

Kubernetes Horizontal Pod Autoscaler Implementation

Project Overview

I developed a Kubernetes deployment with automatic scaling capabilities using Horizontal Pod Autoscaler (HPA) to demonstrate container orchestration and resource management skills.

Deployment Configuration :

- **Container** : nginx image with memory requests/limits
- **Availability** : Minimum 2 pod replicas for high availability

HPA Configuration :

- **Replicas** : 2-5 pods (min-max scaling range)
- **Scaling Metric** : Memory utilization at 75% threshold
- **Behavior** : Automatically scales up/down based on memory usage

Project Deliverables :

1. **YAML Manifests** : Deployment, HPA, and Service configurations
2. **Documentation** : HPA functionality, memory-based scaling mechanics, and resource management best practices
3. **Implementation Evidence** : kubectl outputs and live monitoring screenshots

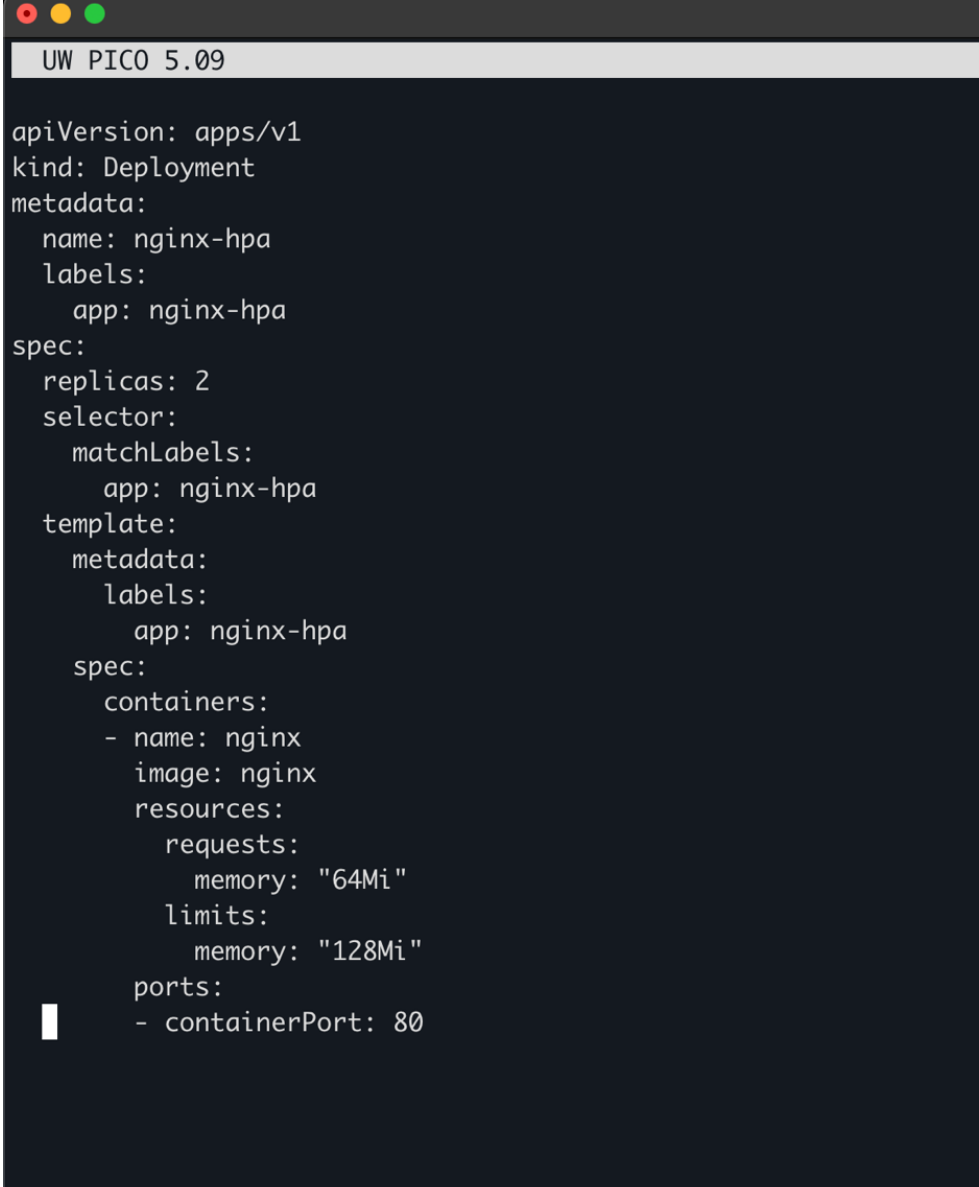
Key Learning Outcomes :

