

**Lab Manual- application autoscaling with Kubernetes Event-driven Autoscaling (KEDA) add-on**

**Prepared for**:

**Date:** 18th Dec 2023

**Prepared by:**

Document Name: Lab Manual **Document Number** AZLabn916

**Contributor:**

Contents

[1. Objective 3](#_Toc159493866)

[1. Components: 3](#_Toc159493867)

[2. Relationships: 3](#_Toc159493868)

[2. KEDA SAMPLE ARCHITECTURE 3](#_Toc159493869)

[3. AKS Cluster with single node 4](#_Toc159493870)

[4. Install KEDA in Kubernetes using Helm Charts 5](#_Toc159493871)

[5. Create Azure Service Bus and Queue 6](#_Toc159493872)

[6. Replace the primaryConnectionString in the NGNIX-Deploy YAML file and Deploy the File 9](#_Toc159493873)

[7. Add and remove messages in the queue and watch the number of pods 11](#_Toc159493874)

# Objective

Kubernetes Event-driven Autoscaling (KEDA) is a single-purpose and lightweight component that strives to make application autoscaling simple and is a CNCF (Cloud Native Computing Foundation ) Graduate project.

It applies event-driven autoscaling to scale your application to meet demand in a sustainable and cost-efficient manner with scale-to-zero.

<https://keda.sh/docs/2.13/concepts/>

## Components:

* **Event Source:** This is the external system generating events, such as a message queue (Kafka, RabbitMQ), cloud platform event stream (AWS SQS, Azure Event Hubs), database change logs, or custom sources.
* **Scaler:** This component translates events from the source into metrics that Kubernetes understands. KEDA comes with many built-in scalers for popular sources, and you can create custom ones.
* **Metrics Adapter:** This component converts the scaler's custom metrics into a format compatible with the Horizontal Pod Autoscaler (HPA).
* **Horizontal Pod Autoscaler (HPA):** This Kubernetes built-in component scales pods based on specified metrics.
* **Kubernetes Deployment/Job:** The workload that KEDA scales based on event-driven triggers.
* **keda-operator:** This operator manages the KEDA deployment and ensures all components are running correctly.

## Relationships:

* The event source sends events to the scaler.
* The scaler translates events into custom metrics and sends them to the metrics adapter.
* The metrics adapter converts the custom metrics into HPA-compatible format.
* The HPA uses the translated metrics to scale the Kubernetes deployment/job up or down.
* The keda-operator manages the entire process and ensures smooth communication between components.

# KEDA SAMPLE ARCHITECTURE

+-------------------+ +----------------+ +-------------------+

| Event Source | -----> | Scaler | -----> | Metrics Adapter |

| (Kafka, RabbitMQ, | | (Translates | | (Converts metrics |

| etc.) | | events to | | to HPA format) |

+-------------------+ | custom metrics) | +-------------------+

|

+-------------------+

| Horizontal Pod |

| Autoscaler (HPA) |

+-------------------+

|

+-------------------+

| Kubernetes |

| Deployment/Job |

+-------------------+

|

+-------------------+

| keda-operator |

+-------------------+

This diagram represents the core components of KEDA and their interactions:

* **Event Source:** Generates events (e.g., messages in a queue).
* **Scaler:** Translates events into custom metrics specific to the source.
* **Metrics Adapter:** Converts custom metrics into a format understood by HPA.
* **Horizontal Pod Autoscaler (HPA):** Scales the deployment/job based on the provided metrics.
* **Kubernetes Deployment/Job:** The workload being scaled by KEDA.
* **keda-operator:** Manages the KEDA deployment and ensures everything runs smoothly.

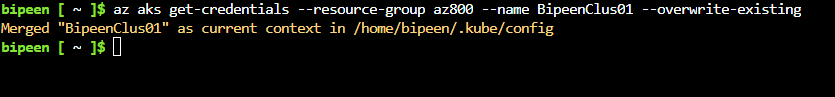
# AKS Cluster with single node

creates an AKS cluster named  in the resource group

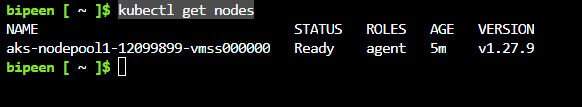
az aks create --resource-group az800 --name BipeenClus01 --node-count 1 --generate-ssh-keys



