Horizontal Scaling

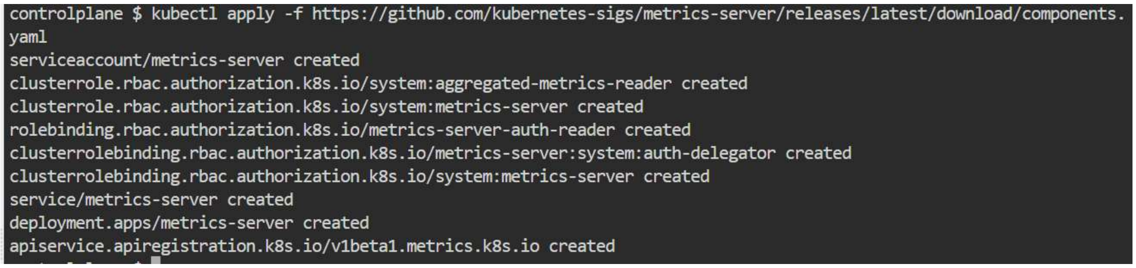
- Horizontal scaling, also known as scaling out, involves adding more machines or nodes to a system to handle increased load.
- It distributes the workload across multiple servers to improve performance and availability.
- This approach enhances capacity by expanding resources rather than upgrading existing hardware.

Key features of horizontal scaling include:

1. **Improved Fault Tolerance**: Distributes the load across multiple machines, reducing the risk of a single point of failure.
2. **Enhanced Performance**: Increases capacity to handle more requests simultaneously by adding more nodes.
3. **Scalability and Flexibility**: Easily adapts to growing workloads by incorporating additional servers or nodes as needed.

To be Install the Metric server K8s

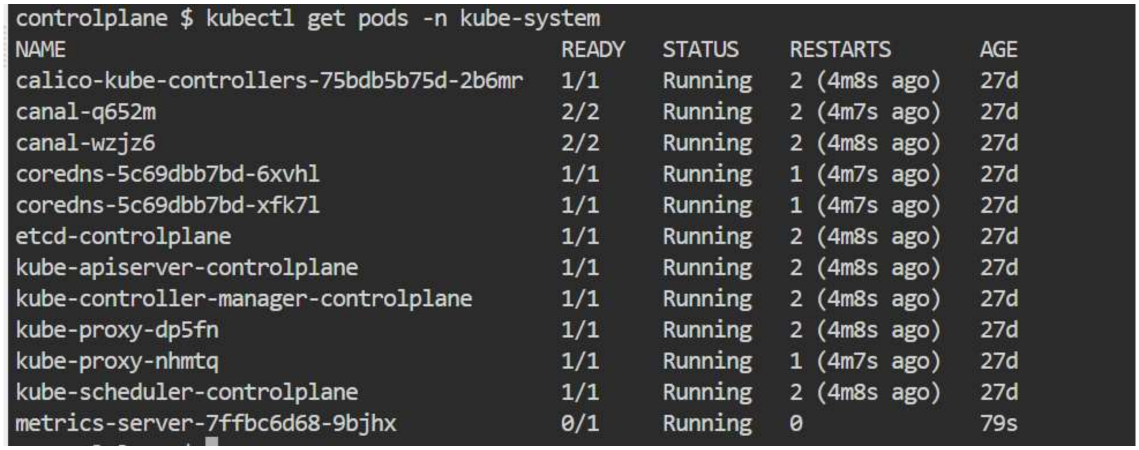
1.Kubectl apply –f https://github.com/kubernetes-sigs/metrics-server/releases/latest/download/componets



```
controlplane $ kubectl apply -f https://github.com/kubernetes-sigs/metrics-server/releases/latest/download/components.
yaml
serviceaccount/metrics-server created
clusterrole.rbac.authorization.k8s.io/system:aggregated-metrics-reader created
clusterrole.rbac.authorization.k8s.io/system:metrics-server created
rolebinding.rbac.authorization.k8s.io/metrics-server-auth-reader created
clusterrolebinding.rbac.authorization.k8s.io/metrics-server:system:auth-delegator created
clusterrolebinding.rbac.authorization.k8s.io/system:metrics-server created
service/metrics-server created
deployment.apps/metrics-server created
apiservice.apiregistration.k8s.io/v1beta1.metrics.k8s.io created
```

