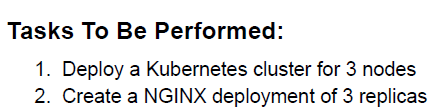
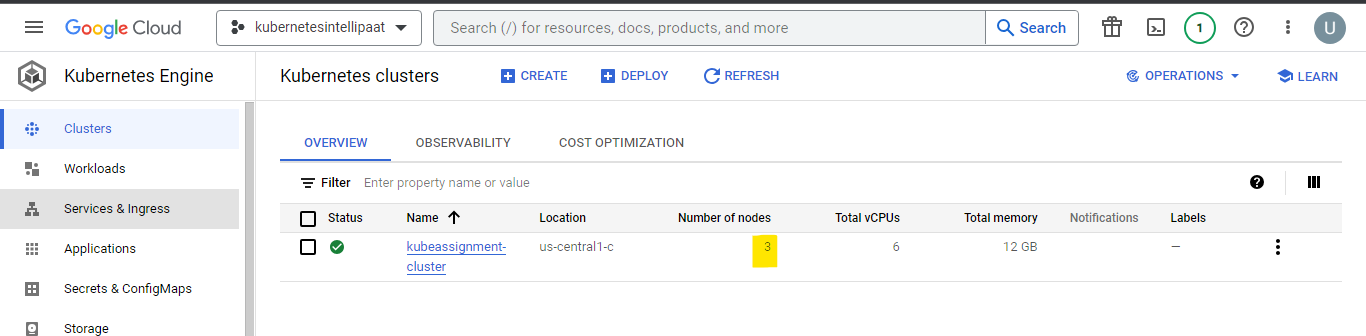
**Kubernetes Assignment – 1**

****

1. Create a Kubernetes Cluster of 3 Nodes. (I am using GCP since deploying in AWS failed every time and I do not have a domain to try with kops. Got in touch with the intellipaat support team, they approved it stating we can perform the same in GCP)



1. Connect to the Cluster (Master) via Cloud Shell
   1. gcloud container clusters get-credentials cluster\_name --zone zone --project project\_id
2. Create a deployment file to deploy nginx with 3 replicas. PFB yaml file for reference.

ukuduwa@cloudshell:~ (kubernetesintellipaat-400507)$ sudo nano deployment.yaml

ukuduwa@cloudshell:~ (kubernetesintellipaat-400507)$ cat deployment.yaml

apiVersion: apps/v1

kind: Deployment

metadata:

name: nginx-deployment

spec:

selector:

matchLabels:

app: nginx

replicas: 3

template:

metadata:

labels:

app: nginx

spec:

containers:

- name: nginx

image: nginx:1.14.2

ports:

- containerPort: 80

1. Create the deployment by using kubectl create command.

ukuduwa@cloudshell:~ (kubernetesintellipaat-400507)$ kubectl create -f deployment.yaml

deployment.apps/nginx-deployment created

1. Check if the deployment is running on all the 3 Nodes by using below commands. AS observed, each replica is running on each node.

ukuduwa@cloudshell:~ (kubernetesintellipaat-400507)$ kubectl get pods -o wide

NAME READY STATUS RESTARTS AGE IP NODE NOMINATED NODE READINESS GATES

nginx-deployment-cbdccf466-m5nqm 1/1 Running 0 3m20s 10.96.0.7 gke-kubeassignment-cluster-kube-nodes-f2e23eda-lr1j <none> <none>

nginx-deployment-cbdccf466-n4l8z 1/1 Running 0 3m20s 10.96.1.6 gke-kubeassignment-cluster-kube-nodes-f2e23eda-p25s <none> <none>

nginx-deployment-cbdccf466-wt5vc 1/1 Running 0 3m20s 10.96.2.14 gke-kubeassignment-cluster-kube-nodes-f2e23eda-08rc <none> <none>