az aks get-credentials --resource-group az800 --name BipeenClus01 --overwrite-existing



kubectl get nodes

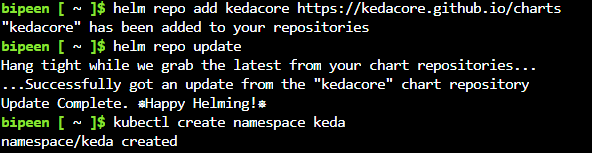


# Install KEDA in Kubernetes using Helm Charts

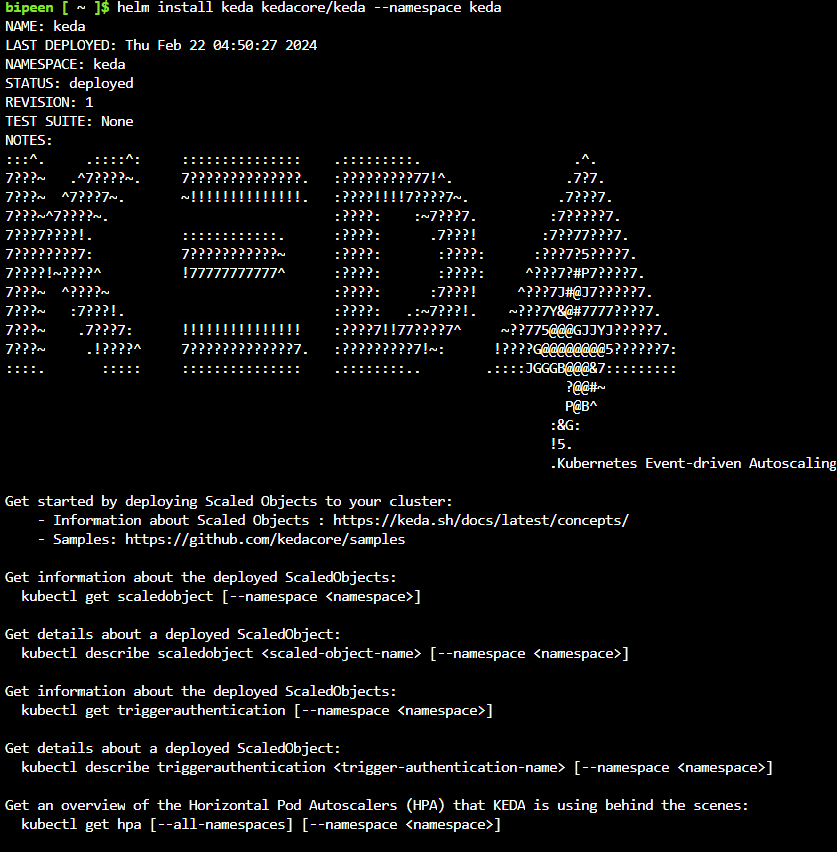
helm repo add kedacore https://kedacore.github.io/charts

helm repo update

kubectl create namespace keda



helm install keda kedacore/keda --namespace keda



# Create Azure Service Bus and Queue

We Perform this activity in Azuree PowerShell . Open duplicate Portal and launch azure PowerShell

**# Create Azure Service Bus Variable in PowerShell**

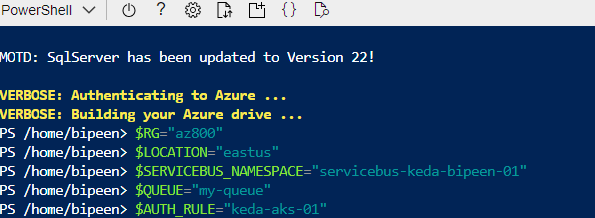
$RG="az800"

$LOCATION="eastus"

$SERVICEBUS\_NAMESPACE="servicebus-keda-bipeen-01"

