Name: Hrishikesh Kumbhar

Div: D15A

Roll no: 32

Sub: Advanced DevOps

Experiment No: 4

Date: 30/08/2022

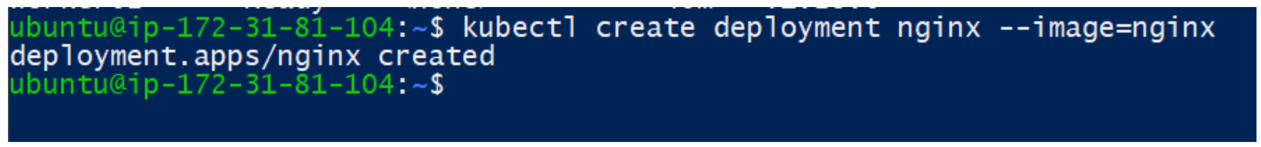
Aim: To install Kubectl and execute Kubectl commands to manage the Kubernetes cluster and deploy Your First Kubernetes Application.

Steps:

Running An Application on the Cluster

You can now deploy any containerized application to your cluster. To keep things familiar, let's deploy Nginx using Deployments and Services to see how this application can be deployed to the cluster. You can use the commands below for other containerized applications as well, provided you change the Docker image name and any relevant flags (such as ports and volumes).
Still within the master node, execute the following command to create a deployment named nginx:

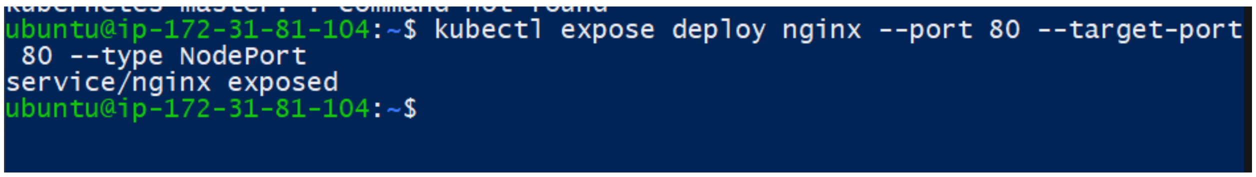
```
kubernetes-master:~$kubectl create deployment nginx --image=nginx
```



```
ubuntu@ip-172-31-81-104:~$ kubectl create deployment nginx --image=nginx
deployment.apps/nginx created
ubuntu@ip-172-31-81-104:~$
```

A deployment is a type of Kubernetes object that ensures there's always a specified number of pods running based on a defined template, even if the pod crashes during the cluster's lifetime. The above deployment will create a pod with one container from the Docker registry's Nginx Docker Image.
Next, run the following command to create a service named nginx that will expose the app publicly. It will do so through a NodePort, a scheme that will make the pod accessible through an arbitrary port opened on each node of the cluster:

```
kubernetes-master:~$kubectl expose deploy nginx --port 80
--target-port 80 --type NodePort
```

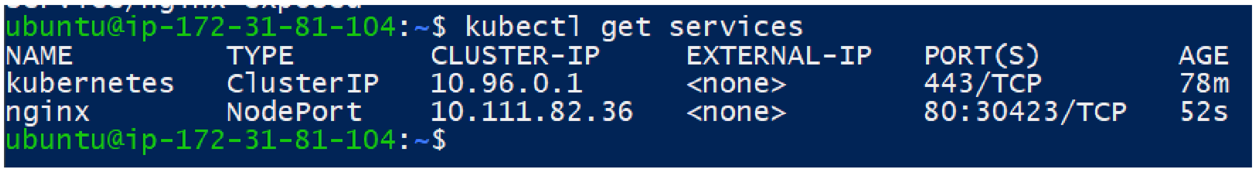


```
ubuntu@ip-172-31-81-104:~$ kubectl expose deploy nginx --port 80 --target-port
 80 --type NodePort
service/nginx exposed
ubuntu@ip-172-31-81-104:~$
```

Services are another type of Kubernetes object that expose cluster internal services to clients, both internal and external. They are also capable of load balancing requests to multiple pods, and are an integral component in Kubernetes, frequently interacting with other components.
Run the following command:

```
kubernetes-master:~$kubectl get services
```

