Oracle® Communications

Operations Services Overlay (OSO) Installation Guide

Release 1.6.0
F33692-01
August 2020

Oracle Communications Operations Services Overlay (OSO) Installation Guide, Release 1.6.0

F33692-01

Contents

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.

What's New in This Guide

This is the first release of the document.

1 Introduction

This document details the procedure for installing the ***Oracle Communication's Signaling and Network Function: Operations Services Overlay (OCOSO)***. The intended audiences for this document are Oracle engineers who work with customers to install an OSO service on-site at customer facilities.

The Operations Services Overlay installs and configures common operation services (Prometheus and it's components alertmanager, pushgateway) in a previously installed Kubernetes cluster.

The Operations Services Overlay is an independent deliverable, distinct from Oracle Communications Cloud Native Environment (OCCNE).

Acronyms

The following table provides information about the acronyms and the terminology used in the document.

Table 1-1 Acronyms

Acronym	Description
CSAR	Cloud Service Archive
OCCNE	Oracle Communications Cloud Native Environment
ONAP	Open Network Automation Platform
OSO	Operations Services Overlay
OSDC	Oracle Software Download Center

2
Installing OSO

The installation procedures in this document provision and configure the Oracle Communications Operations Services Overlay (OCOSO). OSO is packaged as a Cloud Service Archive (CSAR) file. The OSO CSAR is a ZIP file that contains the following components:

- All required OSO images (including open source software) as a tar file.
- All required OSO Helm charts
- Custom values.yaml file named "prometheus-custom-values.yaml" under Artifacts -> Scripts in the CSAR.

> **Note:**
>
> The name of the *prometheus-custom-values.yaml* file will change as per the below format in each release:
> "`oso-<release-version>-custom-values.yaml`"

Installing OSO using CSAR

The CSAR format is intended to be used by an Open Network Automation Platform (ONAP) compliant Orchestrator to onboard, validate, and install OSO. However in the absence of orchestrator, manual installation is possible using the CSAR file contents.

Follow the procedure to install OSO using CSAR artifacts:

1. Download OSO CSAR zip file from Oracle Software Download Center (OSDC).
2. Unpack the CSAR zip file:

> **Note:**
>
> In case "unzip" is not available on the linux system where the CSAR zip is stored, download and installed the "unzip" using the command: "*yum install unzip"*.

```
unzip csar_oso_1_6_0.zip
```

3. Load **Artifacts** → **Images** in the repository.
4. Go to **Artifacts** → **Scripts** directory and update the *prometheus-custom-values.yaml* file with the required values as mentioned in OSO Configuration Parameter.

5. Go to **Artifacts** → **Scripts** directory where the helm charts tgz file is available and untar it.

   ```
   cd Artifacts/Scripts
   tar -xvzf oso-1.6.0-charts.tgz
   ```

6. Install OSO using helm charts provided and updated *prometheus-custom-values.yaml* file with below command:

   a. Installation using helm2:

      ```
      helm install -f <prometheus-custom-values.yaml> --
      namespace=<deployment-namespace-name> --name=<deployment-name> ./
      prometheus
      ```

   b. Installation using helm3 if custom labels are given. Skip the flag (--disable-openapi-validation) if custom labels are not provided:

      ```
      helm3 install -f <prometheus-custom-values.yaml> --
      namespace=<deployment-namespace-name> --name=<deployment-name> ./
      prometheus --disable-openapi-validation
      ```

   Example:

   ```
   helm install -f prometheus-custom-values.yaml --namespace=ocoso --
   name=ocoso ./prometheus
   ```

3
Guidelines for OSO Installation

Container Naming Convention and limitations

OSO supports having custom prefix and suffix to be configured and used in the container names. The name field in the chart currently supports up to 63 characters only, including the custom "prefix", "suffix" and default "fixed name". Currently the "fixed-name" configured by default for containers without prefix and suffix is as mentioned in Container Naming Convention.

> **Note:**
>
> Users are advised to limit the number of characters up to ***63*** in the prefix and suffix configuration accordingly to avoid name truncation.

