Oracle® Communications

Cloud Native Core, Operations Services Overlay User Guide

Release 25.2.200
G45390-01
March 2026

Oracle Communications Cloud Native Core, Operations Services Overlay User Guide, Release 25.2.200

G45390-01

Contents

Preface

Documentation Accessibility

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

Access to Oracle Support

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

Diversity and Inclusion

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

Conventions

The following text conventions are used in this document:

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

My Oracle Support

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

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

- 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**.
- For Hardware, Networking and Solaris Operating System Support, select **3**.

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 provides information about the acronyms used in the document.

Table Acronyms

Acronym	Description
CNE	Cloud Native Environment
CNC	Cloud Native Core
CRD	Custom Resource Definition
CSAR	Cloud Service Archive
NF	Network Function
OCCNE	Oracle Communications Cloud Native Environment
ONAP	Open Network Automation Platform
OSO	Operations Services Overlay
OSDC	Oracle Software Download Center
TSDB	Time Series Database

What's New in This Guide

This section lists the documentation updates for release 25.2.2xx.

Release 25.2.200- G45390-01, March 2026

Added the feature description in the Alert Forwarding to Kafka section.

1 Overview

The Oracle Communications Operations Services Overlay (OSO) installs and configures common operation services. For example, you can install and configure Prometheus and its components like AlertManager in a previously installed Kubernetes cluster.

OSO is an independent deliverable distinct from Oracle Communications Cloud Native Core, Cloud Native Environment (CNE). OSO assists efficient integration of network measurement mechanisms, resource allocation, and dynamic decision-making based on a common set of measurements.

This document guides the OSO users to operate and maintain OSO services.

1.1 References

Following are the reference documents:

- *Oracle Communications Cloud Native Core, Operations Services Overlay Installation and Upgrade Guide*
- *Oracle Communications Cloud Native Core, Operations Services Overlay Network Impact Report*
- *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*

2

Installing and Configuring OSO

Oracle Communications Operations Services Overlay (OSO) can be installed on any supported version of Kubernetes. For more information about how to install and configure OSO, see *Oracle Communications Cloud Native Core, Operations Services Overlay Installation and Upgrade Guide*.

3
OSO Services

Oracle Communications Operations Services Overlay (OSO) provides the following observability services to help users observe the behavior of applications running in their cloud native environment.

3.1 Metrics

This section describes the observability metrics provided by the OSO services.

3.1.1 Metrics Collection

OSO uses Prometheus to collect metrics from all the Cloud Native Core (CNC) applications deployed in the user's CNE. Prometheus collects metrics from the following:

- OSO services: Prometheus collects and stores metrics generated by the OSO services, such as Prometheus and AlertManager.
- CNC Network Functions: OSO Prometheus receives customer metrics generated and delivered by the CNC NFs.

3.1.2 Storing Metrics

OSO Prometheus stores all the metrics in an internal Time Series Database (TSDB).

3.1.3 Support for Time Series Database (TSDB) Snapshot

Prometheus uses Time Series Database (TSDB) to store the metrics. Along with metric storage, OSO captures a snapshot at a specific point of time with the available data in the Prometheus data store. OSO allows you to capture these snapshots without shutting down or disrupting the Prometheus instance.

This feature can be used for the following:

- Backups
- Recovery
- Debugging

For more information about the procedure for capturing TSDB snapshots, see the "Creating Backup of Prometheus Time Series Database (TSDB) Using Snapshot Utility" section in *Oracle Communications Cloud Native Core, Operations Services Overlay Installation and Upgrade Guide*.

3.2 OSO Alerts

OSO uses AlertManager to raise alerts. These alerts inform the user about the aspect of OSO that requires attention.

The applications deployed on OSO define their alerts to inform the user about the problems specific to each application. For more information on how applications can load alerting rules, see the Updating Alert Rules for an NF section.

Automated Configuration of NF Alerts

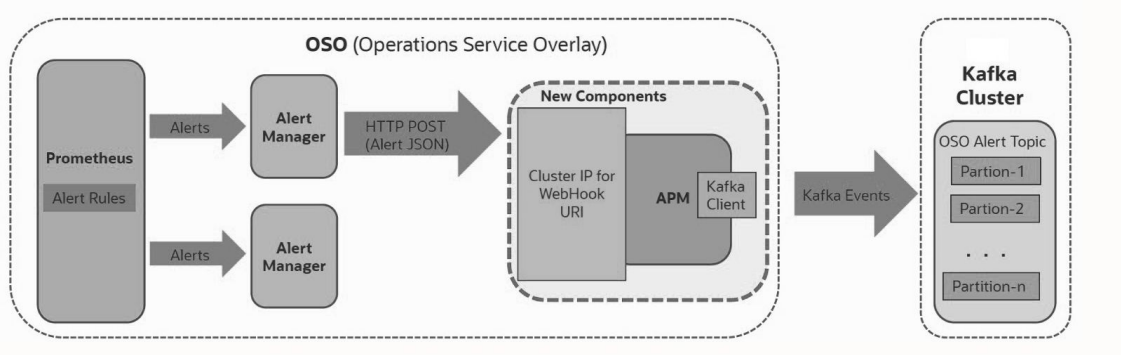
From release 25.1.200 onwards, OSO offers an automated mechanism to configure NF alerts after the successful deployment of all respective NFs and OSO components. This feature streamlines the management of OSO alert rules by dynamically updating the Prometheus ConfigMap during the Helm chart installation or upgrade process. As a result, manual intervention is minimized, ensuring consistent and efficient alert rule deployment across environments.

