Deploy Prometheus and Grafana on OCI Container Engine for Kubernetes

Feb 2023, Version 1.1

Desmond Muriu
EMEA Cloud Specialist Engineer - Compute

Table of contents

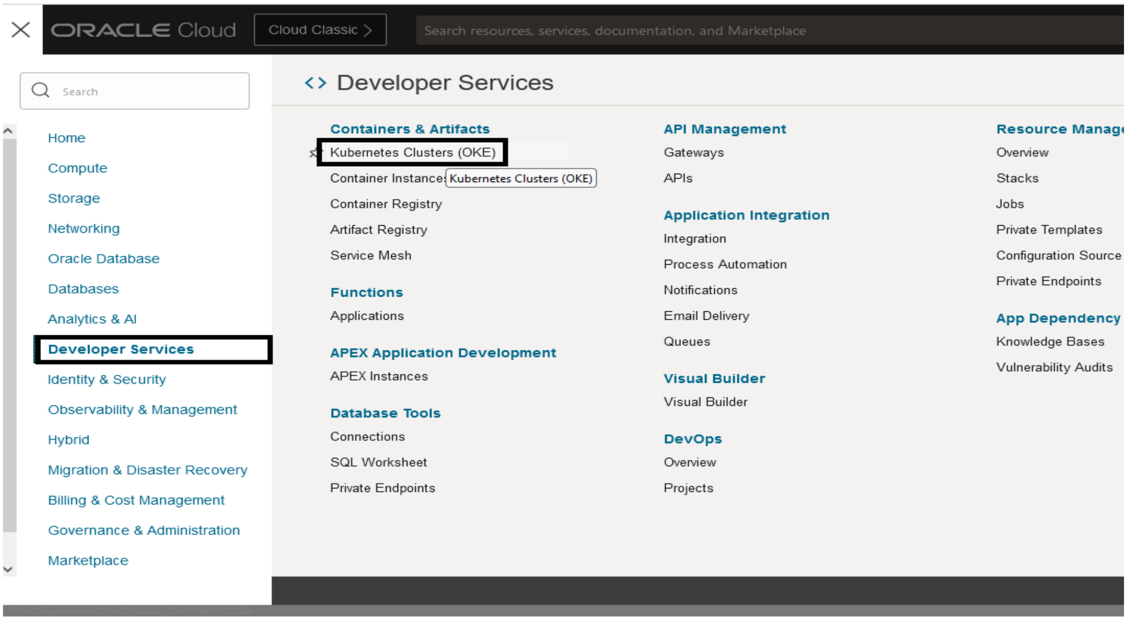
***NB:* Avoid directly doing copy/paste from this document since it could include hidden characters resulting into command lines failures.**

1. Log In to OCI

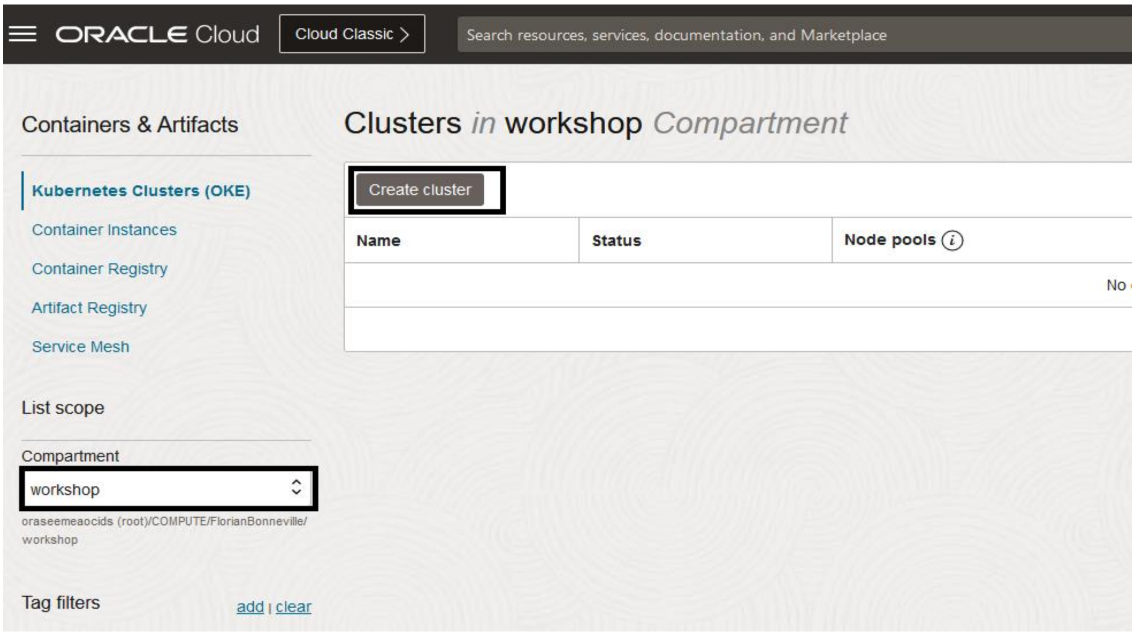
Check this blog post on the details of How to access your OCI console -> https://docs.oracle.com/en-us/iaas/Content/GSG/Tasks/signingin.htm

2. Create Cluster

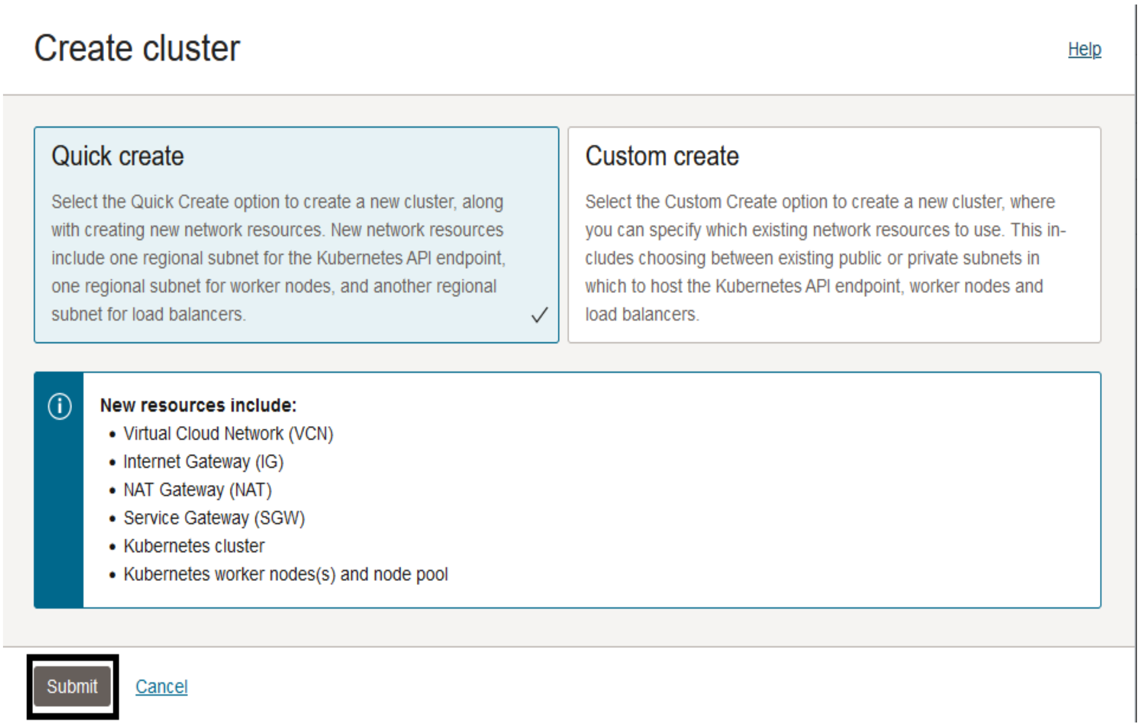
1. From the OCI Services menu (top left hamburger button), click Developer Services > Kubernetes Clusters (OKE).