Table 3-1 Container Naming Convention

POD NAME	FIXED CONTAINER NAME
Prometheus-server	• prometheus-server-configmap-reload • prometheus
Alertmanager	• prometheus-alertmanager • prometheus-alertmanager-configmap-reload
Pushgateway	• prometheus-pushgateway

These are the values user need to populate in order to provide suffix and prefix in their container names. These values can be found inside *prometheus-custom-values.yaml* file.

```
global:
# prefix & suffix that will be added to k8Resources
  k8Resource:
    container:
      prefix:
      suffix:
```

Label naming convention guide and limitations

OSO provides three types of label definitions as below to be configured in *prometheus-custom-values.yaml* file.

1. **Global Labels(allResources)**: These labels are attached to all K8s resources except those which are service or deployment/statefulsets. Depicted with TYPE1 in the provided sample *prometheus-custom-values.yaml* file for reference.
2. **LB & NON-LB TYPE label**: These labels are attached to load-balancer & non-load balancer types deployments, statefulsets & services. Depicted with TYPE2 in the provided sample *prometheus-custom-values.yaml* file for reference.

3. **Service specific label**: These labels are attached with each service in their specific service-label metadata sections and can be used to uniquely label services regardless of them being load-balancer/non-loadbalancer. Depicted with TYPE3 in the provided sample *prometheus-custom-values.yaml* file for reference.

Note:

1. While installing OSO with HELM2, custom/unique labels can be assigned to services.
2. While installing OSO with HELM3, custom/unique labels can be assigned, if you skip the helm3's Kubernetes OpenAPI Validation that it attempts to perform before installation. While installation with helm3 and specifying custom labels, refer to Installing OSO using CSAR. Users need to be well-versed with these notions and use the labels accordingly. Go through the official helm3 documentation for more information on this. These are the values user need to populate in order to provide custom labels for Kubernates resources. These values can be found inside *prometheus-custom-values.yaml* file.

```
 customExtension:
# TYPE1 Label
    allResources:
      labels: {}
# TYPE2 Labels
    lbServices:
      labels: {}

    nonlbServices:
      labels: {}

    lbDeployments:
      labels: {}

    nonlbDeployments:
      labels: {}

    lbStatefulSets:
      labels: {}
..

 alertmanager:
# TYPE-3 Labels: specific to alertmanager service are added
here.
  service:
    customExtension:
      labels: {}
# TYPE-3 Labels: specific to alertmanager deployement are
added here.
  deployment:
    customExtension:
      labels: {}
```

> **Note:**
>
> Similar to alertmanager service specific labels, prometheus and pushgateway will have their own TYPE3 labels.

3. Users would need to create and configure a Service Account along with needed Role and RoleBindings for the intended namespaces. These service accounts be configured in the *prometheus-custom-values.yaml* file in below parameters to be used by prometheus.

```
serviceAccountNamePromSvr: ""
serviceAccountNameAlertMgr: ""
serviceAccountNamePushGw: ""
```

4
OSO Configuration Parameter

Overview

This section provides the configuration parameters details that are used to install the OSO (Operations Service Overlay) CSAR packages manually.

Prerequisites

1. Docker is installed
2. Helm Repository is set up and have access to it.

Images

The 1.6.0 OSO CSAR is packaged with some open source images as below:

- prometheus
- pushgateway
- alertmanager
- configmap-reload
- busybox

Table 4-1 OSO Configuration Parameter

Parameter	Description	Default Values	M/O/C	Range	Notes
ALERTMANAGER-REPO-ADDRESS	Docker Image Repo URL of alertmanager where it is located		M		For example: Bastion:5000/docker.io/prom/alertmanager
BUSYBOX-REPO-ADDRESS	Docker Image Repo URL of busybox where it is located		M		For example: Bastion:5000/docker.io/busybox
CONFIGMAP-RELOAD-ADDRESS	Docker Image Repo URL of configmap-reload where it is located		M		For example: Bastion:5000/docker.io/jimmidyson/configmap-reload
PROMETHEUS-REPO-ADDRESS	Docker Image Repo URL of prometheus where it is located		M		For example: Bastion:5000/docker.io/prom/prometheus
PROMETHEUS-PUSHGATEWAY-REPO-ADDRESS	Docker Image Repo URL of prometheus-pushgateway where it is located		M		For example: Bastion:5000/docker.io/prom/pushgateway

