

Lab 13. Setting up Container Clusters in OC

Objectives:

- Create a Kubernetes Clusters and access the dashboard from local system.

Pre-Requisite

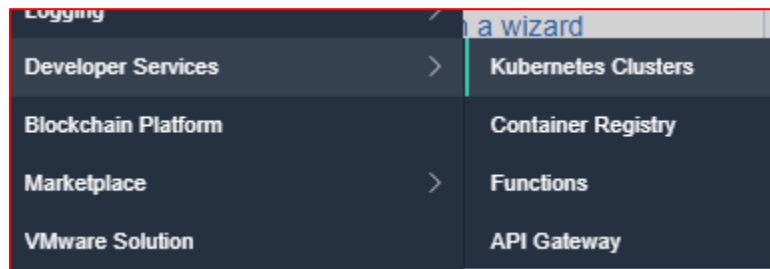
- A Trial Account on OCI

Sequence 1. Create a Kubernetes Clusters and Access the Dashboard

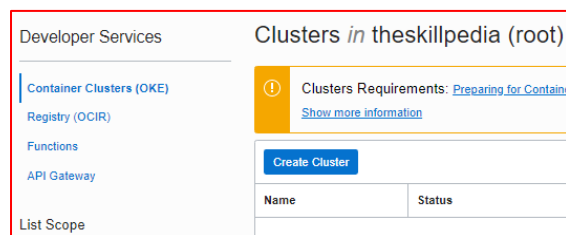
Anyone with a trial account for Oracle Cloud can use Oracle Cloud Infrastructure (OCI) to get herself a three-node Kubernetes Cluster instance, running on Oracle's managed Kubernetes Engine Cloud Service called Kubernetes Clusters (Subject to Service Limits).

Prerequisite:

1. From the menu shown on the left side of the screen select Developer Services | Kubernetes Clusters.



2. From the page that open, click on Create Cluster.



3. You will see a popup with two options. Keep default option "Quick Start" and click on Launch Workflow.

Create Cluster [Help](#)

☒ Quick Create

☐ Custom 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 worker nodes , and another regional subnet for load balancers.

New resources include:

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

[Launch Workflow](#)
[Cancel](#)
[Cancel](#)

4. Provide the details such as cluster Name and Version and Click on **Next**.

Enter Following Details :

- Name: tsp-cluser1
- Compartment: root
- Kubernetes Version: latest
- Visibility Type: Public
- Shape: VM Standard2.2
- Number of Nodes: 3

Name

Compartment

Kubernetes Version ⓘ

Choose Visibility Type

Private
The Kubernetes worker nodes that are created will be hosted in private subnet(s)

Public
The Kubernetes worker nodes that are created will be hosted in public subnet(s) ✓

Shape ⓘ

Number of nodes ⓘ

☐ Specify a custom boot volume size
[Volume performance](#) varies with volume size. Default boot volume size: 46.6 GB

[Show Advanced Options](#)

5. Click on Next and Review the Details. Click on **Create Cluster**

Create Cluster

Review

Basic Information

Cluster Name: tsp-cluster1

Compartment: 6354688-C18

Version: v1.18.10

Network

Compartment: 6354688-C18

Kubernetes CIDR Block: 10.96.0.0/16

VCN Name: oke-vcn-quick-tsp-cluster1-20210212164151

Pods CIDR Block: 10.244.0.0/16

Node Pools

Node Pool1

Node Pool Name: pool1

Version: v1.18.10

Compartment: 6354688-C18

Number of Nodes: 3

Image: Oracle-Linux-7.9-2020.11.10-1

Shape: VM.Standard.B1.1

Kubernetes Labels

Key: name

Value: pool1

Back

Create Cluster

Cancel

- You will see a Request processing popup. Click on Close once the request is complete.