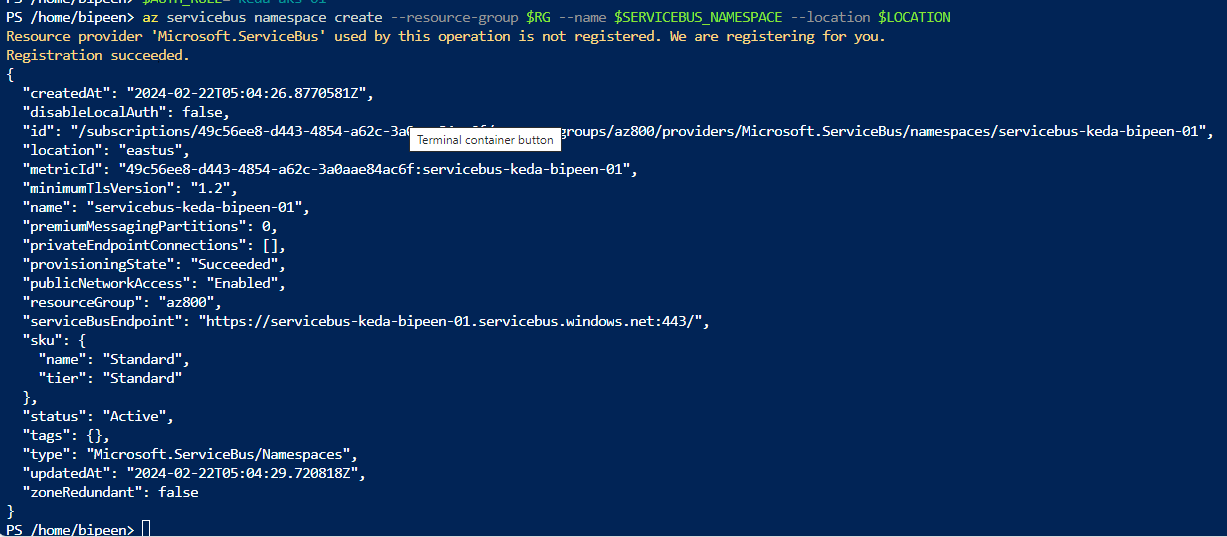
$QUEUE="my-queue"

$AUTH\_RULE="keda-aks-01"



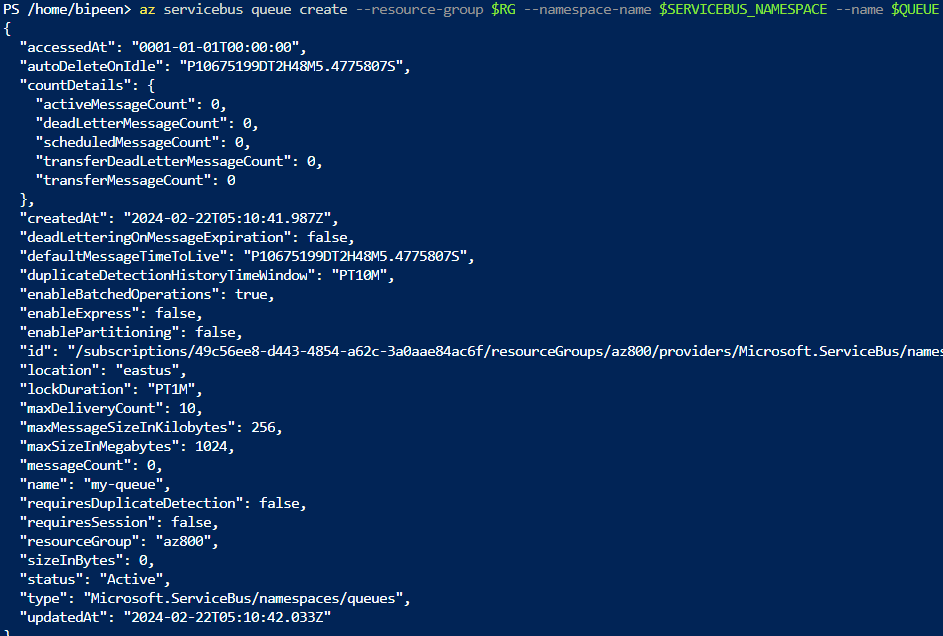
**# Create Azure Service Bus Namespace**

az servicebus namespace create --resource-group $RG --name $SERVICEBUS\_NAMESPACE --location $LOCATION