- Kubernetes resource management and optimization
- Automated scaling strategies for production workloads
- Cost-effective cloud resource allocation
- Performance monitoring and troubleshooting

Technologies Used : Kubernetes, Docker, nginx, HPA, YAML

1. Create YAML manifests for Deployment and **Horizontal Pod Autoscalers** :

deployment1.yml



```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-hpa
  labels:
    app: nginx-hpa
spec:
  replicas: 2
  selector:
    matchLabels:
      app: nginx-hpa
  template:
    metadata:
      labels:
        app: nginx-hpa
    spec:
      containers:
      - name: nginx
        image: nginx
        resources:
          requests:
            memory: "64Mi"
          limits:
            memory: "128Mi"
        ports:
        - containerPort: 80
```

- **Replicas** : The number of pods running at the start.

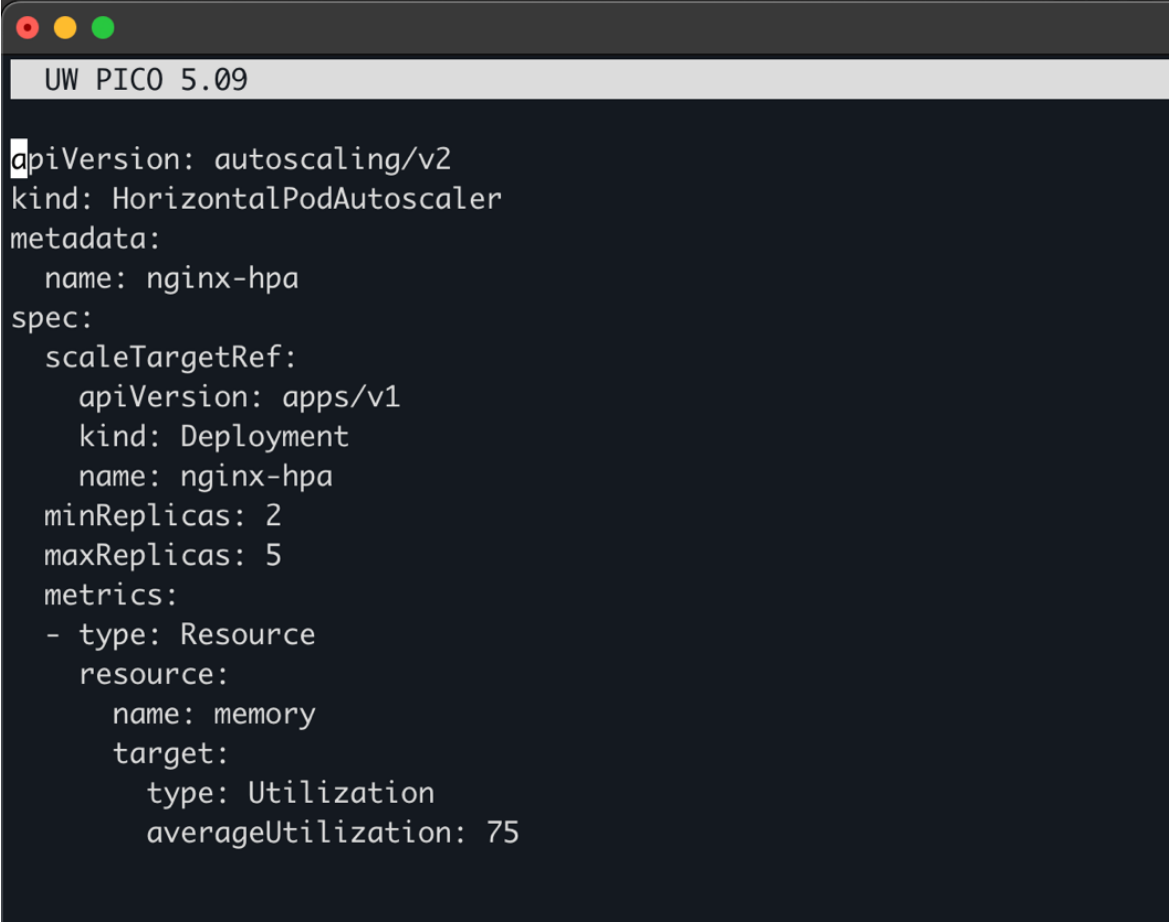
- **Resources.requests.memory** : "64Mi"