ukuduwa@cloudshell:~ (kubernetesintellipaat-400507)$ kubectl get nodes

NAME STATUS ROLES AGE VERSION

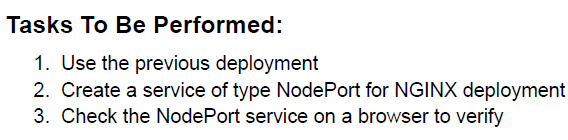
gke-kubeassignment-cluster-kube-nodes-f2e23eda-08rc Ready <none> 4m9s v1.27.3-gke.100

gke-kubeassignment-cluster-kube-nodes-f2e23eda-lr1j Ready <none> 4m8s v1.27.3-gke.100

gke-kubeassignment-cluster-kube-nodes-f2e23eda-p25s Ready <none> 4m9s v1.27.3-gke.100

ukuduwa@cloudshell:~ (kubernetesintellipaat-400507)$

**Kubernetes Assignment – 2**



1. To create a service of the nginx deployment. Use the below command :

ukuduwa@cloudshell:~ (kubernetesintellipaat-400507)$ kubectl create service nodeport nginx --tcp=80:80

service/nginx created

ukuduwa@cloudshell:~ (kubernetesintellipaat-400507)$ kubectl get svc

NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE

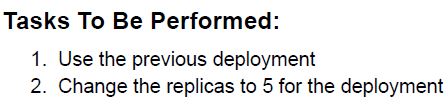
kubernetes ClusterIP 10.100.0.1 <none> 443/TCP 35m

nginx NodePort 10.100.1.254 <none> 80:32661/TCP 5s

ukuduwa@cloudshell:~ (kubernetesintellipaat-400507)$

1. I was unable to verify it in browser since in GKE (Google Kubernetes Engine) we don’t have control over Security Group. I am suspecting no traffic is allowed, but the above command did open up port 32661 and the service nginx. You have to use public IP of master followed by the opened port to get the desired result.

**Kubernetes Assignment – 3**

****

1. To change the replicas to 5, edit the deployment.yaml file and change the value of replicas to 5.

ukuduwa@cloudshell:~ (kubernetesintellipaat-400507)$ sudo nano deployment.yaml

ukuduwa@cloudshell:~ (kubernetesintellipaat-400507)$ cat deployment.yaml

apiVersion: apps/v1

kind: Deployment

metadata:

name: nginx-deployment

spec:

selector:

matchLabels:

app: nginx

replicas: 5

template:

metadata:

labels:

app: nginx

spec:

containers:

- name: nginx

image: nginx:1.14.2

ports:

- containerPort: 80

1. After editing the replicas to 5, you need to apply the configuration.

ukuduwa@cloudshell:~ (kubernetesintellipaat-400507)$ kubectl apply -f deployment.yaml

Warning: resource deployments/nginx-deployment is missing the kubectl.kubernetes.io/last-applied-configuration annotation which is required by kubectl apply. kubectl apply should only be used on resources created declaratively by either kubectl create --save-config or kubectl apply. The missing annotation will be patched automatically.

deployment.apps/nginx-deployment configured

1. Once applied, verify it by checking the pods.

ukuduwa@cloudshell:~ (kubernetesintellipaat-400507)$ kubectl get pods -o wide

NAME READY STATUS RESTARTS AGE IP NODE NOMINATED NODE READINESS GATES

nginx-deployment-cbdccf466-7xwrl 1/1 Running 0 7s 10.96.1.7 gke-kubeassignment-cluster-kube-nodes-f2e23eda-p25s <none> <none>

nginx-deployment-cbdccf466-m5nqm 1/1 Running 0 37m 10.96.0.7 gke-kubeassignment-cluster-kube-nodes-f2e23eda-lr1j <none> <none>