This will output text similar to the following:



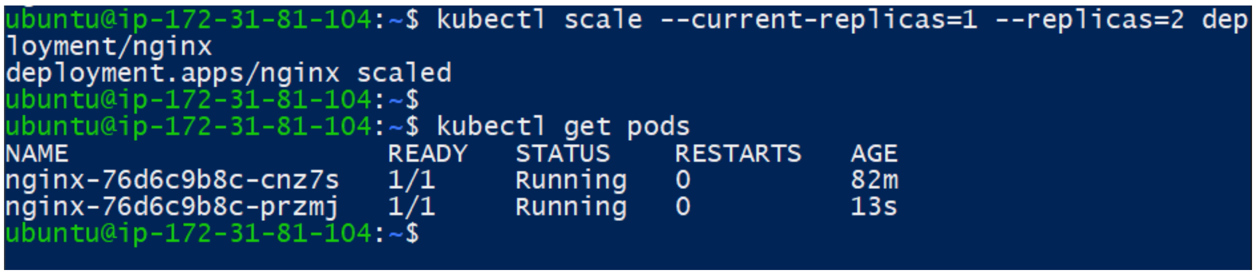
```
ubuntu@ip-172-31-81-104:~$ kubectl get services
NAME         TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)        AGE
kubernetes   ClusterIP   10.96.0.1      <none>        443/TCP        78m
nginx        NodePort    10.111.82.36   <none>        80:30423/TCP   52s
ubuntu@ip-172-31-81-104:~$
```

From the third line of the above output, you can retrieve the port that Nginx is running on.
Kubernetes will assign a random port that is greater than 30000 automatically, while ensuring that
the port is not already bound by another service.
Note: if you're running your setup on ec2 ensure the nginx_port is open under the inbound rules in the security groups.
To test that everything is working, visit http://worker_1_ip:nginx_port
or
http://worker_2_ip:nginx_port
through a browser on your local machine. You will see Nginx's familiar welcome page.
To see the deployed container on worker node switch to worker01
on-slave#docker ps
Output: you will see the container for nginx image running.

If ou want to scale up the replicas for a deployment (nginx in our case) the use the following command:

```
kubernetes-master:~$kubectl scale --current-replicas=1 --replicas=2
deployment/nginx
```

```
kubernetes-master:~$kubectl get pods
```