2. Under List Scope, select the compartment in which you would like to create a cluster then click create Cluster



3. On the next pop up, choose Quick Create and click Submit.



Create cluster

Help

Quick create

Select the Quick Create option to create a new cluster, along with creating new network resources. New network resources include one regional subnet for the Kubernetes API endpoint, one regional subnet for worker nodes, and another regional subnet for load balancers. ✓

Custom create

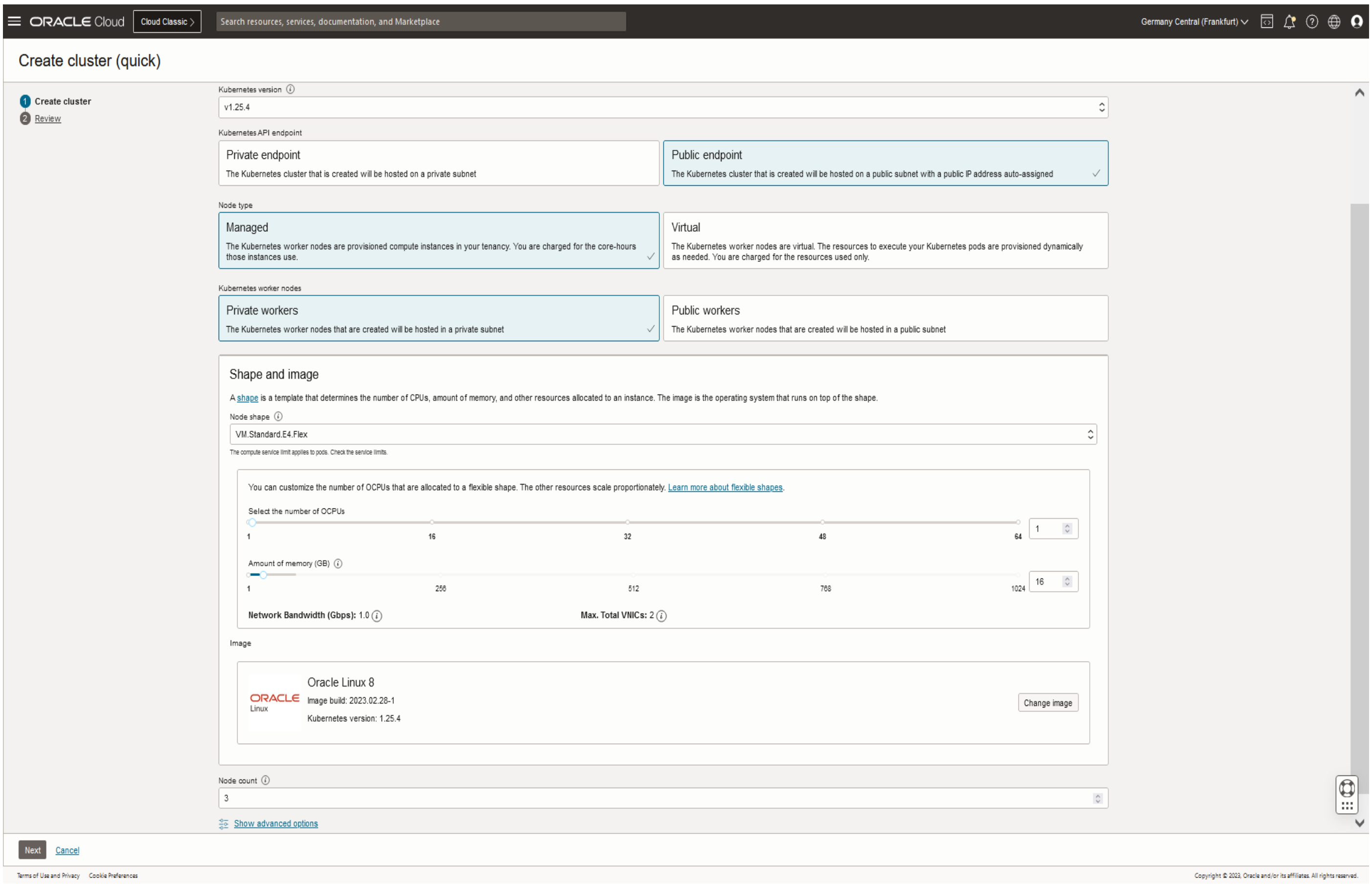
Select the Custom Create option to create a new cluster, where you can specify which existing network resources to use. This includes choosing between existing public or private subnets in which to host the Kubernetes API endpoint, worker nodes and load balancers.

New resources include:

- Virtual Cloud Network (VCN)
- Internet Gateway (IG)
- NAT Gateway (NAT)
- Service Gateway (SGW)
- Kubernetes cluster
- Kubernetes worker nodes(s) and node pool

4. Fill out the Cluster Details on the Form that pops up .i.e.

- **Name**: Provide a name
- **Compartment**: Choose your compartment
- **Kubernetes Version**: Choose the most recent version(At the time of making this tutorial, the most recent version is v1.25.4)
- **Kubernetes API Endpoint**: Public Endpoint(this will allow cluster access on OCI Cloud shell)
- **Node Type**: Managed
- **Kubernetes Worker Nodes**: Private Workers
- **Shape and Image**: Select a pod shape, Number of OCPUS, Amount of RAM and Image based on your requirement (VM.Standard.E4.Flex, 2 OCPU and 8GB Ram will be used in this example)
- **Node count**: Provide number of nodes(3 in this example)



ORACLE Cloud
Cloud Classic
Search resources, services, documentation, and Marketplace
Germany Central (Frankfurt)

Create cluster (quick)

1 Create cluster

2 Review

Kubernetes version: v1.25.4

Kubernetes API endpoint

- Private endpoint: The Kubernetes cluster that is created will be hosted on a private subnet
- Public endpoint ✓: The Kubernetes cluster that is created will be hosted on a public subnet with a public IP address auto-assigned

Node type

- Managed ✓: The Kubernetes worker nodes are provisioned compute instances in your tenancy. You are charged for the core-hours those instances use.
- Virtual: The Kubernetes worker nodes are virtual. The resources to execute your Kubernetes pods are provisioned dynamically as needed. You are charged for the resources used only.

Kubernetes worker nodes

- Private workers ✓: The Kubernetes worker nodes that are created will be hosted in a private subnet
- Public workers: The Kubernetes worker nodes that are created will be hosted in a public subnet

Shape and image

A shape is a template that determines the number of CPUs, amount of memory, and other resources allocated to an instance. The image is the operating system that runs on top of the shape.

Node shape: VM.Standard.E4.Flex

The compute service limit applies to pods. Check the service limits.

You can customize the number of OCPUs that are allocated to a flexible shape. The other resources scale proportionately. Learn more about flexible shapes.

Select the number of OCPUs: 1

Amount of memory (GB): 16

Network Bandwidth (Gbps): 1.0

Max. Total VNICs: 2

Image

Oracle Linux 8
Image build: 2023.02.28-1
Kubernetes version: 1.25.4

Node count: 3

5. Click Next. Review the cluster details and finally create cluster.

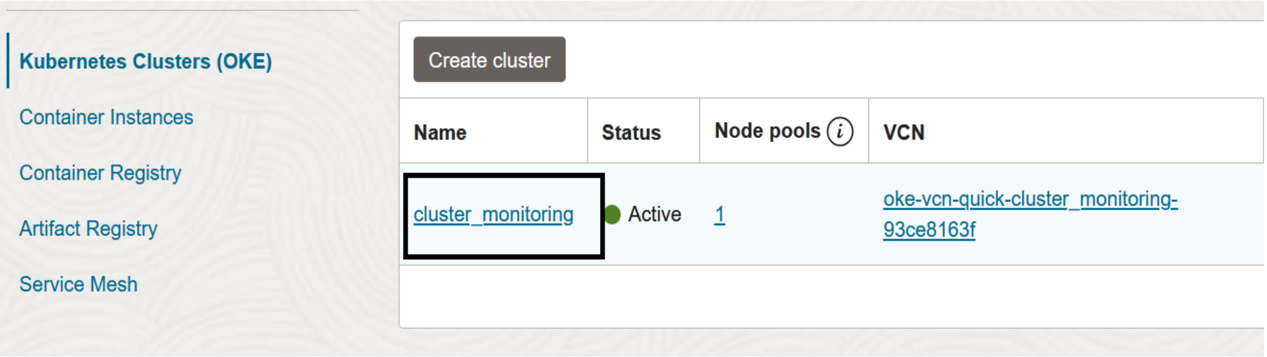
3. Accessing the Cluster

You can use the Kubernetes command line tool **kubectl** to perform operations on a cluster you've created with Container Engine for Kubernetes. You can use the kubectl installation included in **OCI Cloud Shell**, or you can use a local installation of kubectl. In both cases, before you can use kubectl to access a cluster, you have to specify the cluster on which to perform operations by setting up the cluster's kubeconfig file.

