

QUES 4 ;

Managing Kubernetes with Azure Kubernetes Service (AKS), Creating and managing AKS clusters, Scaling and upgrading AKS clusters

SOLN:

In the previous QUESTIONS, I had already created different types of Kubernetes Services (ClusterIP, NodePort, LoadBalancer), deployed applications using ReplicaSet, ReplicationController, and Deployments, and managed them manually. In this task, I extended that learning by creating and managing a Kubernetes cluster using Azure Kubernetes Service (AKS), and explored how to scale and upgrade it efficiently in a managed cloud environment.

SO LETS FOCUS ON UPGRADING AND SCALING THE CLUSTERS THROUGH :

Cluster Autoscaler (Node Autoscaling) :

Scales the number of nodes (VMs) in the cluster when more pods need to be scheduled and no space is available.

- Automatically adds/removes nodes based on pod resource demands.
- Enabled in AKS using the `--enable-cluster-autoscaler` flag

It deals with node-level scaling, which is part of cluster management.

```
PS C:\WINDOWS\system32> kubectl get nodes
NAME                                STATUS    ROLES    AGE    VERSION
aks-agentpool-92868701-vmss000000  Ready    <none>    8h    v1.31.8
aks-agentpool-92868701-vmss000001  Ready    <none>    8h    v1.31.8
aks-userpool-92868701-vmss000000    Ready    <none>    8h    v1.31.8
aks-userpool-92868701-vmss000001    Ready    <none>    8h    v1.31.8
PS C:\WINDOWS\system32> kubectl get nodes
NAME                                STATUS    ROLES    AGE    VERSION
aks-agentpool-92868701-vmss000000  Ready    <none>    8h    v1.31.8
aks-agentpool-92868701-vmss000001  Ready    <none>    8h    v1.31.8
aks-userpool-92868701-vmss000000    Ready    <none>    8h    v1.31.8
aks-userpool-92868701-vmss000001    Ready    <none>    8h    v1.31.8
PS C:\WINDOWS\system32>
```

This is what we have deployed it till now

we have **two node pools**: agentpool and userpool

- Each pool has **2 nodes** and is built on **VMSS** (since the names have vmss, i.e., Virtual Machine Scale Set)

This means we **can enable cluster autoscaler** on your existing AKS cluster **without recreating it**.

1. Enable autoscaler on a node pool

Choose one pool to enable (e.g., agentpool):