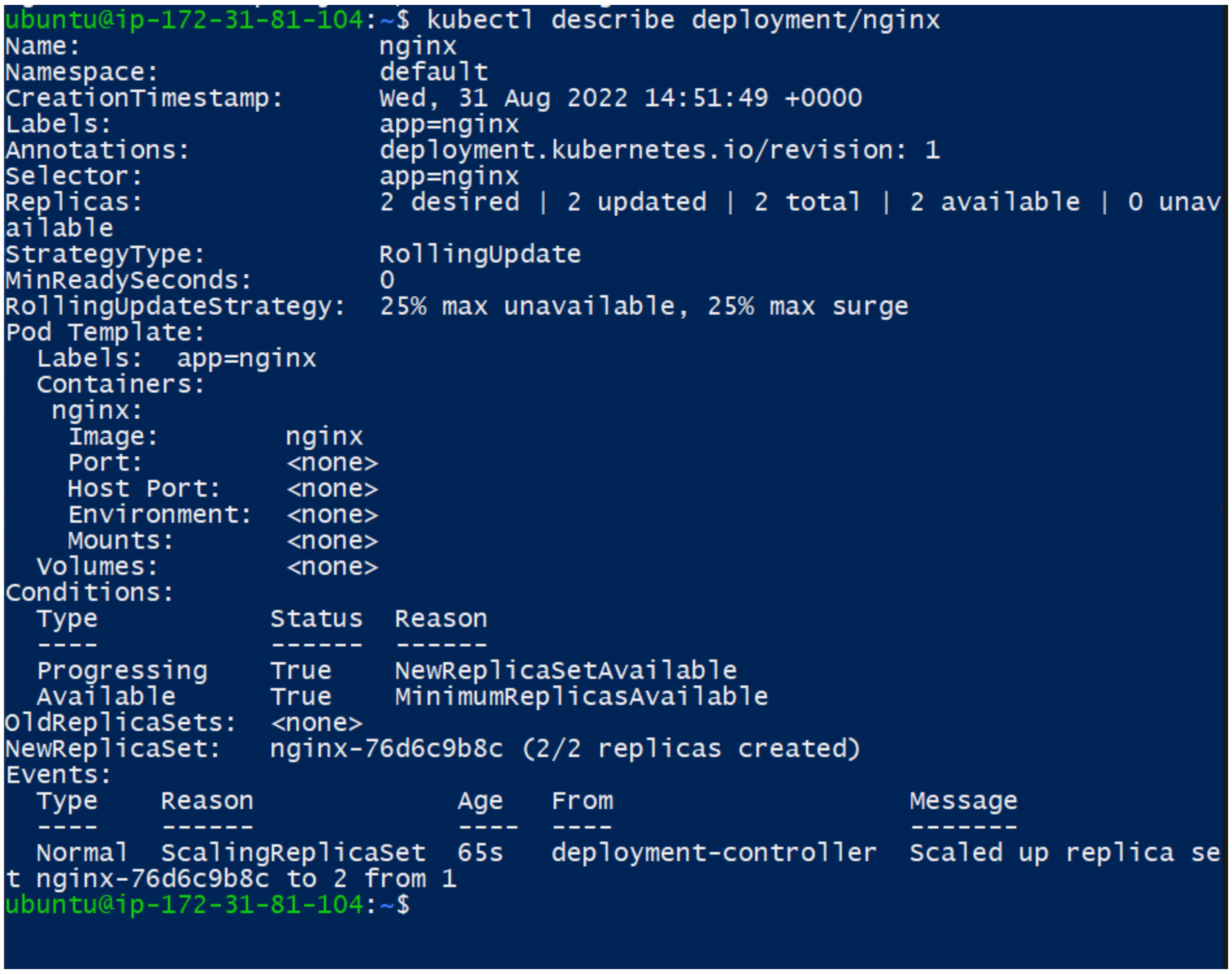


```
ubuntu@ip-172-31-81-104:~$ kubectl scale --current-replicas=1 --replicas=2 dep
loyment/nginx
deployment.apps/nginx scaled
ubuntu@ip-172-31-81-104:~$
ubuntu@ip-172-31-81-104:~$ kubectl get pods
NAME                     READY   STATUS    RESTARTS   AGE
nginx-76d6c9b8c-cnz7s    1/1     Running   0          82m
nginx-76d6c9b8c-przmj    1/1     Running   0          13s
ubuntu@ip-172-31-81-104:~$
```

Output: you will see 2/2 as output in nginx deployment.

```
kubernetes-master:~$kubectl describe deployment/nginx
```

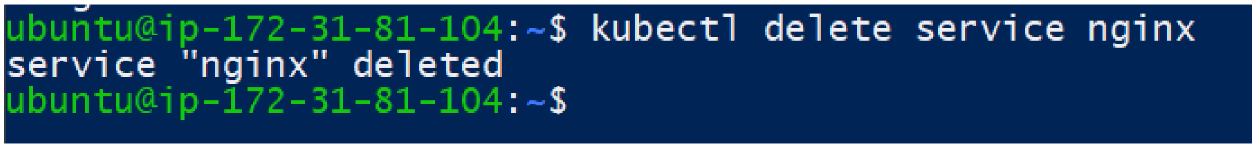
Output: give details about the service deployed



```
ubuntu@ip-172-31-81-104:~$ kubectl describe deployment/nginx
Name:                   nginx
Namespace:              default
CreationTimestamp:      Wed, 31 Aug 2022 14:51:49 +0000
Labels:                 app=nginx
Annotations:            deployment.kubernetes.io/revision: 1
Selector:               app=nginx
Replicas:               2 desired | 2 updated | 2 total | 2 available | 0 unav
ailable
StrategyType:           RollingUpdate
MinReadySeconds:        0
RollingUpdateStrategy:  25% max unavailable, 25% max surge
Pod Template:
  Labels:  app=nginx
  Containers:
   nginx:
    Image:        nginx
    Port:         <none>
    Host Port:    <none>
    Environment:  <none>
    Mounts:       <none>
  Volumes:        <none>
Conditions:
  Type           Status  Reason
  ----           ------  ------
  Progressing    True    NewReplicaSetAvailable
  Available      True    MinimumReplicasAvailable
OldReplicaSets:  <none>
NewReplicaSet:   nginx-76d6c9b8c (2/2 replicas created)
Events:
  Type    Reason             Age   From                   Message
  ----    ------             ----  ----                   -------
  Normal  ScalingReplicaSet  65s   deployment-controller  Scaled up replica se
t nginx-76d6c9b8c to 2 from 1
ubuntu@ip-172-31-81-104:~$
```

