



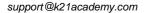
Debugging Application Failure

[Edition 1]

[Last Update 200822]

1

For any issues/help contact: support@k21academy.com







Contents

1 Introduction	7
2 Documentation	
2.1 Kubernetes Documentation	
3 Debugging Application Failure Scenario	
3.1 Running NGINX Server as Scalable Deployment in Cluster	
3.2 Creating Service to Expose Deployed NGINX Web-Server	
3.3 Debugging & Fixing Issue	
4 Summary	





INTRODUCTION 1

A Kubernetes cluster is a set of node machines for running containerized applications. If you're running Kubernetes, you're running a cluster. At a minimum, a cluster contains a control plane and one or more compute machines, or nodes.

This guide Covers:





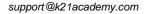


2 DOCUMENTATION

2.1 Kubernetes Documentation

- Cluster Node Maintenance
 https://kubernetes.io/docs/tasks/administer-cluster/safely-drain-node/
- 2. Cluster Maintenance https://kubernetes.io/docs/tasks/administer-cluster/cluster-management/
- 3. Troubleshooting kubeadm

 https://kubernetes.io/docs/setup/production-environment/tools/kubeadm/troubleshooting-kubeadm/
- 4. Developing and debugging services locally https://kubernetes.io/docs/tasks/debug-application-cluster/local-debugging/
- 5. Troubleshoot Applications
 https://kubernetes.io/docs/tasks/debug-application-cluster/debug-application/
- Monitoring, Logging, and Debugging https://kubernetes.io/docs/tasks/debug-application-cluster/







3 DEBUGGING APPLICATION FAILURE SCENARIO

3.1 Running NGINX Server as Scalable Deployment in Cluster

1. Viewing the nginx-deployment.yaml file to see the nginx server definition

\$ vim nginx-deployment.yaml

```
apiVersion: apps/v1
kind: Deployment
metadata:
    name: nginx-deployment
labels:
    app: nginx
spec:
    replicas: 2
    selector:
        matchLabels:
        app: nginx
template:
        metadata:
        labels:
            app: nginx
spec:
        containers:
            - name: nginx
        image: nginx:1.12
        ports:
            - containerPort: 80
```

2. Creating resources using the nginx-deployment.yaml file

\$ kubectl create -f nginx-deployment.yaml

```
root@kubeadm-master:/home/ubuntu/Kubernetes# kubectl create -f nginx-deployment.yaml
deployment.apps/nginx-deployment created
```

3. Listing the deployments in the cluster

\$ kubectl get deployment

```
root@kubeadm-master:/home/ubuntu/Kubernetes# kubectl get deployment
NAME READY UP-TO-DATE AVAILABLE AGE
nginx-deployment 2/2 2 2 71s
root@kubeadm-master:/home/ubuntu/Kubernetes#
```





4. Listing all resources running in default namespace in the cluster

\$ kubectl get all

```
root@kubeadm-master:/home/ubuntu/Kubernetes-basics# kubectl get all
                                         READY STATUS
                                                            RESTARTS
                                                                       AGE
pod/nginx-deployment-7c68bd84c6-d4155
                                         1/1
                                                  Running
                                                                       13s
pod/nginx-deployment-7c68bd84c6-qd47j
                                         1/1
                                                  Running
                     TYPE
                                  CLUSTER-IP
                                               EXTERNAL-IP
                                                                        AGE
                                                              PORT(S)
service/kubernetes
                     ClusterIP
                                  10.96.0.1
                                                              443/TCP
                                                                        18m
                                               <none>
                                    READY
                                            UP-TO-DATE
                                                          AVAILABLE
                                                                      AGE
deployment.apps/nginx-deployment
                                            2
                                                                      13s
NAME
                                               DESIRED
                                                          CURRENT
                                                                    READY
                                                                             AGE
replicaset.apps/nginx-deployment-7c68bd84c6
                                                                            135
```

5. Listing all the pod and using -o wide option to list the nodes hosting the nginx replicas

\$ kubectl get pods -o wide

```
root@kubeadm-master:/home/ubuntu/Kubernetes-basics# kubectl get pods -o wide
                                    READY
                                            STATUS
                                                                                                              READINESS GATES
                                                                       10.32.0.3
nginx-deployment-7c68bd84c6-d4155
                                    1/1
                                            Running
                                                                                   worker2
                                                                                             <none>
                                                                                                              <none>
nginx-deployment-7c68bd84c6-qd47j
                                    1/1
                                            Running
                                                      0
                                                                       10.32.0.2
                                                                                   worker2
                                                                                                              <none>
                                                                                             <none>
root@kubeadm-master:/home/ubuntu/Kubernetes-basics#
```

3.2 Creating Service to Expose Deployed NGINX Web-Server

1. Viewing the nginx-svc.yaml file to see the nginx service definition

\$ vim nginx-svc.yaml





2. Create the service and list

\$ kubectl create -f nginx-svc.yaml

\$ kubectl get svc

```
root@kubeadm-master:/home/ubuntu/Kubernetes-basics# kubectl create -f nginx-svc.yaml
service/my-service created
root@kubeadm-master:/home/ubuntu/Kubernetes-basics# kubectl get svc
NAME
             TYPE
                         CLUSTER-IP
                                          EXTERNAL-IP
                                                        PORT(S)
kubernetes
             ClusterIP
                        10.96.0.1
                                          <none>
                                                        443/TCP
                                                                  24m
                       10.105.200.234
                                                        80/TCP
my-service
             ClusterIP
                                          <none>
                                                                  5s
root@kubeadm-master:/home/ubuntu/Kubernetes-basics#
```