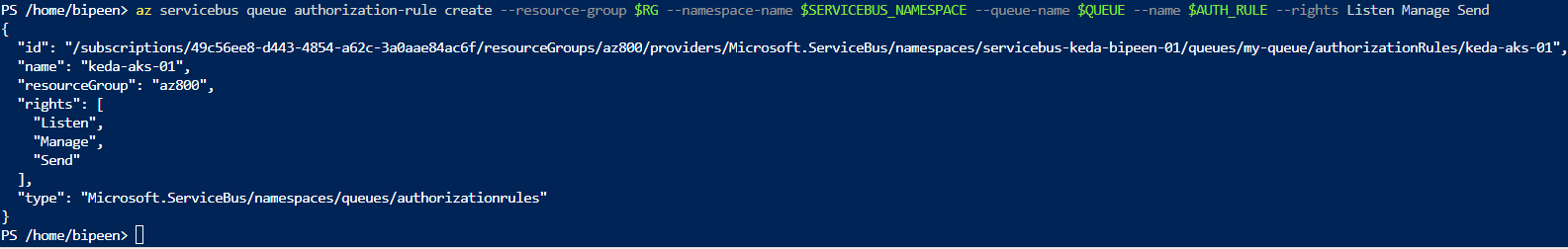
**# Create Queue**

az servicebus queue create --resource-group $RG --namespace-name $SERVICEBUS\_NAMESPACE --name $QUEUE



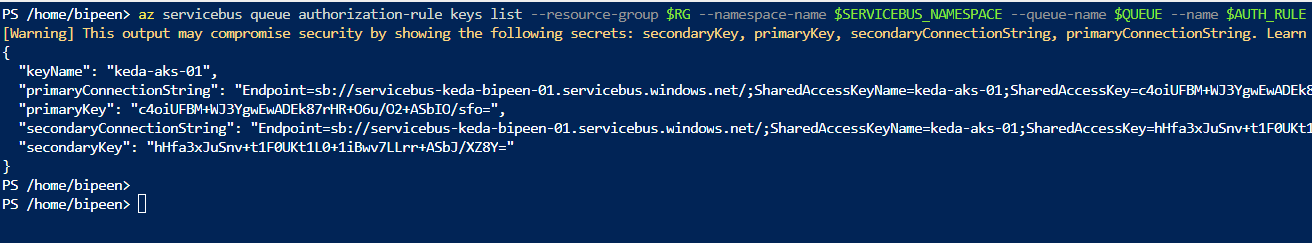
**# create Authorisation Rule for the Queue**

az servicebus queue authorization-rule create --resource-group $RG --namespace-name $SERVICEBUS\_NAMESPACE --queue-name $QUEUE --name $AUTH\_RULE --rights Listen Manage Send

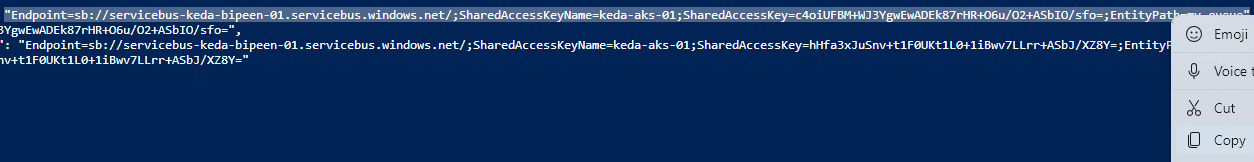
****

**# list Authorisation Rule keys**

az servicebus queue authorization-rule keys list --resource-group $RG --namespace-name $SERVICEBUS\_NAMESPACE --queue-name $QUEUE --name $AUTH\_RULE



# Replace the primaryConnectionString in the NGNIX-Deploy YAML file and Deploy the File

****

**Ngnix-Deploy.yaml**

�apiVersion: apps/v1

kind: Deployment

metadata:

  name: nginx-deployment

  labels:

    app: nginx-deployment

spec:

  replicas: 3

  selector:

    matchLabels:

      app: nginx-deployment

  strategy: {}

  template:

    metadata:

      labels:

        app: nginx-deployment

    spec:

      containers:

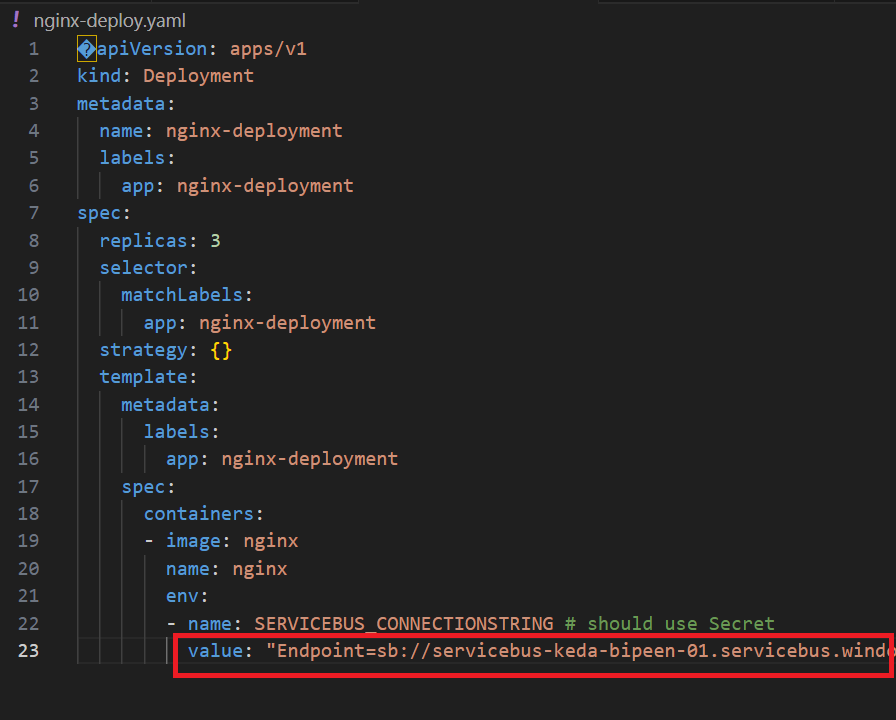
      - image: nginx

        name: nginx

        env:

        - name: SERVICEBUS\_CONNECTIONSTRING # should use Secret

          value: "Endpoint=sb://servicebus-keda-bipeen-01.servicebus.windows.net/;SharedAccessKeyName=keda-aks-01;SharedAccessKey=c4oiUFBM+WJ3YgwEwADEk87rHR+O6u/O2+ASbIO/sfo=;EntityPath=my-queue" # to be replaced

****

**scaledObject.yaml**

**‘**

apiVersion: keda.sh/v1alpha1

kind: ScaledObject

metadata:

  name: azure-servicebus-queue-scaledobject

  namespace: default

spec:

  scaleTargetRef:

    name: nginx-deployment # must be in the same namespace as the ScaledObject

  pollingInterval: 3  # Optional. Default: 30 seconds

  cooldownPeriod: 2 # Optional. Default: 300 seconds

  minReplicaCount: 0 # Optional. Default: 0

  maxReplicaCount: 10 # Optional. Default: 100

  triggers:

  - type: azure-servicebus

    metadata:

      queueName: my-queue # to be replaced

      messageCount: "3" # Optional. Count of messages to trigger scaling on. Default: 5 messages

      connectionFromEnv: SERVICEBUS\_CONNECTIONSTRING # Name of the environment variable your deployment uses to get the connection string of the Azure Service Bus namespace. (Optional) This must be a connection string for a queue itself, and not a namespace level (e.g. RootAccessPolicy) connection string [#215](https://github.com/kedacore/keda/issues/215)

      # Required: queueName OR topicName and subscriptionName

      # topicName: functions-sbtopic

      # subscriptionName: sbtopic-sub1

      # Optional, required when pod identity is used

      # namespace: service-bus-namespace

      # Optional, can use TriggerAuthentication as well

      # Optional

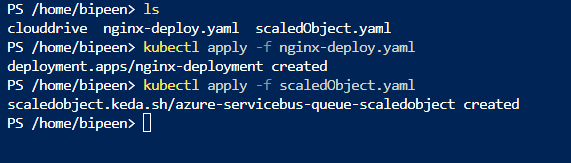
      # cloud: Private # Optional. Default: AzurePublicCloud