If you would like to remove the Nginx application, first delete the nginx service from the master node:

```
kubernetes-master:~$kubectl delete service nginx
```

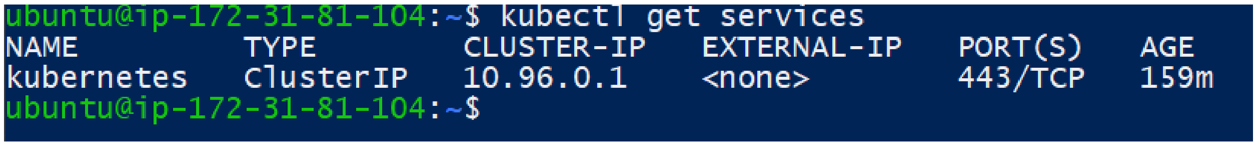


```
ubuntu@ip-172-31-81-104:~$ kubectl delete service nginx
service "nginx" deleted
ubuntu@ip-172-31-81-104:~$
```

Run the following to ensure that the service has been deleted:

```
kubernetes-master:~$kubectl get services
```

You will see the following output:
Output



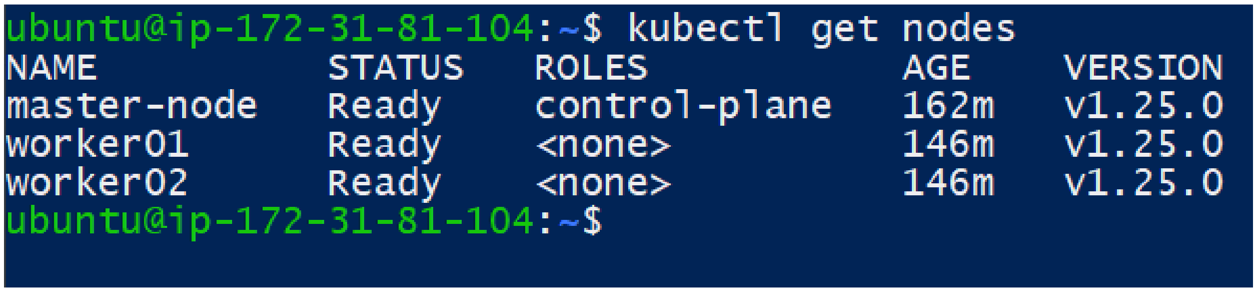
```
ubuntu@ip-172-31-81-104:~$ kubectl get services
NAME         TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
kubernetes   ClusterIP   10.96.0.1    <none>        443/TCP   159m
ubuntu@ip-172-31-81-104:~$
```

How to gracefully remove a node from Kubernetes?