Resources to be created

Basic Information

Cluster Name: tsp-cluster1

Compartment: 6354688-C18

Version: v1.18.10

Network

Compartment: 6354688-C18

VCN Name: oke-vcn-quick-tsp-cluster1-20210212164151

Node Pools

Node Pool1

Node Pool Name: pool1

Version: v1.18.10

Compartment: 6354688-C18

Number of Nodes: 3

Image: Oracle-Linux-7.9-2020.11.10-1

Shape: VM.Standard.B1.1

Kubernetes Labels

Key: name

Value: pool1

Creating cluster and associated network resources

Create Virtual Cloud Network

The Virtual Cloud Network was created: oke-vcn-quick-tsp-cluster1-6e83245d

Create Internet Gateway

The Internet Gateway was created: oke-igw-quick-tsp-cluster1-6e83245d

Create Route Tables

The Route Table was created: oke-routetable-tsp-cluster1-6e83245d

Create Security Lists

The Security List was created: oke-sec1-quick-tsp-cluster1-6e83245d

The Security List was created: oke-sec2-quick-tsp-cluster1-6e83245d

Create Subnets

Subnet was created: oke-svbsubnet-quick-tsp-cluster1-6e83245d-regional

Subnet was created: oke-svbsubnet-quick-tsp-cluster1-6e83245d-regional

Create Cluster

Requesting Cluster: tsp-cluster1

Create Node Pool

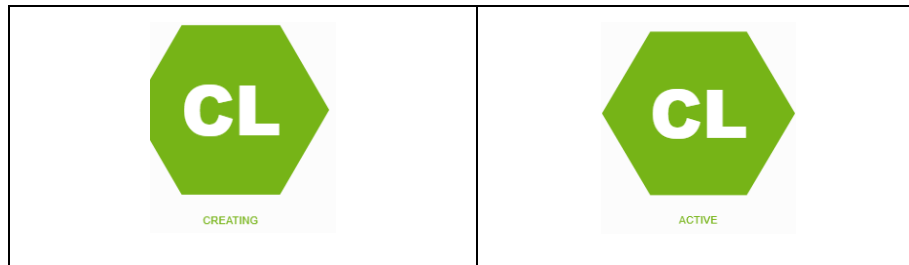
Requesting Node Pool: pool1

Cluster and associated network resources created.

Close

Cancel

- You should see the status **Creating**. After some time the cluster will be ready and status will change to **Active**.



8. To access the Cluster from your Virtual Machine,

- Create another instance **master** in your root compartment, if you are using trial account.
- Or use existing master if working as a normal user.
- Once the instance is ready, come back to OKE console.
- You can also use your VM, **server**.

9. Click on Cluster Name "tsp-cluster1" as shown below.

Developer Services

Container Clusters (OKE)

Registry (OCIR)

Functions

API Gateway

List Scope

COMPARTMENT

theskillpedia (root)

Clusters in theskillpedia (root) Compartment

Clusters Requirements: [Preparing for Container Engine for Kubernetes](#)
[Show more information](#)

Create Cluster

Name	Status	Node Pools	VCN
tsp-cluster1	Active	1 View	oke-vcn-quick-t

10. Click on Quick Start from sidebar f

Resources

Metrics

Node Pools

Work Requests

Quick Start

[Access Kubernetes Dashboard](#)

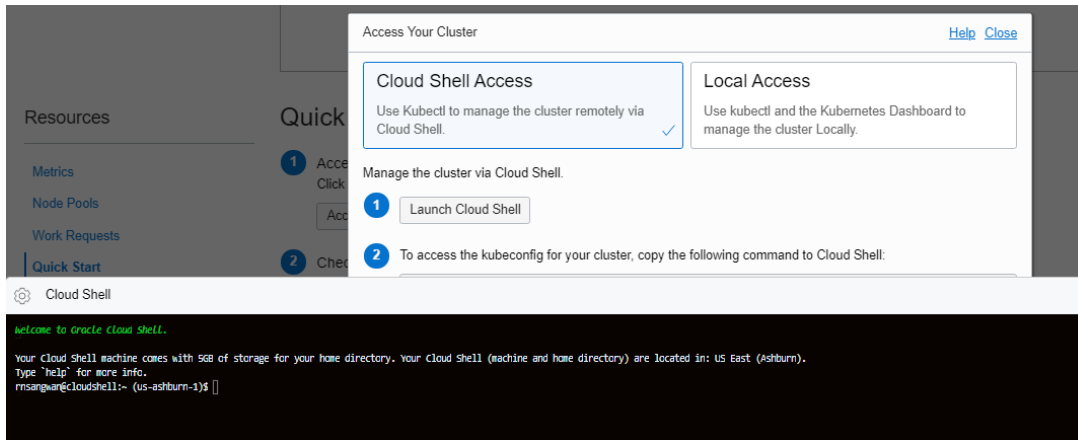
Quick Start: Deploy Sample App

- 1 Access Your Cluster**
Click the button below to learn how to setup access to your Kubernetes cluster.
[Access Cluster](#)
- 2 Check Version**
Verify that kubernetes is available by entering the following command in your terminal

```
$ kubectl version
```
- 3 Deploy Application**
Deploy a sample hello world application by running the following command in your terminal.

```
$ kubectl create -f https://k8s.io/examples/application/deployment.yaml
```
- 4 [Access Kubernetes Dashboard](#)**