In this tutorial OCI cloud shell will be used to interact with the OKE cluster.

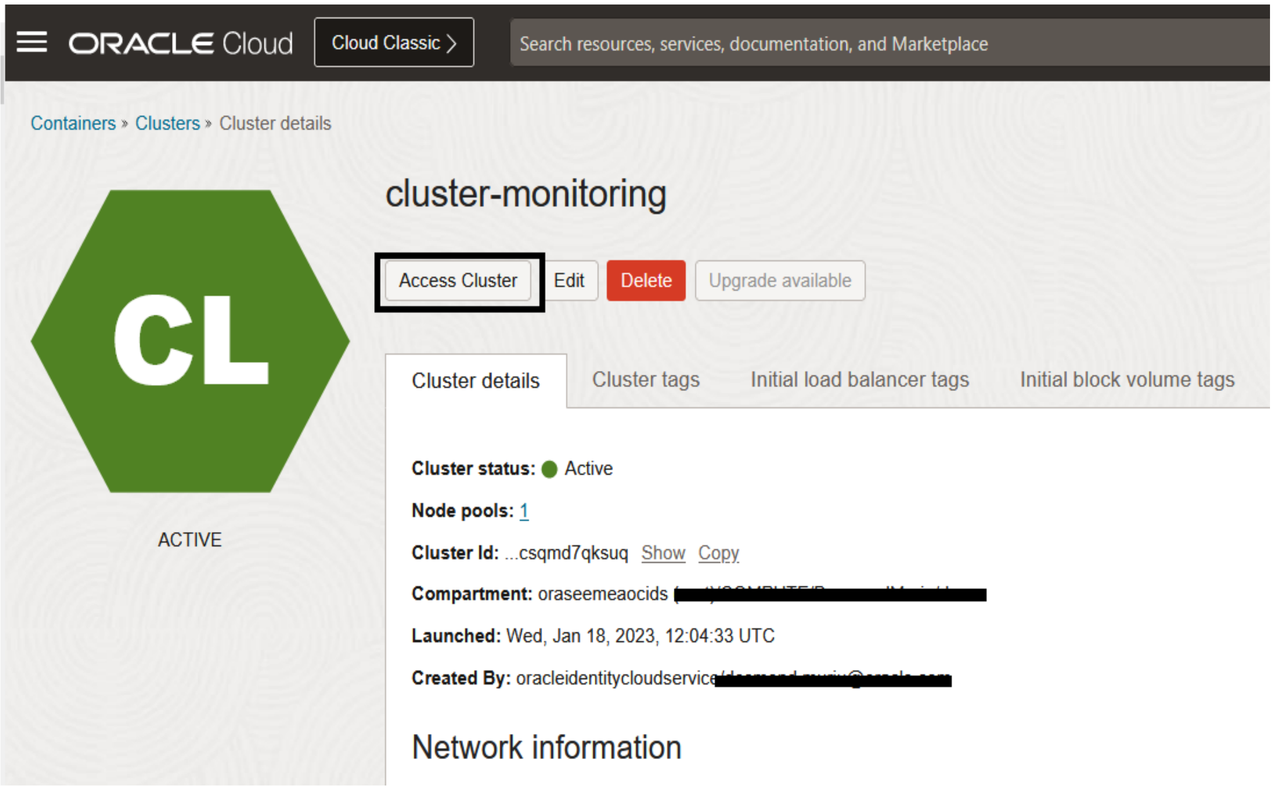
To access the cluster from the cloud shell follow the below.

1. Select the deployed cluster from the Clusters list page.



Name	Status	Node pools	VCN
cluster_monitoring	● Active	1	oke-vcn-quick-cluster_monitoring-93ce8163f

2. Click on the **Access Cluster** Link


ORACLE Cloud
Cloud Classic >
Search resources, services, documentation, and Marketplace
Containers » Clusters » Cluster details
CL
ACTIVE
Access Cluster
Edit
Delete
Upgrade available
Cluster details
Cluster tags
Initial load balancer tags
Initial block volume tags

cluster-monitoring

Cluster status: ● Active

Node pools: 1

Cluster Id: ...csqmd7qksuq Show Copy

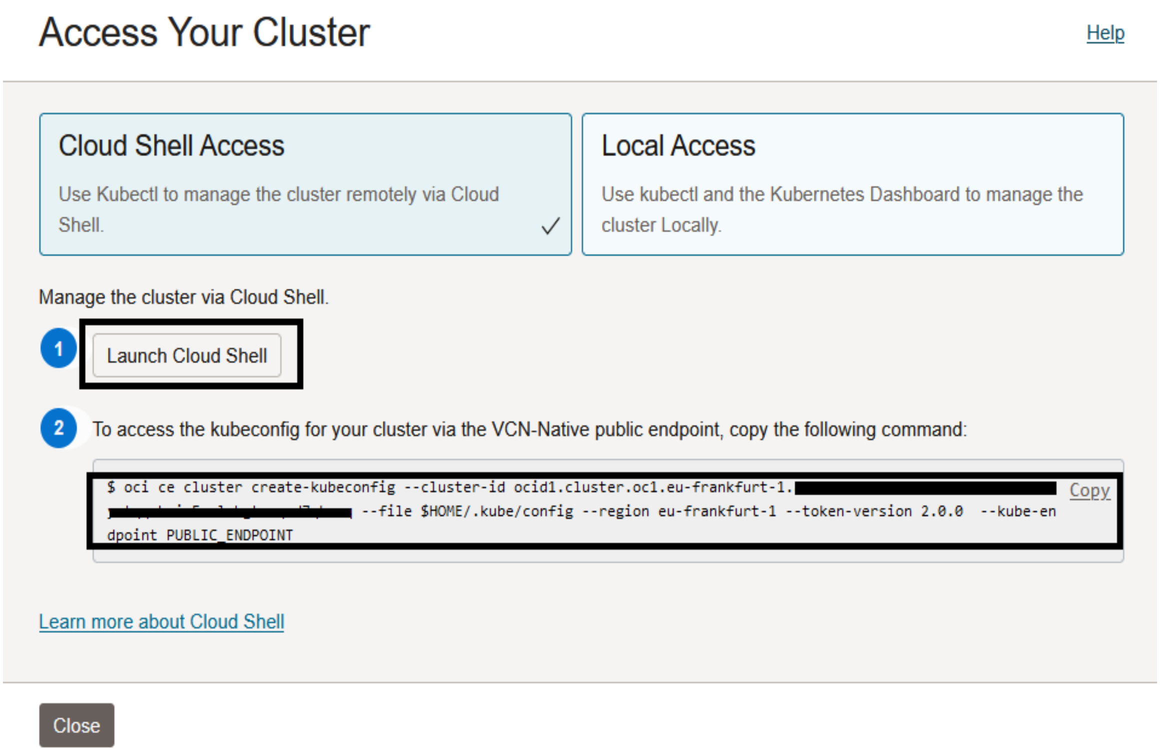
Compartment: oraseemeaocids [redacted]

Launched: Wed, Jan 18, 2023, 12:04:33 UTC

Created By: oracleidentitycloudservice[redacted]

Network information

3. Launch **Cloud shell** from the top right of the OCI console and copy the command to access the cluster.



Access Your Cluster

Cloud Shell Access

Use Kubectl to manage the cluster remotely via Cloud Shell. ✓

Local Access

Use kubectl and the Kubernetes Dashboard to manage the cluster Locally.

