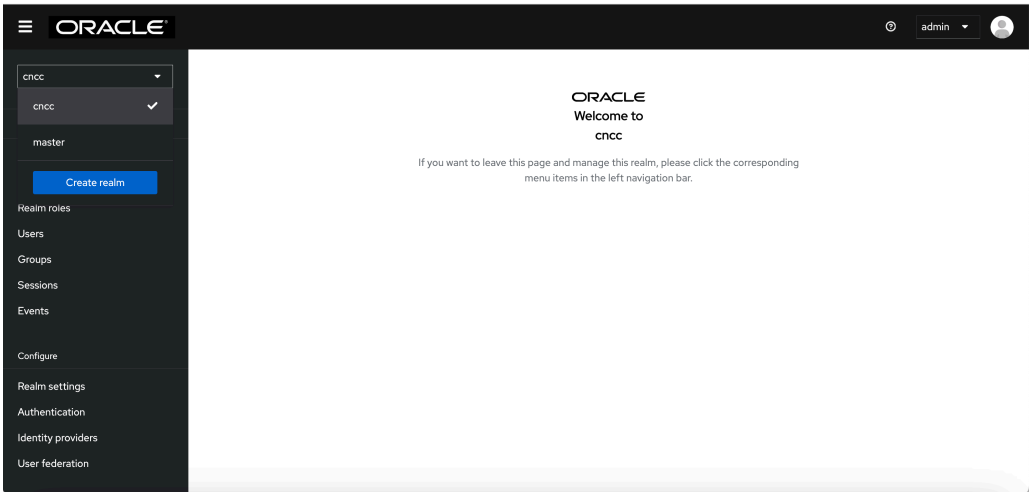
2. Select the appropriate realm based on where you want to import users:
   a. To grant LDAP users access to CNCC IAM, choose the default realm.

Figure 4-40 default realm



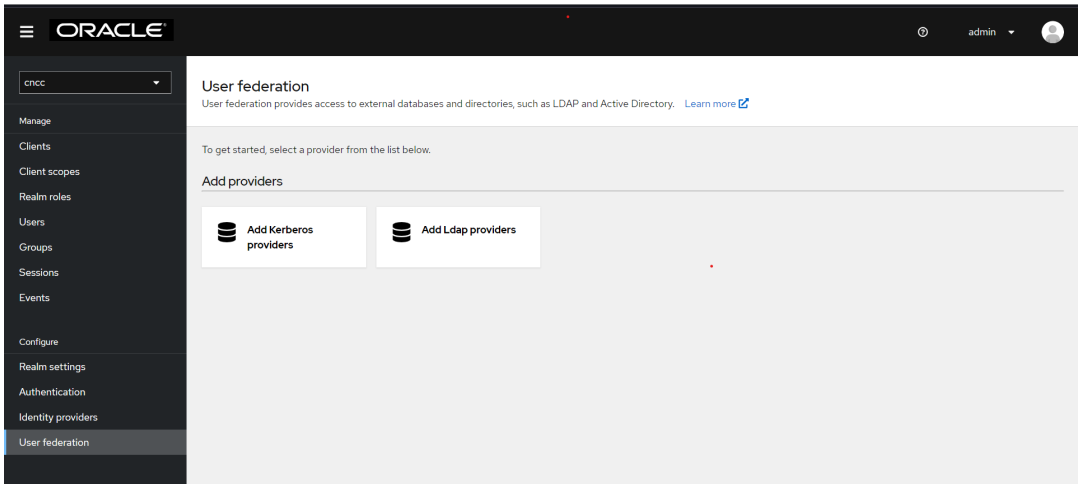
b. To grant LDAP users access to CNCC Core, choose the **cncc** realm.

Figure 4-41 cncc realm



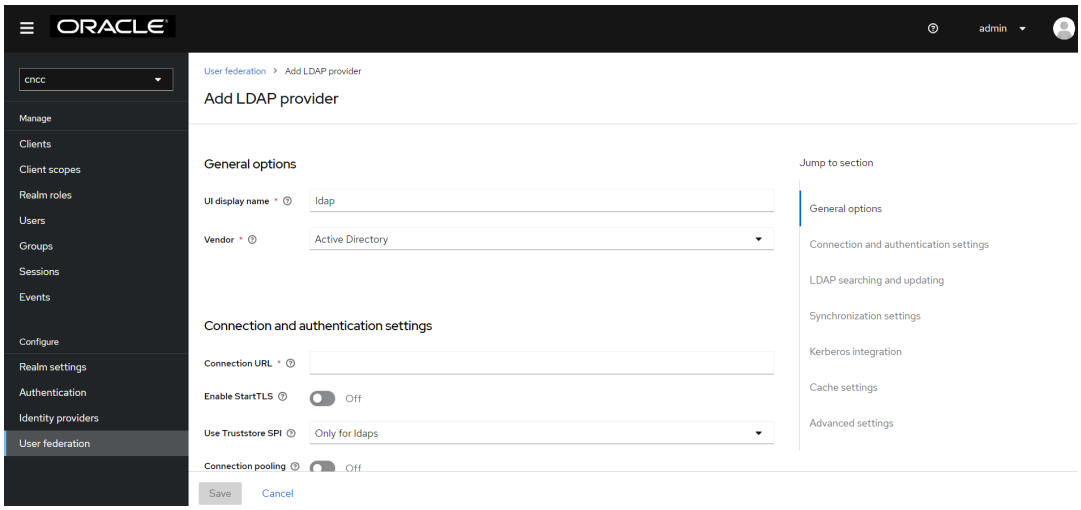
3. Click **Realm Settings** and click **Add realm** under **cncc**. Click **User Federation**on the left pane. The **User Federation** screen appears in the right pane.

Figure 4-42 User Federation



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 4-43 Add LDAP providers



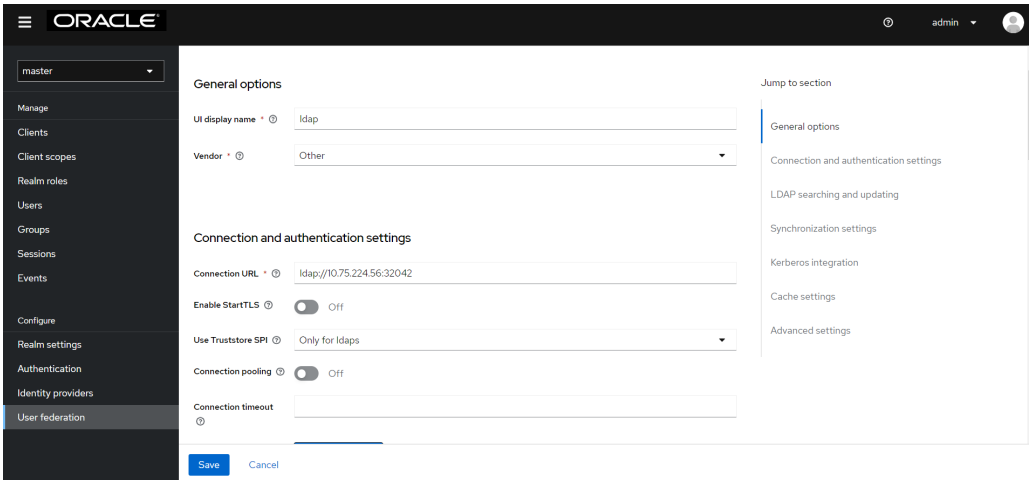
5. Enter the values for the following fields:
   - **UI Display Name:** Enter the display name.
   - **Vendor:** Enter the LDAP server provider name for the company.

> **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 alread. That is, the connection URL (hostname

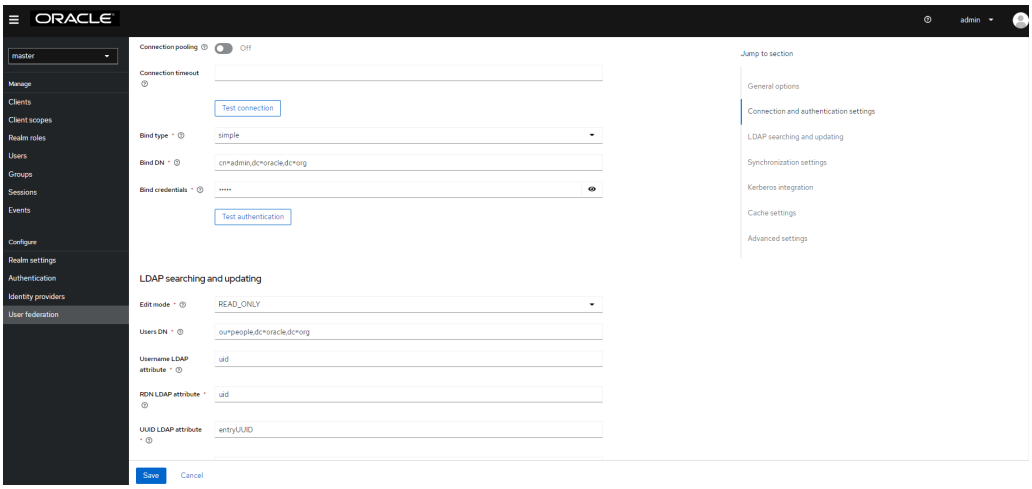
prefixed with ldap:// OR when LDAP Secure connection enabled (LDAPS) hostname prefix should be ldaps://), and the port.

Figure 4-44 General Options



- If your LDAP is secured then select **simple** from the **Bind Type** drop-down, and add the admin bind username and password, or select Bind-type as **none**. Sample data for the field Bind DN: **"cn=admin,dc=oracle,dc=org"**
- Click **Test Connection** and **Test Authentication**. Both these tests will be successful.
- Proceed to 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.**

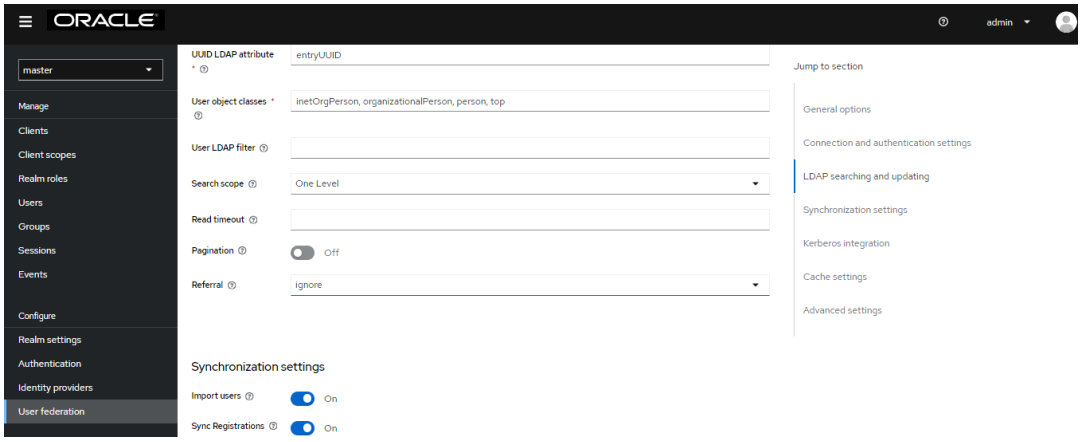
Figure 4-45 User Federation



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

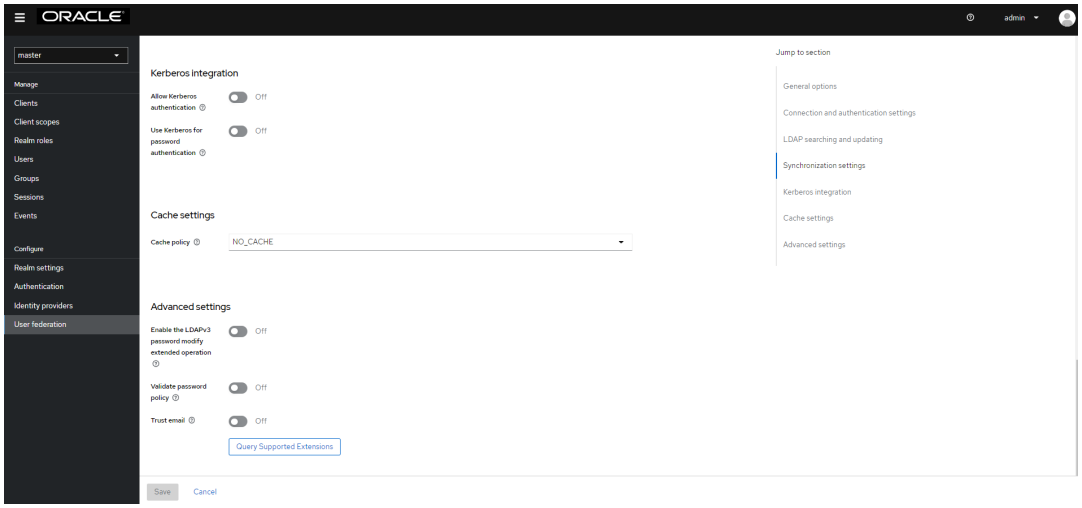
6. After populating the required fields, the following screen appears:

Figure 4-46 User Federation



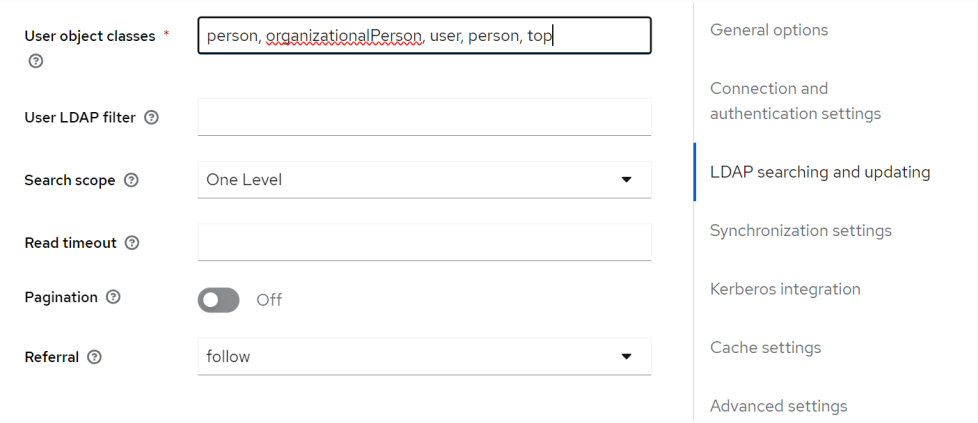
7. Click **Save**.

Figure 4-47 User Federation



> **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 towards LDAP Server.
>
> Manage DSA IT Control is enabled by default as **Refferal** is set to **Ignore** in the **User Federation Setup**.
>
> This can be disabled by setting **Referral** to **Follow**.

Figure 4-48 Enabling and Disabling the Manage DSA IT Control in LDAP Requests



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.

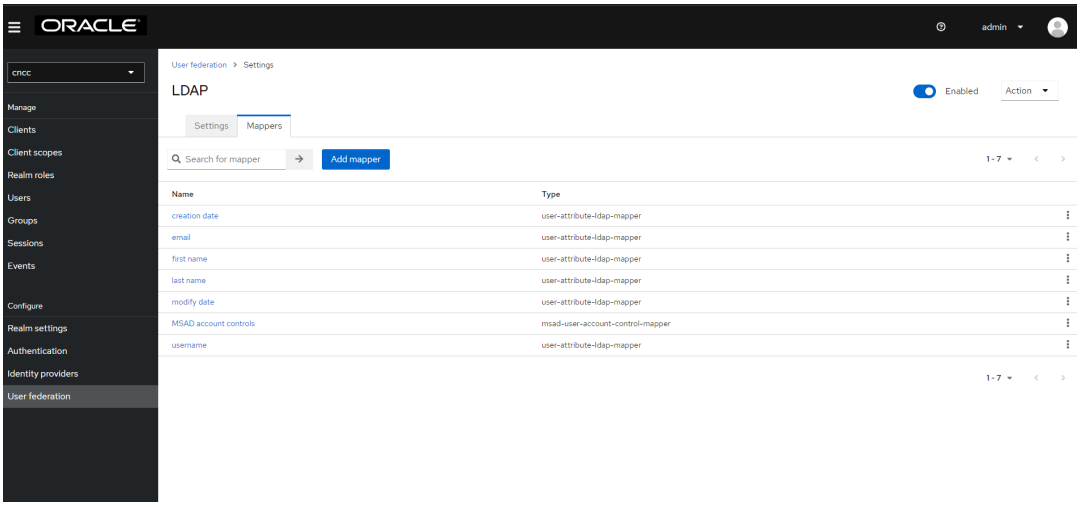
Group Mapper

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:

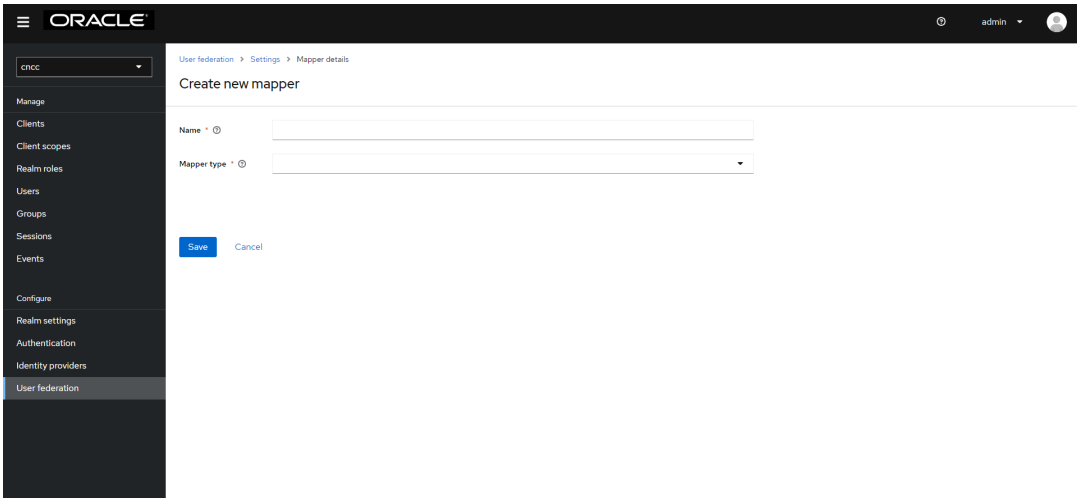
1. Under **Configure** in the left pane, click **User Federation**. Click **ldap** and select the **Mappers** tab, and then click **Add Mapper**.