2. To verify the command:

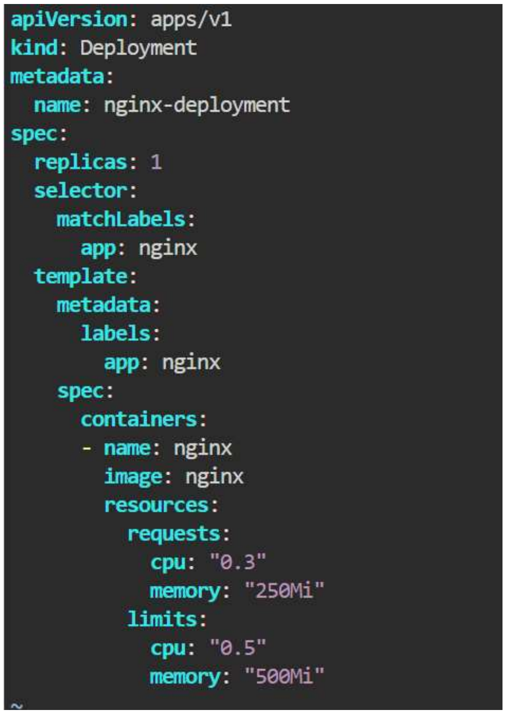
Kubectl get pods –n kube-system



```
controlplane $ kubectl get pods -n kube-system
NAME                                      READY   STATUS    RESTARTS       AGE
calico-kube-controllers-75bdb5b75d-2b6mr  1/1     Running   2 (4m8s ago)   27d
canal-q652m                               2/2     Running   2 (4m7s ago)   27d
canal-wzjz6                               2/2     Running   2 (4m8s ago)   27d
coredns-5c69dbb7bd-6xvhl                  1/1     Running   1 (4m7s ago)   27d
coredns-5c69dbb7bd-xfk7l                  1/1     Running   1 (4m7s ago)   27d
etcd-controlplane                         1/1     Running   2 (4m8s ago)   27d
kube-apiserver-controlplane               1/1     Running   2 (4m8s ago)   27d
kube-controller-manager-controlplane      1/1     Running   2 (4m8s ago)   27d
kube-proxy-dp5fn                          1/1     Running   2 (4m8s ago)   27d
kube-proxy-nhmtq                          1/1     Running   1 (4m7s ago)   27d
kube-scheduler-controlplane               1/1     Running   2 (4m8s ago)   27d
metrics-server-7ffbc6d68-9bjhx            0/1     Running   0              79s
```

3. To create the yaml file:

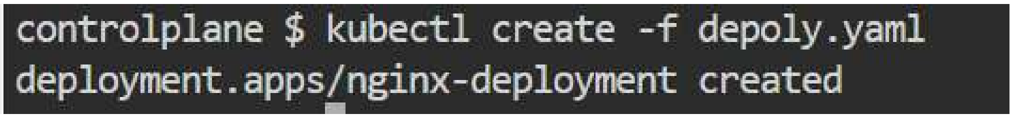
vi depoly.yaml



```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
spec:
  replicas: 1
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx
        resources:
          requests:
            cpu: "0.3"
            memory: "250Mi"
          limits:
            cpu: "0.5"
            memory: "500Mi"
```

4. To create the pods:

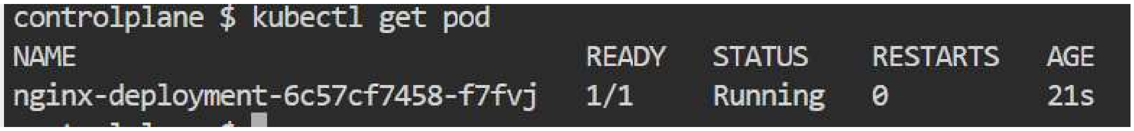
Kubectl create –f deploy.yaml



```
controlplane $ kubectl create -f depoly.yaml
deployment.apps/nginx-deployment created
```

5. To check the pods:

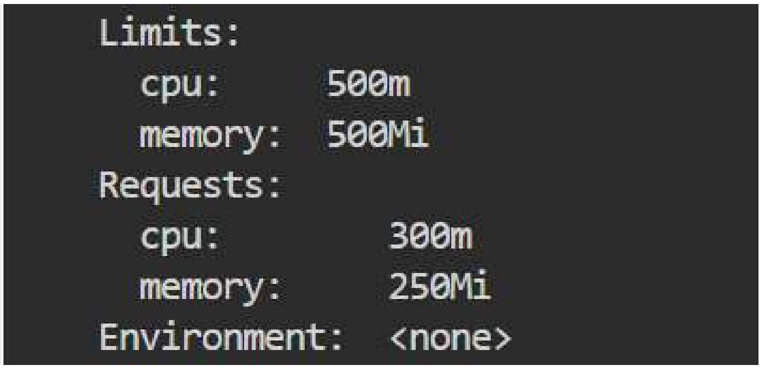
Kubectl get pod



```
controlplane $ kubectl get pod
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-6c57cf7458-f7fvj   1/1     Running   0          21s
```