11. Follow the Instruction given to Access Cluster using **Cloud Shell**.



Copy paste the command as shown in screen shot to access the cluster.

OPTION 2.

12. To get started click on **Access Cluster**. Execute the commands on the VM, **server** (Assuming that you have already completed the OCI CLI Lab).

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.

You must have [downloaded and installed](#) OCI CLI version 2.24.0 (or later) and [configured](#) it for use. If your version of the OCI CLI is earlier than version 2.24.0, download and install a newer version from [here](#). If you are not sure of the version of the OCI CLI you currently have installed, check with this command:

```
$ oci -v
```

1 Create a directory to contain the kubeconfig file.

```
$ mkdir -p $HOME/.kube
```

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.iad.aaaaaaa5wbyoy6kjbszaa643kmtj5dyiuqwk5yrpjl1satxhbc5f5awteqa --file $HOME/.kube/config --region us-ashburn-1 --token-version 2.0.0 --kube-endpoint PUBLIC_ENDPOINT
```

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

```
$ oci ce cluster create-kubeconfig --cluster-id ocid1.cluster.oc1.iad.aaaaaaa5wbyoy6kjbszaa643kmtj5dyiuqwk5yrpjl1satxhbc5f5awteqa --file $HOME/.kube/config --region us-ashburn-1 --token-version 2.0.0 --kube-endpoint PRIVATE_ENDPOINT
```

3 To set your KUBECONFIG environment variable to the file for this cluster, use:

```
$ export KUBECONFIG=$HOME/.kube/config
```

i If you wish to save your new kubeconfig to a different location, please change the --file argument in the CLI command above with the new location path. You may also have to set or update your KUBECONFIG environment variable with this new location path. To persist the environment variable, your shell initiation script may have to be updated as well.

For more information on managing kubeconfig files, please refer to the official [Kubernetes documentation](#). More information on the available commands for OCI's Container Engine for Kubernetes CLI can be found [here](#).

Sequence 1. Install and Configure OCI CLI on Linux.

The Lab outcome will be used in other labs. So please be careful and complete all steps.

1. Login to Linux Machine **server**, as root user.
2. To install OCI CLI on the compute instance, Enter Command:

```
# bash -c "$(curl -L https://raw.githubusercontent.com/oracle/oci-cli/master/scripts/install/install.sh)"
```

```
[root@server ~]# bash -c "$(curl -L https://raw.githubusercontent.com/oracle/oci-cli/master/scripts/install/install.sh)"
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           Dload  Upload   Total   Spent    Left     Speed
  0     0    0     0    0     0      0      0  --:--:-- --:--:-- --:--:--    0curl: (6) Could not resolve host: xn--l-5gn; Unknown error
100 14738 100 14738    0     0  18237    0  --:--:-- --:--:-- --:--:--  299k

*****
You have started the OCI CLI Installer in interactive mode. If you do not wish
to run this in interactive mode, please include the --accept-all-defaults option.
If you have the script locally and would like to know more about
input options for this script, then you can run:
./install.sh -h
If you would like to know more about input options for this script, refer to:
https://github.com/oracle/oci-cli/blob/master/scripts/install/README.rst
*****
Downloading Oracle Cloud Infrastructure CLI install script from https://raw.githubusercontent.com/oracle/oci-cli/v2.9.3/scripts/install/install.py to /tmp/oci_cli_install/tmp_1oH0.
##### 100.0%
Python3 not found on system PATH
Running install script.
```

*If you are using Oracle Linux, you can simply use **yum install python36-oci-cli***

3. When prompted for Install directory, Press Enter (choose default)
4. When prompted for 'oci' directory, Press Enter (choose default)
5. When prompted for 'Y/N' for \$Path, Enter Y, when prompted for path for rc file Press Enter (choose default)