      # endpointSuffix: servicebus.airgap.example # Required when cloud=Private

**# deploy the app**

kubectl apply -f nginx-deploy.yaml

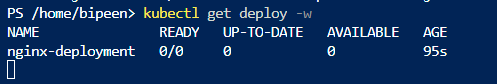
kubectl apply -f scaledObject.yaml

****

# Add and remove messages in the queue and watch the number of pods

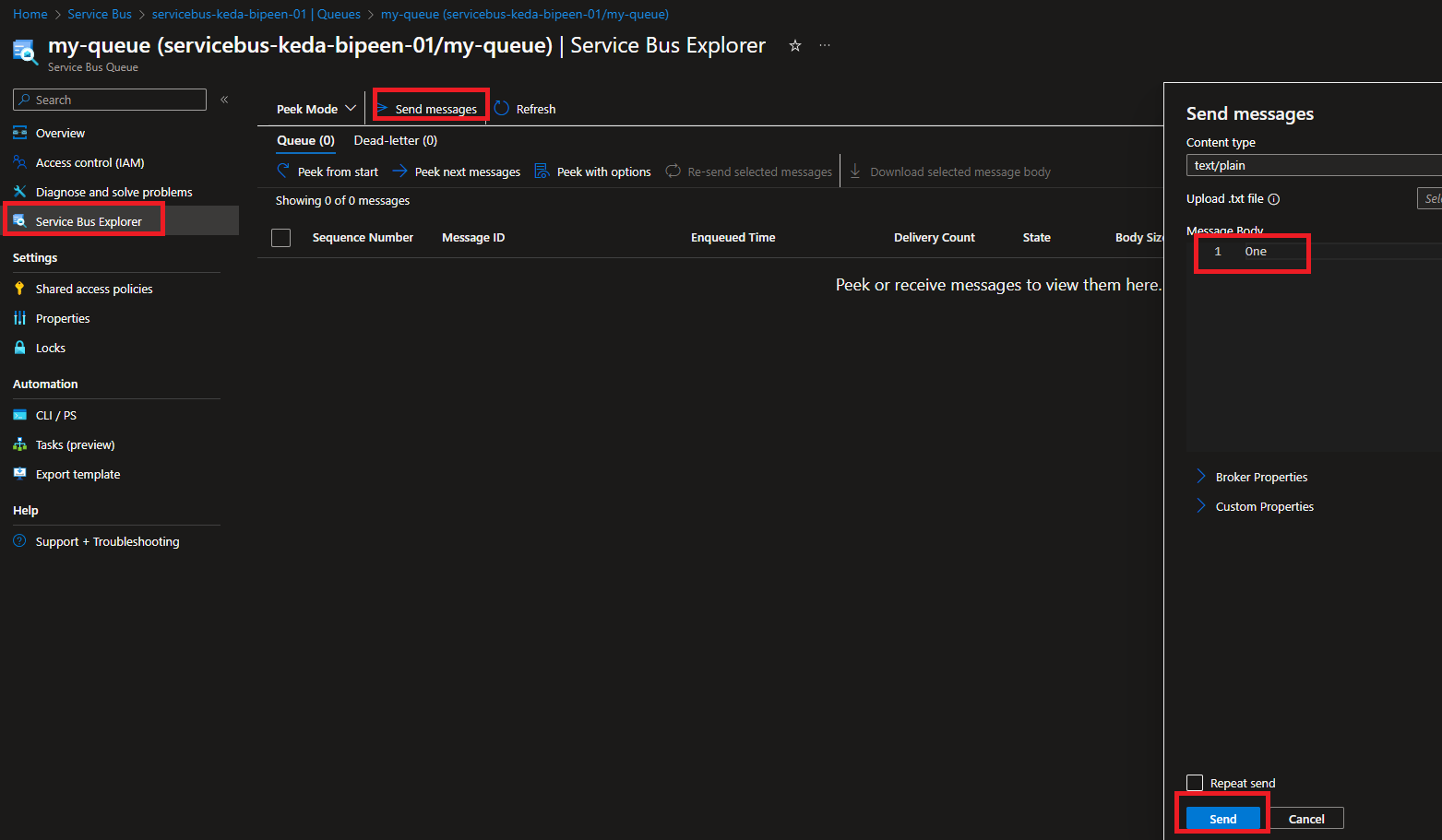
**# watch the deployment number of pods**