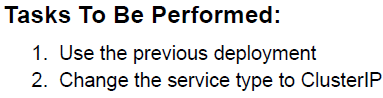
nginx-deployment-cbdccf466-n4l8z 1/1 Running 0 37m 10.96.1.6 gke-kubeassignment-cluster-kube-nodes-f2e23eda-p25s <none> <none>

nginx-deployment-cbdccf466-ppgdk 1/1 Running 0 7s 10.96.0.8 gke-kubeassignment-cluster-kube-nodes-f2e23eda-lr1j <none> <none>

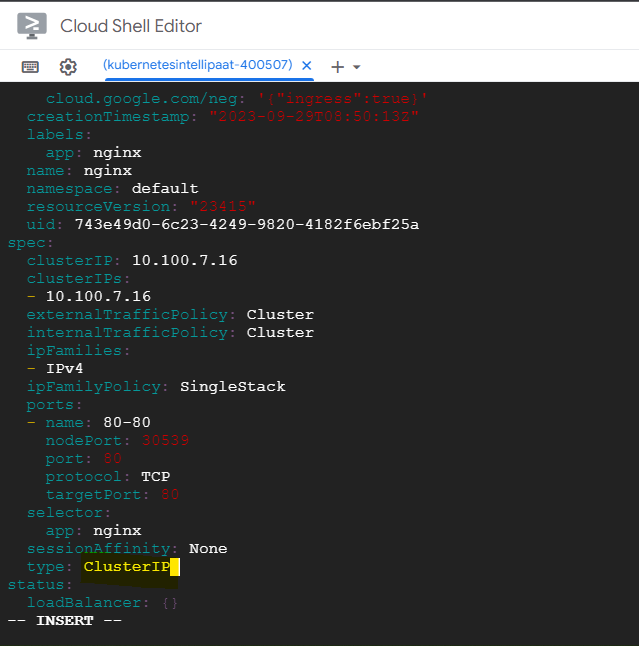
nginx-deployment-cbdccf466-wt5vc 1/1 Running 0 37m 10.96.2.14 gke-kubeassignment-cluster-kube-nodes-f2e23eda-08rc <none> <none>

ukuduwa@cloudshell:~ (kubernetesintellipaat-400507)$

**Kubernetes Assignment – 4**

****

1. You need to edit the type in the Service to change the Service from NodePort to ClusterIP by using command kubectl edit service service\_name.



1. Once edited, save and exit. Check the Service Status by using kubectl get svc command, you can see the changes.

ukuduwa@cloudshell:~ (kubernetesintellipaat-400507)$ kubectl get svc

NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE

kubernetes ClusterIP 10.100.0.1 <none> 443/TCP 48m

nginx NodePort 10.100.7.16 <none> 80:30539/TCP 5s

ukuduwa@cloudshell:~ (kubernetesintellipaat-400507)$ kubectl edit service nginx

service/nginx edited

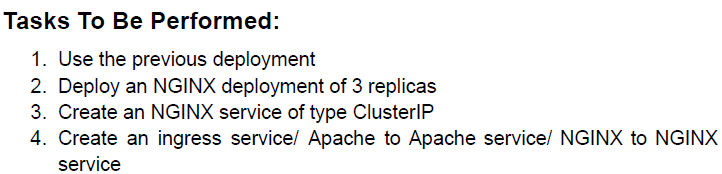
ukuduwa@cloudshell:~ (kubernetesintellipaat-400507)$ kubectl get svc

NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE

kubernetes ClusterIP 10.100.0.1 <none> 443/TCP 52m

nginx ClusterIP 10.100.7.16 <none> 80/TCP 4m35s

**Kubernetes Assignment – 5**

****

1. Edit the replicas back to 3 from 5, and apply the changes.

ukuduwa@cloudshell:~ (kubernetesintellipaat-400507)$ kubectl apply -f deployment.yaml