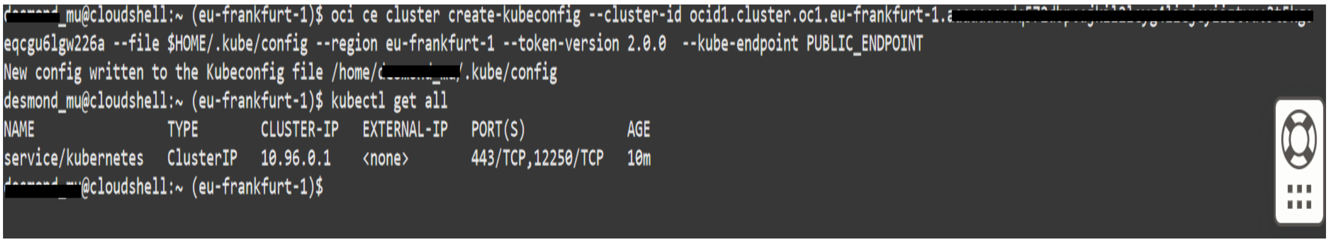
Manage the cluster via Cloud Shell.

1 Launch Cloud Shell

2 To access the kubeconfig for your cluster via the VCN-Native public endpoint, copy the following command:

```
$ oci ce cluster create-kubeconfig --cluster-id ocid1.cluster.oc1.eu-frankfurt-1.[redacted]
[redacted] --file $HOME/.kube/config --region eu-frankfurt-1 --token-version 2.0.0  --kube-en
dpoint PUBLIC_ENDPOINT
```

4. Enter a simple kubectl command to check you have access to the cluster eg **kubectl get all**

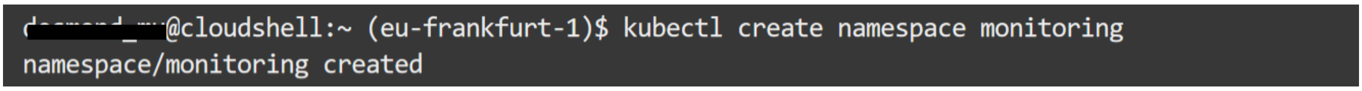


```
[redacted]_mu@cloudshell:~ (eu-frankfurt-1)$ oci ce cluster create-kubeconfig --cluster-id ocid1.cluster.oc1.eu-frankfurt-1.a[redacted]
eqcgu6lgw226a --file $HOME/.kube/config --region eu-frankfurt-1 --token-version 2.0.0  --kube-endpoint PUBLIC_ENDPOINT
New config written to the Kubeconfig file /home/[redacted].kube/config
desmond_mu@cloudshell:~ (eu-frankfurt-1)$ kubectl get all
NAME                 TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)            AGE
service/kubernetes   ClusterIP   10.96.0.1    <none>        443/TCP,12250/TCP  10m
[redacted]@cloudshell:~ (eu-frankfurt-1)$
```

4. Install Prometheus and Grafana

1. Create monitoring namespace. A namespace could be thought of as a virtual cluster. It provides a means of organizing a group of resources within single cluster. In this case, we create a monitoring namespaces where all monitoring resources can be isolated in.

   kubectl create namespace monitoring

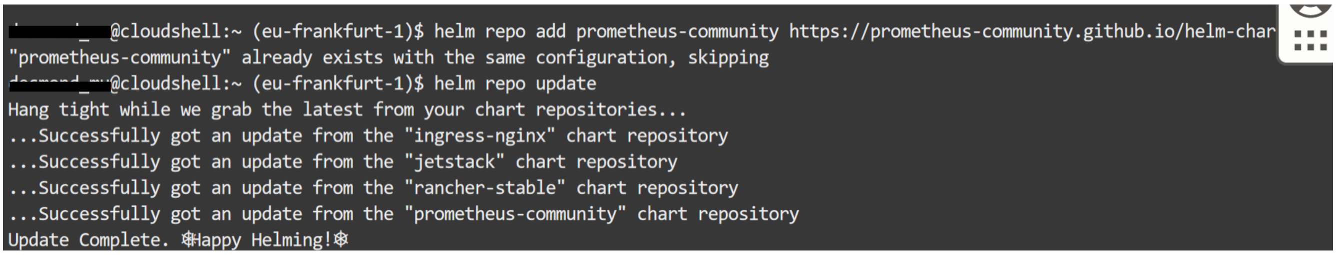


```
[redacted]@cloudshell:~ (eu-frankfurt-1)$ kubectl create namespace monitoring
namespace/monitoring created
```

2. Add helm repos

   helm repo add prometheus-community https://prometheus-community.github.io/helm-charts

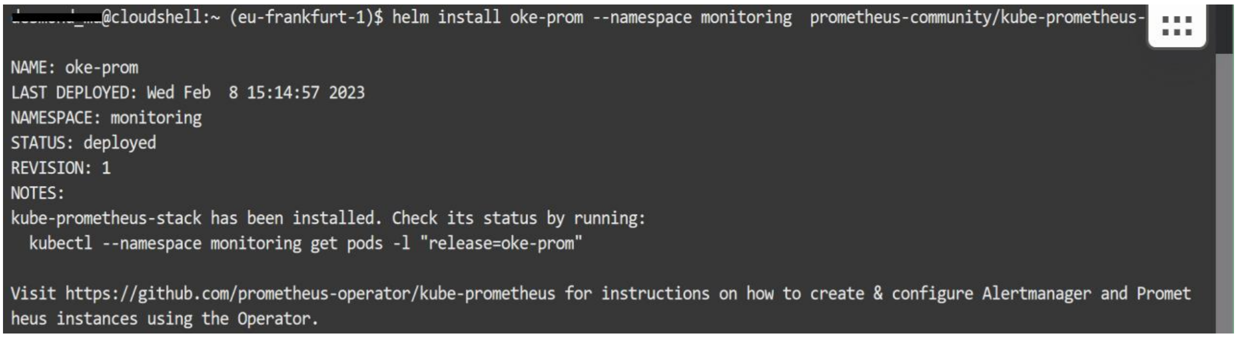
   helm repo update



```
[redacted]@cloudshell:~ (eu-frankfurt-1)$ helm repo add prometheus-community https://prometheus-community.github.io/helm-char
"prometheus-community" already exists with the same configuration, skipping
[redacted]@cloudshell:~ (eu-frankfurt-1)$ helm repo update
Hang tight while we grab the latest from your chart repositories...
...Successfully got an update from the "ingress-nginx" chart repository
...Successfully got an update from the "jetstack" chart repository
...Successfully got an update from the "rancher-stable" chart repository
...Successfully got an update from the "prometheus-community" chart repository
Update Complete. ⎈Happy Helming!⎈
```

3. Install chart in the monitoring namespace

   helm install oke-prom --namespace monitoring prometheus-community/kube-prometheus-stack