Figure 4-49 LDAP Mapper Page



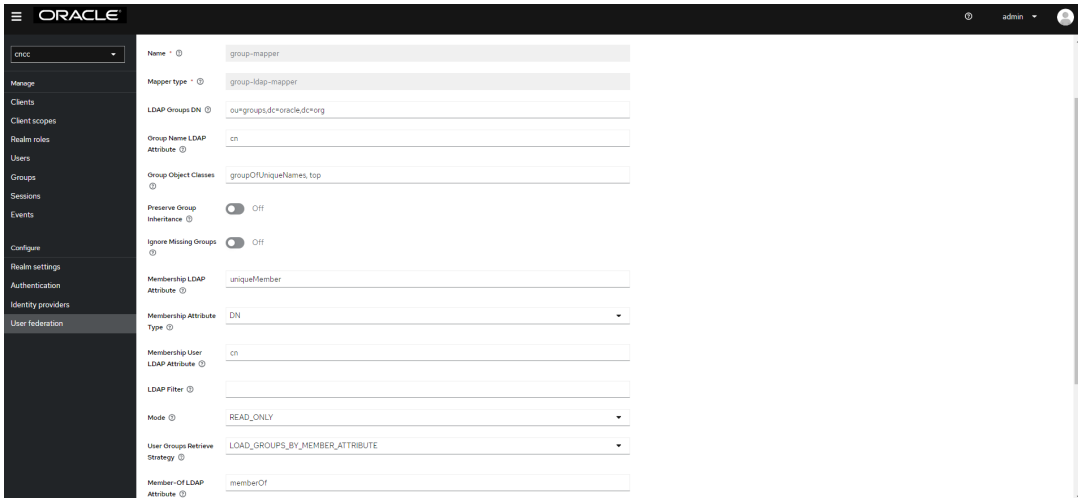
2. The **Create New Mapper** page appears. Give an appropriate name for the field **Name.** Select group-ldap-mapper as **Mapper Type** drop-down menu. Click **Save.**

Figure 4-50 User Federation Mapper Page



The following screen appears:

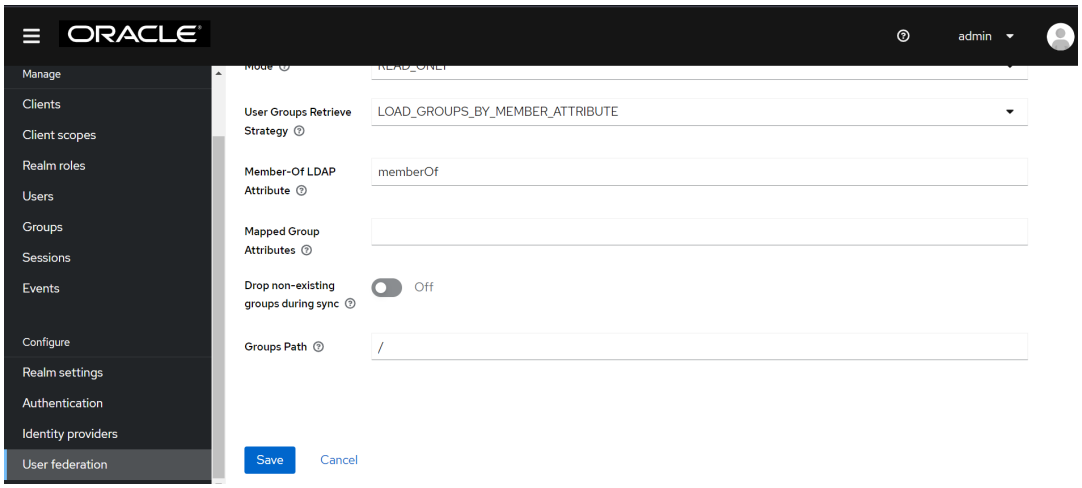
Figure 4-51 LDAP Mapper Filled Form



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

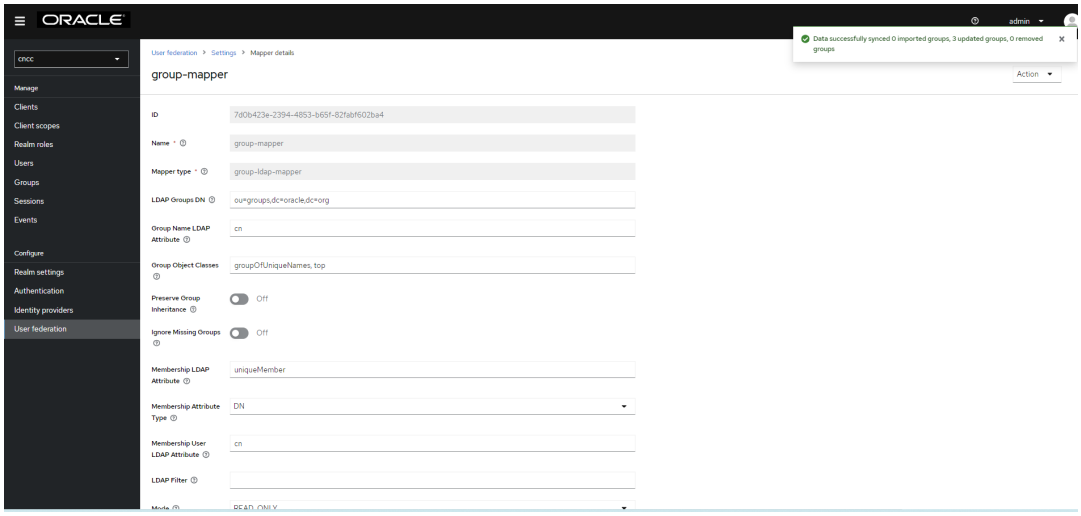
3. Click **Save**.

Figure 4-52 Save



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

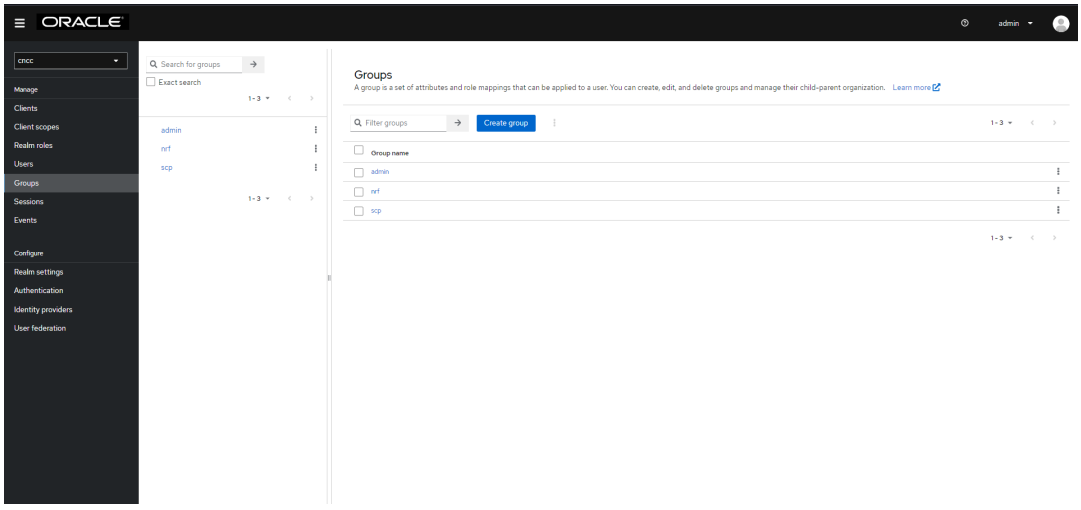
Figure 4-53 Group Mapper



> **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 CNC Console Logs section in the *Oracle Communication Cloud Native Configuration Console Troubleshooting Guide* for further details.

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

Figure 4-54 Groups



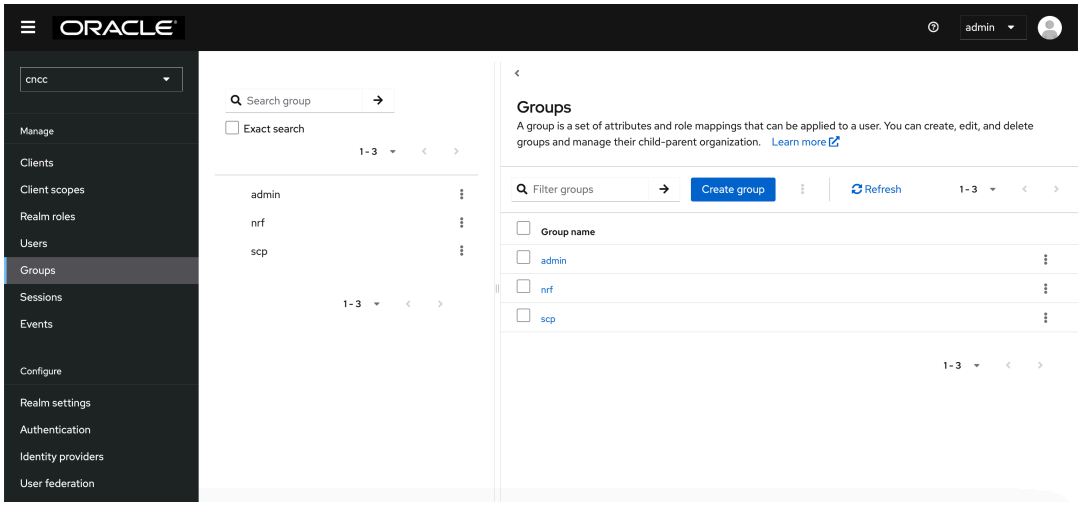
6. Click any group and click **Edit**. The following tabs appear: **Child groups, Attributes, Role Mappings,** and **Members**.
   - Select the **Role Mapping** tab to see a list of roles that are predefined in CNC Console IAM.
   - Select one or more roles from **Available Roles** and assign it to the group.

- When you're 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.

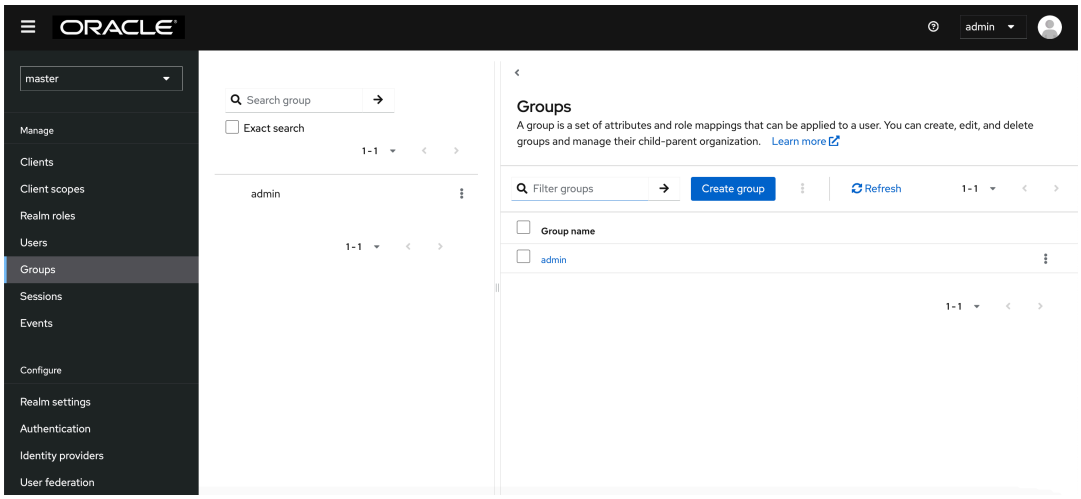
-

Figure 4-55 Role mapping to LDAP Group



- 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 4-56 CNCC IAM Role mapping to LDAP Group for default Realm



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 5the 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 them.

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 doesn't 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 adding 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.

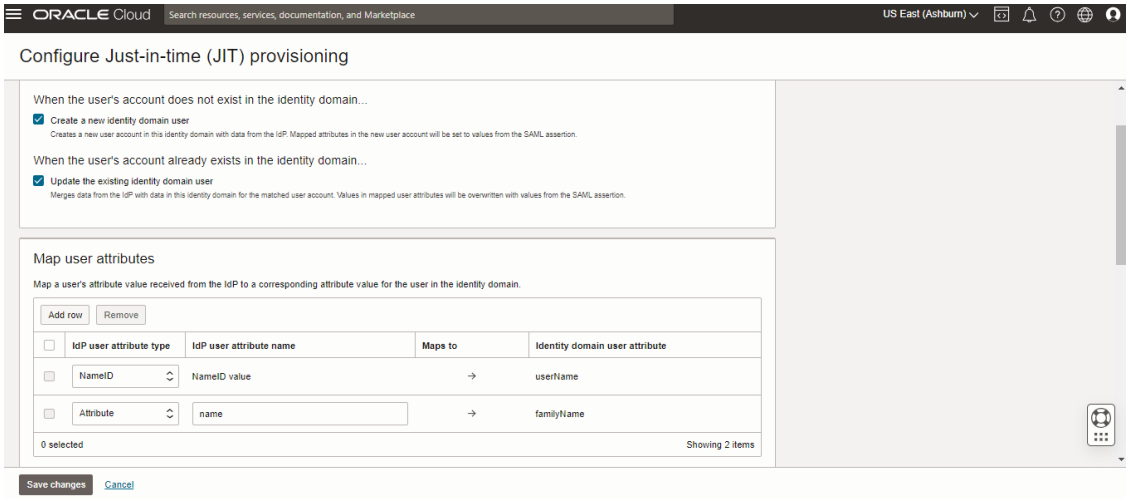
> **Note**
>
> - After following the steps from the above guide, the SAML IdP is configured, but any user created in SAML IdP must be created in OCI IAM to allow a successful login.
> - To directly log in to CNC Console Core using SAML IdP, SAML JIT (Just-In-Time) must be configured.
> - Role Mappings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



Configuration Example:

The **NameID** value to the **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:

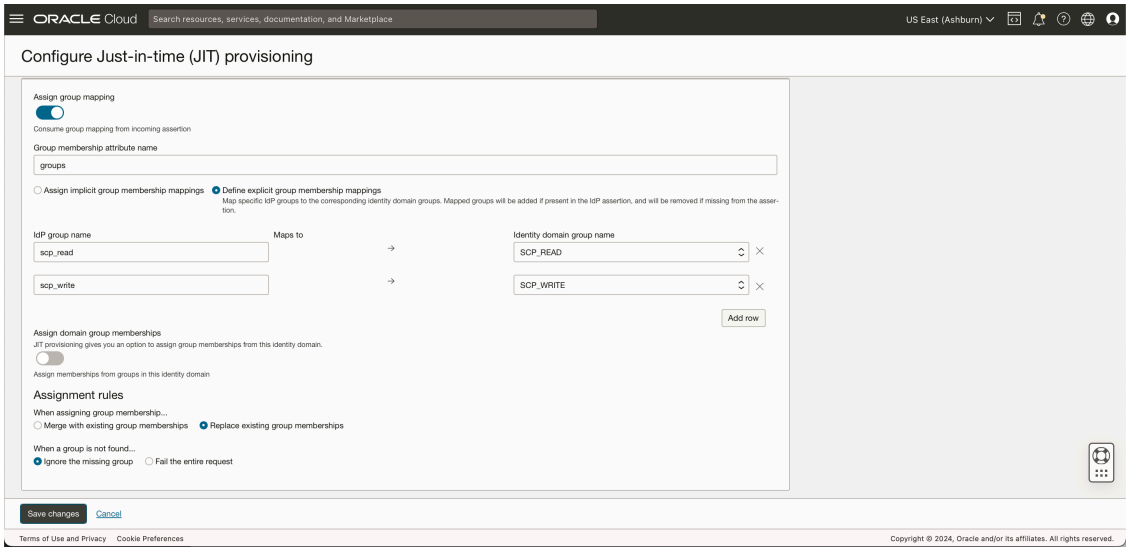
```
curl --location --request PUT 'http://{host}:{port}/occm-config/v1/occm/
logging' \
--header 'oc-cncc-id: Cluster1' \
--header 'oc-cncc-instance-id: Cluster1-occm-instance1' \
--header 'Authorization: Bearer eyJhbG...Yh8bJI_Owc_nb_hA' \
--header 'Content-Type: application/json' \
--data-raw '{
    "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 configuring will not work, and the 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 per in the following screenshot:

Figure 5-2 Sample Role Mapping



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 the users created in a particular identity domain, or the passwords of users present 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 OCI default password policy (**Custom**) will be applied.

> **Note**
>
> If required, OCI recommends the user to review each and every criteria, and modify the existing or create 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**
>
> Role(s) are referred as Group(s) 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:
The user assigned with 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 user has POLICY_READ group, then the user can only read configurations of any MOs configurations 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 user has POLICY_WRITE then the user can read or write or update or delete any MOs configurations within the NF.

> **Note**
>
> CNCC_READ/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 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.

The 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 access only 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. In case 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 NFs 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

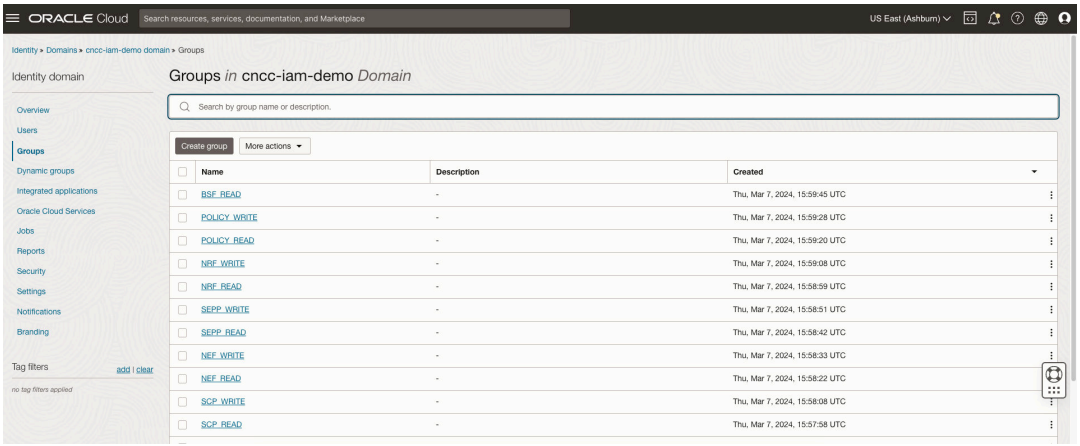
For information on exporting groups, see the Exporting Groups section in Oracle Cloud Infrastructure Documentation.

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



5.3.5 Assigning Groups to User Using CSV

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

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 in the `groups.csv` file.

> **Note**
>
> For multiple users use semi-colons (;) 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 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

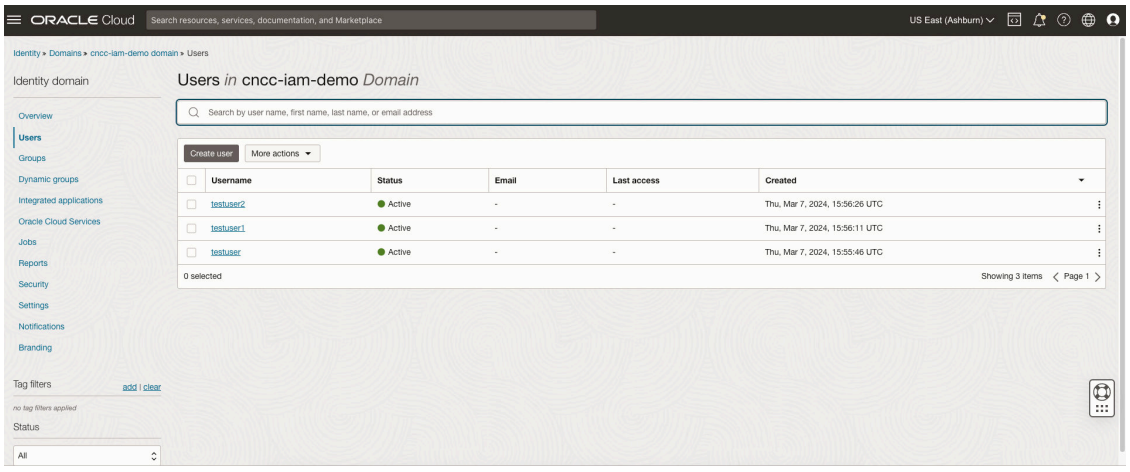
For information on exporting users, see the Exporting Users section in Oracle Cloud Infrastructure Documentation.

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 the users in your tenancy appears.

Figure 5-4 View Users



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 user's must beassigned 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 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.

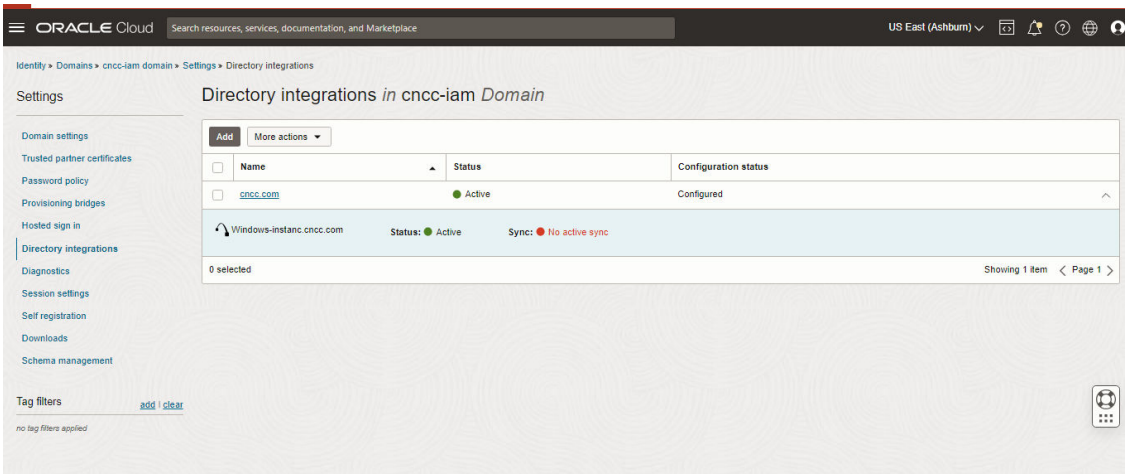
> **Note**
>
> You must enable the **Use SSL** flag while providing the active directory credentials.

5.5.3 Importing Users and Groups From Active Directory

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

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

Figure 5-5 Directory integration in cncc-iam Domain



> **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.

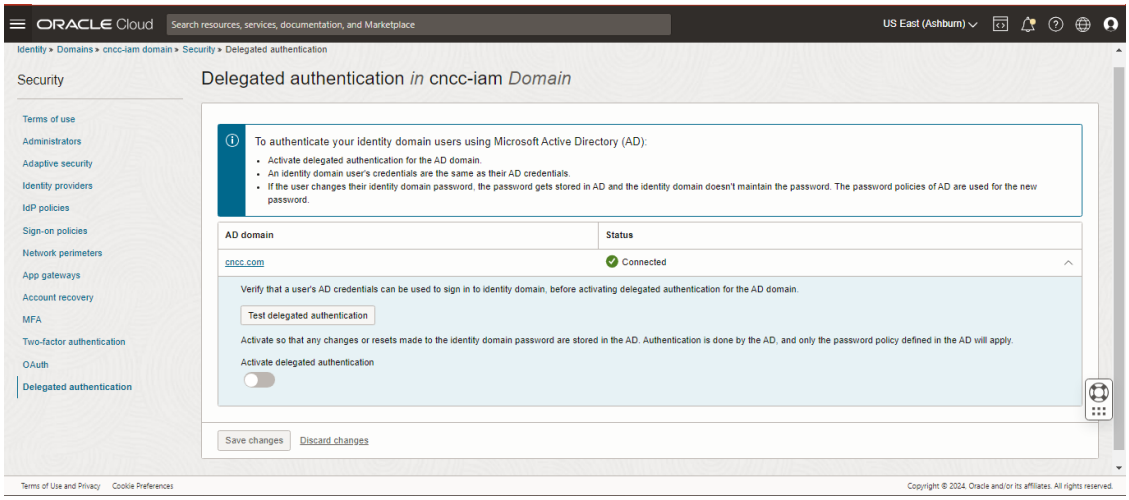
Procedure
To import users from active directory, see the Importing Users from Active Directory section in the Oracle Cloud Infrastructure Documentation.

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 Delegate Authentication.**

Figure 5-6 Delegated Authentication in CNC Console IAM Domain



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

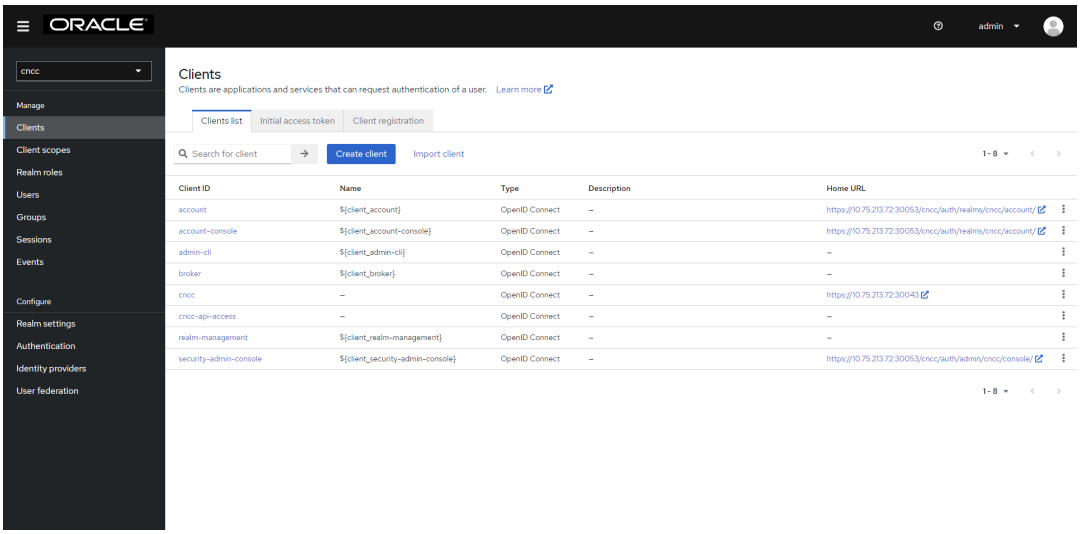
CNC Console IAM provides a REST API for generating and refreshing access tokens.

You must use the *cncc-api-access* client for accessing 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:

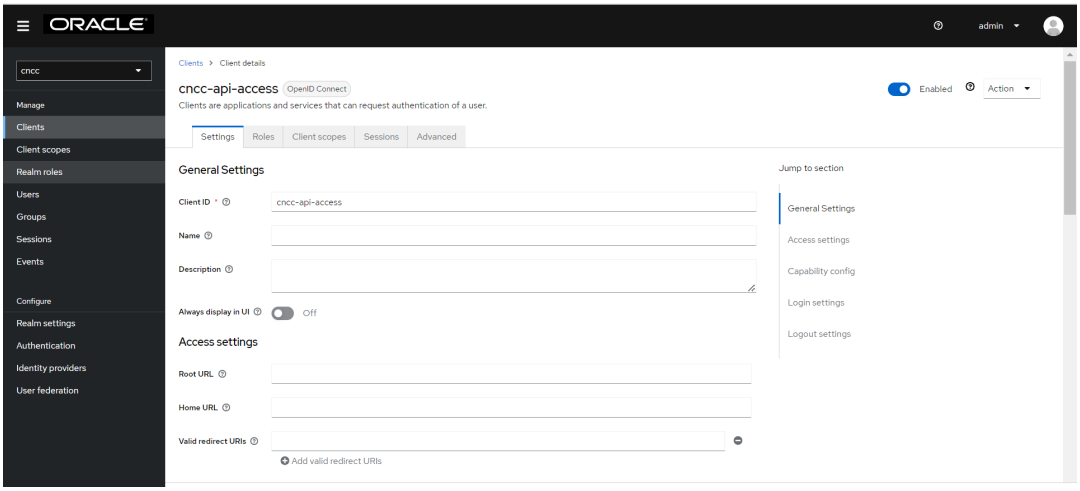
1. Log in to CNC Console IAM with valid credentials.
2. Click the **cncc** realm.
3. On the right pane, click **Clients**. The following screen appears:

Figure 6-1 Clients



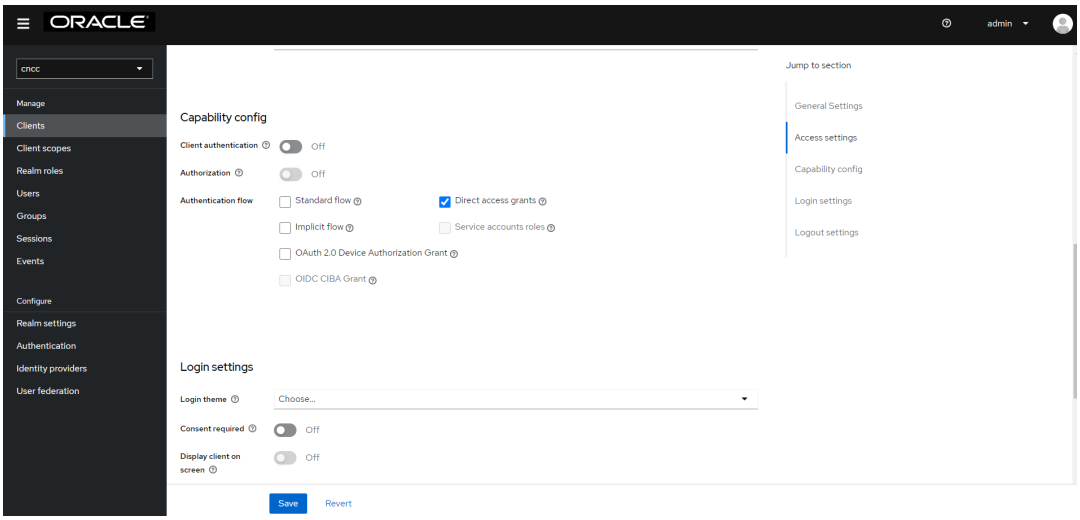
4. Click **cncc-api-access**. The following screen appears:

Figure 6-2 cncc-api-access



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



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://${cncc-iam-ingress-extrenal-ip}:${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 be the same.

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; but the POST method must have the refresh token instead of username and password. The format is as follows:

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