```
====> Modify profile to update your $PATH and enable shell/tab completion now? (Y/n): y
====> Enter a path to an rc file to update (leave blank to use '/root/.bashrc'):
-- Backed up '/root/.bashrc' to '/root/.bashrc.backup'
-- Tab completion set up complete.
-- If tab completion is not activated, verify that '/root/.bashrc' is sourced by your shell.
--
-- ** Run `exec -l $SHELL` to restart your shell. **
--
-- Installation successful.
-- Run the CLI with /root/bin/oci --help
[root@server ~]#
```

6. Check oci CLI installed version, Enter command:

```
# oci -v
```

```
[root@server ~]# oci -v
2.12.0
[root@server ~]#
```

```
[root@server ~]# oci -v
2.12.0
[root@server ~]# mkdir -p $HOME/.kube
[root@server ~]# oci ce cluster create-kubeconfig --cluster-id ocid1.cluste
a3t --file $HOME/.kube/config --region us-ashburn-1 --token-version 2.0.0
New config written to the Kubeconfig file /root/.kube/config
[root@server ~]# export KUBECONFIG=$HOME/.kube/config
```

13. Setup OCI CLI

```
# oci setup config
```

Accept all Defaults and Provide

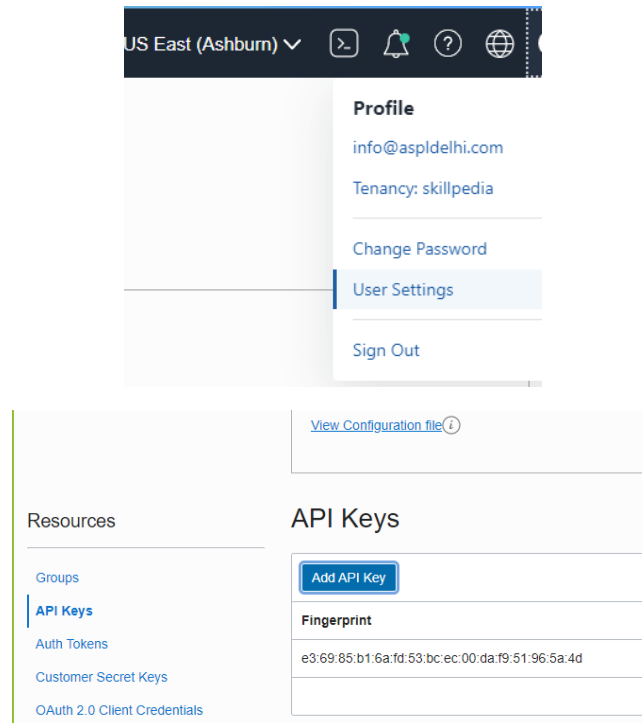
User OCID

Tenancy OCID

Select the Region name and asked for.

14. This will also generate a Public Key in PEM format located under .oci folder in your home directory. Add this key under API Key in User Settings on OCI

```
# cat .oci/oci_api_key_public.pem
```



15. Verify kubectl

- Confirm that you've already installed kubectl.
- Verify that you can use kubectl to connect to the new cluster you've created. In a terminal window, enter the following command:

```
# kubectl get nodes
```

```
[root@server ~]# kubectl get nodes
NAME             STATUS    ROLES    AGE      VERSION
10.0.10.2        Ready    node     6m44s    v1.16.8
10.0.10.3        Ready    node     6m36s    v1.16.8
10.0.10.4        Ready    node     6m10s    v1.16.8
[root@server ~]#
```

Sequence 2 Kubernetes Dashboard.

kubectl apply -f

<https://raw.githubusercontent.com/kubernetes/dashboard/v2.0.3/aio/deploy/recommended.yaml>