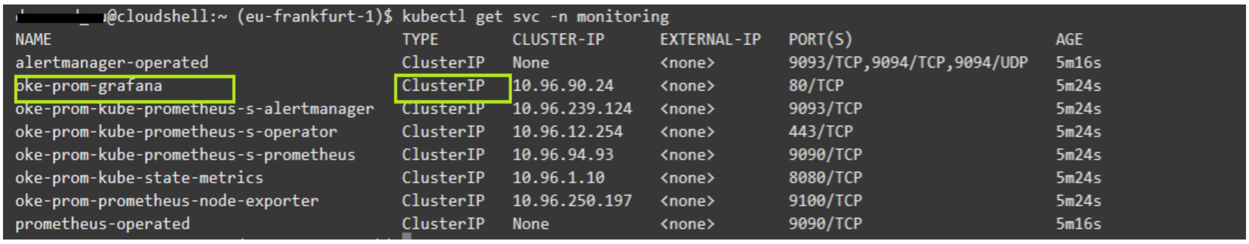
```
[redacted]@cloudshell:~ (eu-frankfurt-1)$ helm install oke-prom --namespace monitoring  prometheus-community/kube-prometheus-

NAME: oke-prom
LAST DEPLOYED: Wed Feb  8 15:14:57 2023
NAMESPACE: monitoring
STATUS: deployed
REVISION: 1
NOTES:
kube-prometheus-stack has been installed. Check its status by running:
  kubectl --namespace monitoring get pods -l "release=oke-prom"

Visit https://github.com/prometheus-operator/kube-prometheus for instructions on how to create & configure Alertmanager and Promet
heus instances using the Operator.
```

4. Expose the grafana service to allow for external access.

   To get the service name use the *kubectl get svc –n monitoring* to check the list of services. The current service is a clusterIP type meaning it cannot be accessed outside of the cluster. The next task will be to edit this service to allow for external access via an OCI native Load balancer.

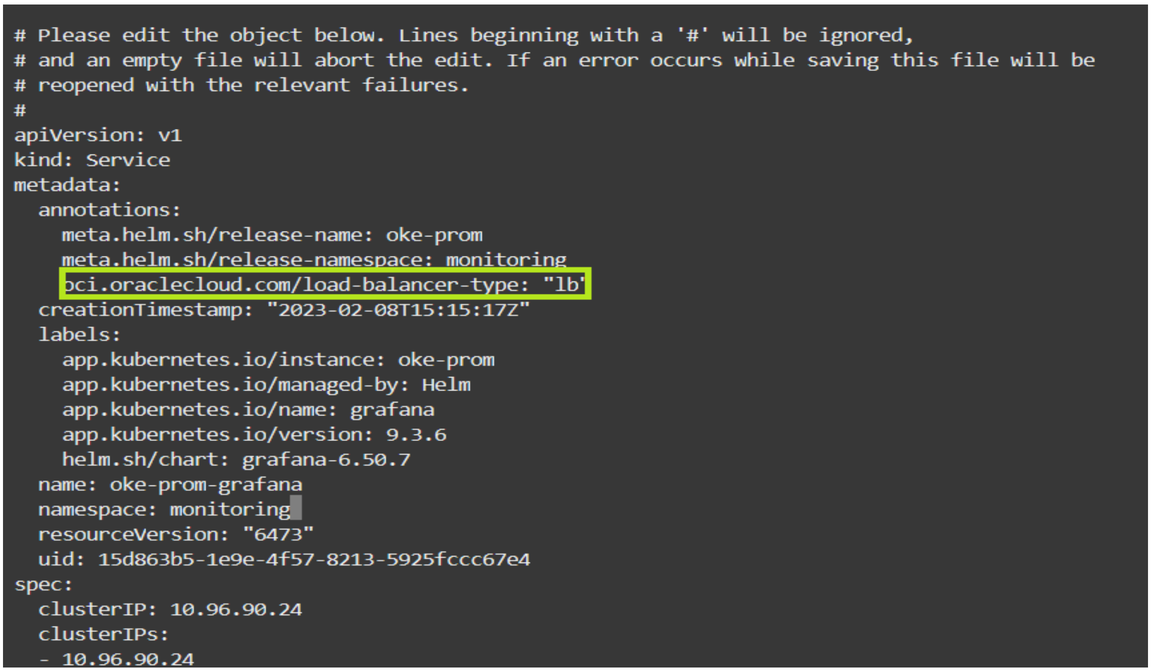


```
[redacted]@cloudshell:~ (eu-frankfurt-1)$ kubectl get svc -n monitoring
NAME                                      TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)                      AGE
alertmanager-operated                     ClusterIP   None            <none>        9093/TCP,9094/TCP,9094/UDP   5m16s
oke-prom-grafana                          ClusterIP   10.96.90.24     <none>        80/TCP                       5m24s
oke-prom-kube-prometheus-s-alertmanager   ClusterIP   10.96.239.124   <none>        9093/TCP                     5m24s
oke-prom-kube-prometheus-s-operator       ClusterIP   10.96.12.254    <none>        443/TCP                      5m24s
oke-prom-kube-prometheus-s-prometheus     ClusterIP   10.96.94.93     <none>        9090/TCP                     5m24s
oke-prom-kube-state-metrics               ClusterIP   10.96.1.10      <none>        8080/TCP                     5m24s
oke-prom-prometheus-node-exporter         ClusterIP   10.96.250.197   <none>        9100/TCP                     5m24s
prometheus-operated                       ClusterIP   None            <none>        9090/TCP                     5m16s
```

   To do this run the following command:

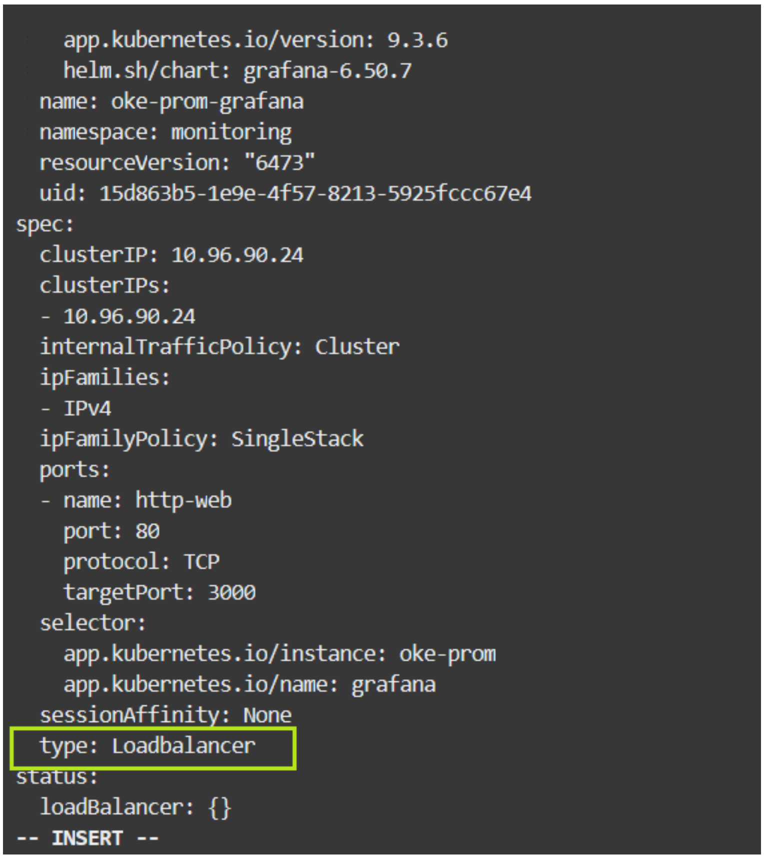
   kubectl edit svc oke-prom-grafana -n monitoring

   Under the **annotations** section add *oci.oraclecloud.com/load-balancer-type: "lb"*