```
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>'
```

6.2 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:

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

<Base64 encoding of client credentials>	1. For information on getting clientId and clientSecret, see the Access ClientId and ClientSecret for Confidential Application section in *Oracle Communication Cloud Native Configuration Console Installation, Upgrade, and Fault Recovery Guide.*
	2. Base64 encoding of client credentials
	
	language: bash
	title: Command to encode client credentials in base64
	
	Format:
	`'clientId:clientSecret'`
	
	Command:
	`echo -n 'clientId:clientSecret' | base64`
	
	For Example:
	`echo -n 'asdfxxxxq3rF:Q3r4fsdxxxxxxxfv' | base64`
	`YXNkZnEzckY6UTNyNGZzZGZ2`

grant_type	The way to gets an access token. **Value:** password
username	Username of the OCI IAM user used for login.
password	Password of the OCI IAM user used for login.
scope	The way to limit the amount of access that is granted to an access token. **Values:** 1. **urn:opc:idm:__myscopes__** 2. **offline_access**

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.*.

<Base64 encoding of client credentials>	1. For information on getting clientId and clientSecret, see the Access ClientId and ClientSecret for Confidential Application section in *Oracle Communication Cloud Native Configuration Console Installation, Upgrade, and Fault Recovery Guide.*
	2. Base64 encoding of client credentials
	
	language: bash
	title: Command to encode client credentials in base64.
	
	Format:
	`'clientId:clientSecret'`
	
	Command:
	`echo -n 'clientId:clientSecret' | base64`
	
	For example:
	`echo -n 'asdfxxxxq3rF:Q3r4fsdxxxxxxxfv' | base64`
	`YXNkZnEzckY6UTNyNGZzZGZ2`

grant_type	The way to get an access token. **Value**: refresh_token
refresh_token	The actual refresh_token received while generating access_token as part of Access Token generation using User Credentials.

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.

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_ingressgateway"} **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_ingressgateway",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

7.2 M-CNCC Core Metrics

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

7.2.1 M-CNCC Core Requests

Table 7-13 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()

Table 7-13 (Cont.) M-CNCC Core Requests

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

7.2.2 M-CNCC Core Responses

Table 7-14 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 7-15 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()

Table 7-15 (Cont.) M-CNCC Core Success Responses

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

7.2.4 M-CNCC Core 5xx Responses

Table 7-16 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 7-17 M-CNCC Core 4xx Responses

Field	Description
Metric Details	Total number of error responses (4xx) for M-CNCC Core requests

Table 7-17 (Cont.) M-CNCC Core 4xx Responses

Field	Description
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 7-18 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 7-19 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*
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 7-20 M-CNCC Core Access Token Granted Responses

Field	Description
Metric Details	Total number of access token granted for M-CNCC Core

Table 7-20 (Cont.) M-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

7.2.9 M-CNCC Core Access Token Not Granted Responses

Table 7-21 M-CNCC Core Access Token Granted Responses

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

Table 7-21 (Cont.) M-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=~"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 7-22 M-CNCC Core User Login Failure Responses

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

Table 7-22 (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 7-23 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 7-23 (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 7-24 M-CNCC Core BSF Requests

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

Table 7-24 (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 7-25 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 7-25 (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 7-26 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 7-27 M-CNCC Core DD Responses

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

Table 7-27 (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 7-28 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 7-29 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 7-30 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 7-30 (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 7-31 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 7-32 M-CNCC Core NSSF Requests

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

Table 7-32 (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 7-33 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 NWDAF Requests

Table 7-34 M-CNCC Core NWDAF Requests

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

7.2.23 M-CNCC Core NWDAF Responses

Table 7-35 M-CNCC Core NWDAF Responses

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

7.2.24 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

7.2.25 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

7.2.26 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

7.2.27 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

7.2.28 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

7.2.29 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

7.2.30 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

7.2.31 M-CNCC Core UDR Responses

Table 7-43 M-CNCC Core UDR Responses

Field	Description
Metric Details	Total number of responses sent by M-CNCC Core for UDR
Metric Filter	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()
Dimensions	• Host • Method • Route_Path • InstanceIdentifier • InstanceId • ResourcePath • ResourceType • UserId • UserName
Metric Type	Counter

7.2.32 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

7.2.33 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

7.2.34 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

7.2.35 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

7.2.36 M-CNCC Core NEF Requests

Table 7-48 M-CNCC Core NEF Requests

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

7.2.37 M-CNCC Core NEF Responses

Table 7-49 M-CNCC Core NEF Responses

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

7.2.38 M-CNCC Core CAPIF Requests

Table 7-50 M-CNCC Core CAPIF Requests

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

7.2.39 M-CNCC Core CAPIF Responses

Table 7-51 M-CNCC Core CAPIF Responses

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

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 7-52 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 7-53 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 7-54 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 7-55 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 7-56 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 7-57 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 7-58 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 7-59 A-CNCC Core Access Token Granted Responses

Field	Description
Metric Details	Total number of access token granted for A-CNCC Core

Table 7-59 (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 7-60 A-CNCC Core Access Token Not Granted Responses

Field	Description
Metric Details	Total number of access token not granted for A-CNCC Core

Table 7-60 (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 7-61 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 7-61 (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 7-62 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 7-62 (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 7-63 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 7-64 A-CNCC Core NRF Requests

Field	Description
Metric Details	Total number of requests received by A-CNCC Core for NRF

Table 7-64 (Cont.) A-CNCC Core NRF Requests

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

7.3.14 A-CNCC Core NRF Responses

Table 7-65 A-CNCC Core NRF Responses

Field	Description
Metric Details	Total number of responses sent by A-CNCC Core for NRF
Metric Filter	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()

Table 7-65 (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 7-66 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 7-67 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 7-68 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 7-68 (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 7-69 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 7-70 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 7-71 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 7-71 (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 7-72 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 7-73 A-CNCC Core SEPP Responses

Field	Description
Metric Details	Total number of responses sent by A-CNCC Core for SEPP

Table 7-73 (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 7-74 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 7-75 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 7-76 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 7-77 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 7-78 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 7-79 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 NWDAF Requests

Table 7-80 A-CNCC Core NWDAF Requests

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

7.3.30 A-CNCC Core NWDAF Responses

Table 7-81 A-CNCC Core NWDAF Responses

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

7.3.31 A-CNCC Core cnDBTier Requests

Table 7-82 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

7.3.32 A-CNCC Core cnDBTier Responses

Table 7-83 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

7.3.33 A-CNCC Core OCCM Requests

Table 7-84 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

7.3.34 A-CNCC Core OCCM Responses

Table 7-85 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

7.3.35 A-CNCC Core NEF Requests

Table 7-86 A-CNCC Core NEF Requests

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

7.3.36 A-CNCC Core NEF Responses

Table 7-87 A-CNCC Core NEF Responses

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

7.3.37 A-CNCC Core CAPIF Requests

Table 7-88 A-CNCC Core CAPIF Requests

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

7.3.38 A-CNCC Core CAPIF Responses

Table 7-89 A-CNCC Core CAPIF Responses

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

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*

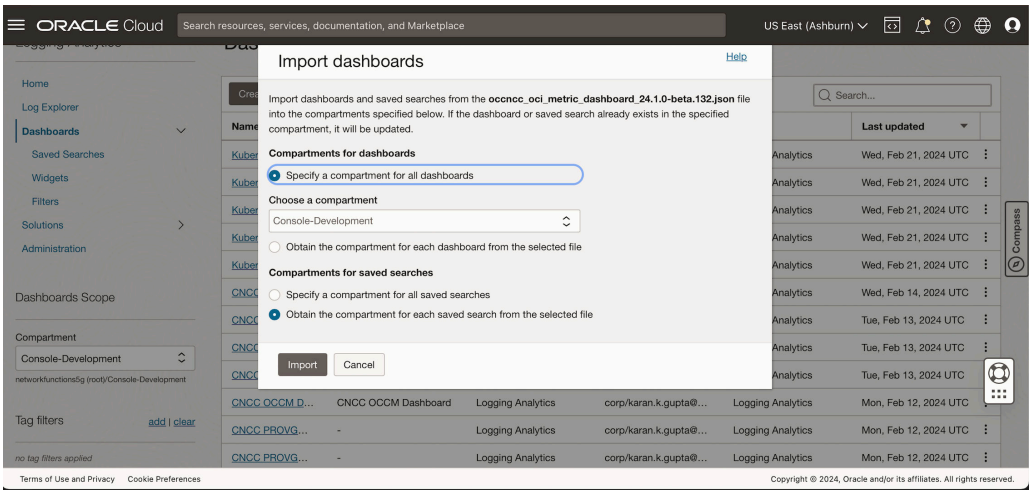
Viewing Dashboards on OCI Console

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_nef_metric_dashboard_<version>.json
occncc_oci_nrf_metric_dashboard_<version>.json
occncc_oci_nssf_metric_dashboard_<version>.json
occncc_oci_nwdaf_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
occncc_oci_capif_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**



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**
>
> **For OCI:**
> The only section applicable for OCI is CNC Console Alerts on OCI.

> **Note**
>
> - The user must use updated `occncc_agent_alertrules_<version>.yaml` file for agent cluster, in case of multi cluster deployment.
> - Use `occncc_manager_alertrules_<version>.yaml` file for single cluster deployment and in the manager cluster, in case of multi cluster deployment.

> **Note**
>
> Added the four sample alert files:
>
> - CNCC alert rules file for CNE without Prometheus Operator: `occncc_manager_alertrules_<version>.yaml` file for manager only or manager plus agent deployments and `occncc_agent_alertrules_<version>.yaml` file for agent only deployments.
> - CNCC alert rules file supporting CNE with Prometheus HA Operator: `occncc_manager_alerting_rules_promha_<version>.yaml` file for deployments in manager cluster and `occncc_agent_alertingrules_promha_<version>.yaml` file for deployments in agent only cluster.

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.

Table 8-1 (Cont.) Alerts Levels or Severity Types

Alerts Levels / Severity Types	Definition
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 CncclamTotalIngressTrafficRateAboveMinorThreshold

Table 8-2 CncclamTotalIngressTrafficRateAboveMinorThreshold

Field	Details
Trigger Condition	The total CNCC IAM Ingress Message rate has crossed the configured minor threshold of 700 TPS. Default value of this alert trigger point in *occncc_alertrules_<version>.yaml*is when CNCC IAM Ingress Rate crosses 70 % of 1000 (Maximum ingress request rate)
Severity	Minor
Alert details provided	**Description**: CNCC IAM Ingress traffic Rate is above the configured minor threshold i.e. 700 requests per second (current value is: {{ $value }}) **For CNE with Prometheus HA Operator:** **summary**: 'namespace: {{$labels.namespace}}, podname: {{$labels.pod}}, timestamp: {{ with query "time()" }}{{ . \| first \| value \| humanizeTimestamp }}{{ end }}: Traffic Rate is above 70 Percent of Max requests per second(1000)' **For CNE without Prometheus Operator :** **summary**: 'namespace: {{$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)'
OID used for SNMP Traps	1.3.6.1.4.1.323.5.3.51.1.2.7001
Metric Used	oc_ingressgateway_http_requests_total
Resolution	The alert is cleared either when the total Ingress Traffic rate falls below the Minor threshold or when the total traffic rate crosses the Major threshold, in which case the *CncclamTotalIngressTrafficRateAboveMajorThreshold* alert is raised. **Note:** The threshold is configurable in the *occncc_alertrules_<version>.yaml* file. Steps: 1. Reassess why the CNCC IAM is receiving additional traffic. 2. If this is unexpected, contact My Oracle Support.

8.1.2 CncclamTotalIngressTrafficRateAboveMajorThreshold

Table 8-3 CncclamTotalIngressTrafficRateAboveMajorThreshold

Field	Details
Trigger Condition	The total CNCC IAM Ingress Message rate has crossed the configured major threshold of 800 TPS. Default value of this alert trigger point ino *ccncc_alertrules_<version>.yaml* is when CNCC IAM Ingress Rate crosses 80 % of 1000 (Maximum ingress request rate)
Severity	Major
Alert details provided	**Description**: 'CNCC IAM Ingress traffic Rate is above the configured major threshold i.e. 800 requests per second (current value is: {{ $value }})' **For CNE with Prometheus HA Operator:** **summary**: 'namespace: {{$labels.namespace}}, podname: {{$labels.pod}}, timestamp: {{ with query "time()" }}{{ . \| first \| value \| humanizeTimestamp }}{{ end }}: Traffic Rate is above 80 Percent of Max requests per second(1000)' **For CNE without Prometheus Operator :** **summary**: 'namespace: {{$labels.kubernetes_namespace}}, podname: {{$labels.kubernetes_pod_name}}, timestamp: {{ with query "time()" }}{{ . | first | value | humanizeTimestamp }}{{ end }}: Traffic Rate is above 80 Percent of Max requests per second(1000)'
OID used for SNMP Traps	1.3.6.1.4.1.323.5.3.51.1.2.7001
Metric Used	oc_ingressgateway_http_requests_total
Resolution	The alert is cleared when the total Ingress Traffic rate falls below the Major threshold or when the total traffic rate crosses the Critical threshold, in which case the *CncclamTotalIngressTrafficRateAboveCriticalThreshold* alert is raised. **Note:** The threshold is configurable in the *occncc_alertrules_<version>.yaml* file. Steps: 1. Reassess why the CNCC IAM is receiving additional traffic. 2. If this is unexpected, contact My Oracle Support.

8.1.3 CncclamTotalIngressTrafficRateAboveCriticalThreshold

Table 8-4 CncclamTotalIngressTrafficRateAboveCriticalThreshold

Field	Details
Trigger Condition	The total CNCC IAM Ingress Message rate has crossed the configured critical threshold of 900TPS. Default value of this alert trigger point in *occncc_alertrules_<version>.yaml* is when CNCC IAM Ingress Rate crosses 90 % of 1000 (Maximum ingress request rate)
Severity	Critical

Table 8-4 (Cont.) CncclamTotalIngressTrafficRateAboveCriticalThreshold

Field	Details
Alert details provided	**Description**:CNCC IAM Ingress traffic Rate is above the configured critical threshold, that is, 900 requests per second (current value is: {{ $value }}) **For CNE with Prometheus HA Operator:** **summary**: 'namespace: {{$labels.namespace}}, podname: {{$labels.pod}}, timestamp: {{ with query "time()" }}{{ . \| first \| value \| humanizeTimestamp }}{{ end }}: Traffic Rate is above 90 Percent of Max requests per second(1000)' **For CNE without Prometheus Operator:** **summary**: 'namespace: {{$labels.kubernetes_namespace}}, podname: {{$labels.kubernetes_pod_name}}, timestamp: {{ with query "time()" }}{{ . | first | value | humanizeTimestamp }}{{ end }}: Traffic Rate is above 90 Percent of Max requests per second(1000)'
OID used for SNMP Traps	1.3.6.1.4.1.323.5.3.51.1.2.7001
Metric Used	oc_ingressgateway_http_requests_total
Resolution	The alert is cleared when the Ingress Traffic rate falls below the Critical threshold. **Note:** The threshold is configurable in the *occncc_alertrules_<version>.yaml* file. Steps: 1. Reassess why the CNCC IAM is receiving additional traffic. 2. If this is unexpected, contact My Oracle Support.

8.1.4 CncclamMemoryUsageCrossedMinorThreshold

Table 8-5 CncclamMemoryUsageCrossedMinorThreshold

Field	Details
Trigger Condition	A pod has reached the configured minor threshold(70%) of its memory resource limits.
Severity	Minor
Alert details provided	**Description**: 'CNCC IAM Memory Usage for pod {{ $labels.pod }} has crossed the configured minor threshold (70%) (value={{ $value }}) of its limit.' **Summary**: 'namespace: {{$labels.namespace}}, podname: {{$labels.pod}}, timestamp: {{ with query "time()" }}{{ . | first | value | humanizeTimestamp }}{{ end }} : Memory Usage of pod exceeded 70% of its limit.'
OID used for SNMP Traps	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
Resolution	The alert gets cleared when the memory utilization falls below the Minor Threshold or crosses the major threshold, in which case *CncclamMemoryUsageCrossedMajorThreshold* alert is raised. **Note:** The threshold is configurable in the *occncc_alertrules_<version>.yaml* file. If guidance is required, contact My Oracle Support.

8.1.5 CncclamMemoryUsageCrossedMajorThreshold

Table 8-6 CncclamMemoryUsageCrossedMajorThreshold

Field	Details
Trigger Condition	A pod has reached the configured major threshold(80%) of its memory resource limits.
Severity	Major
Alert details provided	**Description**: 'CNCC IAM Memory Usage for pod {{ $labels.pod }} has crossed the configured major threshold (80%) (value = {{ $value }}) of its limit.' **Summary**: 'namespace: {{$labels.namespace}}, podname: {{$labels.pod}}, timestamp: {{ with query "time()" }}{{ . | first | value | humanizeTimestamp }}{{ end }} : Memory Usage of pod exceeded 80% of its limit.'
OID used for SNMP Traps	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.
Resolution	The alert gets cleared when the memory utilization falls below the Major Threshold or crosses the critical threshold, in which case *CncclamMemoryUsageCrossedCriticalThreshold* alert shall be raised. **Note:** The threshold is configurable in the *occncc_alertrules_<version>.yaml* file. If guidance is required, contact My Oracle Support.

8.1.6 CncclamMemoryUsageCrossedCriticalThreshold

Table 8-7 CncclamMemoryUsageCrossedCriticalThreshold

Field	Details
Trigger Condition	A pod has reached the configured critical threshold (90%) of its memory resource limits
Severity	Critical

Table 8-7 (Cont.) CncclamMemoryUsageCrossedCriticalThreshold

Field	Details
Alert details provided	**Description**: 'CNCC IAM Memory Usage for pod {{ $labels.pod }} has crossed the configured critical threshold (90%) (value = {{ $value }}) of its limit.' **Summary**: 'namespace: {{$labels.namespace}}, podname: {{$labels.pod}}, timestamp: {{ with query "time()" }}{{ . | first | value | humanizeTimestamp }}{{ end }} : Memory Usage of pod exceeded 90% of its limit.'
OID used for SNMP Traps	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.
Resolution	The alert gets cleared when the memory utilization falls below the Critical Threshold. Note : The threshold is configurable in the *occncc_alertrules_<version>.yaml* file. If guidance is required, contact My Oracle Support.

8.1.7 CncclamTransactionErrorRateAbove0.1Percent

Table 8-8 CncclamTransactionErrorRateAbove0.1Percent

Field	Details
Trigger Condition	The number of failed transactions is above 0.1 percent of the total transactions.
Severity	Warning
Alert details provided	**Description**: 'CNCC IAM transaction Error rate is above 0.1 Percent of Total Transactions (current value is {{ $value }})' **Summary**: 'CNCC IAM transaction Error Rate detected above 0.1 Percent of Total Transactions'
OID used for SNMP Traps	1.3.6.1.4.1.323.5.3.51.1.2.7003
Metric Used	oc_ingressgateway_http_responses_total
Resolution	The alert is cleared when the number of failed transactions is below 0.1% of the total transactions or when the number of failed transactions crosses the 1% threshold in which case the *CncclamTransactionErrorRateAbove1Percent* is raised. Steps: 1. Check the Service specific metrics to understand the specific service request errors. 2. If guidance is required, contact My Oracle Support.

8.1.8 CncclamTransactionErrorRateAbove1Percent

Table 8-9 CncclamTransactionErrorRateAbove1Percent

Field	Details
Trigger Condition	The number of failed transactions is above 1 percent of the total transactions.
Severity	Warning
Alert details provided	**Description**: 'CNCC IAM transaction Error rate is above 1 Percent of Total Transactions (current value is {{ $value }})' **Summary**: 'CNCC IAM transaction Error Rate detected above 1 Percent of Total Transactions'
OID used for SNMP Traps	1.3.6.1.4.1.323.5.3.51.1.2.7003
Metric Used	oc_ingressgateway_http_responses_total
Resolution	The alert is cleared when the number of failed transactions is below 1% of the total transactions or when the number of failed transactions crosses the 10% threshold in which case the CncclamTransactionErrorRateAbove10Percent is raised. Steps: 1. Check the Service specific metrics to understand the specific service request errors. 2. If guidance is required, contact My Oracle Support.

8.1.9 CncclamTransactionErrorRateAbove10Percent

Table 8-10 CncclamTransactionErrorRateAbove10Percent

Field	Details
Trigger Condition	The number of failed transactions is above 10 percent of the total transactions.
Severity	Minor
Alert details provided	**Description**: CNCC IAM transaction Error rate is above 10 Percent of Total Transactions (current value is {{ $value }})' **Summary**: 'CNCC IAM transaction Error rate is above 10 Percent of Total Transactions (current value is {{ $value }})'
OID used for SNMP Traps	1.3.6.1.4.1.323.5.3.51.1.2.7003
Metric Used	oc_ingressgateway_http_responses_total
Resolution	The alert is cleared when the number of failed transactions is below 10% of the total transactions or when the number of failed transactions crosses the 25% threshold in which case the *CncclamTransactionErrorRateAbove25Percent* is raised. Steps: 1. Check the Service specific metrics to understand the specific service request errors. 2. If guidance is required, contact My Oracle Support.

8.1.10 CncclamTransactionErrorRateAbove25Percent

Table 8-11 CncclamTransactionErrorRateAbove25Percent

Field	Details
Trigger Condition	The number of failed transactions is above 25 percent of the total transactions.
Severity	Major
Alert details provided	**Description**: 'CNCC IAM transaction Error Rate detected above 25 Percent of Total Transactions (current value is {{ $value }})' **Summary**: 'CNCC IAM transaction Error Rate detected above 25 Percent of Total Transactions'
OID used for SNMP Traps	1.3.6.1.4.1.323.5.3.51.1.2.7003
Metric Used	oc_ingressgateway_http_responses_total
Resolution	TThe alert is cleared when the number of failed transactions are below 25% of the total transactions or when the number of failed transactions cross the 50% threshold in which case the *CncclamTransactionErrorRateAbove50Percent* is raised. Steps: 1. Check the Service specific metrics to understand the specific service request errors. 2. If guidance is required, contact My Oracle Support.

8.1.11 CncclamTransactionErrorRateAbove50Percent

Table 8-12 CncclamTransactionErrorRateAbove50Percent

Field	Details
Trigger Condition	The number of failed transactions is above 50 percent of the total transactions.
Severity	Critical
Alert details provided	**Description**: The number of failed transactions is above 50 percent of the total transactions. **Summary**: 'CNCC IAM transaction Error Rate detected above 50 Percent of Total Transactions'.
OID used for SNMP Traps	1.3.6.1.4.1.323.5.3.51.1.2.7003
Metric Used	oc_ingressgateway_http_responses_total
Resolution	The alert is cleared when the number of failed transactions is below 50 percent of the total transactions. The threshold is configurable in the *occncc_alertrules_<version>.yaml* file. Steps: 1. Check the Service specific metrics to understand the specific service request errors. 2. If guidance is required, contact My Oracle Support.

8.1.12 CncclamIngressGatewayServiceDown

Table 8-13 CncclamIngressGatewayServiceDown

Field	Details
Trigger Condition	The pods of the CNCC IAM Ingress Gateway microservice is available.
Severity	Critical
Alert details provided	**Description**:'CNCC IAM Ingress-Gateway service InstanceIdentifier=~".*cncc-iam_ingressgateway" is down' **For CNE with Prometheus HA Operator:** **summary**: 'namespace: {{$labels.namespace}}, podname: {{$labels.pod}}, timestamp: {{ with query "time()" }}{{ . | first | value | humanizeTimestamp }}{{ end }} : Ingress-gateway service down' **For CNE without Prometheus Operator:** **summary**: 'namespace: {{$labels.kubernetes_namespace}}, podname: {{$labels.kubernetes_pod_name}}, timestamp: {{ with query "time()" }}{{ . | first | value | humanizeTimestamp }}{{ end }} : Ingress-gateway service down'