```
[root@server ~]# kubectl apply -f https://raw.githubusercontent.com/kubernetes/dashboard/v2.0.3/aio/deploy/recommended.yaml
namespace/kubernetes-dashboard created
serviceaccount/kubernetes-dashboard created
service/kubernetes-dashboard created
secret/kubernetes-dashboard-certs created
secret/kubernetes-dashboard-csrf created
secret/kubernetes-dashboard-key-holder created
configmap/kubernetes-dashboard-settings created
role.rbac.authorization.k8s.io/kubernetes-dashboard created
clusterrole.rbac.authorization.k8s.io/kubernetes-dashboard created
rolebinding.rbac.authorization.k8s.io/kubernetes-dashboard created
clusterrolebinding.rbac.authorization.k8s.io/kubernetes-dashboard created
deployment.apps/kubernetes-dashboard created
service/dashboard-metrics-scraper created
deployment.apps/dashboard-metrics-scraper created
```

1. In a text editor, create a file called **oke-admin-service-account.yaml** with following content:

vi **oke-admin-service-account.yaml**

apiVersion: v1

kind: ServiceAccount

metadata:

name: oke-admin

namespace: kube-system

apiVersion: rbac.authorization.k8s.io/v1beta1

kind: ClusterRoleBinding

metadata:

name: oke-admin

roleRef:

apiGroup: rbac.authorization.k8s.io

kind: ClusterRole

name: cluster-admin

subjects:

- kind: ServiceAccount

name: oke-admin

namespace: kube-system

The file defines an administrator service account and a clusterrolebinding, both called oke-admin.

2. Create the service account and the clusterrolebinding in the cluster by entering:

clusterrolebinding.rbac.authorization.k8s.io "oke-admin" created

You can now use the `oke-admin` service account to view and control the cluster, and to connect to the Kubernetes dashboard.

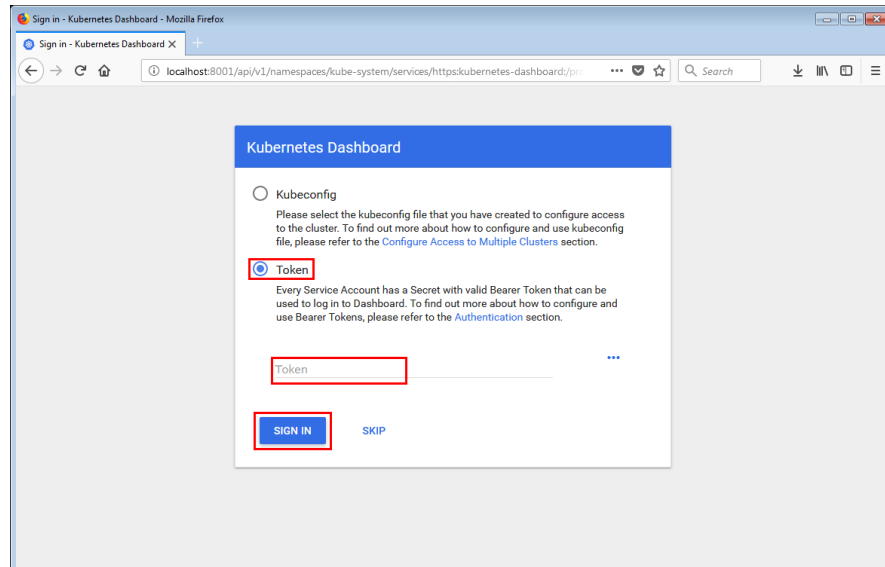
- ```
$ kubectl -n kube-system describe secret $(kubectl -n kube-system get secret | grep oke-admin | awk '{print $1}')
```

**token:** eyJh\_\_\_\_\_px1Q

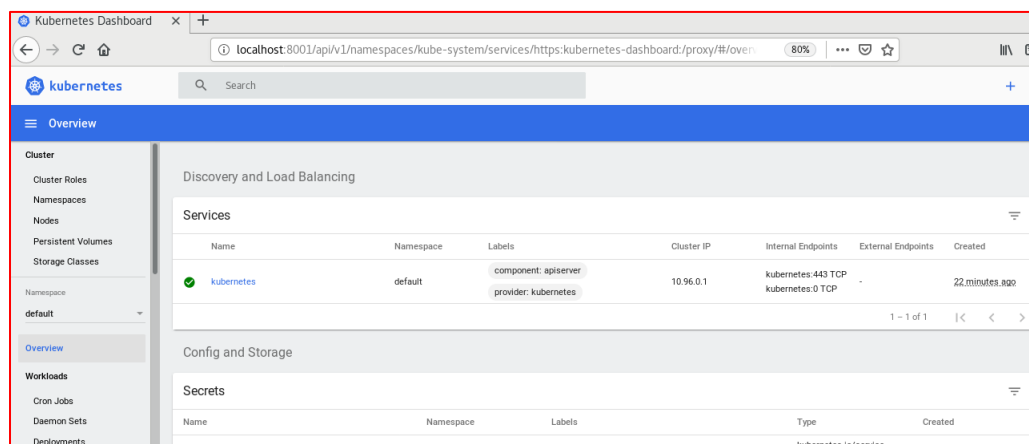
- ```
$ kubectl proxy .
```