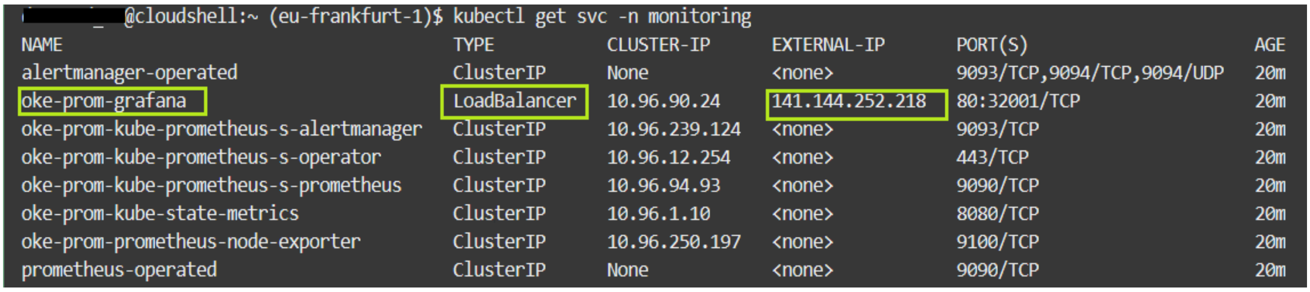
```
# Please edit the object below. Lines beginning with a '#' will be ignored,
# and an empty file will abort the edit. If an error occurs while saving this file will be
# reopened with the relevant failures.
#
apiVersion: v1
kind: Service
metadata:
  annotations:
    meta.helm.sh/release-name: oke-prom
    meta.helm.sh/release-namespace: monitoring
    oci.oraclecloud.com/load-balancer-type: "lb"
  creationTimestamp: "2023-02-08T15:15:17Z"
  labels:
    app.kubernetes.io/instance: oke-prom
    app.kubernetes.io/managed-by: Helm
    app.kubernetes.io/name: grafana
    app.kubernetes.io/version: 9.3.6
    helm.sh/chart: grafana-6.50.7
  name: oke-prom-grafana
  namespace: monitoring
  resourceVersion: "6473"
  uid: 15d863b5-1e9e-4f57-8213-5925fccc67e4
spec:
  clusterIP: 10.96.90.24
  clusterIPs:
  - 10.96.90.24
```

Also change the **type** from ClusterIp to Loadbalancer at the bottom of the config file



```
    app.kubernetes.io/version: 9.3.6
    helm.sh/chart: grafana-6.50.7
  name: oke-prom-grafana
  namespace: monitoring
  resourceVersion: "6473"
  uid: 15d863b5-1e9e-4f57-8213-5925fccc67e4
spec:
  clusterIP: 10.96.90.24
  clusterIPs:
  - 10.96.90.24
  internalTrafficPolicy: Cluster
  ipFamilies:
  - IPv4
  ipFamilyPolicy: SingleStack
  ports:
  - name: http-web
    port: 80
    protocol: TCP
    targetPort: 3000
  selector:
    app.kubernetes.io/instance: oke-prom
    app.kubernetes.io/name: grafana
  sessionAffinity: None
  type: Loadbalancer
status:
  loadBalancer: {}
-- INSERT --
```

After the update view the services again and this time note that the type will change to Loadbalancer and an external IP will also be provided. The Loadbalancer createad is a native OCI Load balancer which can also be viewed from the OCI console by going to OCI Services menu (top left hamburger button) > Networks > Loadbalancers



```
[redacted]@cloudshell:~ (eu-frankfurt-1)$ kubectl get svc -n monitoring
NAME                                     TYPE           CLUSTER-IP      EXTERNAL-IP       PORT(S)                      AGE
alertmanager-operated                    ClusterIP      None            <none>            9093/TCP,9094/TCP,9094/UDP   20m
oke-prom-grafana                         LoadBalancer   10.96.90.24     141.144.252.218   80:32001/TCP                 20m
oke-prom-kube-prometheus-s-alertmanager  ClusterIP      10.96.239.124   <none>            9093/TCP                     20m
oke-prom-kube-prometheus-s-operator      ClusterIP      10.96.12.254    <none>            443/TCP                      20m
oke-prom-kube-prometheus-s-prometheus    ClusterIP      10.96.94.93     <none>            9090/TCP                     20m
oke-prom-kube-state-metrics              ClusterIP      10.96.1.10      <none>            8080/TCP                     20m
oke-prom-prometheus-node-exporter        ClusterIP      10.96.250.197   <none>            9100/TCP                     20m
prometheus-operated                      ClusterIP      None            <none>            9090/TCP                     20m
```

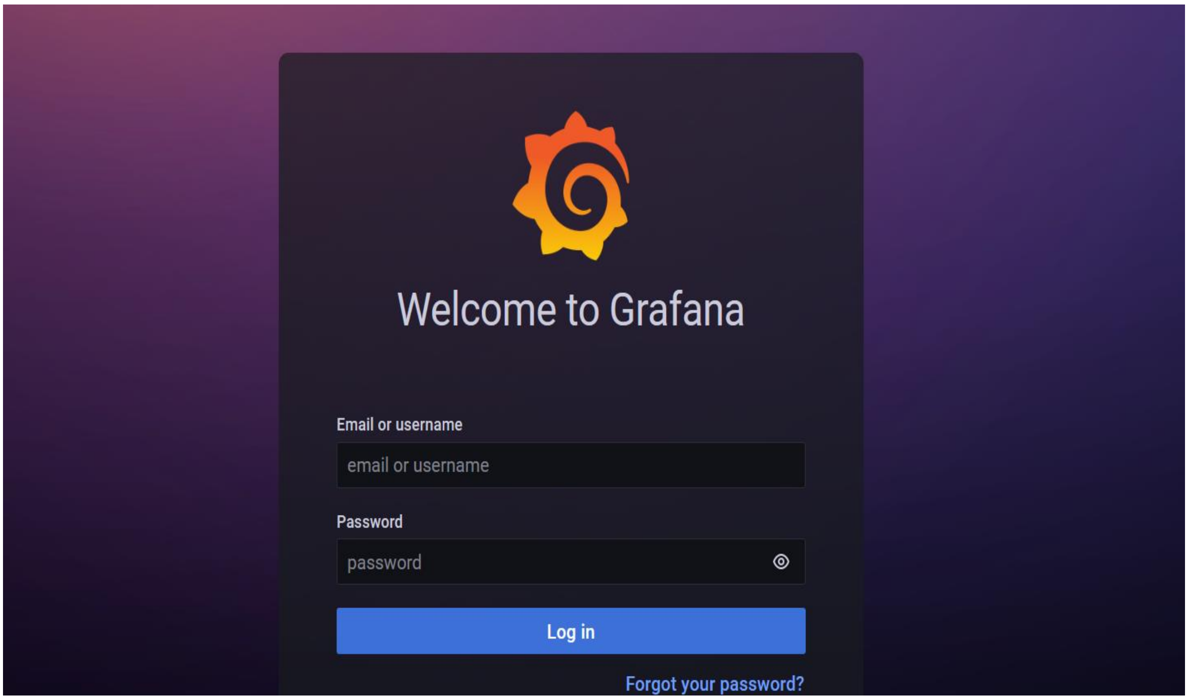
The external-ip will be the endpoint to access the Grafana UI