The minimum memory requested.

- **Resources.limits.memory** : "128Mi"

The maximum memory requested/allowed.

hpa.yml

A screenshot of a terminal window with a dark background. The window title bar shows three colored circles (red, yellow, green) and the text "UW PICO 5.09". The terminal displays the following YAML configuration for a HorizontalPodAutoscaler:

```
apiVersion: autoscaling/v2
kind: HorizontalPodAutoscaler
metadata:
  name: nginx-hpa
spec:
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: nginx-hpa
  minReplicas: 2
  maxReplicas: 5
  metrics:
  - type: Resource
    resource:
      name: memory
      target:
        type: Utilization
        averageUtilization: 75
```

How it works is simple :

- If the average memory usage is **>75%**, the **HPA** will add pods (scale up).
- If the average usage is **<75%**, the **HPA** will reduce pods (scale down).
- But it remains within the minimum limit of 2 pods, maximum 5 pods.

What is a **Horizontal Pod Autoscaler (HPA)** ?

Horizontal Pod Autoscaler (HPA) is a Kubernetes component that automatically increases or decreases the number of pods in a **Deployment/ReplicaSet** based on specific metrics, such as CPU or memory usage.

How **Horizontal Pod Autoscalers (HPA)** work :

- The **HPA** reads memory usage metrics from pods.

If the average memory usage of a pod is **>75%** of requests.memory, then :

- **HPA** will add pods (scale out)
- If memory usage is **<75%**, then :
- **HPA** will reduce pods (scale in), to a minimum (2 pods)

What happens if memory usage exceeds or falls short of the target :

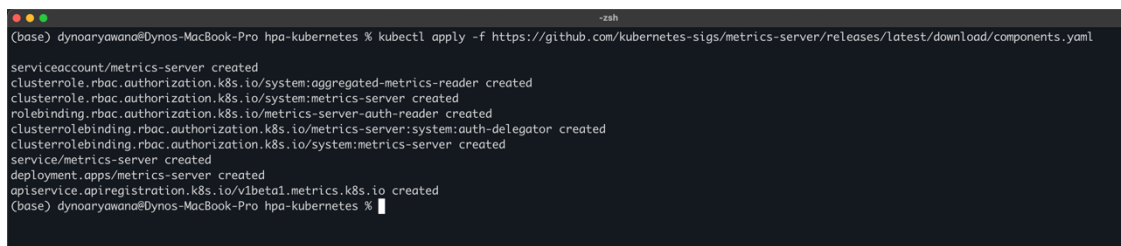
- **Memory usage >75%** : Increase the number of pods (no greater than maxReplicas)
- **Memory usage <75%** : Reduce the number of pods (no less than minReplicas)

Real-life example :

I have a news website that receives a lot of visitors in the morning :

- When traffic increases, memory usage increases, **HPA** will automatically scale out
- When traffic decreases, memory usage decreases, **HPA** will scale in, the benefit of which is resource and cost savings in a cloud environment.