deployment.apps/nginx-deployment configured

ukuduwa@cloudshell:~ (kubernetesintellipaat-400507)$ kubectl get pods

NAME READY STATUS RESTARTS AGE

nginx-deployment-cbdccf466-m5nqm 1/1 Running 0 53m

nginx-deployment-cbdccf466-n4l8z 1/1 Running 0 53m

nginx-deployment-cbdccf466-wt5vc 1/1 Running 0 53m

1. Create a deployment file to install httpd (Apache), and run kubectl create command and deploy it.

ukuduwa@cloudshell:~ (kubernetesintellipaat-400507)$ sudo nano deployment.yaml

ukuduwa@cloudshell:~ (kubernetesintellipaat-400507)$ cat deployment.yaml

apiVersion: apps/v1

kind: Deployment

metadata:

name: apache-deployment

spec:

selector:

matchLabels:

app: httpd

replicas: 3

template:

metadata:

labels:

app: httpd

spec:

containers:

- name: httpd

image: httpd

ports:

- containerPort: 80

ukuduwa@cloudshell:~ (kubernetesintellipaat-400507)$ kubectl create -f deployment.yaml

deployment.apps/apache-deployment created

ukuduwa@cloudshell:~ (kubernetesintellipaat-400507)$ kubectl get pods -o wide

NAME READY STATUS RESTARTS AGE IP NODE NOMINATED NODE READINESS GATES

apache-deployment-57bb565c84-ccbbk 1/1 Running 0 20s 10.96.2.15 gke-kubeassignment-cluster-kube-nodes-f2e23eda-08rc <none> <none>

apache-deployment-57bb565c84-p6tls 1/1 Running 0 20s 10.96.1.8 gke-kubeassignment-cluster-kube-nodes-f2e23eda-p25s <none> <none>

apache-deployment-57bb565c84-rkgsk 1/1 Running 0 20s 10.96.0.9 gke-kubeassignment-cluster-kube-nodes-f2e23eda-lr1j <none> <none>

nginx-deployment-cbdccf466-m5nqm 1/1 Running 0 55m 10.96.0.7 gke-kubeassignment-cluster-kube-nodes-f2e23eda-lr1j <none> <none>

nginx-deployment-cbdccf466-n4l8z 1/1 Running 0 55m 10.96.1.6 gke-kubeassignment-cluster-kube-nodes-f2e23eda-p25s <none> <none>

nginx-deployment-cbdccf466-wt5vc 1/1 Running 0 55m 10.96.2.14 gke-kubeassignment-cluster-kube-nodes-f2e23eda-08rc <none> <none>

ukuduwa@cloudshell:~ (kubernetesintellipaat-400507)$

1. nginx Service was already created, create one for Apache.

ukuduwa@cloudshell:~ (kubernetesintellipaat-400507)$ kubectl create service clusterip httpd --tcp=80:80

service/httpd created

ukuduwa@cloudshell:~ (kubernetesintellipaat-400507)$ kubectl get svc

NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE

httpd ClusterIP 10.100.2.53 <none> 80/TCP 4s

kubernetes ClusterIP 10.100.0.1 <none> 443/TCP 64m

nginx ClusterIP 10.100.7.16 <none> 80/TCP 15m

ukuduwa@cloudshell:~ (kubernetesintellipaat-400507)$

1. Install the nginx ingress controller via github.
2. Once installed, deploy a yaml file with ingress rules :

ukuduwa@cloudshell:~ (kubernetesintellipaat-400507)$ sudo nano demo.yaml

ukuduwa@cloudshell:~ (kubernetesintellipaat-400507)$ cat demo.yaml

apiVersion: networking.k8s.io/v1

kind: Ingress

metadata:

name: ingress-kube

annotations:

nginx.ingress.kubernetes.io/rewrite-target: /

spec:

rules:

- http:

paths:

- path: /nginx

backend:

serviceName: nginx

servicePort: 80

- path: /apache

backend:

serviceName: httpd

serviePort: 80

1. If the path is /apache it will route to httpd, if /nginx it will route to nginx.