5. Access Grafana UI

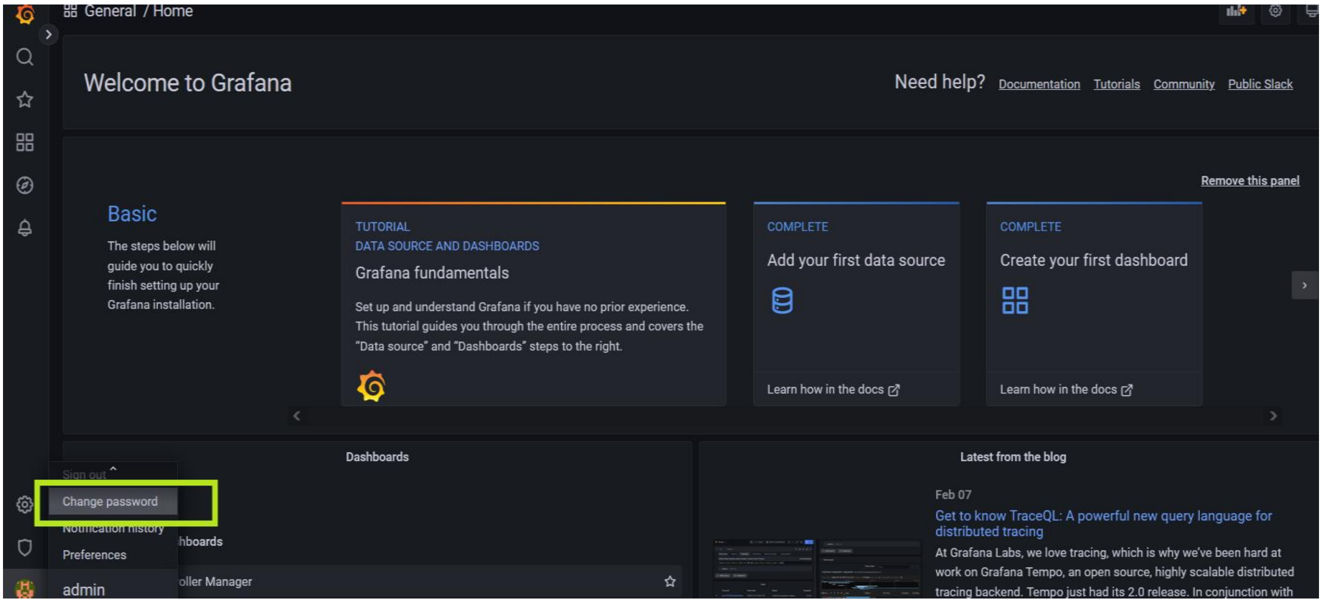
1. Enter the external IP on your browser to get access to the grafana UI.

   Default username: **admin**

   Default Password: **prom-operator(remember to change password after login)**

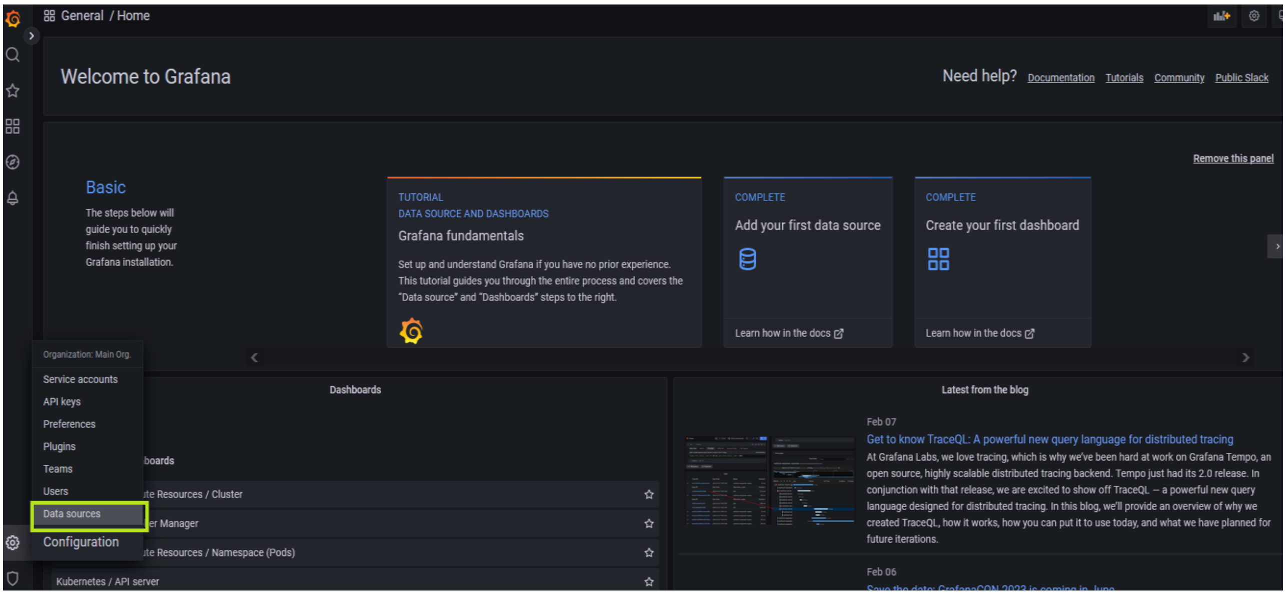


2. Changing Password. This can be done by clicking the bottom left Admin icon and selecting the **change password** option.



General / Home
Welcome to Grafana
Need help? Documentation Tutorials Community Public Slack
Remove this panel
Basic
The steps below will guide you to quickly finish setting up your Grafana installation.
TUTORIAL
DATA SOURCE AND DASHBOARDS
Grafana fundamentals
Set up and understand Grafana if you have no prior experience. This tutorial guides you through the entire process and covers the "Data source" and "Dashboards" steps to the right.
COMPLETE
Add your first data source
Learn how in the docs
COMPLETE
Create your first dashboard
Learn how in the docs
Dashboards
Latest from the blog
Sign out
Change password
Notification history
Preferences
admin
hboards
oller Manager
Feb 07
Get to know TraceQL: A powerful new query language for distributed tracing
At Grafana Labs, we love tracing, which is why we've been hard at work on Grafana Tempo, an open source, highly scalable distributed tracing backend. Tempo just had its 2.0 release. In conjunction with

3. View **Data Sources** by selecting it from the **Configuration** options on the bottom left bar.



The steps below will guide you to quickly finish setting up your Grafana installation.

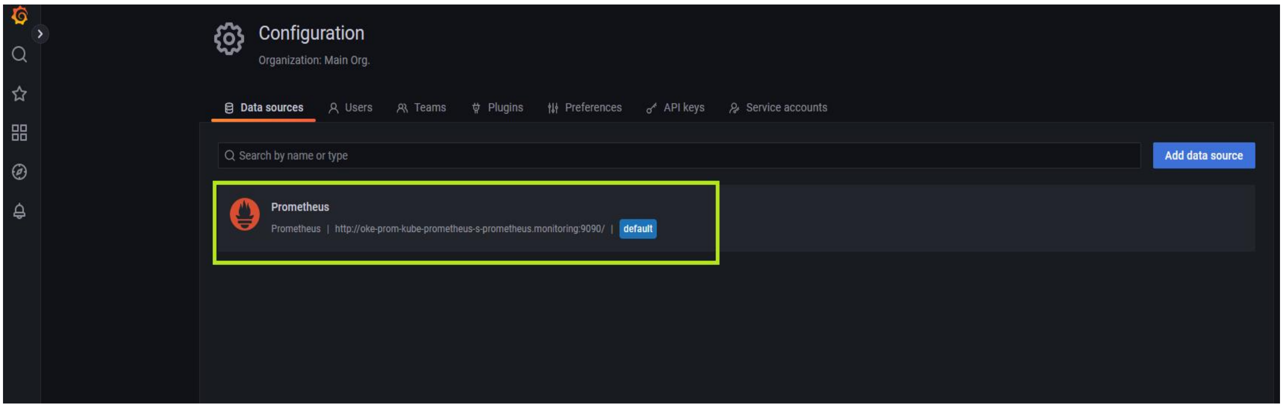
Set up and understand Grafana if you have no prior experience. This tutorial guides you through the entire process and covers the "Data source" and "Dashboards" steps to the right.

Get to know TraceQL: A powerful new query language for distributed tracing

At Grafana Labs, we love tracing, which is why we've been hard at work on Grafana Tempo, an open source, highly scalable distributed tracing backend. Tempo just had its 2.0 release. In conjunction with that release, we are excited to show off TraceQL — a powerful new query language designed for distributed tracing. In this blog, we'll provide an overview of why we created TraceQL, how it works, how you can put it to use today, and what we have planned for future iterations.