Installing **metrics-server**

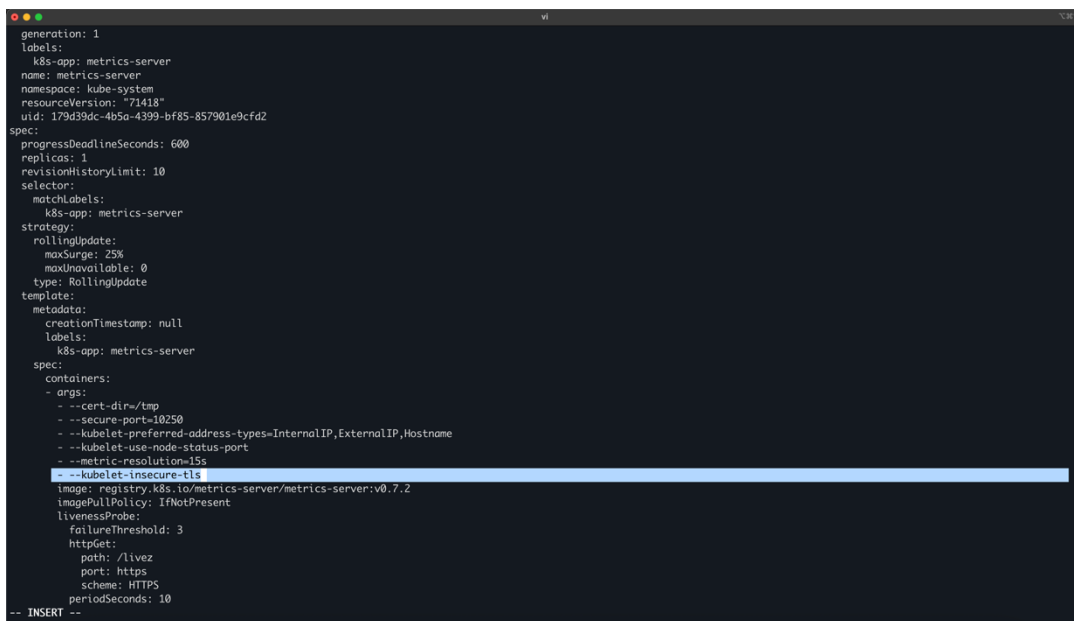


```
(base) dynoaryawana@Dynos-MacBook-Pro hpa-kubernetes % kubectl apply -f https://github.com/kubernetes-sigs/metrics-server/releases/latest/download/components.yaml
serviceaccount/metrics-server created
clusterrole.rbac.authorization.k8s.io/system:aggregated-metrics-reader created
clusterrole.rbac.authorization.k8s.io/system:metrics-server 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
(base) dynoaryawana@Dynos-MacBook-Pro hpa-kubernetes %
```

Modify **metrics-server** to work locally :

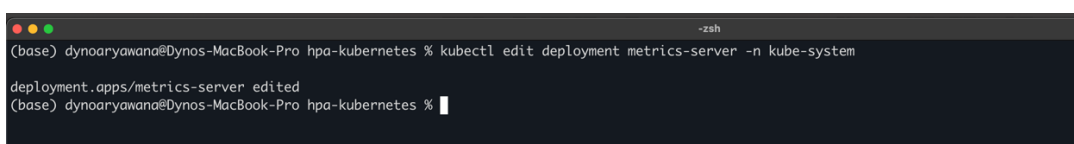
For **Horizontal Pod Autoscaler (HPA)** to function properly, Kubernetes requires a **metrics-server** capable of reading resource usage (such as CPU and memory) from pods. However, in on-premises environments like a Macbook running **Docker Desktop**, there's a problem because the kubelet uses a self-signed **TLS certificate**, which is invalid according to the **metrics-server**. To address this issue, modify the **metrics-server** deployment by adding the following argument to the args section :