6. To verify the resource utilization:

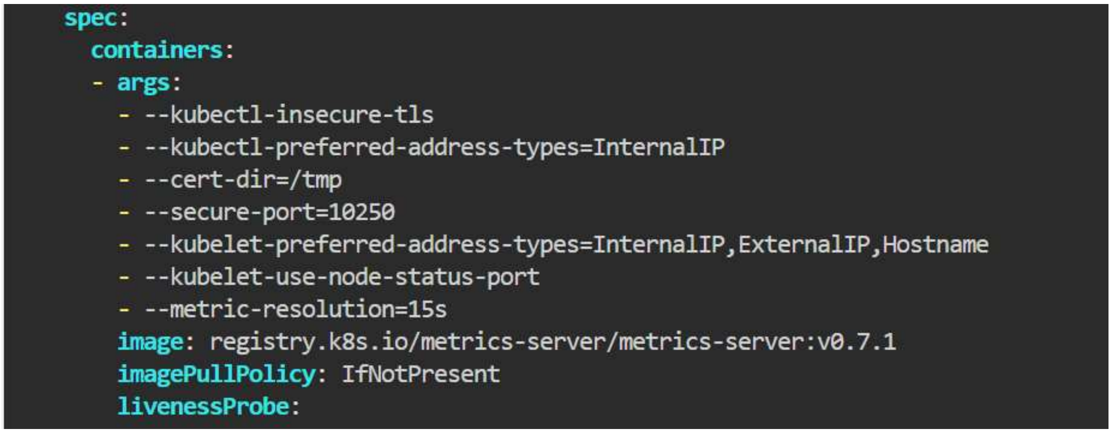
Kubectl describe pod nginx-deployment-6c57cf7458-f7fvj



```
    Limits:
      cpu:     500m
      memory:  500Mi
    Requests:
      cpu:        300m
      memory:     250Mi
    Environment:  <none>
```

7. To Edit the yaml file:

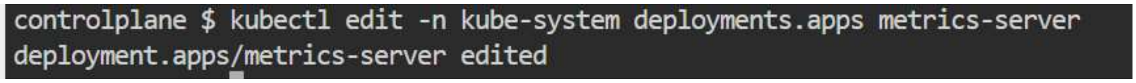
Kubectl edit -n kube-system deployments.apps metrics-server



```
    spec:
      containers:
      - args:
        - --kubectl-insecure-tls
        - --kubectl-preferred-address-types=InternalIP
        - --cert-dir=/tmp
        - --secure-port=10250
        - --kubelet-preferred-address-types=InternalIP,ExternalIP,Hostname
        - --kubelet-use-node-status-port
        - --metric-resolution=15s
        image: registry.k8s.io/metrics-server/metrics-server:v0.7.1
        imagePullPolicy: IfNotPresent
        livenessProbe:
```

8. And edit the images:

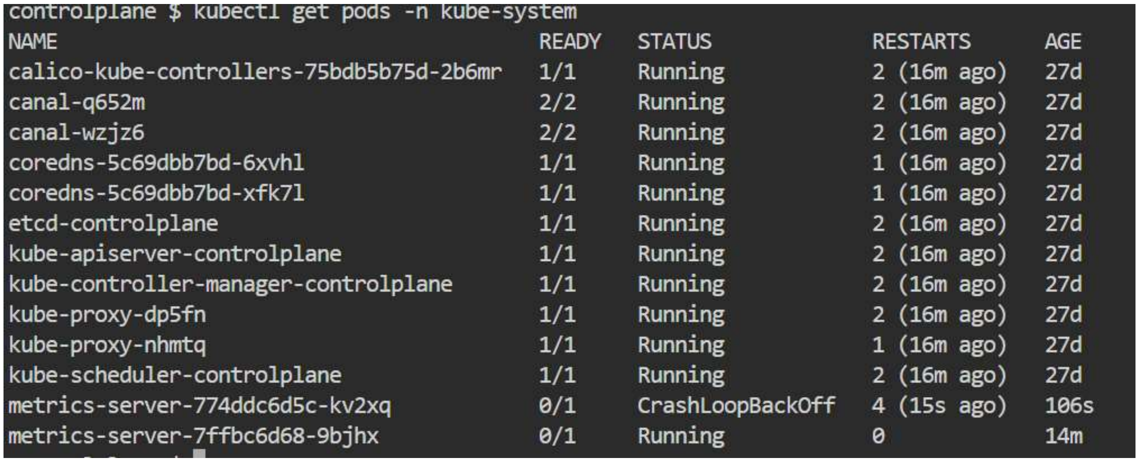
Kubectl edit –n kube-system deployments.apps metrics-server



```
controlplane $ kubectl edit -n kube-system deployments.apps metrics-server
deployment.apps/metrics-server edited
```