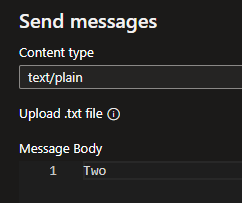
kubectl get deploy -w

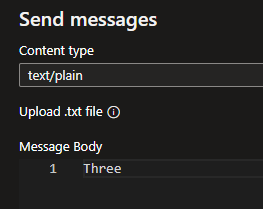
****

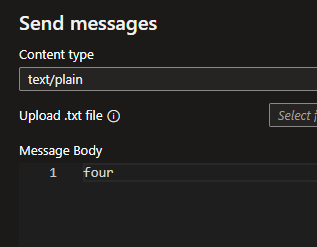
**Add Message in Queue**

Add 4 Message :

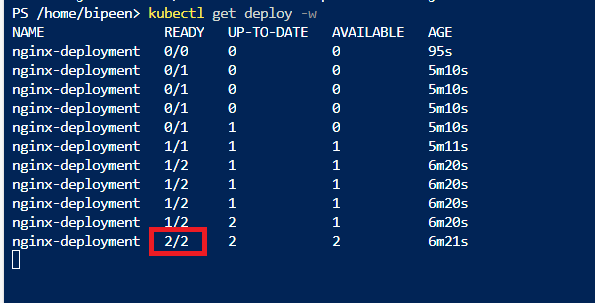




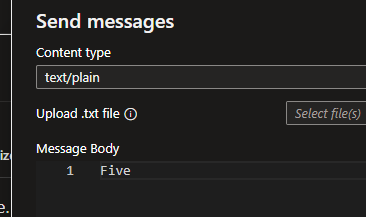


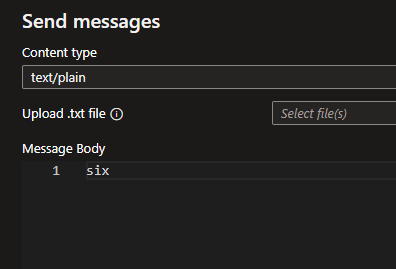


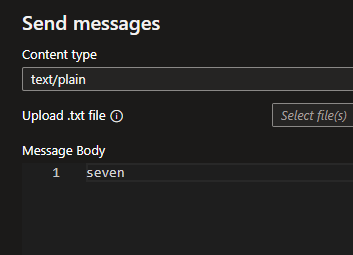
See the New PODs get created once the total number of message greater than **3**