Table 4-1 (Cont.) OSO Configuration Parameter

Parameter	Description	Default Values	M/O/C	Range	Notes
PREFIX	Configure this value with specific prefix, if there is any. otherwise the example can be used.	oso	M		For example: PREFIX-prometheus-alerts = oso-prometheus-alerts
OSO_CORTEX_URL	The URL where Prometheus will write metrics data.		C		For example: http://172.16.5.20:9001/api/prom/push
OSO_REMOTE_WRITE_TIMEOUT	The maximum amount of time that Prometheus will wait for a response to the remote write request.	30s	C		For example: Default is 30s
PV_SIZE	The size for persistence volume should be configured here.	8Gi	M		For example: 8Gi
STORAGE_SIZE	This size is calculated based upon PV_SIZE. i.e (PV_SIZE*0.9)	7.2GB	M		STORAGE_SIZE = PV_SIZE*0.9 For example: If PV = 8Gi, STORAGE_SIZE = 8Gi *0.9 that is 7.2GB
serviceAccountNamePromSvr serviceAccountNameAlertMgr serviceAccountNamePushGw	The ServiceAccount to be used by Prometheus/AlertManager/PushGateway resources.	""	M		Users need to create ServiceAccount with required namespace access defined along with the Role/RoleBinding for each of the services(Prometheus/Alertmanager/PushGateway) to be used. If no external ServiceAccount is configured or empty string (""), a default ServcieAccount with ClusterRole/ClusterRoleBinding will be created for use by Prometheus/Alertmanager/PushGateway.

Table 4-1 (Cont.) OSO Configuration Parameter

Parameter	Description	Default Values	M/O/C	Range	Notes
`namespaces`	If using external Service Account with namespace specific Role/RoleBinding Access granted as above, those namespaces need to be configured in Prometheus for scraping the given allowed namespaces.		C		A commented sample is provided in the *prometheus-custom-values.yaml* file. Uncomment below lines and replace ns1, ns2 with namespaces you wish to configure for scraping in each of the Prometheus jobs. *- job_name: 'oracle-cnc-pod'* *kubernetes_sd_configs:* *- role: pod* *#namespaces:* *# names:* *# - ns1* *# - ns2* Can add more in the list as per need.

`prefix`
`suffix`
	Prefix and suffix that will be added to Kubernates Resources.		C		**NOTE**: Only containers have been configured with prefix and suffix as of now. For example:
					`k8Resource:`
					`container:`
					`prefix: Verizon1`
					`suffix: Verizon2`

Table 4-1 (Cont.) OSO Configuration Parameter

Parameter	Description	Default Values	M/O/C	Range	Notes

`allResources:`
`labels: {}`
	Global Labels(allResources) - These labels will be attached to ALL Kubernates resources. Depicted with TYPE1 in the provided sample *prometheus-custom-values.yaml* file for reference.		C		For example:
					
					`allResources`
					`:`
					
					`labels:`
					
					`env :`
					`production`

Table 4-1 (Cont.) OSO Configuration Parameter

Parameter	Description	Default Values	M/O/C	Range	Notes
lbServices: labels: {} nonlbServices: labels: {} lbDeployments: labels: {} nonlbDeployments: labels: {} lbStatefulSets: labels: {}	**LB & NON-LB TYPE label** - These labels will be attached to load-balancer & non-load balancer types deployments, statefulsets and services. Depicted with TYPE2 in the provided sample *prometheus-custom-values.yaml* file for reference.		C		For example: lbServices: labels: key1 : value1 nonlbServices: labels: key2 : value2 lbDeployments: labels: key3 : value3 nonlbDeployments: labels: key4 : value4 lbStatefulSets: labels: key5 : value5

Table 4-1 (Cont.) OSO Configuration Parameter

Parameter	Description	Default Values	M/O/C	Range	Notes

`alertmanager:`
`service:`
 	
`customExtension:`
 	
`labels: {}`
 	
`deployment:`
 	
`customExtension:`
 	
`labels: {}`
	Service specific label - These labels will be attached with each service in their specific service-label metadata sections and can be used to uniquely label services regardless of them being load-balancer/non-loadbalancer. Depicted with TYPE3 in the provided sample prometheus-custom-values.yaml file for reference. These will be similarly defined for all other services as Prometheus and Pushgateway inside oso.		C		For example:
					
					`alertmanager:`
					`service:`
					
					`customExtension:`
					
					`labels:`
					`app: nginx`
					
					`deployment:`
					
					`customExtension:`
					
					`labels:`
					`name: oso`