9. To verify the pods on kubesystem:

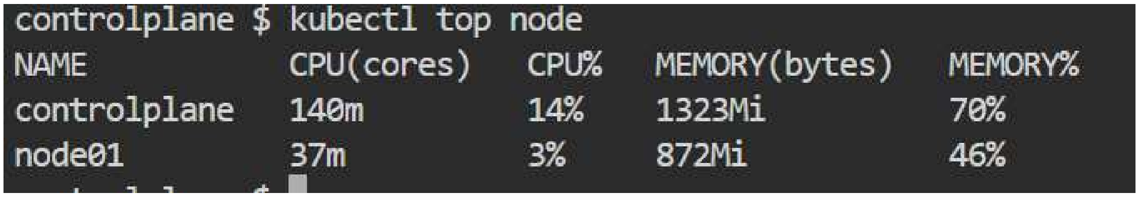
Kubectl get pods –n kube-system



```
controlplane $ kubectl get pods -n kube-system
NAME                                      READY   STATUS             RESTARTS       AGE
calico-kube-controllers-75bdb5b75d-2b6mr  1/1     Running            2 (16m ago)    27d
canal-q652m                               2/2     Running            2 (16m ago)    27d
canal-wzjz6                               2/2     Running            2 (16m ago)    27d
coredns-5c69dbb7bd-6xvhl                  1/1     Running            1 (16m ago)    27d
coredns-5c69dbb7bd-xfk7l                  1/1     Running            1 (16m ago)    27d
etcd-controlplane                         1/1     Running            2 (16m ago)    27d
kube-apiserver-controlplane               1/1     Running            2 (16m ago)    27d
kube-controller-manager-controlplane      1/1     Running            2 (16m ago)    27d
kube-proxy-dp5fn                          1/1     Running            2 (16m ago)    27d
kube-proxy-nhmtq                          1/1     Running            1 (16m ago)    27d
kube-scheduler-controlplane               1/1     Running            2 (16m ago)    27d
metrics-server-774ddc6d5c-kv2xq           0/1     CrashLoopBackOff   4 (15s ago)    106s
metrics-server-7ffbc6d68-9bjhx            0/1     Running            0              14m
```

10. To using the view to node:

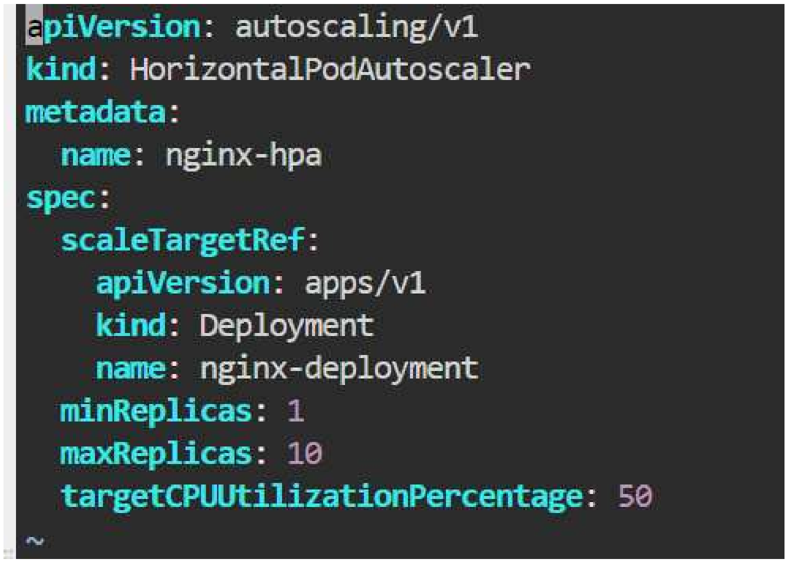
Kubectl top node



```
controlplane $ kubectl top node
NAME           CPU(cores)   CPU%   MEMORY(bytes)   MEMORY%
controlplane   140m         14%    1323Mi          70%
node01         37m          3%     872Mi           46%
```