- ```
http://localhost:8001/api/v1/namespaces/kubernetes-dashboard/services/https:kubernetes-
```

7. It will display the Kubernetes Login Page like this



8. In the Kubernetes Dashboard as shown above, select **Token** and paste the value of the **token:** element you copied earlier into the **Token** field.
9. In the Kubernetes Dashboard, click **Sign In**, and then click **Overview** to see the applications deployed on the cluster.



## Sequence 3 Deploy an Application

1. Deploy nginx Web Server on your Cluster by executing  
# kubectl create deployment nginx --image=nginx
2. Verify the Deployment  
# kubectl get deployments

```
[root@server ~]# kubectl create deployment nginx --image=nginx
deployment.apps/nginx created
[root@server ~]# kubectl get deployments
NAME READY UP-TO-DATE AVAILABLE AGE
nginx 1/1 1 1 72s
[root@server ~]#
```

3. If you'd like to see more detail about your deployment, run the describe command.

# kubectl describe deployment nginx

```
[root@server ~]# kubectl describe deployment nginx
Name: nginx
Namespace: default
CreationTimestamp: Sat, 15 Aug 2020 02:09:32 -0400
Labels: app=nginx
Annotations: deployment.kubernetes.io/revision: 1
Selector: app=nginx
Replicas: 1 desired | 1 updated | 1 total | 1 available | 0 unavailable
StrategyType: RollingUpdate
MinReadySeconds: 0
RollingUpdateStrategy: 25% max unavailable, 25% max surge
Pod Template:
 Labels: app=nginx
 Containers:
 nginx:
 Image: nginx
```

4. Kubernetes offers several options when exposing your service based on a feature called Kubernetes Service-types:

- **ClusterIP** – To exposes the service on an internal IP, reachable only within the cluster.
- **NodePort** – To expose your service to be accessible outside of your cluster, on a specific port (called the NodePort) on every node in the cluster.
- **LoadBalancer** – Leverages on external Load-Balancing services offered by various providers to allow access to your service.

Create a Service Type NodePort for Nginx

# kubectl create service nodeport nginx --tcp=80:80

```
[root@server ~]# kubectl create service nodeport nginx --tcp=80:80
service/nginx created
[root@server ~]#
```

5. Run the **get svc** command to see a summary of the service and the ports exposed.

# kubectl get svc

```
[root@server ~]# kubectl get svc
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
kubernetes ClusterIP 10.96.0.1 <none> 443/TCP 81m
nginx NodePort 10.96.198.225 <none> 80:31054/TCP 51s
```

Take note of the Port corresponding to port 80

6. Visit Compute -> Instances page and take note of public IP of any of your node

Compute

Instances

Dedicated Virtual Machine Hosts

Instance Configurations

Instance Pools

Cluster Networks

Autoscaling Configurations

### Instances *in theskillpedia (root) Compartment*

The [Compute service](#) helps you provision VMs and bare metal instances to meet your compute and application needs. The image that you use to launch an instance determines its operating system and other software.

Create Instance

| Name                                                      | Status  | Public IP      | Shape          |
|-----------------------------------------------------------|---------|----------------|----------------|
| <a href="#">oke-c4tgyrsonqt-ngtim3dggzd-sfknm2rmhmq-0</a> | Running | 132.145.186.42 | VM.Standard1.1 |
| <a href="#">oke-c4tgyrsonqt-ngtim3dggzd-sfknm2rmhmq-0</a> |         |                |                |

7. Now Open the Browser and Access the Website with Public IP and Port Noted above

Welcome to nginx! x +

132.145.186.42:31054

## Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to [nginx.org](#).  
Commercial support is available at [nginx.com](#).

*Thank you for using nginx.*

8. You have successfully deployed an application on your OKE cluster