3. Curl on the service ClusterIP to see the web server welcome page

```
$ curl <service ClusterIP>
$ curl 10.105.200.234
```

```
root@kubeadm-master:/home/ubuntu/Kubernetes-basics#
root@kubeadm-master:/home/ubuntu/Kubernetes-basics# curl 10.105.200.234
curl: (7) Failed to connect to 10.105.200.234 port 80: Connection refused
root@kubeadm-master:/home/ubuntu/Kubernetes-basics#
```

3.3 Debugging & Fixing Issue

1. Describe the service to check if Pod to Service connection is proper

\$ kubectl describe svc my-service

```
root@kubeadm-master:/home/ubuntu/Kubernetes-basics#
root@kubeadm-master:/home/ubuntu/Kubernetes-basics# kubectl describe svc my-service
Name:
                   my-service
Namespace:
                   default
Labels:
                   <none>
                   Selector: app=MyApp
Annotations:
Type:
                   ClusterIP
                   10.105.200.234
Port:
                   <unset> 80/TCP
TargetPort:
                   82/TCP
Endpoints:
                   <none>
Session Affinity:
                   None
Events:
root@kubeadm-master:/home/ubuntu/Kubernetes-basics#
```





We see that Pod IP Addresses are not listing in the service Endpoint list. Pod and Service connection is dependent on the Label and Selector values. Match the Pod label to the Service selector value

\$ kubectl describe svc my-service

```
root@kubeadm-master:/home/ubuntu/Kubernetes-basics#
root@kubeadm-master:/home/ubuntu/Kubernetes-basics# kubectl describe svc my-service
Name:
                   my-service
Namespace:
                   default
Labels:
                   <none>
Annotations:
                   Selector:
                             app=MyApp
Type:
                   ClusterIP
IP:
                   10.105.200.234
Port:
                   <unset> 80/TCP
TargetPort:
                   82/TCP
                   <none>
Endpoints:
Session Affinity:
                  None
                   <none>
root@kubeadm-master:/home/ubuntu/Kubernetes-basics#
```

3. Pod label is marked as app=nginx and Service selector is app=MyApp. Correct the selector in nginx-syc.yaml file to app: nginx

\$ vim nginx-svc.yaml

4. Apply the changes and describe the service to see that Pod IP Addresses gets populated in the Service Endpoint list

```
$ kubectl apply -f nginx-svc.yaml
```

\$ kubectl describe svc my-service





root@kubeadm-master:/home/ubuntu/Kubernetes-basics# kubectl apply -f nginx-svc.yaml service/mv-service configured root@kubeadm-master:/home/ubuntu/Kubernetes-basics# kubectl describe svc my-service Name: my-service Namespace: default Labels: <none> Annotations: Selector: app=nginx Type: ClusterIP IP: 10.105.200.234 Port: <unset> 80/TCP TargetPort: 82/TCP Endpoints: 10.32.0.2:82,10.32.0.3:82 Session Affinity: None

Events: <none>

[root@kubeadm-master:/home/ubuntu/Kubernetes-basics#

5. Now try to curl on Service IP Address and open the Nginx web server

\$ curl 10.105.200.234

```
root@kubeadm-master:/home/ubuntu/Kubernetes-basics#
root@kubeadm-master:/home/ubuntu/Kubernetes-basics# curl 10.105.200.234
curl: (7) Failed to connect to 10.105.200.234 port 80: Connection refused
root@kubeadm-master:/home/ubuntu/Kubernetes-basics#
```

Its again connection refused.

- 6. Look for other parameters in Service yaml file and verify the port numbers. Target Port for Service is Container's port to which it forwards the traffic. Target port in the service vaml file is 82 whereas nginx pod is expecting packet on port 80
- 7. Correct the target port in nginx-svc.yaml file, apply changes and verify if Nginx webserver welcome page is seen

\$ vim nginx-svc.yaml

```
apiVersion: v1
kind: Service
metadata:
  name: my-service
spec:
  selector:
    app: nginx
  ports:
     - protocol: TCP
      port: 80
      targetPort: 80
```





- \$ kubectl apply -f nginx-svc.yaml
- \$ kubectl describe svc my-service

root@kubeadm-master:/home/ubuntu/Kubernetes-basics#

root@kubeadm-master:/home/ubuntu/Kubernetes-basics# vim nginx-svc.yaml

root@kubeadm-master:/home/ubuntu/Kubernetes-basics# kubectl apply -f nginx-svc.yaml

service/my-service configured

root@kubeadm-master:/home/ubuntu/Kubernetes-basics# kubectl describe svc my-service

Name: my-service Namespace: default Labels: <none>

Annotations: Selector: app=nginx

Type: ClusterIP
IP: 10.105.200.234
Port: <unset> 80/TCP

TargetPort: 80/TCP

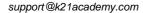
Endpoints: 10.32.0.2:80,10.32.0.3:80

Session Affinity: None Events: <none>

root@kubeadm-master:/home/ubuntu/Kubernetes-basics#

8. Now try to curl on Service IP Address and open the Nginx web server

\$ curl 10.105.200.234







4 SUMMARY

In this guide we Covered:

• Debugging application failure scenario