1. Create the Yaml file:

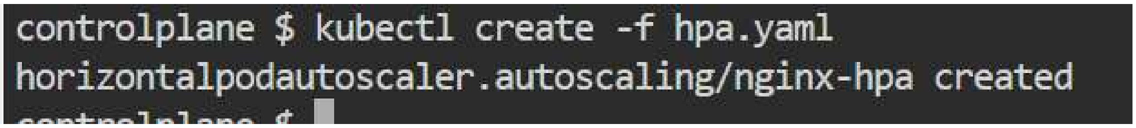
vi hpa.yaml



```
apiVersion: autoscaling/v1
kind: HorizontalPodAutoscaler
metadata:
  name: nginx-hpa
spec:
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: nginx-deployment
  minReplicas: 1
  maxReplicas: 10
  targetCPUUtilizationPercentage: 50
~
```

2. And create the pods:

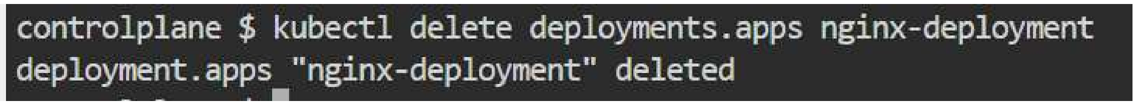
Kubectl create –f hpa.yaml



```
controlplane $ kubectl create -f hpa.yaml
horizontalpodautoscaler.autoscaling/nginx-hpa created
controlplane $
```

3. You have delete the pods:

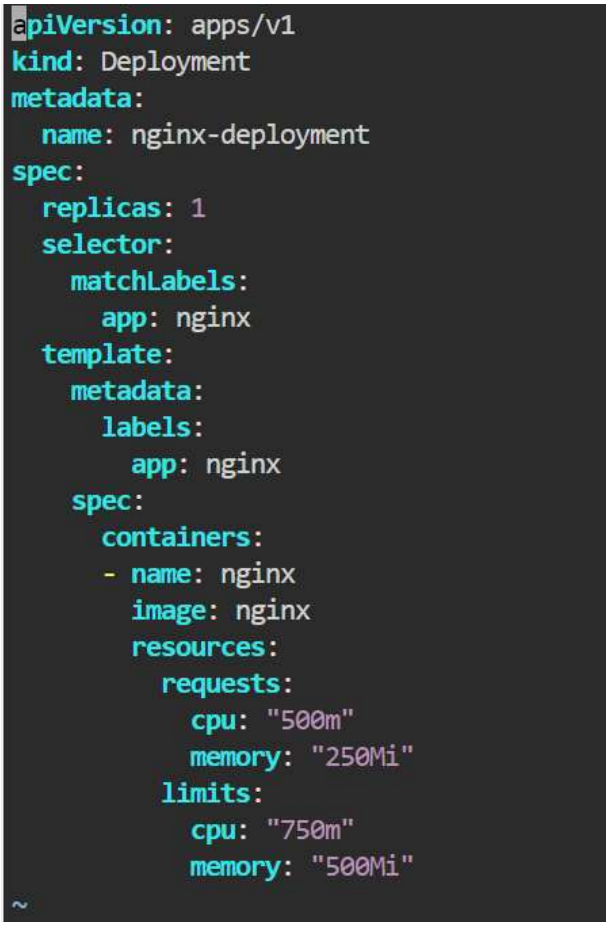
Kubectl delete deployments.apps nginx-deployment



```
controlplane $ kubectl delete deployments.apps nginx-deployment
deployment.apps "nginx-deployment" deleted
```

4. Create the yaml file:

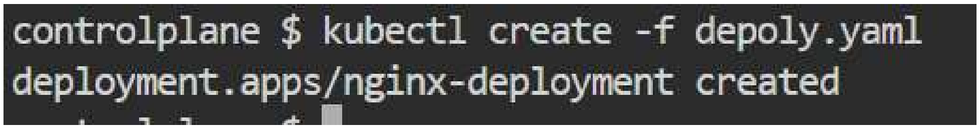
vi depoly.yaml



```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
spec:
  replicas: 1
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx
        resources:
          requests:
            cpu: "500m"
            memory: "250Mi"
          limits:
            cpu: "750m"
            memory: "500Mi"
~
```