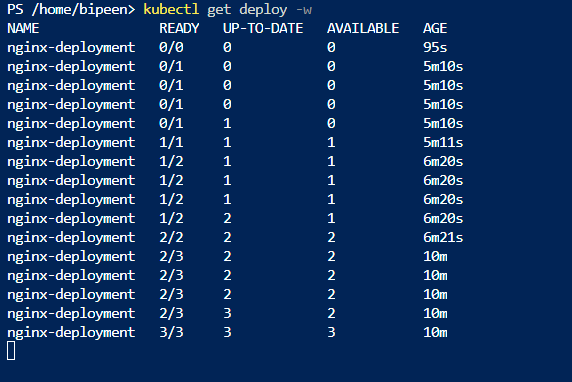
Add 3 addional Message



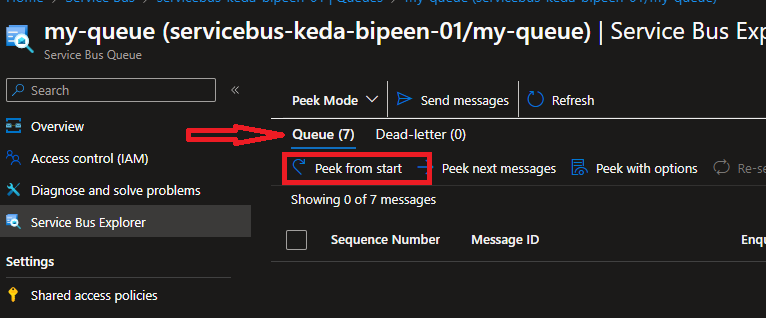


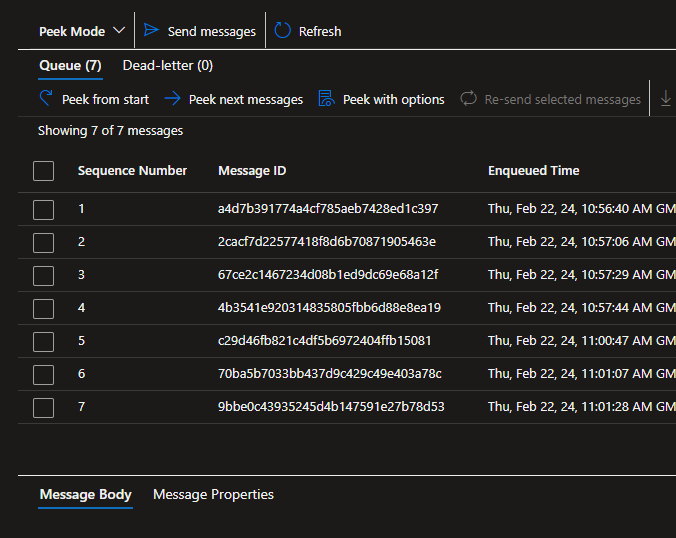


**Now You can see 3rd Pod get Launched**

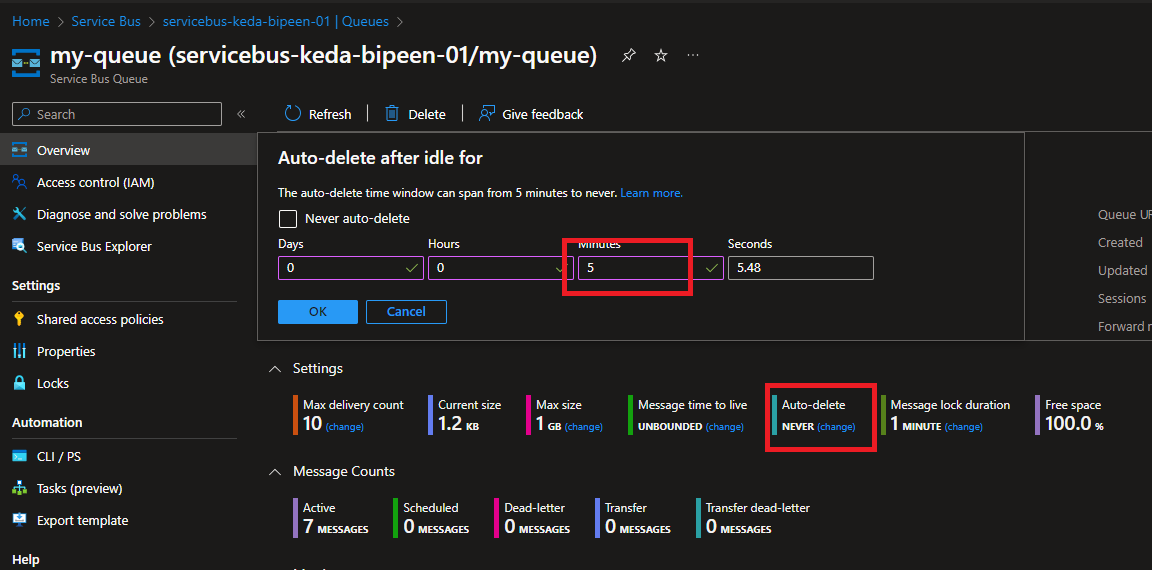


Now Peak the message and POD start getting Scale down



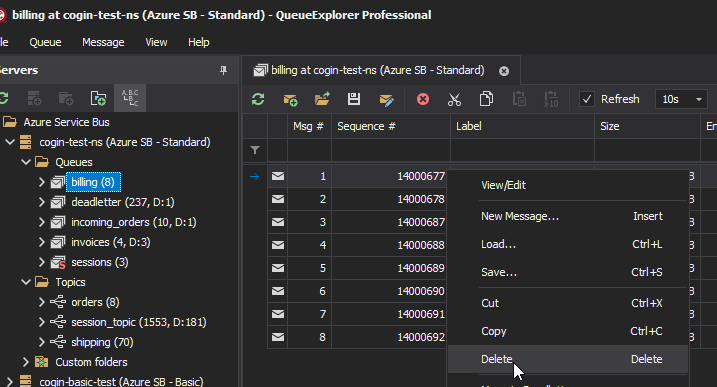


Change Autodelete setting for five minute



Or use Query Explorer

<https://www.cogin.com/QueueExplorer/azure-service-bus/>



kubectl get all -n keda