az aks nodepool update \

--resource-group myResourceGroup \

--cluster-name myAKSCluster \

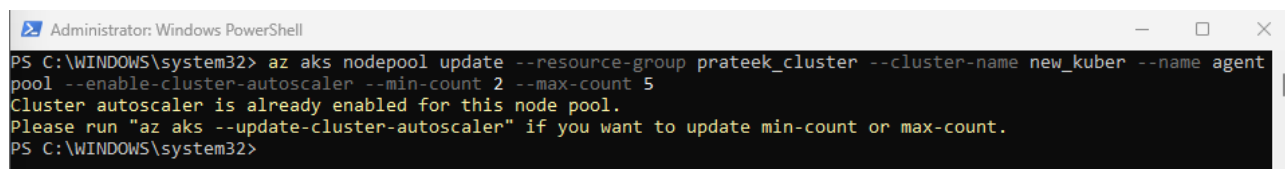
--name agentpool \

--enable-cluster-autoscaler \

--min-count 2 \

--max-count 5

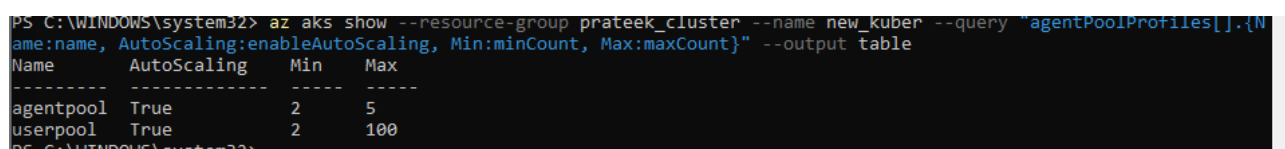
our autoscaler was enabled before formatting :



```
Administrator: Windows PowerShell
PS C:\WINDOWS\system32> az aks nodepool update --resource-group prateek_cluster --cluster-name new_kuber --name agentpool --enable-cluster-autoscaler --min-count 2 --max-count 5
Cluster autoscaler is already enabled for this node pool.
Please run "az aks --update-cluster-autoscaler" if you want to update min-count or max-count.
PS C:\WINDOWS\system32>
```

Check if autoscaler is enabled

az aks show --resource-group prateek_cluster --name new_kuber --query "agentPoolProfiles[0].{Name:name, AutoScaling:enableAutoScaling, Min:minCount, Max:maxCount}" --output table

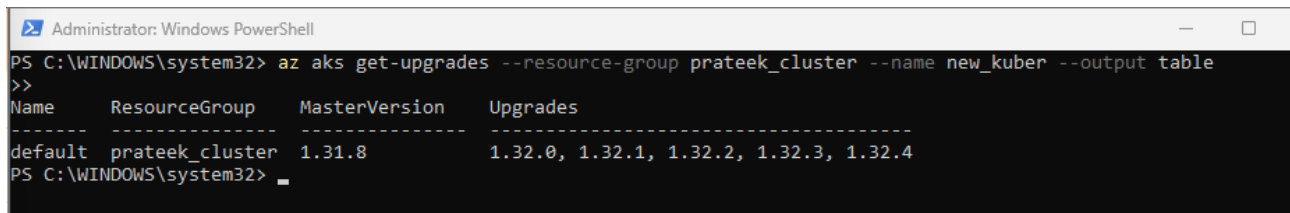


```
PS C:\WINDOWS\system32> az aks show --resource-group prateek_cluster --name new_kuber --query "agentPoolProfiles[0].{Name:name, AutoScaling:enableAutoScaling, Min:minCount, Max:maxCount}" --output table
Name      AutoScaling  Min  Max
-----
agentpool  True         2    5
userpool   True         2    100
PS C:\WINDOWS\system32>
```

NOW LETS US UPGRADE THE CLUSTER IF NEEDED AND THE VERSIONS ARE AVAILABLE FOR THE SAME :

Step 1: Check Available Upgrade Versions

az aks get-upgrades --resource-group prateek_cluster --name new_kuber --output table



```
Administrator: Windows PowerShell
PS C:\WINDOWS\system32> az aks get-upgrades --resource-group prateek_cluster --name new_kuber --output table
>>
Name      ResourceGroup  MasterVersion  Upgrades
-----
default   prateek_cluster  1.31.8         1.32.0, 1.32.1, 1.32.2, 1.32.3, 1.32.4
PS C:\WINDOWS\system32> _
```

Step 2: Upgrade Your Cluster

az aks upgrade --resource-group prateek_cluster --name new_kuber --kubernetes-version 1.32.4 --yes

During upgrade:

- The control plane is upgraded first.
- Node pools are then cordoned and upgraded one by one.
- No data loss, but short downtime might happen.

In a real-world scenario like **an e-commerce website during festive sales or flash deals (e.g., Flipkart Big Billion Days or Amazon Great Indian Festival)**, user traffic spikes drastically in a short time.

Using **Cluster Autoscaler**, the AKS cluster can automatically add more nodes to handle the surge in user requests, preventing app crashes or latency. Once traffic reduces, nodes scale down automatically — saving infrastructure cost.

On the other hand, **upgrading the AKS cluster** ensures that the platform runs on the latest Kubernetes version with enhanced security patches, better performance, and support for new features — which is critical for large-scale production systems where downtime or vulnerabilities could cause massive losses.

“A practical example is e-commerce platforms during flash sales, where autoscaling ensures reliability during peak loads, and cluster upgrades provide long-term security and performance in production ”