5. Create the pod:

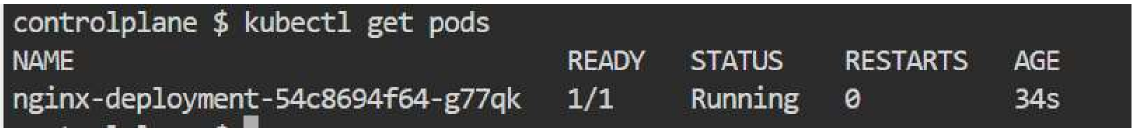
Kubectl create –f deploy.yaml



```
controlplane $ kubectl create -f depoly.yaml
deployment.apps/nginx-deployment created
```

6. To verify the pods:

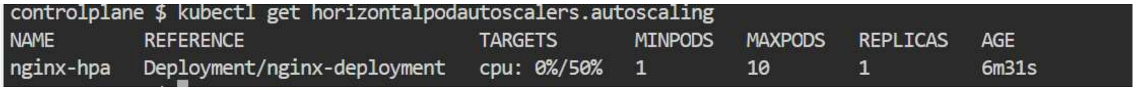
Kubectl get pods



```
controlplane $ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-54c8694f64-g77qk   1/1     Running   0          34s
```

7. To see the details of scaling:

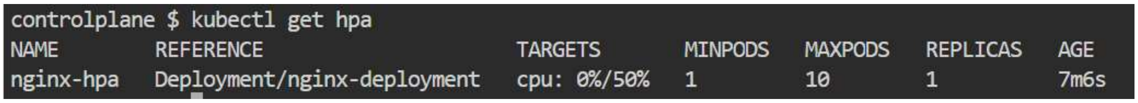
Kubectl get horizontalpodautoscalers.autoscaling



```
controlplane $ kubectl get horizontalpodautoscalers.autoscaling
NAME        REFERENCE                       TARGETS      MINPODS   MAXPODS   REPLICAS   AGE
nginx-hpa   Deployment/nginx-deployment    cpu: 0%/50%  1         10        1          6m31s
```

8. To check the status:

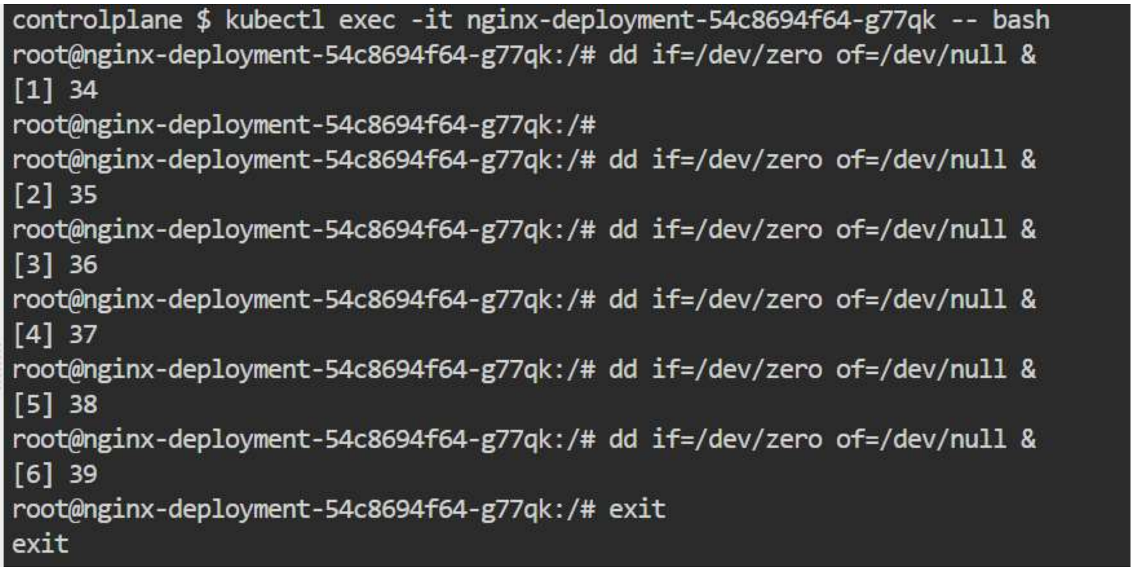
Kubectl get hpa



```
controlplane $ kubectl get hpa
NAME        REFERENCE                       TARGETS      MINPODS   MAXPODS   REPLICAS   AGE
nginx-hpa   Deployment/nginx-deployment    cpu: 0%/50%  1         10        1          7m6s
```