On Master Node

Find the node

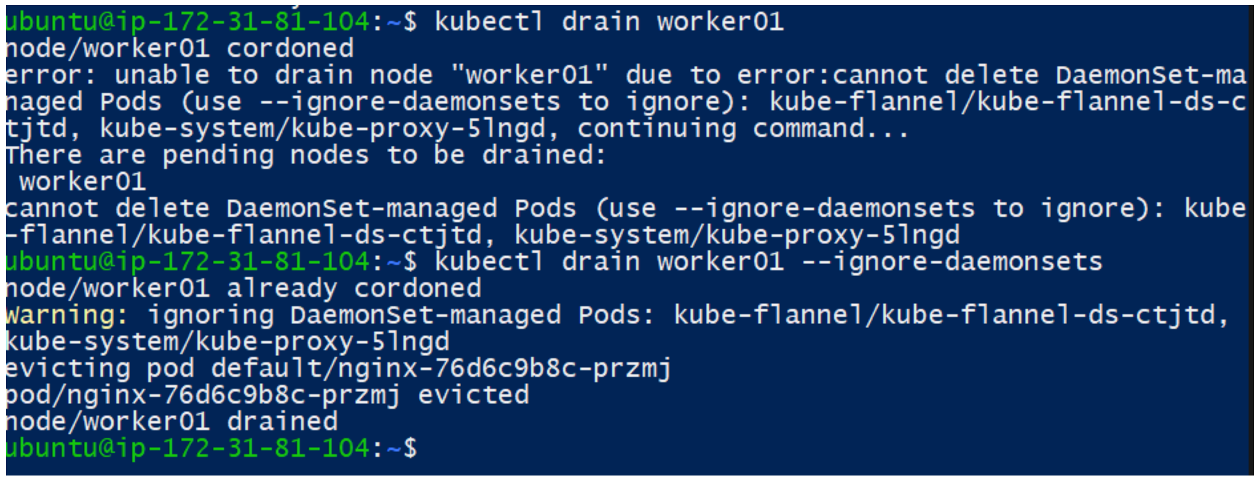
```
kubernetes-master:~$kubectl get nodes
```



```
ubuntu@ip-172-31-81-104:~$ kubectl get nodes
NAME          STATUS   ROLES           AGE    VERSION
master-node   Ready    control-plane   162m   v1.25.0
worker01      Ready    <none>          146m   v1.25.0
worker02      Ready    <none>          146m   v1.25.0
ubuntu@ip-172-31-81-104:~$
```

Drain it

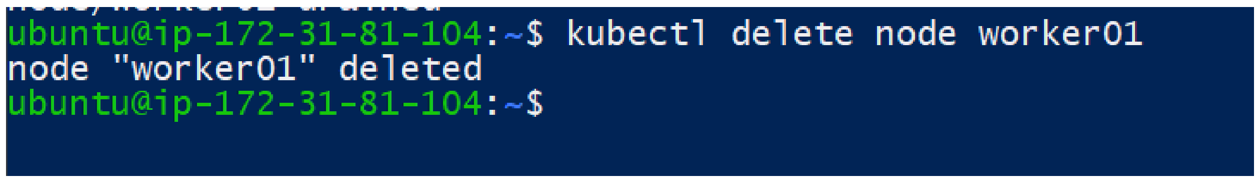
```
kubernetes-master:~$kubectl drain nodetoberemoved
```



```
ubuntu@ip-172-31-81-104:~$ kubectl drain worker01
node/worker01 cordoned
error: unable to drain node "worker01" due to error:cannot delete DaemonSet-ma
naged Pods (use --ignore-daemonsets to ignore): kube-flannel/kube-flannel-ds-c
tjtd, kube-system/kube-proxy-5lngd, continuing command...
There are pending nodes to be drained:
 worker01
cannot delete DaemonSet-managed Pods (use --ignore-daemonsets to ignore): kube
-flannel/kube-flannel-ds-ctjtd, kube-system/kube-proxy-5lngd
ubuntu@ip-172-31-81-104:~$ kubectl drain worker01 --ignore-daemonsets
node/worker01 already cordoned
Warning: ignoring DaemonSet-managed Pods: kube-flannel/kube-flannel-ds-ctjtd,
kube-system/kube-proxy-5lngd
evicting pod default/nginx-76d6c9b8c-przmj
pod/nginx-76d6c9b8c-przmj evicted
node/worker01 drained
ubuntu@ip-172-31-81-104:~$
```