OID used for SNMP Traps	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.
Resolution	The alert is cleared when the cncc-iam_ingressgateway service is available. Steps: 1. Check the orchestration logs of cncc-iam_ingressgateway service and check for liveness or readiness probe failures. 2. Refer to the application logs on Kibana and filter based on cncc-iam_ingressgateway service names. Check for ERROR WARNING logs related to thread exceptions. 3. Depending on the failure reason, take the resolution steps. 4. In case the issue persists, contact My Oracle Support.

8.1.13 CncclamFailedLogin

Table 8-14 CncclamFailedLogin

Field	Details
Trigger Condition	The count of failed login attempts in CNCC-IAM by a user goes above '3'
Severity	Warning

Table 8-14 (Cont.) CncclamFailedLogin

Field	Details
Alert details provided	**Description**:'{{ $value }} failed Login attempts have been detected in CNCC IAM for user {{$labels.UserName}}, the configured threshold value is 3 failed login attempts for every 5 min' **For CNE with Prometheus HA Operator:** **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' **For CNE without Prometheus Operator:** **summary**: 'namespace: {{$labels.kubernetes_namespace}}, podname: {{$labels.kubernetes_pod_name}}, timestamp: {{ with query "time()" }}{{ . | first | value | humanizeTimestamp }}{{ end }}: failed login attempts are more than the configured threshold value'
OID used for SNMP Traps	1.3.6.1.4.1.323.5.3.51.1.2.7005
Metric Used	oc_ingressgateway_http_responses_total
Resolution	The alert gets cleared when the total failed login attempts for a particular user goes below the threshold value (default value is '3') in the last 5 min (default value is 5 m). **Note:** The threshold and time is configurable in the *alerts.yaml* file. If guidance is required, contact My Oracle Support.

8.1.14 AdminUserCreation

Table 8-15 AdminUserCreation

Field	Details
Trigger Condition	If a new admin account is created in the last 5 min
Severity	Warning
Alert details provided	**For CNE with Prometheus HA Operator:** **Description**: '{{ $value }} admin users have been created by {{$labels.UserName}} ' summary: 'namespace: {{$labels.namespace}} **summary**: {{$labels.pod}}, user: {{$labels.UserName}}, timestamp: {{ with query "time()" }}{{ . | first | value | humanizeTimestamp }}{{ end }}: Admin users have been created ' **For CNE without Prometheus Operator:** **Description**: '{{ $value }} admin users have been created by {{$labels.UserName}} ' summary: 'namespace: {{$labels.kubernetes_namespace}} **summary**: {{$labels.kubernetes_pod_name}}, user: {{$labels.UserName}}, timestamp: {{ with query "time()" }}{{ . | first | value | humanizeTimestamp }}{{ end }}: Admin users have been created '
OID used for SNMP Traps	1.3.6.1.4.1.323.5.3.51.1.2.7006
Metric Used	oc_ingressgateway_http_requests_total

Table 8-15 (Cont.) AdminUserCreation

Field	Details
Resolution	The alert gets cleared when the total failed login attempts for a particular user go below the threshold value (default value is '3') in the last 5 min (default value is 5 m) **Note:** The threshold and time is configurable in the *occncc_alertrules_<version>.yaml* file. Login to admin GUI and review the user created. If guidance is required, contact My Oracle Support.

8.1.15 CncclamAccessTokenFailure

Table 8-16 CncclamAccessTokenFailure

Field	Details
Trigger Condition	If the count of failed token for CNCC-IAM goes above configured value of '3'
Severity	Warning
Alert details provided	**Description**: 'CNCC Iam Access Token Failure count is above the configured value i.e. 3 for every 5 min. Failed access token request count per second is (current value is: {{ $value }})' **For CNE with Prometheus HA Operator:** **summary**: 'namespace: {{$labels.namespace}}, podname: {{$labels.pod}}, timestamp: {{ with query "time()" }}{{ . \| first \| value \| humanizeTimestamp }}{{ end }}: Access Token Failure count is above the configured threshold value' **For CNE without Prometheus Operator:** **summary**: 'namespace: {{$labels.kubernetes_namespace}}, podname: {{$labels.pod}}, timestamp: {{ with query "time()" }}{{ . | first | value | humanizeTimestamp }}{{ end }}: Access Token Failure count is above the configured threshold value'
OID used for SNMP Traps	1.3.6.1.4.1.323.5.3.51.1.2.7007
Metric Used	oc_ingressgateway_http_responses_total
Resolution	The alert gets cleared when the total failed tokens for a particular user go below the threshold value (default value is '3') in the last 5 min (default value is 5 m) **Note:** The threshold and time is configurable in the *occncc_alertrules_<version>.yaml* file. If guidance is required, contact My Oracle Support.

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
Trigger Condition	The total CNCC Core Ingress Message rate has crossed the configured minor threshold of 700 TPS. Default value of this alert trigger point in cncc_alert_rules.yaml is when CNCC Core Ingress Rate crosses 70 % of 1000 (Maximum ingress request rate)
Severity	Minor
Alert details provided	**Description**: 'CNCC Core Ingress traffic Rate is above the configured minor threshold i.e. 700 requests per second (current value is: {{ $value }})' **For CNE with Prometheus HA Operator:** **summary**: 'namespace: {{$labels.namespace}}, podname: {{$labels.pod}}, timestamp: {{ with query "time()" }}{{ . \| first \| value \| humanizeTimestamp }}{{ end }}: Traffic Rate is above 70 Percent of Max requests per second(1000)' **For CNE without Prometheus Operator:** **summary**: 'namespace: {{$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)'
OID used for SNMP Traps	1.3.6.1.4.1.323.5.3.51.1.2.8001
Metric Used	oc_ingressgateway_http_requests_total
Resolution	The alert is cleared either when the total Ingress traffic rate falls below the minor threshold or when the total traffic rate crosses the major threshold, in which case the *CnccCoreTotalIngressTrafficRateAboveMajorThreshold* alert is raised. **Note:** The threshold is configurable in the *occncc_alertrules_<version>.yaml* file. Steps: 1. Reassess why the CNCC Core is receiving additional traffic. 2. If this is unexpected, contact My Oracle Support.

8.2.2 CnccCoreTotalIngressTrafficRateAboveMajorThreshold

Table 8-18 CnccCoreTotalIngressTrafficRateAboveMajorThreshold

Field	Details
Trigger Condition	The total CNCC Core Ingress Message rate has crossed the configured major threshold of 800 TPS. Default value of this alert trigger point in cncc_alert_rules.yaml is when CNCC Core Ingress Rate crosses 80 % of 1000 (Maximum ingress request rate)
Severity	Major

Table 8-18 (Cont.) CnccCoreTotalIngressTrafficRateAboveMajorThreshold

Field	Details
Alert details provided	**Description**: 'CNCC Core Ingress traffic Rate is above the configured major threshold i.e. 800 requests per second (current value is: {{ $value }})' **For CNE with Prometheus HA Operator:** **summary**: 'namespace: {{$labels.namespace}}, podname: {{$labels.pod}}, timestamp: {{ with query "time()" }}{{ . \| first \| value \| humanizeTimestamp }}{{ end }}: Traffic Rate is above 80 Percent of Max requests per second(1000)' **For CNE without Prometheus Operator :** **summary**: 'namespace: {{$labels.kubernetes_namespace}}, podname: {{$labels.kubernetes_pod_name}}, timestamp: {{ with query "time()" }}{{ . | first | value | humanizeTimestamp }}{{ end }}: Traffic Rate is above 80 Percent of Max requests per second(1000)'
OID used for SNMP Traps	1.3.6.1.4.1.323.5.3.51.1.2.8001
Metric Used	oc_ingressgateway_http_requests_total
Resolution	The alert is cleared when the total Ingress Traffic ratefalls below the Major threshold or when the total traffic rate cross the Critical threshold, in which case the CnccCoreTotalIngressTrafficRate Above CriticalThreshold. **Note:** The threshold is configurable in the *alerts.yaml* file. Steps: 1. Reassess why the CNCC Core is receiving additional traffic. 2. If this is unexpected, contact My Oracle Support.

8.2.3 CnccCoreTotalIngressTrafficRateAboveCriticalThreshold

Table 8-19 CnccCoreTotalIngressTrafficRateAboveCriticalThreshold

Field	Details
Trigger Condition	The total CNCC Core Ingress Message rate has crossed the configured critical threshold of 900TPS. Default value of this alert trigger point in cncc_alert_rules.yaml is when CNCC Core Ingress Rate crosses 90 % of 1000 (Maximum ingress request rate)
Severity	Critical

Table 8-19 (Cont.) CnccCoreTotalIngressTrafficRateAboveCriticalThreshold

Field	Details
Alert details provided	**Description**: 'CNCC Core Ingress traffic Rate is above the configured critical threshold i.e. 900 requests per second (current value is: {{ $value }})' **For CNE with Prometheus HA Operator:** **summary**: 'namespace: {{$labels.namespace}}, podname: {{$labels.pod}}, timestamp: {{ with query "time()" }}{{ . \| first \| value \| humanizeTimestamp }}{{ end }}: Traffic Rate is above 90 Percent of Max requests per second(1000)' **For CNE without Prometheus Operator :** **summary**: 'namespace: {{$labels.kubernetes_namespace}}, podname: {{$labels.kubernetes_pod_name}}, timestamp: {{ with query "time()" }}{{ . | first | value | humanizeTimestamp }}{{ end }}: Traffic Rate is above 90 Percent of Max requests per second(1000)'
OID used for SNMP Traps	1.3.6.1.4.1.323.5.3.51.1.2.8001
Metric Used	oc_ingressgateway_http_requests_total
Resolution	The alert is cleared when the Ingress Traffic rate falls below the Critical threshold. **Note:** The threshold is configurable in the *occncc_alertrules_<version>.yaml* file. Steps: 1. Reassess why the CNCC IAM is receiving additional traffic. 2. If this is unexpected, contact My Oracle Support.

8.2.4 CnccCoreMemoryUsageCrossedMinorThreshold

Table 8-20 CnccCoreMemoryUsageCrossedMinorThreshold

Field	Details
Trigger Condition	A pod has reached the configured minor threshold(70%) of its memory resource limits.
Severity	Minor
Alert details provided	**Description**: 'CNCC Core Memory Usage for pod {{ $labels.pod }} has crossed the configured minor threshold (70%) (value={{ $value }}) of its limit.' **Summary**: 'namespace: {{$labels.namespace}}, podname: {{$labels.pod}}, timestamp: {{ with query "time()" }}{{ . | first | value | humanizeTimestamp }}{{ end }} : Memory Usage of pod exceeded 70% of its limit.'
OID used for SNMP Traps	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 the similar metric as exposed by the monitoring system.

Table 8-20 (Cont.) CnccCoreMemoryUsageCrossedMinorThreshold

Field	Details
Resolution	The alert gets cleared when the memory utilization falls below the Minor Threshold or crosses the major threshold, in which case CnccCoreMemoryUsageCrossedMajorThreshold alert is raised. **Note:** The threshold is configurable in the *occncc_alertrules_<version>.yaml* file. If guidance is required, contact My Oracle Support.

8.2.5 CnccCoreMemoryUsageCrossedMajorThreshold

Table 8-21 CnccCoreMemoryUsageCrossedMajorThreshold

Field	Details
Trigger Condition	A pod has reached the configured major threshold(80%) of its memory resource limits.
Severity	Major
Alert details provided	**Description**: 'CNCC Core Memory Usage for pod {{ $labels.pod }} has crossed the configured major threshold (80%) (value = {{ $value }}) of its limit.' **Summary**: 'namespace: {{$labels.namespace}}, podname: {{$labels.pod}}, timestamp: {{ with query "time()" }}{{ . | first | value | humanizeTimestamp }}{{ end }} : Memory Usage of pod exceeded 80% of its limit.'
OID used for SNMP Traps	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 the similar metric as exposed by the monitoring system.
Resolution	The alert gets cleared when the memory utilization falls below the Major Threshold or crosses the critical threshold, in which case CnccCoreMemoryUsageCrossedCriticalThreshold alert is raised **Note:** The threshold is configurable in the *occncc_alertrules_<version>.yaml* file. If guidance is required, contact My Oracle Support.

8.2.6 CnccCoreMemoryUsageCrossedCriticalThreshold

Table 8-22 CnccCoreMemoryUsageCrossedCriticalThreshold

Field	Details
Trigger Condition	A pod has reached the configured critical threshold (90%) of its memory resource limits
Severity	Critical

Table 8-22 (Cont.) CnccCoreMemoryUsageCrossedCriticalThreshold

Field	Details
Alert details provided	**Description**: 'CNCC Core Memory Usage for pod {{ $labels.pod }} has crossed the configured critical threshold (90%) (value = {{ $value }}) of its limit.' **Summary**: 'namespace: {{$labels.namespace}}, podname: {{$labels.pod}}, timestamp: {{ with query "time()" }}{{ . | first | value | humanizeTimestamp }}{{ end }} : Memory Usage of pod exceeded 90% of its limit.'
OID used for SNMP Traps	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 the similar metric as exposed by the monitoring system.
Resolution	The alert gets cleared when the memory utilization falls below the Critical Threshold. **Note:** The threshold is configurable in the *occncc_alertrules_<version>.yaml* file. If guidance is required, contact My Oracle Support.

8.2.7 CnccCoreTransactionErrorRateAbove0.1Percent

Table 8-23 CnccCoreTransactionErrorRateAbove0.1Percent

Field	Details
Trigger Condition	The number of failed transactions is above 0.1 percent of the total transactions..
Severity	Warning
Alert details provided	**Description**:'CNCC Core transaction Error rate is above 0.1 Percent of Total Transactions (current value is {{ $value }})' **Summary**:'CNCC Core transaction Error rate is above 0.1 Percent of Total Transactions (current value is {{ $value }})'
OID used for SNMP Traps	1.3.6.1.4.1.323.5.3.51.1.2.8003
Metric Used	oc_ingressgateway_http_responses_total
Resolution	The alert is cleared when the number of failed transactions are below 0.1% of the total transactions or when the number of failed transactions cross the 1% threshold in which case the *CnccCoreTransactionErrorRateAbove1Percent* is raised. The threshold is configurable in the *alerts.yaml* file. Steps: 1. Check the Service specific metrics to understand the specific service request errors. 2. If guidance is required, contact My Oracle Support.

8.2.8 CnccCoreTransactionErrorRateAbove1Percent

Table 8-24 CnccCoreTransactionErrorRateAbove1Percent

Field	Details
Trigger Condition	The number of failed transactions is above 1 percent of the total transactions.
Severity	Warning
Alert details provided	**Description**: 'CNCC Core transaction Error rate is above 1 Percent of Total Transactions (current value is {{ $value }})' **Summary**:'CNCC Core transaction Error Rate detected above 1 Percent of Total Transactions'
OID used for SNMP Traps	1.3.6.1.4.1.323.5.3.51.1.2.8003
Metric Used	oc_ingressgateway_http_responses_total
Resolution	The alert is cleared when the number of failed transactions are below 1% of the total transactions or when the number of failed transactions crosses the 10% threshold in which case the *CnccCoreTransactionErrorRateAbove10Percent* is raised. Steps: 1. Check the Service specific metrics to understand the specific service request errors. 2. If guidance is required,contact My Oracle Support.

8.2.9 CnccCoreTransactionErrorRateAbove10Percent

Table 8-25 CnccCoreTransactionErrorRateAbove10Percent

Field	Details
Trigger Condition	The number of failed transactions is above 10 percent of the total transactions.
Severity	Minor
Alert details provided	**Description**: 'CNCC Core transaction Error rate is above 10 Percent of Total Transactions (current value is {{ $value }})' **summary**: 'CNCC Core ransaction Error Rate detected above 10 Percent of Total Transactions'
OID used for SNMP Traps	1.3.6.1.4.1.323.5.3.51.1.2.8003
Metric Used	oc_ingressgateway_http_responses_total
Resolution	The alert is cleared when the number of failed transactions are below 10% of the total transactions or when the number of failed transactions crosses the 25% threshold in which case the *CnccCoreTransactionErrorRateAbove25Percent* is raised. Steps: 1. Check the Service specific metrics to understand the specific service request errors. 2. If guidance is required, contact My Oracle Support.

8.2.10 CnccCoreTransactionErrorRateAbove25Percent

Table 8-26 CnccCoreTransactionErrorRateAbove25Percent

Field	Details
Trigger Condition	The number of failed transactions is above 25 percent of the total transactions.
Severity	Major
Alert details provided	**Description**: 'CNCC Core transaction Error Rate detected above 25 Percent of Total Transactions (current value is {{ $value }})' **Summary**: 'CNCC Core transaction Error Rate detected above 25 Percent of Total Transactions'
OID used for SNMP Traps	1.3.6.1.4.1.323.5.3.51.1.2.8003
Metric Used	oc_ingressgateway_http_responses_total
Resolution	The alert is cleared when the number of failed transactions are below 25% of the total transactions or when the number of failed transactions crosses the 50% threshold in which case the *CnccCoreTransactionErrorRateAbove50Percent* is raised. Steps: 1. Check the Service specific metrics to understand the specific service request errors. 2. If guidance is required, contact My Oracle Support.

8.2.11 CnccCoreTransactionErrorRateAbove50Percent

Table 8-27 CnccCoreTransactionErrorRateAbove50Percent

Field	Details
Trigger Condition	The number of failed transactions is above 50 percent of the total transactions.
Severity	Critical
Alert details provided	**Description**: 'CNCC Core transaction Error Rate detected above 50 Percent of Total Transactions (current value is {{ $value }})' **Summary**: 'CNCC Core transaction Error Rate detected above 50 Percent of Total Transactions'
OID used for SNMP Traps	1.3.6.1.4.1.323.5.3.51.1.2.8003
Metric Used	oc_ingressgateway_http_responses_total
Resolution	The alert is cleared when the number of failed transactions are below 50 percent of the total transactions Steps: 1. Check the Service specific metrics to understand the specific service request errors. 2. If guidance is required, contact My Oracle Support.

8.2.12 CnccCoreIngressGatewayServiceDown

Table 8-28 CnccCoreIngressGatewayServiceDown

Field	Details
Trigger Condition	Cncc Core Ingress Gateway service is down
Severity	Critical
Alert details provided	**Description**: 'CNCC Core Ingress-Gateway service InstanceIdentifier=~".*core_ingressgateway" is down' **For CNE with Prometheus HA Operator:** **summary**: 'namespace: {{$labels.namespace}}, podname: {{$labels.pod}}, timestamp: {{ with query "time()" }}{{ . | first | value | humanizeTimestamp }}{{ end }} : Ingress-gateway service down' **For CNE without Prometheus Operator:** **summary**: 'namespace: {{$labels.kubernetes_namespace}}, podname: {{$labels.kubernetes_pod_name}}, timestamp: {{ with query "time()" }}{{ . | first | value | humanizeTimestamp }}{{ end }} : Ingress-gateway service down'
OID used for SNMP Traps	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.
Resolution	The alert is cleared when the cncc-core_ingressgateway service is available. Steps: 1. Check the orchestration logs of cncc-core_ingressgateway service and check for liveness or readiness probe failures. 2. Refer the application logs on Kibana and filter based on cncc-core_ingressgateway service names. Check for ERROR WARNING logs related to thread exceptions. 3. Depending on the failure reason, take the resolution steps. 4. In case the issue persists, contact My Oracle Support.

8.2.13 CnccCoreFailedLogin

Table 8-29 CnccCoreFailedLogin

Field	Details
Trigger Condition	The count of failed login attempts in CNCC-Core by a user goes above '3'
Severity	Warning

Table 8-29 (Cont.) CnccCoreFailedLogin

Field	Details
Alert details provided	**Description**:'{{ $value }} failed Login attempts have been detected in CNCC Core for user {{$labels.UserName}}, the configured threshold value is 3 failed login attempts for every 5 min' **For CNE with Prometheus HA Operator:** **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' **For CNE without Prometheus Operator :** **summary**: 'namespace: {{$labels.kubernetes_namespace}}, podname: {{$labels.kubernetes_pod_name}}, timestamp: {{ with query "time()" }}{{ . | first | value | humanizeTimestamp }}{{ end }}: failed login attempts are more than the configured threshold value'
OID used for SNMP Traps	1.3.6.1.4.1.323.5.3.51.1.2.8005
Metric Used	oc_ingressgateway_http_responses_total
Resolution	The alert gets cleared when the total failed login attempts for a particular user go below the threshold value (default value is '3') in the last 5 min (default value is 5 m) **Note:** The threshold and time is configurable in the *alerts.yaml* file. If guidance is required, contact My Oracle Support.

8.2.14 CnccCoreUnauthorizedAccess

Table 8-30 CnccCoreUnauthorizedAccess

Field	Details
Trigger Condition	The count of failed login attempts in CNCC-Core by a user goes above '3'
Severity	Warning
Alert details provided	**Description**:'{{ $value }} Unauthorized Accesses have been detected in CNCC-Core for {{$labels.ResourceType}} for {{$labels.Method}} request. The configured threshold value is 3 for every 5 min' **For CNE with Prometheus HA Operator:** **summary**: 'namespace: {{$labels.namespace}}, podname: {{$labels.pod}}, timestamp: {{ with query "time()" }}{{ . \| first \| value \| humanizeTimestamp }}{{ end }}: Unauthorized Access for CNCC-Core are more than threshold value' **For CNE without Prometheus Operator :** **summary**: 'namespace: {{$labels.kubernetes_namespace}}, podname: {{$labels.kubernetes_pod_name}}, timestamp: {{ with query "time()" }}{{ . | first | value | humanizeTimestamp }}{{ end }}: Unauthorized Access for CNCC-Core are more than threshold value'
OID used for SNMP Traps	1.3.6.1.4.1.323.5.3.51.1.2.8006
Metric Used	oc_ingressgateway_http_responses_total

Table 8-30 (Cont.) CnccCoreUnauthorizedAccess

Field	Details
Resolution	The alert gets cleared when the total failed login attempts for a particular user go below the threshold value (default value is '3') in the last 5 min (default value is 5 m) **Note:** The threshold and time is configurable in the alerts.yaml If guidance is required, contact My Oracle Support.

8.2.15 CnccCoreAccessTokenFailure

Table 8-31 CnccCoreAccessTokenFailure

Field	Details
Trigger Condition	If the count of failed token for CNCC-Core goes above configured value of '3'
Severity	Warning
Alert details provided	**Description**: 'CNCC Core Access Token Failure count is above the configured value i.e. 3 for every 5 min. Failed access token request count per second is (current value is: {{ $value }})' **For CNE with Prometheus HA Operator:** **summary:** 'namespace: {{$labels.namespace}}, podname: {{$labels.kubernetes_pod_name}}, timestamp: {{ with query "time()" }}{{ . \| first \| value \| humanizeTimestamp }}{{ end }}: Access Token Failure count is above the configured threshold value' **For CNE without Prometheus Operator :** **summary**: 'namespace: {{$labels.kubernetes_namespace}}, podname: {{$labels.kubernetes_pod_name}}, timestamp: {{ with query "time()" }}{{ . | first | value | humanizeTimestamp }}{{ end }}: Access Token Failure count is above the configured threshold value'
OID used for SNMP Traps	1.3.6.1.4.1.323.5.3.51.1.2.8007
Metric Used	oc_ingressgateway_http_responses_total
Resolution	The alert gets cleared when the total failed tokens for a particular user go below the threshold value (default value is '3') in the last 5 min (default value is 5 m) **Note:** The threshold and time is configurable in the *alerts.yaml* file. If guidance is required, Contact My Oracle Support.

8.3 CNC Console Alert configuration in Prometheus

8.3.1 Applying Alerts Rule to CNE without Prometheus Operator

This section describes the measurement based Alert rules configuration for CNCC in Prometheus. Use updated `occncc_agent_alertrules_<version>.yaml` file for agent only deployments and `ccncc_manager_alertrules_<version>.yaml` for manager only as well as manager plus agent deployments.

This section includes the process of applying alert rules to Prometheus for Cloud Native Core Console (CNCC) using both the manual approach and the automated approach using the `oso-alr-config` Helm chart.

> **Note**
>
> - Default namespace configured in `occncc_manager_alerting_rules_promha_<version>.yaml` and `occncc_agent_alerting_rules_promha_<version>.yaml` is "cncc". User must update the namespace as per the deployment.
> - The Prometheus server takes an updated configuration map, which is automatically reloaded after approximately 60 seconds. To verify that the CNCC alerts have been reloaded, refresh the Prometheus GUI.

Manual Approach: Applying Alert Rules Using kubectl

Prerequisites

- Access to a Kubernetes cluster with OSO Prometheus installed.
- The CNCC alert rules yaml file (`occncc_manager_alertrules_<version>.yaml and occncc_agent_alertrules_<version>.yaml`).
- The necessary permissions to run `kubectl` commands.

Procedure

NAME :- Helm Release of Prometheus

Namespace :- Kubernetes NameSpace in which Prometheus is installed

1. Run the following command to take Backup of current config map of prometheus server:

   ```
   $ kubectl get configmaps <NAME>-server -o yaml -n <Namespace> > /tmp/
   tempConfig.yaml
   ```

   where, <Namespace> is the prometheus server namespace used in helm install command. This ensures you have a backup before making changes.
2. Verify and Add CNCC Alert file name in Prometheus ConfigMap:

   ```
   sed -i '/etc\/config\/alertscncc/d' /tmp/tempConfig.yaml
   sed -i '/rule_files:/a\    \   - /etc/config/alertscncc' /tmp/
   tempConfig.yaml
   ```

3. Run the following command to update the ConfigMap with the new Alert file name:

   ```
   kubectl replace configmap <NAME>-server -f /tmp/tempConfig.yaml
   ```

4. Run the following command to add CNCC Alert Rules in the ConfigMap:

   ```
   $ kubectl patch configmap <NAME>-server -n <Namespace> --type merge --
   patch "$(cat ~/occncc_manager_alertrules_<version>.yaml)"
   ```

This step merges the CNCC alert rules into the Prometheus ConfigMap.

Example:

```
$ kubectl get configmaps occne-prometheus-server -o yaml -n occne-infra
> /tmp/tempConfig.yaml
$ sed -i '/etc\/config\/alertscncc/d' /tmp/tempConfig.yaml
$ sed -i '/rule_files:/a\ \   - /etc/config/alertscncc' /tmp/
tempConfig.yaml
$ kubectl replace configmap occne-prometheus-server -f /tmp/tempConfig.yaml
$ kubectl patch configmap occne-prometheus-server -n occne-infra --type
merge --patch "$(cat ~/occncc_manager_alertrules_<version>.yaml)"
```

Automated Approach: Applying Alert Rules Using Helm Chart (oso-alr-config)

To optimize the CNC Console alert rule configuration process, the `oso-alr-config` Helm chart can be deployed. Customers can run a Helm upgrade to deploy the alert rules automatically.

For OSO alerts configuration installation and upgrade procedure, see "Automation of Alerts" section in *Operations Services Overlay Installation and Upgrade Guide*.

Prerequisite

OSO, `oso-alr-config` should be up and running. For more details, see *Operations Services Overlay Installation and Upgrade Guide*.

Applying Alerts

Run the following Helm upgrade command on `ocoso_alr_config_csar_<version>_alert_config_charts.tgz` helm chart using `ocoso_alr_config_csar_<version>_alert_config_custom_values.yaml` file to apply CNC Console `alertRule` files (`occncc_manager_alertrules_<version>.yaml and occncc_agent_alertrules_<version>.yaml`).

> **Note**
>
> Helm upgrade can be performed just with one input alertRule file or multiple input alertRule files.