9. To be inside the pod:

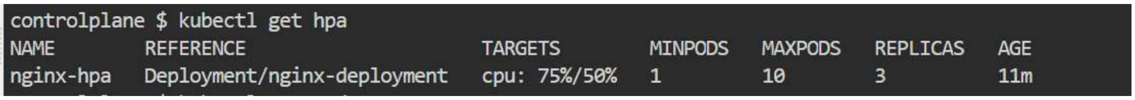
Kubectl exec –it nginx-deployment-54c8694f64-g77qk -- bash



```
controlplane $ kubectl exec -it nginx-deployment-54c8694f64-g77qk -- bash
root@nginx-deployment-54c8694f64-g77qk:/# dd if=/dev/zero of=/dev/null &
[1] 34
root@nginx-deployment-54c8694f64-g77qk:/#
root@nginx-deployment-54c8694f64-g77qk:/# dd if=/dev/zero of=/dev/null &
[2] 35
root@nginx-deployment-54c8694f64-g77qk:/# dd if=/dev/zero of=/dev/null &
[3] 36
root@nginx-deployment-54c8694f64-g77qk:/# dd if=/dev/zero of=/dev/null &
[4] 37
root@nginx-deployment-54c8694f64-g77qk:/# dd if=/dev/zero of=/dev/null &
[5] 38
root@nginx-deployment-54c8694f64-g77qk:/# dd if=/dev/zero of=/dev/null &
[6] 39
root@nginx-deployment-54c8694f64-g77qk:/# exit
exit
```

10. To check the status:

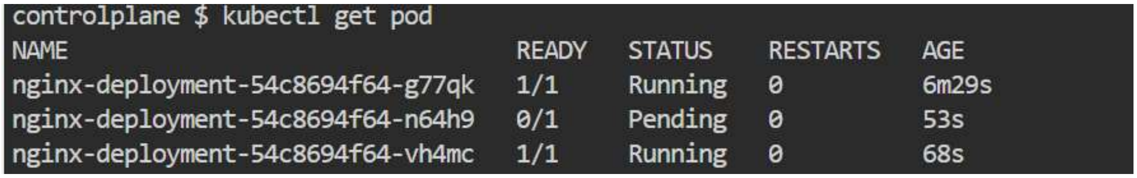
Kubectl get hpa



```
controlplane $ kubectl get hpa
NAME        REFERENCE                       TARGETS       MINPODS   MAXPODS   REPLICAS   AGE
nginx-hpa   Deployment/nginx-deployment    cpu: 75%/50%  1         10        3          11m
```

11. To check the pods:

Kubectl get pod

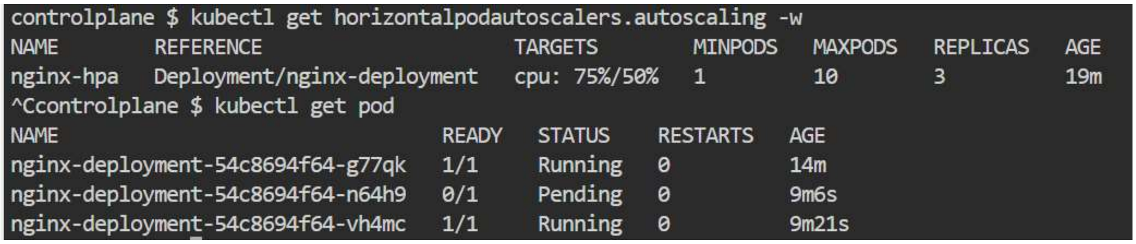


```
controlplane $ kubectl get pod
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-54c8694f64-g77qk   1/1     Running   0          6m29s
nginx-deployment-54c8694f64-n64h9   0/1     Pending   0          53s
nginx-deployment-54c8694f64-vh4mc   1/1     Running   0          68s
```