- Add **--kubelet-insecure-tls** in this line:



```
generation: 1
labels:
  k8s-app: metrics-server
name: metrics-server
namespace: kube-system
resourceVersion: "71418"
uid: 179d39ac-4b5a-4399-bf85-857901e9cfa2
spec:
  progressDeadlineSeconds: 600
  replicas: 1
  revisionHistoryLimit: 10
  selector:
    matchLabels:
      k8s-app: metrics-server
  strategy:
    rollingUpdate:
      maxSurge: 25%
      maxUnavailable: 0
    type: RollingUpdate
  template:
    metadata:
      creationTimestamp: null
      labels:
        k8s-app: metrics-server
    spec:
      containers:
        - args:
            - --cert-dir=/tmp
            - --secure-port=10250
            - --kubelet-preferred-address-types=InternalIP,ExternalIP,Hostname
            - --kubelet-use-node-status-port
            - --metric-resolution=15s
            - --kubelet-insecure-tls
          image: registry.k8s.io/metrics-server/metrics-server:v0.7.2
          imagePullPolicy: IfNotPresent
          livenessProbe:
            failureThreshold: 3
            httpGet:
              path: /livez
              port: https
              scheme: HTTPS
            periodSeconds: 10
        -- INSERT --
```

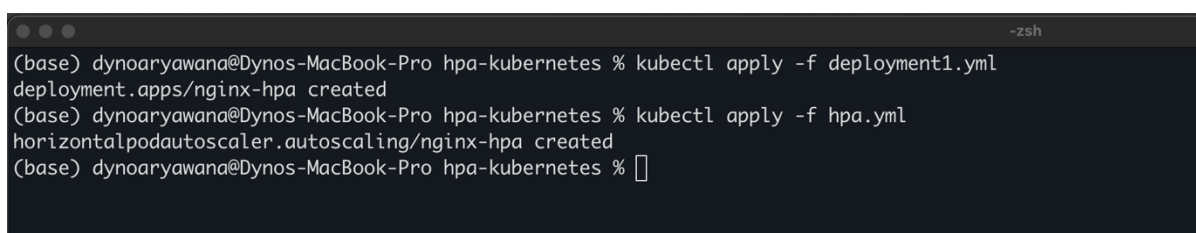
(save file and exit)



```
(base) dyoaryawana@Dynos-MacBook-Pro hpa-kubernetes % kubectl edit deployment metrics-server -n kube-system
deployment.apps/metrics-server edited
(base) dyoaryawana@Dynos-MacBook-Pro hpa-kubernetes %
```

This argument instructs metrics-server to **ignore TLS certificate verification when trying to access the kubelet API**, so it can still read metrics from the node even if the certificate is not trusted.

- Running YAML configuration with **kubectl apply -f <filename>**



```
(base) dyoaryawana@Dynos-MacBook-Pro hpa-kubernetes % kubectl apply -f deployment1.yml
deployment.apps/nginx-hpa created
(base) dyoaryawana@Dynos-MacBook-Pro hpa-kubernetes % kubectl apply -f hpa.yml
horizontalpodautoscaler.autoscaling/nginx-hpa created
(base) dyoaryawana@Dynos-MacBook-Pro hpa-kubernetes %
```

- Monitoring CPU metrics across multiple pods **kubectl top pod**

```
(base) dynoaryawana@Dynos-MacBook-Pro hpa-kubernetes % kubectl top pod
NAME                                CPU(cores)   MEMORY(bytes)
nginx-deployment-8f7498794-5pxkl    0m           7Mi
nginx-deployment-8f7498794-ssmzx    0m           6Mi
nginx-deployment-8f7498794-zn9n8    0m           7Mi
nginx-hpa-6bb9b4b45f-lb7m5         0m           7Mi
nginx-hpa-6bb9b4b45f-qlxlc         0m           7Mi
```

- Monitoring active Autoscalers HPA metrics running in real time with **-w**

```
(base) dynoaryawana@Dynos-MacBook-Pro hpa-kubernetes % kubectl get hpa -w
NAME      REFERENCE          TARGETS          MINPODS  MAXPODS  REPLICAS  AGE
nginx-hpa  Deployment/nginx-hpa  memory: 11%/75%    2         5         2         35m
```

THANK YOU