For more information about automation of alert configuration in OSO, see the "Automated Configuration of NF Alerts" section in *Oracle Communications Cloud Native Core, Operations Services Overlay Installation and Upgrade Guide*.

3.2.1 Alert Forwarding to Kafka

With this feature, OSO introduces a Alert Processing Microservice (APM) that consumes HTTP-based alerts from OSO (Alert manager), converts them into Kafka-compatible JSON messages, and publishes them to a designated Kafka cluster. This feature can be deployed independently using a dedicated Helm chart, offering flexibility to add or remove it without affecting the core OSO platform components.

Figure 3-1 Alert Forwarding to Kafka



The image illustrates the alert processing flow enabled by the APM within OSO:

1. Prometheus continuously monitors the system metrics and applies alert rules to detect issues or threshold breaches.
2. When an alert condition is met, Prometheus sends the alert to the Alert Manager within OSO. The Alertmanager aggregates the incoming alerts for further processing based on the configuration.
3. The Alert Manager sends the processed alert as a JSON payload via an HTTP POST request to the Cluster IP.
4. The APM receives the HTTP POSTed alert.
5. APM's Kafka client converts the alert into a Kafka-compatible JSON message. It manages retries, backoff, and delivery timeouts for reliable delivery.
6. APM publishes the alert to the Kafka Cluster.

7. The Kafka Cluster stores the alerts.

Managing the Alert Forwarding to Kafka

The APM feature is **optional** in OSO. It is not enabled by default and must be explicitly installed and configured if alert forwarding to Kafka is needed. If omitted, core OSO functionality is unaffected.

This section lists the Helm configuration details for this feature.

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

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

1. Open the `ocoso_csar_25_2_200_0_0_0_apm_custom_values.yaml` file.
2. Set the value of `apmEnabled` to *true* to enable the alert forwarding to Kafka cluster using APM.
3. Configure the values for the following parameters:
   - `env.kafkaIp` -Indicates the external bootstrap service IP, which maps to the Kafka brokers.
   - `env.kafkaPort` -Indicates the external bootstrap service port which maps to the Kafka brokers.
   - `env.kafkaTopic` -Indicates the Kafka topic to which APM will publish alert messages.
   - `env.kafkaGroupId` -Indicates a name that the user provides for the Kafka group id.

   Note

   Configure the following parameters in the `ocoso_csar_25_2_200_0_0_0_alm_custom_values.yaml` file.

   ```
   config:
      receivers:
       - name: default-receiver
          webhook_configs:
            - url: 'http://APM-SVC-NAME.NAMESPACE.svc:APM-SVC-PORT/
   webhook'
              send_resolved: true
   ```

   For more information about the parameters, see *Oracle Communications Cloud Native Core, Operations Services Overlay Installation and Upgrade Guide*.
4. Save the file.
5. Install OSO. For more information about the installation procedure, see *Oracle Communications Cloud Native Core, Operations Services Overlay Installation and Upgrade Guide*.

Observability

Metrics

There are no metrics related to this feature.

Alerts

There are no alerts related to this feature.

KPIs

There are no new KPIs related to this feature.

4 Maintenance Procedures

This section provides details about the OSO maintenance procedures.

4.1 Postinstallation CNE Configuration

4.1.1 Changing Metrics Storage Allocation

The following procedure describes how to increase or decrease the amount of persistent storage allocated to Prometheus for metrics storage.

Prerequisites

The user must calculate the revised amount of persistent storage required by the metrics.

Procedure

1. A Prometheus resource is used to configure all Prometheus instances running in OSO. Run the following command to identify the Prometheus resource:

   ```
   kubectl get prometheus -n <namespace>
   ```

2. Run the following command to resize the Prometheus metric allocation size by setting the value of `allowVolumeExpansion` to true.

   ```
   $ kubectl -n <namespace> get sc
   $ '{"allowVolumeExpansion": true}'
   ```

3. Run the following command to scale the Prometheus pod down.

   ```
   $ kubectl -n <namespace> scale deploy oso-prom-svr --replicas 0
   ```

4. Run the following command to change the pvc size of Prometheus pods:

   ```
   $ kubectl -n <namespace> edit pvc oso-prom-svr
   ```

> **Note**
>
> You will be placed in a *vi* editor session that contains all of the configurations for the OSO Prometheus pvc. Scroll down to the line that contains the "spec.Capacity" key, then update the value to the <desired increased pv size> as configured in the above step. The file must look similar to the following example:
>
> ```
> spec.Capacity: 10Gi
> ```
>
> Type "`:wq`" to exit the editor session and save the changes.