12. To verify the scaling:

Kubectl get horizontalpodautoscalers.autoscaling –w

Kubectl get pod

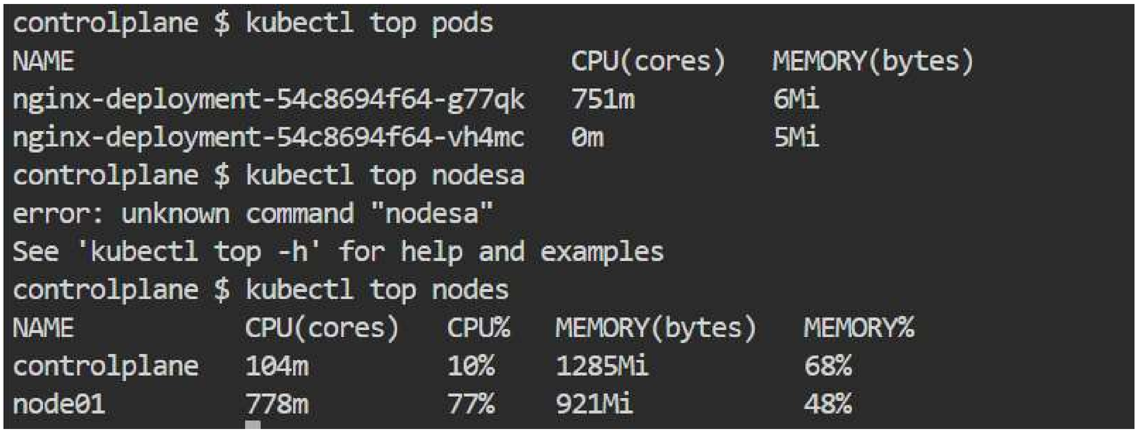


```
controlplane $ kubectl get horizontalpodautoscalers.autoscaling -w
NAME        REFERENCE                     TARGETS        MINPODS   MAXPODS   REPLICAS   AGE
nginx-hpa   Deployment/nginx-deployment   cpu: 75%/50%   1         10        3          19m
^Ccontrolplane $ kubectl get pod
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-54c8694f64-g77qk   1/1     Running   0          14m
nginx-deployment-54c8694f64-n64h9   0/1     Pending   0          9m6s
nginx-deployment-54c8694f64-vh4mc   1/1     Running   0          9m21s
```

13. To verify the pods & Nodes:

Kubectl top pods

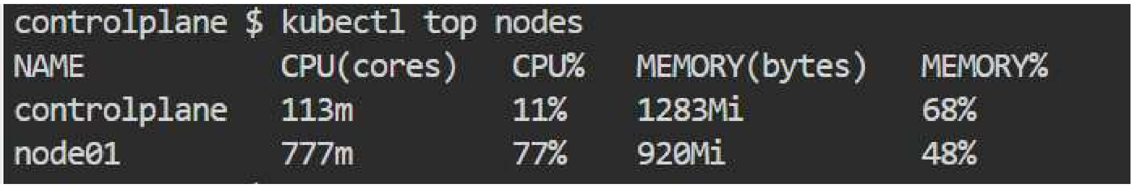
Kubectl top nodes



```
controlplane $ kubectl top pods
NAME                                CPU(cores)   MEMORY(bytes)
nginx-deployment-54c8694f64-g77qk   751m         6Mi
nginx-deployment-54c8694f64-vh4mc   0m           5Mi
controlplane $ kubectl top nodesa
error: unknown command "nodesa"
See 'kubectl top -h' for help and examples
controlplane $ kubectl top nodes
NAME           CPU(cores)   CPU%   MEMORY(bytes)   MEMORY%
controlplane   104m         10%    1285Mi          68%
node01         778m         77%    921Mi           48%
```

14. To check the nodes:

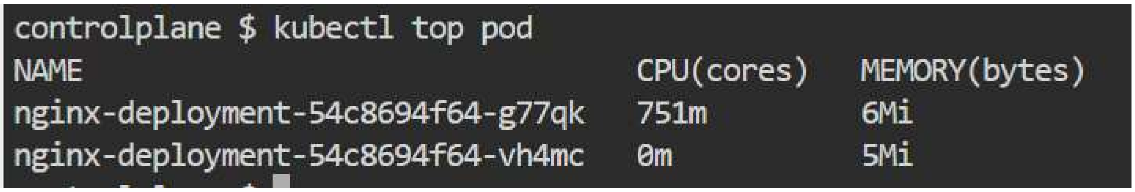
Kubectl top nodes



```
controlplane $ kubectl top nodes
NAME           CPU(cores)   CPU%   MEMORY(bytes)   MEMORY%
controlplane   113m         11%    1283Mi          68%
node01         777m         77%    920Mi           48%
```

15. To check the pod:

Kubectl top pod



```
controlplane $ kubectl top pod
NAME                                CPU(cores)   MEMORY(bytes)
nginx-deployment-54c8694f64-g77qk   751m         6Mi
nginx-deployment-54c8694f64-vh4mc   0m           5Mi
```