```
# Command:
$ helm upgrade <oso-alr-config-release-name>
<ocoso_alr_config_csar_<version>_alert_config_charts.tgz> -f
<ocoso_alr_config_csar_<version>_alert_config_custom_values.yaml> -f
occncc_manager_alertrules_<version>.yaml -n <oso_namespace>

# Example 1: Sample command with one input alertRule file:
$ helm upgrade oso-alr-config
ocoso_alr_config_csar_<version>_alert_config_charts.tgz -f
ocoso_alr_config_csar_<version>_alert_config_custom_values.yaml -f
occncc_manager_alertrules_<version>.yaml -n oso

# Example 2: Sample command with two input alertRule file:
$ helm upgrade oso-alr-config
ocoso_alr_config_csar_<version>_alert_config_charts.tgz -f
ocoso_alr_config_csar_<version>_alert_config_custom_values.yaml -f
```

```
occncc_manager_alertrules_<version>.yaml -f <NF>_alertrules_<version>.yaml -n
oso
```

Field	Description
<oso-alr-config-release-name>	Release name of the oso-alr-config
<ocoso_alr_config_csar_<version>_alert_config_charts.tgz>	oso-alr-config helm chart package
<ocoso_alr_config_csar_<version>_alert_config_custom_values.yaml>	Custom values.yaml file of oso-alr-config helm chart
occncc_manager_alertrules_<version>.yaml and occncc_agent_alertrules_<version>.yaml	AlertRule files of CNC Console
<oso_namespace>	Namespace where oso-alr-config helm chart is deployed

Cleaning Up Alerts

An empty CNCC alert rule file, `occncc_empty_alertrules_<version>.yaml`, is included in the CNC Console package. This file can be used to remove all CNC Console alerts by running the helm upgrade command and providing `occncc_empty_alertrules_<version>.yaml` as the input file.

Sample helm upgrade command to clean up CNC Console alerts:

```
# Command:
$ helm upgrade <oso-alr-config-release-name>
<ocoso_alr_config_csar_<version>_alert_config_charts.tgz> -f
<ocoso_alr_config_csar_<version>_alert_config_custom_values.yaml> -f
occncc_empty_alertrules_<version>.yaml -n <oso_namespace>

# Example:
$ helm upgrade oso-alr-config
ocoso_alr_config_csar_<version>_alert_config_charts.tgz -f
ocoso_alr_config_csar_<version>_alert_config_custom_values.yaml -f
occncc_empty_alertrules_<version>.yaml -n oso
```

8.3.2 Applying Alerts Rule to CNE with Prometheus HA Operator

This section describes the measurement based Alert rules configuration for CNCC in Prometheus. Use updated `occncc_agent_alerting_rules_promha_<version>.yaml` file for agent only deployments and `occncc_manager_alerting_rules_promha_<version>.yaml` for manager only as well as manager plus agent deployments.

> **Note**
>
> Default namespace configured in `occncc_manager_alerting_rules_promha_<version>.yaml` and `occncc_agent_alerting_rules_promha_<version>.yaml` is "cncc-ns". User must update the namespace as per the deployment.

Run the following command to apply CNCC alerts file to create Prometheus rules Custom Resource Definition (CRD):

```
$ kubectl apply -f <file_name> -n <cncc namespace>
```

Where, <file_name> is the CNCC alerts file and <cncc namespace> is the CNCC namespace.

Example:

```
$ kubectl apply -f occncc_manager_alerting_rules_promha_<version>.yaml -n cncc
```

The following sample file delivered with CNC Console package:

```
occncc_manager_alerting_rules_promha_<version>.yaml
```

8.4 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

8.5 Disabling Alerts

This section explains how to disable the alerts in CNCC.

1. Edit *occncc_alerting_rules_promha_<version>.yaml* file to remove specific alert.
2. Remove complete content of the specific alert from the *occncc_alerting_rules_promha_<version>.yaml* file.

   cncc_alert_rules_<version>.yaml

   ```
   For example: If you want to remove
   CnccIamTotalIngressTrafficRateAboveMinorThreshold alert, remove the
   complete content:

   ## 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.

8.6 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
   ```

8.7 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.

8.8 CNC Console Alerts on OCI

This section provides information about CNC Console Alerts on OCI:

8.8.1 Configure CNC Console Alerts on OCI

> **Note**
>
> You must ensure that the user has the required roles and permissions to view or modify the alerts. For more information, see the Creating OCI User Management section in the *Oracle Communications Cloud Native Core OCI Adaptor, NF Deployment on OCI Guide.*

To configure CNC Console Alerts on OCI:

1. CNC Console CSAR package *occncc_csar_<version>.zip includes alarm files specific to OCI deployment.* These files are zipped as `occncc_oci_alertrules_<version>.zip` *and* placed in the Scripts directory of CSAR package.
2. Unzip the file to get `/occncc_oci_alertrules_<version>/occncc_oci` and `/occncc_oci_alertrules_<version>/occncc_oci_resources` these are the terraform folders to be applied in OCI console to create CNC Console related alarms.
3. Review the `occncc_oci_alertrules_<version>occncc_oci/alarms.tf` and `/occncc_oci_alertrules_<version>/occncc_oci_resources/alarms.tf` file, edit the value of the parameters in the file (if needed to be changed from default values) before configuring the alerts.
4. In `/occncc_oci_alertrules_<version>/occncc_oci/alarms.tf` k8Namespace is configured as kubernetes namespace in which CNC Console is deployed. Default value is cncc-ns. Update the `/occncc_oci/alarms.tf` file to reflect the correct CNC Console kubernetes namespace.
5. In `/occncc_oci_alertrules_<version>/occncc_oci_resources/alarms.tf` namespace is configured as kubernetes namespace in which CNC Console is deployed. Default value is cncc-ns. Update the /occncc_oci/alarms.tf file to reflect the correct CNC Console kubernetes namespace.
6. In `/occncc_oci_alertrules_<version>/occncc_oci/notifications.tf` and `/occncc_oci_alertrules_<version>/occncc_oci_resources/notifications.tf` update the subscription email, where resource `"email_subscription_1"` endpoint parameter should be updated with user email to which alarm trigger notification will be sent. Example:

```
resource "oci_ons_notification_topic" "notification_topic" {
  compartment_id = var.compartment_id
  name = var.topic_name
  description = "Send an email to the subscribed stakeholders"
}

resource "oci_ons_subscription" "email_subscription_1" {
  compartment_id = var.compartment_id
  endpoint = "xxx.oracle.com"
  protocol = "EMAIL"
```

```
  topic_id = oci_ons_notification_topic.notification_topic.id
}
```

To configure two or more email subscriptions, add new entries. For example:

```
resource "oci_ons_notification_topic" "notification_topic" {
  compartment_id = var.compartment_id
  name = var.topic_name
  description = "Send an email to the subscribed stakeholders"
}

resource "oci_ons_subscription" "email_subscription_1" {
  compartment_id = var.compartment_id
  endpoint = "xxx.oracle.com"
  protocol = "EMAIL"
  topic_id = oci_ons_notification_topic.notification_topic.id
}

resource "oci_ons_subscription" "email_subscription_2" {
  compartment_id = var.compartment_id
  endpoint = "yyy.oracle.com"
  protocol = "EMAIL"
  topic_id = oci_ons_notification_topic.notification_topic.id
}
```

7. Once alarms.tf and notifications.tf files are updated, these /occncc_oci and /occncc_oci_resources terraform folders can be applied in OCI console to create alarms.
8. Login to OCI Console GUI and search for "Stacks" or click on Hamburger menu, click on Developer Services, then click Stacks in the Resource Manager section → A page will open containing table with existing stacks if any along with Create stack button.
9. To apply these updated terraform folders /occncc_oci and /occncc_oci_resources, click on Create Stack button → This will open Create stack configuration page which includes three steps:
    a. Stack information: Choose the origin of the Terraform configuration to "My configuration" if not checked. Select "Folder" terraform configuration source in Stack Configuration to Brose and upload the updated terraform folders(one at a time: /occncc_oci or /occncc_oci_resources) Provide Name and Description (optional) and select the Create in compartment parameter in which terraform needs to applied, alarms should be created and click Next.
    b. Configure variables:
        i. Compartment Name: Choose the compartment in which metric namespace is present (namespace created for metrics as part of adapters installation).
        ii. Metric namespace: Input Namespace in which your Metrics will be populated. Note1: In case of /occncc_oci_alertrules_<version>/occncc_oci user should input the metric namespace created for metrics as part of adapters installation.

            Note2: In case of /occncc_oci_alertrules_<version>/occncc_oci_resources metric namespace is prepopulated.
        iii. Topic Name: Input name of the topic to be created.

> **Note**
>
> - Topic name must contain fewer than 256 characters. Only alphanumeric characters plus hyphens (-) and underscores (_) are allowed.
> - Topic name should be different for both the terraforms / occncc_oci_alertrules_<version>/occncc_oci and / occncc_oci_alertrules_<version>/occncc_oci_resources when applied on OCI console.

iv. Message Format: Select the message format as required, Format in which the notification will be sent.

v. Click Next.

c. Review: Verify your configuration variables and then Click Create.

10. A Stack will be created upon clicking on create button, Click on Plan button and wait for Plan job to succeed.
11. Once the Plan job succeeded, Click on Apply button on which right pane will pop up in which select previously ran "Apply job plan resolution" and then click on Apply.
12. Wait for the Apply job to get succeeded. Once Apply job is succeeded Alerts are created successfully.
13. Once Plan and Apply job is successfully completed, check the email Notification triggered to confirm on the topic subscription. Click on "Confirm subscription" link to confirm.
14. To Check the created alarms Click on Hamburger menu, click "Observability and Management", then click "Alarm Definitions" in Monitoring section to view all the alarms.

For more details, see Managing alarms in Oracle Cloud Infrastructure Documentation

Delete Created Alarms and Stack (terraforms applied for Alarms creation)

To delete created alarms:

1. Login to OCI Console GUI, click on Hamburger menu, click "Observability and Management", then click "Alarm Definitions" under Monitoring section and make sure right compartment is selected in left pane under "List scope". Check the alarms to be deleted, Click on "Actions" and then "Delete alarms".
2. Once all the alarms deleted, search for "Stacks" on OCI console search bar or click on Hamburger menu, click on Developer Services, then click on Stacks in Resource Manager section → A page will open containing table with existing stacks.
3. Click the stack names which was applied to create alarms and then click "Destroy".
4. Wait for "Destroy" job to complete and come back to Resources Manager → Stacks, click more options(three dots) for which destroy job was executed and perform "Delete" to delete the stack.

8.8.2 CnccCoreTotalIngressTrafficRateAboveMinorThreshold

Table 8-32 CnccCoreTotalIngressTrafficRateAboveMinorThreshold

Field	Details
Trigger Condition	The total CNCC Core Ingress Message rate has crossed the configured minor threshold of 700 TPS. Default value of this alert trigger point in cncc_alert_rules.yaml is when CNCC Core Ingress Rate crosses 70 % of 1000 (Maximum ingress request rate)
Severity	minor
Alert details provided	CNCC Core Ingress traffic Rate is above the configured minor threshold i.e. 700 requests per second
Metric Used	oc_ingressgateway_http_requests_total
Resolution	The alert is cleared either when the total Ingress Traffic rate falls below the Minor threshold or when the total traffic rate cross the Major threshold, inwhich case the CnccCoreTotalIngressTrafficRateAboveMajorThresholdalert shall be raised.Note: The threshold is configurable in the alerts.yamlSteps:Reassess why the CNCC Core is receiving additionaltraffic.If this is unexpected, contact My Oracle Support.

8.8.3 CnccCoreTotalIngressTrafficRateAboveMajorThreshold

Table 8-33 CnccCoreTotalIngressTrafficRateAboveMajorThreshold

Field	Details
Trigger Condition	The total CNCC Core Ingress Message rate has crossed the configured major threshold of 800 TPS. Default value of this alert trigger point in cncc_alert_rules.yaml is when CNCC Core Ingress Rate crosses 80 % of 1000 (Maximum ingress request rate)
Severity	major
Alert details provided	CNCC Core Ingress traffic Rate is above the configured major threshold i.e. 800 requests per second
Metric Used	oc_ingressgateway_http_requests_total
Resolution	The alert is cleared when the total Ingress Traffic ratefalls below the Major threshold or when the total traffic rate cross the Critical threshold, in which casethe CnccCoreTotalIngressTrafficRateAboveCriticalThresholdNote: The threshold is configurable in the alerts.yaml alert shall be raised.Steps:Reassess why the CNCC Core is receiving additionaltraffic.If this is unexpected, contact My Oracle Support.

8.8.4 CnccCoreTotalIngressTrafficRateAboveCriticalThreshold

Table 8-34 CnccCoreTotalIngressTrafficRateAboveCriticalThreshold

Field	Details
Trigger Condition	The total CNCC Core Ingress Message rate has crossed the configured critical threshold of 900TPS. Default value of this alert trigger point in cncc_alert_rules.yaml is when CNCC Core Ingress Rate crosses 90 % of 1000 (Maximum ingress request rate)
Severity	critical
Alert details provided	CNCC Core Ingress traffic Rate is above the configured critical threshold i.e. 900 requests per second
Metric Used	oc_ingressgateway_http_requests_total
Resolution	The alert is cleared when the Ingress Traffic rate falls below the Critical threshold.Note: The threshold is configurable in the alerts.yamlSteps:Reassess why the CNCC Core is receiving additional traffic.If this is unexpected, contact My Oracle Support.

8.8.5 CnccCoreMemoryUsageCrossedMinorThreshold

Table 8-35 CnccCoreMemoryUsageCrossedMinorThreshold

Field	Details
Trigger Condition	A pod has reached the configured minor threshold(70%) of its memory resource limits.
Severity	minor
Alert details provided	CNCC Core Memory Usage for pod has crossed the configured minor threshold (70%) of its limit.
Metric Used	container_memory_usage_bytes Note : This is a kubernetes metric used for instance availability monitoring. If the metric is not available, use the similar metric as exposed by the monitoring system.
Resolution	The alert gets cleared when the memory utilization falls below the Minor Threshold or crosses the major threshold, in which case CnccCoreMemoryUsageCrossedMajorThreshold alert shall be raised Note : The threshold is configurable in the alerts.yaml If guidance required, Contact My Oracle Support.

8.8.6 CnccCoreMemoryUsageCrossedMajorThreshold

Table 8-36 CnccCoreMemoryUsageCrossedMajorThreshold

Field	Details
Trigger Condition	A pod has reached the configured major threshold(80%) of its memory resource limits.
Severity	major

Table 8-36 (Cont.) CnccCoreMemoryUsageCrossedMajorThreshold

Field	Details
Alert details provided	CNCC Core Memory Usage for pod has crossed the configured major threshold (80%) of its limit.
Metric Used	container_memory_usage_bytes Note : This is a kubernetes metric used for instance availability monitoring. If the metric is not available, use the similar metric as exposed by the monitoring system.
Resolution	The alert gets cleared when the memory utilization falls below the Major Threshold or crosses the critical threshold, in which case CnccCoreMemoryUsageCrossedCriticalThreshold alert shall be raised Note : The threshold is configurable in the alerts.yaml If guidance required, Contact My Oracle Support.

8.8.7 CnccCoreMemoryUsageCrossedCriticalThreshold

Table 8-37 CnccCoreMemoryUsageCrossedCriticalThreshold

Field	Details
Trigger Condition	A pod has reached the configured critical threshold (90%) of its memory resource limits
Severity	critical
Alert details provided	CNCC Core Memory Usage for pod has crossed the configured critical threshold (90%) of its limit.
Metric Used	container_memory_usage_bytes Note : This is a kubernetes metric used for instance availability monitoring. If the metric is not available, use the similar metric as exposed by the monitoring system.
Resolution	The alert gets cleared when the memory utilization falls below the Critical Threshold. Note : The threshold is configurable in the alerts.yaml If guidance required, Contact My Oracle Support.

8.8.8 CnccCoreTransactionErrorRateAbovePointOnePercent

Table 8-38 CnccCoreTransactionErrorRateAbovePointOnePercent

Field	Details
Trigger Condition	The number of failed transactions is above 0.1 percent of the total transactions.
Severity	warning
Alert details provided	CNCC Core transaction Error rate is above 0.1 Percent of Total Transactions
Metric Used	oc_ingressgateway_http_responses_total

Table 8-38 (Cont.) CnccCoreTransactionErrorRateAbovePointOnePercent

Field	Details
Resolution	The alert is cleared when the number of failure transactions are below 0.1% of the total transactions or when the number of failure transactions cross the 1% threshold in which case the CnccCoreTransactionErrorRateAbove1Percent shall beraised.Steps:1. Check the Service specific metrics to understand the specific service request errors.2. If guidance required, contact My Oracle Support.

8.8.9 CnccCoreTransactionErrorRateAboveOnePercent

Table 8-39 CnccCoreTransactionErrorRateAboveOnePercent

Field	Details
Trigger Condition	The number of failed transactions is above 1 percent of the total transactions.
Severity	warning
Alert details provided	CNCC Core transaction Error rate is above 1 Percent of Total Transactions
Metric Used	oc_ingressgateway_http_responses_total
Resolution	The alert is cleared when the number of failure transactions are below 1% of the total transactions or when the number of failure transactions cross the 10% threshold in which case the CnccCoreTransactionErrorRateAbove10Percent shall beraised.Steps:1. Check the Service specific metrics to understand the specific service request errors.2. If guidance required, contact My Oracle Support.

8.8.10 CnccCoreTransactionErrorRateAboveTenPercent

Table 8-40 CnccCoreTransactionErrorRateAboveTenPercent

Field	Details
Trigger Condition	The number of failed transactions is above 10 percent of the total transactions.
Severity	minor
Alert details provided	CNCC Core transaction Error rate is above 10 Percent of Total Transactions
Metric Used	oc_ingressgateway_http_responses_total
Resolution	The alert is cleared when the number of failure transactions are below 10% of the total transactions or when the number of failure transactions cross the 25% threshold in which case the CnccCoreTransactionErrorRateAbove25Percent shall beraised.Steps:1. Check the Service specific metrics to understand the specific service request errors.2. If guidance required, contact My Oracle Support.

8.8.11 CnccCoreTransactionErrorRateAboveTwentyFivePercent

Table 8-41 CnccCoreTransactionErrorRateAboveTwentyFivePercent

Field	Details
Trigger Condition	The number of failed transactions is above 25 percent of the total transactions.
Severity	major
Alert details provided	CNCC Core transaction Error Rate detected above 25 Percent of Total Transactions
Metric Used	oc_ingressgateway_http_responses_total
Resolution	The alert is cleared when the number of failure transactions are below 25% of the total transactions or when the number of failure transactions cross the 50% threshold in which case the CnccCoreTransactionErrorRateAbove50Percent shall beraised.Steps:1. Check the Service specific metrics to understand the specific service request errors.2. If guidance required, contact My Oracle Support.

8.8.12 CnccCoreTransactionErrorRateAboveFiftyPercent

Table 8-42 CnccCoreTransactionErrorRateAboveFiftyPercent

Field	Details
Trigger Condition	The number of failed transactions is above 50 percent of the total transactions.
Severity	critical
Alert details provided	CNCC Core transaction Error Rate detected above 50 Percent of Total Transactions
Metric Used	oc_ingressgateway_http_responses_total
Resolution	The alert is cleared when the number of failure transactions are below 50 percent of the total transactions.Steps:1. Check the Service specific metrics to understand the specific service request errors.2. If guidance required, contact My Oracle Support.

8.8.13 CnccCoreUnauthorizedAccess

Table 8-43 CnccCoreUnauthorizedAccess

Field	Details
Trigger Condition	If the count of unauthorized access goes above the configured value of '3'
Severity	warning
Alert details provided	Unauthorized Accesses have been detected in CNCC-Core for request. The configured threshold value is 3 for every 5 min
Metric Used	oc_ingressgateway_http_responses_total

Table 8-43 (Cont.) CnccCoreUnauthorizedAccess

Field	Details
Resolution	The alert gets cleared when the total unauthorized accesses for a particular user go below the threshold value (default value is '3') in the last 5 min (default value is 5 m) Note : The threshold and time is configurable in the alerts.yaml If guidance required, Contact My Oracle Support.

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

Expression	1. **For CNE without Prometheus Operator**: all: sum(irate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*mcore_ingressgateway",kubernetes_namespace="$namespace"}[2m])) **For CNE with Prometheus HA Operator**: all: sum(irate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*mcore_ingressgateway",namespace="$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. nwdaf: sum(irate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*ocnwdaf.*|.*ocnwdafapi.*"} [2m]))

Table 9-5 (Cont.) M-CNCC Core Requests

	12. cndbtier: sum(irate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*ocdbtier.*"} [2m]))
	13. occm: sum(irate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*occm-config.*"} [2m]))
	14. nef: sum(irate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/nef-configuration/.*|.*/nef/.*"} [2m]))
	15. capif: sum(irate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/capif-configuration/.*|.*/capif/.*"} [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()

Table 9-5 (Cont.) M-CNCC Core Requests

	10. nwdaf: oc_ingressgateway_http_requests_total[10m] {InstanceIdentifier=~'"*mcore_ingressgateway",Route_path=~\"*ocnwdaf*|*ocnwdafapi*"}.sum() 11. occm: oc_ingressgateway_http_requests_total[10m] {InstanceIdentifier=~'"*mcore_ingressgateway",Route_path=~"*occm-config*"}.sum() 12. nef: oc_ingressgateway_http_requests_total[10m] {InstanceIdentifier=~'"*mcore_ingressgateway",ResourcePath=~'"*/nef-configuration/*|*/nef/*"}.sum() 13. capif: oc_ingressgateway_http_requests_total[10m] {InstanceIdentifier=~'"*mcore_ingressgateway",ResourcePath=~'"*/capif-configuration/*|*/capif/*"}.sum()

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

	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. nwdaf_success:sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath~".*ocnwdaf.*|.*ocnwdafapi.*",kubernetes_namespace="$namespace"}[5m])) 14. cndbtier_success:sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath~".*ocdbtier.*",kubernetes_namespace="$namespace"}[5m])) 15. occm_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*occm-config.*",kubernetes_namespace="$namespace"}[5m])) 16. nef_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*/nef-configuration/.*\|.*/nef/.*",kubernetes_namespace="$namespace"}[5m])) 17. capif_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*/capif-configuration/.*|.*/capif/.*",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]))

Table 9-6 (Cont.) M-CNCC Core Responses

	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="$namespace"}[5m]))
	5. nrf_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*/nrf-configuration/v1.*|.*/nrf-state-data/.*|.*/ocnrf-swagger/.*|.*/nrf-status-data/.*|.*/nrf/nf-common-component/.*",namespace="$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]))

Table 9-6 (Cont.) M-CNCC Core Responses

	12. provgw_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*provgw-config.*|.*provgw.*",namespace="$namespace"}[5m])) 13. nwdaf_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*ocnwdaf.*|.*ocnwdafapi.*",namespace="$namespace"}[5m])) 14. cndbtier_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*ocdbtier.*",namespace="$namespace"}[5m])) 15. occm_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*occm-config.*",namespace="$namespace"}[5m])) 16. nef_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*/nef-configuration/.*|.*/nef/.*",namespace="$namespace"}[5m])) 17. capif_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*/capif-configuration/.*|.*/capif/.*",namespace="$namespace"}[5m])) **For OCI:** 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()

Table 9-6 (Cont.) M-CNCC Core Responses

	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] {InstanceIdentifier=~"*mcore_ingressgateway",Status=~"2*",Route_path=~"*provgw-config*|*provgw*",k8Namespace="cncc-ns"}.sum() 14. nwdaf_success: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*mcore_ingressgateway",Status=~"2*",Route_path=~"*ocnwdaf*|*ocnwdafapi*",k8Namespace="cncc-ns"}.sum() 15. occm_success: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*mcore_ingressgateway",Status=~"2*",Route_path=~"*occm-config*",k8Namespace="cncc-ns"}.sum() 16. nef_success: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*mcore_ingressgateway",Status=~"2*",ResourcePath=~"*/nef-configuration/*|*/nef/*",k8Namespace="cncc-ns"}.sum() 17. capif_success: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*mcore_ingressgateway",Status=~"2*",ResourcePath=~"*/capif-configuration/*|*/capif/*",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

Expression	**For CNE without Prometheus Operator:** 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

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. nwdaf: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*ocnwdaf.*|.*ocnwdafapi.*",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=~".*ocnwdaf.*|.*ocnwdafapi.*",kubernetes_namespace="$namespace"}[5m]))*100 12. cndbtier: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*ocdbtier.*",kubernetes_namespace="$namespace"}[5m]))/ sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*ocdbtier.*",kubernetes_namespace="$namespace"}[5m]))*100 13. occm: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*occm-config.*",kubernetes_namespace="$namespace"}[5m]))/ sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*occm-config.*",kubernetes_namespace="$namespace"}[5m]))*100 14. nef: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*/nef-configuration/.*\|.*/nef/.*",kubernetes_namespace="$namespace"}[5m]))/ sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/nef-configuration/.*|.*/nef/.*",kubernetes_namespace="$namespace"}[5m]))*100 15. capif: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*/capif-configuration/.*\|.*/capif/.*",kubernetes_namespace="$namespace"}[5m]))/ sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/capif-configuration/.*|.*/capif/.*",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

Table 9-7 (Cont.) M-CNCC Core Success Rate

	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-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

Table 9-7 (Cont.) M-CNCC Core Success Rate

	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. nwdaf: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*ocnwdaf.*|.*ocnwdafapi.*",namespace="$namespace"}[5m]))/ sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*ocnwdaf.*|.*ocnwdafapi.*",namespace="$namespace"}[5m]))*100 12. cndbtier: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*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 13. 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 14. nef: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*/nef-configuration/.*|.*/nef/.*",namespace="$namespace"}[5m]))/

Table 9-7 (Cont.) M-CNCC Core Success Rate

	sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/nef-configuration/.*|.*/nef/.*",namespace="$namespace"}[5m]))*100 15. capif: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"2.*",ResourcePath=~".*/capif-configuration/.*\|.*/capif/.*",namespace="$namespace"}[5m]))/ sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/capif-configuration/.*|.*/capif/.*",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. nwdaf: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*mcore_ingressgateway",Status=~"2*",Route_path=~"*ocnwdaf*|*ocnwdafapi*",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=~"*ocnwdaf*|*ocnwdafapi*",k8Namespace="cncc-ns"}.rate().grouping().sum() * 100 12. 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] {InstanceIdentifier=~"*mcore_ingressgateway",Route_path=~"*occm-config*",k8Namespace="cncc-ns"}.rate().grouping().sum() * 100 13. nef: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*mcore_ingressgateway",Status=~"2*",ResourcePath=~"*/nef-configuration/*|*/nef/*",k8Namespace="cncc-ns"}.rate().grouping().sum()/ oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*mcore_ingressgateway",ResourcePath=~"*/nef-configuration/*|*/nef/*",k8Namespace="cncc-ns"}.rate().grouping().sum() * 100 14. capif: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*mcore_ingressgateway",Status=~"2*",ResourcePath=~"*/capif-configuration/*|*/capif/*",k8Namespace="cncc-ns"}.rate().grouping().sum()/ oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*mcore_ingressgateway",ResourcePath=~"*/capif-configuration/*|*/capif/*",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

Expression	**For CNE without Prometheus Operator:** 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

Table 9-8 (Cont.) M-CNCC Core Error Rate

	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. nwdaf: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"4.*|5.*",ResourcePath=~".*ocnwdaf.*|.*ocnwdafapi.*",kubernetes_

Table 9-8 (Cont.) M-CNCC Core Error Rate

	namespace="$namespace"}[5m]))/ sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*ocnwdaf.*\|.*ocnwdafapi.*",kubernetes_namespace="$namespace"}[5m]))*100 12. cndbtier: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"4.*|5.*",ResourcePath=~".*ocdbtier.*",kubernetes_namespace="$namespace"}[5m]))/ sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*ocdbtier.*",kubernetes_namespace="$namespace"}[5m]))*100 13. 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 14. nef: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"4.*|5.*",ResourcePath=~".*/nef-configuration/.*|.*/nef/.*",kubernetes_namespace="$namespace"}[5m]))/ sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/nef-configuration/.*\|.*/nef/.*",kubernetes_namespace="$namespace"}[5m]))*100 15. capif: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"4.*|5.*",ResourcePath=~".*/capif-configuration/.*|.*/capif/.*",kubernetes_namespace="$namespace"}[5m]))/ sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/capif-configuration/.*\|.*/capif/.*",kubernetes_namespace="$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.*|

Table 9-8 (Cont.) M-CNCC Core Error Rate

	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 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]))/

Table 9-8 (Cont.) M-CNCC Core Error Rate

	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=~".*provgw-config.*|.*provgw.*",namespace="$namespace"}[5m]))*100 11. nwdaf: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"4.*\|5.*",ResourcePath=~".*ocnwdaf.*\|.*ocnwdafapi.*",namespace="$namespace"}[5m]))/ sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*ocnwdaf.*|.*ocnwdafapi.*",namespace="$namespace"}[5m]))*100 12. cndbtier: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"4.*\|5.*",ResourcePath=~".*ocdbtier.*",namespace="$namespace"}[5m]))/ sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*ocdbtier.*",namespace="$namespace"}[5m]))*100 13. occm: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"4.*\|5.*",ResourcePath=~".*occm-config.*",namespace="$namespace"}[5m]))/

Table 9-8 (Cont.) M-CNCC Core Error Rate

	sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*occm-config.*",namespace="$namespace"}[5m]))*100 14. nef: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"4.*\|5.*",ResourcePath=~".*/nef-configuration/.*\|.*/nef/.*",namespace="$namespace"}[5m]))/ sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/nef-configuration/.*|.*/nef/.*",namespace="$namespace"}[5m]))*100 15. capif: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",Status=~"4.*\|5.*",ResourcePath=~".*/capif-configuration/.*\|.*/capif/.*",namespace="$namespace"}[5m]))/ sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*mcore_ingressgateway",ResourcePath=~".*/capif-configuration/.*|.*/capif/.*",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-

Table 9-8 (Cont.) M-CNCC Core Error Rate

	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=~"*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

Table 9-8 (Cont.) M-CNCC Core Error Rate

	10. provgw: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*mcore_ingressgateway",Status=~"4*|5*",Route_path=~"*provgw-config*|*provgw*",k8Namespace="cncc-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. nwdaf: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*mcore_ingressgateway",Status=~"4*|5*",Route_path=~"*ocnwdaf*|*ocnwdafapi*",k8Namespace="cncc-ns"}.increment().grouping().sum()/ oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*mcore_ingressgateway",Route_path=~"*ocnwdaf*|*ocnwdafapi*",k8Namespace="cncc-ns"}.increment().grouping().sum() * 100 12. 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 13. nef: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*mcore_ingressgateway",Status=~"4*|5*",ResourcePath=~"*/nef-configuration/*|*/nef/*",k8Namespace="cncc-ns"}.increment().grouping().sum()/ oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*mcore_ingressgateway",ResourcePath=~"*/nef-configuration/*|*/nef/*",k8Namespace="cncc-ns"}.increment().grouping().sum() * 100 14. capif: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*mcore_ingressgateway",Status=~"4*|5*",ResourcePath=~"*/capif-configuration/*|*/capif/*",k8Namespace="cncc-ns"}.increment().grouping().sum()/ oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*mcore_ingressgateway",ResourcePath=~"*/capif-configuration/*|*/capif/*",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

Expression	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. nwdaf: sum(irate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*ocnwdaf.*|.*ocnwdafapi.*"} [2m]))

Table 9-9 (Cont.) A-CNCC Core Requests

	12. cndbtier: sum(irate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*ocdbtier.*"}[2m])) 13. occm: sum(irate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*occm-config.*"} [2m])) 14. nef: sum(irate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*/nef-configuration/.*|.*/nef/.*"} [2m])) 15. capif: sum(irate(oc_ingressgateway_http_requests_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*/capif-configuration/.*|.*/capif/.*"} [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()

Table 9-9 (Cont.) A-CNCC Core Requests

	9. nwdaf: oc_ingressgateway_http_requests_total[10m] {InstanceIdentifier=~"*acore_ingressgateway",Route_path=~"*ocnwdaf*|*ocnwdafapi*"}.sum() 10. occm: oc_ingressgateway_http_requests_total[10m] {InstanceIdentifier=~"*acore_ingressgateway",Route_path=~"*occm-config*"}.sum() 11. nef: oc_ingressgateway_http_requests_total[10m] {InstanceIdentifier=~"*acore_ingressgateway",Route_path=~"*/nef-configuration/*|*/nef/*"}.sum() 12. capif: oc_ingressgateway_http_requests_total[10m] {InstanceIdentifier=~"*acore_ingressgateway",Route_path=~"*/capif-configuration/*|*/capif/*"}.sum()

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

Expression	**For CNE without Prometheus Operator** 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

Table 9-10 (Cont.) A-CNCC Core Responses

	=~".*/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. nwdaf_success:sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path~".*ocnwdaf.*|.*ocnwdafapi.*",kubernetes_namespace="$namespace"}[5m]))
	14. cndbtier_success:sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path~".*ocdbtier.*",kubernetes_namespace="$namespace"}[5m]))
	15. occm_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path=~".*occm-config.*",kubernetes_namespace="$namespace"}[5m]))
	16. nef_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path=~".*/nef-configuration/.*|.*/nef/.*",kubernetes_namespace="$namespace"}[5m]))
	17. capif_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path=~".*/capif-configuration/.*|.*/capif/.*",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]))

Table 9-10 (Cont.) A-CNCC Core Responses

	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="$namespace"}[5m])) 5. nrf_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path=~".*/nrf-configuration/v1.*\|.*/nrf-state-data/.*\|.*/ocnrf-swagger/.*\|.*/nrf-status-data/.*\|.*/nrf/nf-common-component/.*",namespace="$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]))

Table 9-10 (Cont.) A-CNCC Core Responses

	12. provgw_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path=~".*provgw-config.*|.*provgw.*",namespace="$namespace"}[5m])) 13. nwdaf_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path=~".*ocnwdaf.*|.*ocnwdafapi.*",namespace="$namespace"}[5m])) 14. cndbtier_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path=~".*ocdbtier.*",namespace="$namespace"}[5m])) 15. occm_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path=~".*occm-config.*",namespace="$namespace"}[5m])) 16. nef_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path=~".*/nef-configuration/.*|.*/nef/.*",namespace="$namespace"}[5m])) 17. capif_success: sum(irate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path=~".*/capif-configuration/.*|.*/capif/.*",namespace="$namespace"}[5m])) **For OCI:** 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()

Table 9-10 (Cont.) A-CNCC Core Responses

	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] {InstanceIdentifier=~"*acore_ingressgateway",Status=~"2*",Route_path=~"*provgw-config*|*provgw*",k8Namespace="cncc-ns"}.sum() 14. nwdaf_success: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*acore_ingressgateway",Status=~"2*",Route_path=~"*ocnwdaf*|*ocnwdafapi*",k8Namespace="cncc-ns"}.sum() 15. occm_success: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*acore_ingressgateway",Status=~"2*",Route_path=~"*occm-config*",k8Namespace="cncc-ns"}.sum() 16. nef_success: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*acore_ingressgateway",Status=~"2*",Route_path=~"*/nef-configuration/*|*/nef/*",k8Namespace="cncc-ns"}.sum() 17. capif_success: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*acore_ingressgateway",Status=~"2*",Route_path=~"*/capif-configuration/*|*/capif/*",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

Expression	**For CNE without Prometheus Operator:** 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

Table 9-11 (Cont.) A-CNCC Core Success Rate

	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. nwdaf: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path=~".*ocnwdaf.*|.*ocnwdafapi.*",kubernetes_namespace="$namespace"}[5m]))/ sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*ocnwdaf.*

Table 9-11 (Cont.) A-CNCC Core Success Rate

	|.*ocnwdafapi.*",kubernetes_namespace="$namespace"} [5m]))*100 12. cndbtier: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path=~".*ocdbtier.*",kubernetes_namespace="$namespace"}[5m]))/ sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*ocdbtier.*",kubernetes_namespace="$namespace"}[5m]))*100 13. occm: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path=~".*occm-config.*",kubernetes_namespace="$namespace"}[5m]))/ sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*occm-config.*",kubernetes_namespace="$namespace"}[5m]))*100 14. nef: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path=~".*/nef-configuration/.*\|.*/nef/.*",kubernetes_namespace="$namespace"}[5m]))/ sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*/nef-configuration/.*|.*/nef/.*",kubernetes_namespace="$namespace"}[5m]))*100 15. capif: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"2.*",Route_path=~".*/capif-configuration/.*\|.*/capif/.*",kubernetes_namespace="$namespace"}[5m]))/ sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*/capif-configuration/.*|.*/capif/.*",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

Table 9-11 (Cont.) A-CNCC Core Success Rate

	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="$namespace"}[5m]))/ sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*core_ingressgateway",ResourcePath=~".*/policyapi/.*|.*/oc-cnpolicy-configuration/.*|.*/pcf/.*",namespace="$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

Table 9-11 (Cont.) A-CNCC Core Success Rate

	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. nwdaf: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*core_ingressgateway",Status=~"2.*",Route_path=~".*ocnwdaf.*|.*ocnwdafapi.*",namespace="$namespace"}[5m]))/ sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*core_ingressgateway",Route_path=~".*ocnwdaf.*|.*ocnwdafapi.*",namespace="$namespace"}[5m]))*100 12. cndbtier: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*core_ingressgateway",Status=~"2.*",Route_path=~".*ocdbtier.*",namespace="$namespace"}[5m]))/ sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*core_ingressgateway",Route_path=~".*ocdbtier.*",namespace="$namespace"}[5m]))*100 13. 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 14. nef: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*core_ingressgateway",Status=~"2.*",Route_path=~".*/nef-configuration/.*|.*/nef/.*",namespace="$namespace"}[5m]))/

Table 9-11 (Cont.) A-CNCC Core Success Rate

	sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*core_ingressgateway",Route_path=~".*/nef-configuration/.*|.*/nef/.*",namespace="$namespace"}[5m]))*100 15. capif: sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*core_ingressgateway",Status=~"2.*",Route_path=~".*/capif-configuration/.*\|.*/capif/.*",namespace="$namespace"}[5m]))/ sum(rate(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*core_ingressgateway",Route_path=~".*/capif-configuration/.*|.*/capif/.*",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

Table 9-11 (Cont.) A-CNCC Core Success Rate

	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()/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. nwdaf: oc_ingressgateway_http_responses_total[10m]{InstanceIdentifier=~"*acore_ingressgateway",Status=~"2*",Route_path=~"*ocnwdaf*|*ocnwdafapi*",k8Namespace="cncc-ns"}.rate().grouping().sum()/oc_ingressgateway_http_responses_total[10m]

Table 9-11 (Cont.) A-CNCC Core Success Rate

	{InstanceIdentifier=~"*acore_ingressgateway",Route_path=~"*ocnwdaf*|*ocnwdafapi*",k8Namespace="cncc-ns"}.rate().grouping().sum() * 100 12. 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 13. nef: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*acore_ingressgateway",Status=~"2*",Route_path=~"*/nef-configuration/*|*/nef/*",k8Namespace="cncc-ns"}.rate().grouping().sum()/ oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*acore_ingressgateway",Route_path=~"*/nef-configuration/*|*/nef/*",k8Namespace="cncc-ns"}.rate().grouping().sum() * 100 14. capif: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*acore_ingressgateway",Status=~"2*",Route_path=~"*/capif-configuration/*|*/capif/*",k8Namespace="cncc-ns"}.rate().grouping().sum()/ oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*acore_ingressgateway",Route_path=~"*/capif-configuration/*|*/capif/*",k8Namespace="cncc-ns"}.rate().grouping().sum() * 100

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

Expression	**For CNE without Prometheus Operator:** 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

Table 9-12 (Cont.) A-CNCC Core Error Rate

	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. nwdaf: sum(increase(oc_ingressgateway_http_responses_total{Insta

Table 9-12 (Cont.) A-CNCC Core Error Rate

	nceIdentifier=~".*acore_ingressgateway",Status=~"4.*|5.*",Route_path=~".*ocnwdaf.*|.*ocnwdafapi.*",kubernetes_namespace="$namespace"}[5m]))/ sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",InstanceIdentifier=~".*core_ingressgateway",Route_path=~".*ocnwdaf.*\|.*ocnwdafapi.*",kubernetes_namespace="$namespace"}[5m]))*100 12. cndbtier: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*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 13. 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 14. nef: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"4.*|5.*",Route_path=~".*/nef-configuration/.*|.*/nef/.*",kubernetes_namespace="$namespace"}[5m]))/ sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",InstanceIdentifier=~".*core_ingressgateway",Route_path=~".*/nef-configuration/.*\|.*/nef/.*",kubernetes_namespace="$namespace"}[5m]))*100 15. capif: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"4.*|5.*",Route_path=~".*/capif-configuration/.*|.*/capif/.*",kubernetes_namespace="$namespace"}[5m]))/ sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",InstanceIdentifier=~".*core_ingressgateway",Route_path=~".*/capif-configuration/.*\|.*/capif/.*",kubernetes_namespace="$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

Table 9-12 (Cont.) A-CNCC Core Error Rate

	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="$namespace"}[5m]))/ sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Route_path=~".*/nudr-dr-prov/.*\|.*/nudr-dr-mgm/.*\|.*/nudr-group-id-map-prov/.*\|.*/slf-group-prov/.*\|.*/nudr-config/.*\|.*/udr/nf-common-component/.*\|.*/n5g-eir-prov/.*",namespace="$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{Insta

Table 9-12 (Cont.) A-CNCC Core Error Rate

	nceIdentifier=~".*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{InstanceIdentifier=~".*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. nwdaf: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"4.*|5.*",Route_path=~".*ocnwdaf.*|.*ocnwdafapi.*",namespace="$namespace"}[5m]))/ sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",InstanceIdentifier=~".*core_ingressgateway",Route_path=~".*ocnwdaf.*\|.*ocnwdafapi.*",namespace="$namespace"}[5m]))*100 12. 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

Table 9-12 (Cont.) A-CNCC Core Error Rate

	13. 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 14. nef: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"4.*|5.*",Route_path=~".*/nef-configuration/.*|.*/nef/.*",namespace="$namespace"}[5m]))/ sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",InstanceIdentifier=~".*core_ingressgateway",Route_path=~".*/nef-configuration/.*\|.*/nef/.*",namespace="$namespace"}[5m]))*100 15. capif: sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",Status=~"4.*|5.*",Route_path=~".*/capif-configuration/.*|.*/capif/.*",namespace="$namespace"}[5m]))/ sum(increase(oc_ingressgateway_http_responses_total{InstanceIdentifier=~".*acore_ingressgateway",InstanceIdentifier=~".*core_ingressgateway",Route_path=~".*/capif-configuration/.*\|.*/capif/.*",namespace="$namespace"}[5m]))*100 16. **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()/

Table 9-12 (Cont.) A-CNCC Core Error Rate

	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=~"*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

Table 9-12 (Cont.) A-CNCC Core Error Rate

	9. dd: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*acore_ingressgateway",Status=~"4*|5*",Route_path=~"*ocnadd*|*ocnaddapi*",k8Namespace="cncc-ns"}.increment().grouping().sum()/ 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. nwdaf: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*acore_ingressgateway",Status=~"4*|5*",Route_path=~"*ocnwdaf*|*ocnwdafapi*",k8Namespace="cncc-ns"}.increment().grouping().sum()/ oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*acore_ingressgateway",Route_path=~"*ocnwdaf*|*ocnwdafapi*",k8Namespace="cncc-ns"}.increment().grouping().sum() * 100 12. 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 13. nef: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*acore_ingressgateway",Status=~"4*|5*",Route_path=~"*/nef-configuration/*|*/nef/*",k8Namespace="cncc-ns"}.increment().grouping().sum()/ oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*acore_ingressgateway",Route_path=~"*/nef-configuration/*|*/nef/*",k8Namespace="cncc-ns"}.increment().grouping().sum() * 100 14. capif: oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*acore_ingressgateway",Status=~"4*|5*",Route_path=~"*/capif-configuration/*|*/capif/*",k8Namespace="cncc-ns"}.increment().grouping().sum()/ oc_ingressgateway_http_responses_total[10m] {InstanceIdentifier=~"*acore_ingressgateway",Route_path=~"*/capif-configuration/*|*/capif/*",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
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()