5. Run the following command to verify that the pvc size change was applied:

   ```
   $ kubectl get pv | grep oso-prom-svr
   ```

   > **Note**
   >
   > Wait until the new desired size gets reflected "10Gi".

6. Once both the pv sizes are updated to the new desired size, run the following command to scale up the Prometheus pods:

   ```
   $ kubectl -n <namespace> scale deploy oso-prom-svr --replicas 1
   ```

   > **Note**
   >
   > You will be placed in a *vi* editor session that contains all of the configurations for the OSO Prometheus instances. Scroll down to the line that contains the "replicas" key, then change the value back to 2. This scale backs up both the pods. The file must look similar to the following example:

7. Run the following command to verify that the Prometheus pods are up and running:

   ```
   kubectl get pods -n <namespace> | grep oso-prometheus
   ```

4.2 Managing 5G NFs

This section describes procedures to manage 5G NFs in CNE OSO.

4.2.1 Updating Alert Rules for an NF

This section describes the procedure to add or update the alerting rules for any Cloud Native Core (CNC) 5G Network Functions (NF) in OSO Prometheus GUI.

Prerequisites

- All NFs are required to create a separate `Alert-rules`.
- For OSO Prometheus: A valid OSO release must be installed and an alert file describing all NF alert rules according to old format is required.

Add or Update Alert Rules

Perform the following steps to add alert rules in OSO Prometheus GUI:

1. Take the backup of current configuration map of OSO Prometheus.

   ```
   $ kubectl get configmaps <OSO-prometheus-configmap-name> -o yaml -n
   <namespace> /tmp/tempPrometheusConfig.yaml
   ```

   Where,

- <OSO-prometheus-configmap-name> is the name of the OSO Prometheus configuration map.
- <namespace> is the OSO namespace.

2. Check and add the NF Alert file name inside the Prometheus configuration map. <nf-alertsname> varies from NF to NF, and can be retrieved from each individual NF alert rules file.

   For example, in the following screenshot, "alertscndbtier" is the nf-alertsname for cnDBTier.

Figure 4-1 OSO Alert file

```
[cloud-user@occne2-cgbu-cne-dbtier-bastion-1 Scripts]$ cat dbalert.yaml
apiVersion: v1
data:
  alertscndbtier: |
    groups:
    - name: DB_MONITOR_SERVICE_ALERTS
      rules:
      - alert: NODE_DOWN
        expr: db_tier_node_status == 0
```

After retrieving the nf-alertsname run the following steps:

```
$ sed -i '/etc\/config\/<nf-alertsname>/d' /tmp/tempPrometheusConfig.yaml
$ sed -i '/rule_files:/a\    \- /etc/config/<nf-alertsname>' /tmp/
tempPrometheusConfig.yaml
```

3. Update the configuration map with the updated file.

```
$ kubectl -n <namespace> replace configmap <OSO-prometheus-configmap-name>
-f
    /tmp/tempPrometheusConfig.yaml
```

4. Patch the NF alert rules in OSO Prometheus configuration map by mentioning the Alert-rule file path.

```
$ kubectl patch configmap <OSO-prometheus-configmap-name> -n <namespace> --
type merge --patch "$(cat ./NF_altertrules.yaml)"
```

5
Prometheus Vertical Scaling

This section describes the procedure for vertical scaling of Prometheus.

To scale Prometheus deployments, follow these steps:

1. Get the list of deployments and identify the OSO Prometheus deployment with the suffix `prom-svr`:

   ```
   # To list all the deployments in the OSO namespace
   $ kubectl -n <OSO_namespace> get deployments
   # To filter the deployment name by its suffix
   $ kubectl -n <OSO_namespace> get deployments | grep prom-svr
   ```

2. Edit the OSO deployment using the following command:

   > **Note**
   >
   > This will open a *vi* editor with the deployment's yaml definition.

   ```
   $ kubectl -n <OSO_namespace> edit deployment <oso_deployment_name>-prom-svr
   ```

3. Find the `resources` section for the `prom-svr` container in the edit mode of deployment, and edit the amount of resources as per the requirements.

   ```
   name: prom-svr
   ports:
   ... # ports definitions
   readinessProbe:
   ... # readiness probe definition
   resources:
     limits:
       cpu: "2"
       memory: 4Gi
     requests:
       cpu: "2"
       memory: 4Gi
   ```

4. Save and quit from the editor after making the required changes in the yaml file for the CPU and memory. In case of any errors while editing, the editor opens again and error message appears at the top of the yaml file as a comment.

   > **Note**
   >
   > If any of these objects have two containers each, you will find two resources sections. For more information about how to assign resources, see https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/.