Save the date: GrafanaCON 2023 is coming in June

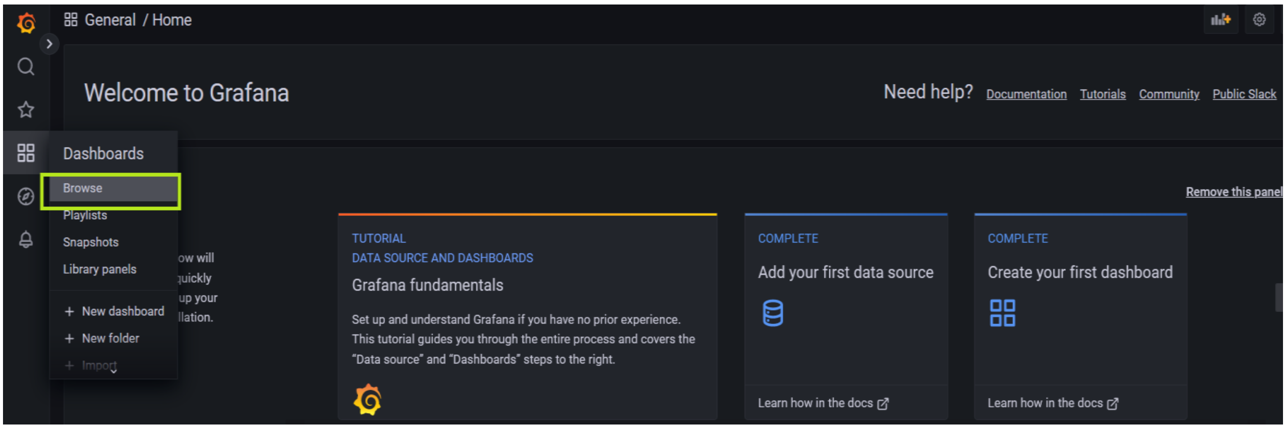
By default Prometheus is already configured.



Prometheus

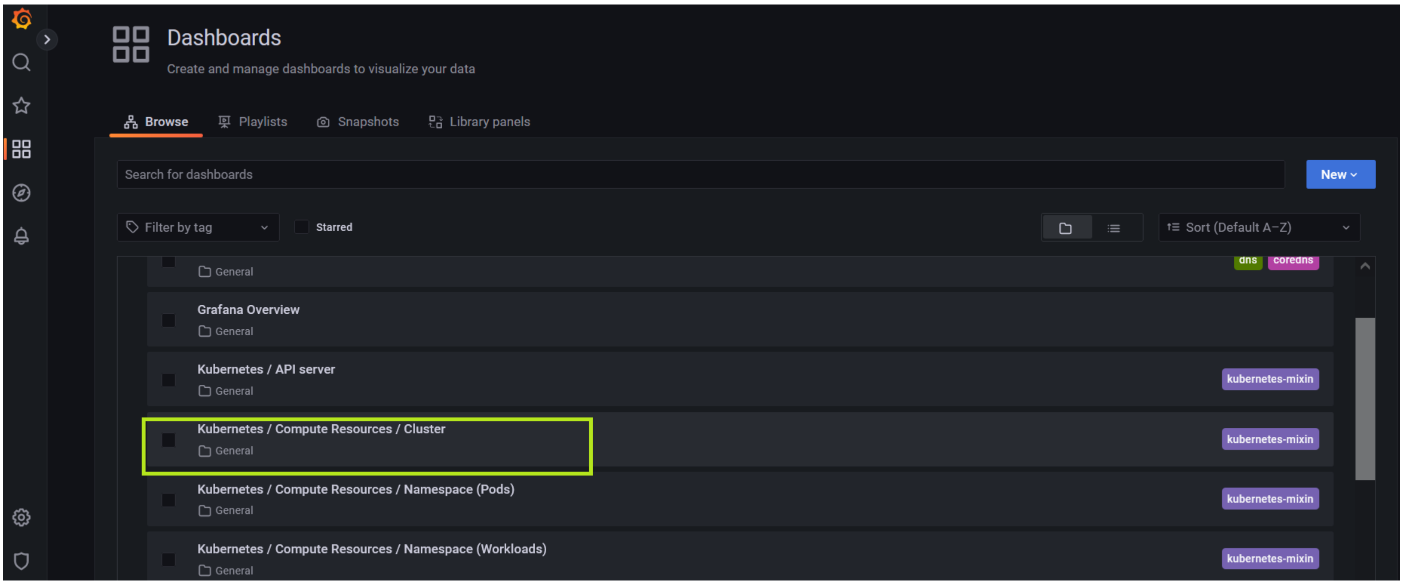
Prometheus | http://oke-prom-kube-prometheus-s-prometheus.monitoring:9090/ | default

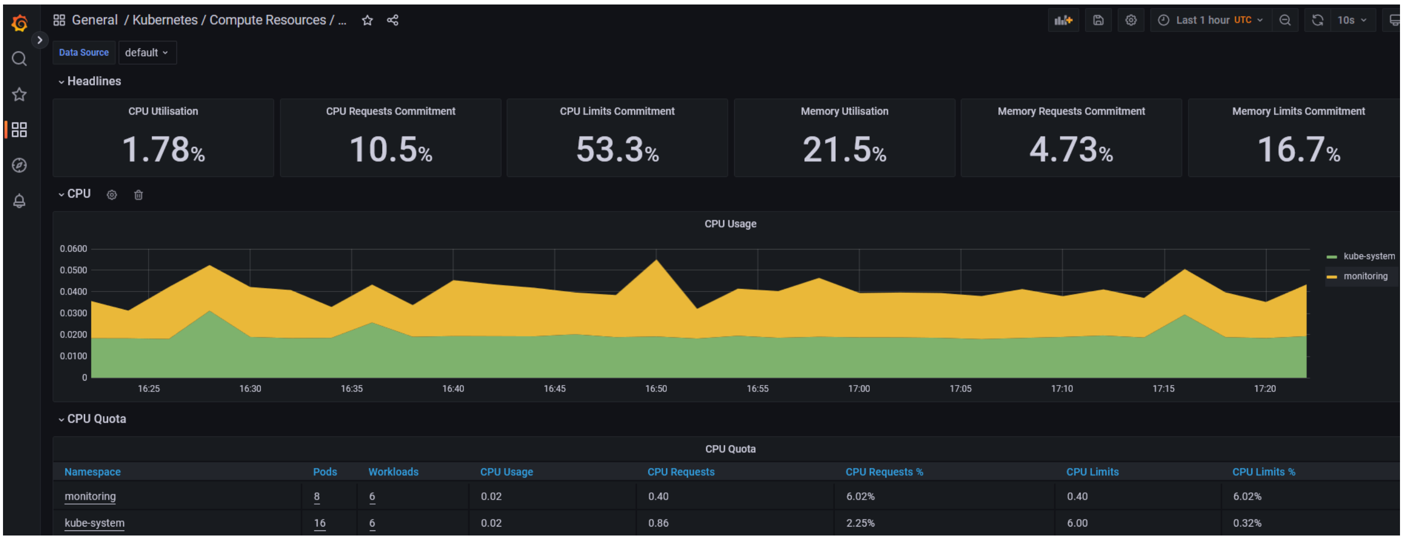
4. Accessing the default dashboards. This is achieved by clicking **Browse** from the **Dashboards** tab on the left



Set up and understand Grafana if you have no prior experience. This tutorial guides you through the entire process and covers the "Data source" and "Dashboards" steps to the right.

Eg. To view a summary of compute resources on your cluster , select **Kubernetes/Compute Reources/Cluster** option





CPU Utilisation	CPU Requests Commitment	CPU Limits Commitment	Memory Utilisation	Memory Requests Commitment	Memory Limits Commitment
1.78%	10.5%	53.3%	21.5%	4.73%	16.7%

CPU Quota

Namespace	Pods	Workloads	CPU Usage	CPU Requests	CPU Requests %	CPU Limits	CPU Limits %
monitoring	8	6	0.02	0.40	6.02%	0.40	6.02%
kube-system	16	6	0.02	0.86	2.25%	6.00	0.32%