Delete it

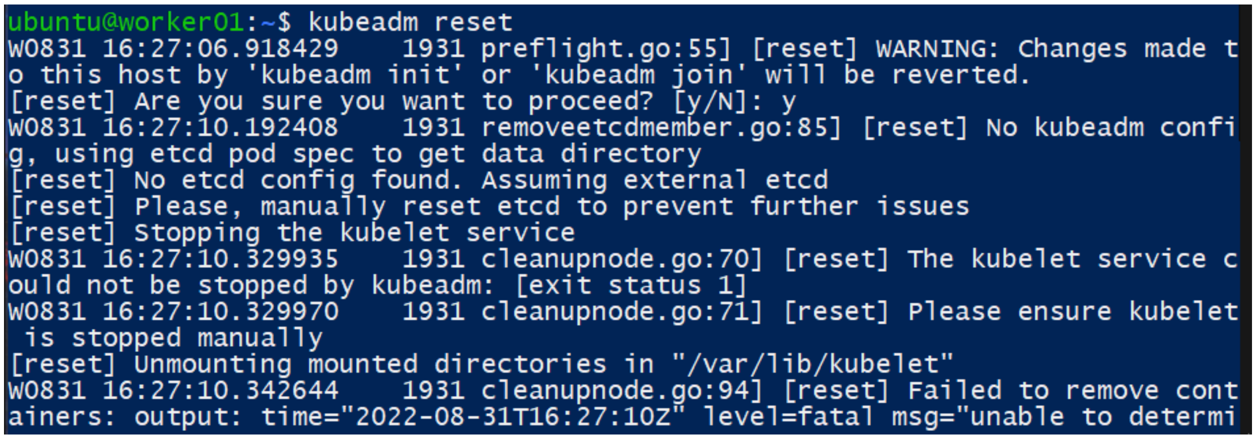
```
kubernetes-master:~$kubectl delete node nodetoberemoved
```



```
ubuntu@ip-172-31-81-104:~$ kubectl delete node worker01
node "worker01" deleted
ubuntu@ip-172-31-81-104:~$
```

On Worker Node (nodetoberemoved). Remove join/init setting from node

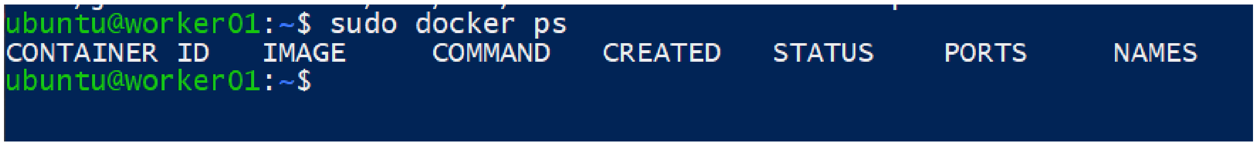
```
kubernetes-slave:~$kubeadm reset
```



```
ubuntu@worker01:~$ kubeadm reset
W0831 16:27:06.918429    1931 preflight.go:55] [reset] WARNING: Changes made t
o this host by 'kubeadm init' or 'kubeadm join' will be reverted.
[reset] Are you sure you want to proceed? [y/N]: y
W0831 16:27:10.192408    1931 removeetcdmember.go:85] [reset] No kubeadm confi
g, using etcd pod spec to get data directory
[reset] No etcd config found. Assuming external etcd
[reset] Please, manually reset etcd to prevent further issues
[reset] Stopping the kubelet service
W0831 16:27:10.329935    1931 cleanupnode.go:70] [reset] The kubelet service c
ould not be stopped by kubeadm: [exit status 1]
W0831 16:27:10.329970    1931 cleanupnode.go:71] [reset] Please ensure kubelet
 is stopped manually
[reset] Unmounting mounted directories in "/var/lib/kubelet"
W0831 16:27:10.342644    1931 cleanupnode.go:94] [reset] Failed to remove cont
ainers: output: time="2022-08-31T16:27:10Z" level=fatal msg="unable to determi
```

Press y to proceed

```
kubernetes-slave:~$docker ps
```



```
ubuntu@worker01:~$ sudo docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
ubuntu@worker01:~$
```

Conclusion:

After following the steps mentioned in this article carefully, you should now have Kubernetes installed on Ubuntu.
This network uses multiple servers to communicate back and forth. Kubernetes allows you to launch and manage Docker containers across multiple servers in the pod.