Lab 3 - Installing KEDA and Setting Up Cron Scaler

Overview:

This lab provides practical experience of installing KEDA in a Kubernetes environment. We will make use of the Cron ScaledObject to scale the application based on a time schedule.

Prerequisites:

Kubernetes cluster with metric server installed as per Lab 1.

Exercise 3.1: Install KEDA Using Helm Chart

In this exercise, we will install KEDA using a Helm chart.

1. Install KEDA using Helm.

Execute the following commands, to get started:

```
helm repo add kedacore https://kedacore.github.io/charts
helm repo update
helm upgrade -i keda kedacore/keda --namespace keda --create-namespace
```

2. Verify KEDA pods are running in the cluster, using the command below:

```
kubectl get deployment -n keda
```

```
NAME                              READY   UP-TO-DATE   AVAILABLE   AGE
keda-admission-webhooks           1/1     1            1           13m
keda-operator                     1/1     1            1           13m
keda-operator-metrics-apiserver   1/1     1            1           13m
```

3. Create a deployment, using the command below:

```
kubectl create deploy myapp --image nginx --replicas=2
```

4. Verify the deployment:

```
kubectl get deployments

NAME    READY   UP-TO-DATE   AVAILABLE   AGE
myapp   2/2     2            2           37s
```

5. Create a `cron.yaml` file with contents below. This file defines a Kubernetes ScaledObject for your sample application.

```
apiVersion: keda.sh/v1alpha1
kind: ScaledObject
metadata:
  name: cron-scaledobject
  namespace: default
spec:
  scaleTargetRef:
    name: myapp
  triggers:
  - type: cron
    metadata:
      timezone: Asia/Kolkata
      start: 30 * * * *
      end: 45 * * * *
      desiredReplicas: "10"
```

This specification describes the **cron** trigger that scales workloads in/out based on a Cron schedule.

- **timezone** - One of the acceptable values from the Internet Assigned Numbers Authority (IANA) Time Zone Database. The list of timezones can be found in the Wikipedia article "List of tz database time zones".
- **start** - Cron expression indicating the start of the Cron schedule.
- **end** - Cron expression indicating the end of the Cron schedule.
- **desiredReplicas** - Number of replicas to which the resource has to be scaled between the start and end of the Cron schedule.

***NOTE**: Start/end supports "Linux format Cron" (MIN for minutes, HOUR for hours, DOM for day of the month, MON for month, and DOW for day of the week).*

6. Create the ScaledObject using the below command:

```
kubectl apply -f cron.yaml
```

7. Verify the ScaledObject:

```
kubectl get scaledobject.keda.sh

NAME                SCALETARGETKIND      SCALETARGETNAME   MIN   MAX   TRIGGERS
AUTHENTICATION   READY   ACTIVE   FALLBACK   PAUSED    AGE
cron-scaledobject   apps/v1.Deployment   myapp                       cron
True    False    Unknown    Unknown   75s
```

8. The ScaledObject will scale the application pods to the desired number of replicas when the schedule triggers.

```
watch kubectl get all
```

```
NAME                            READY   STATUS    RESTARTS   AGE
pod/myapp-747784665b-58vq7      1/1     Running   0          7m38s
pod/myapp-747784665b-5k8mg      1/1     Running   0          7m53s
pod/myapp-747784665b-92c47      1/1     Running   0          7m23s
pod/myapp-747784665b-cc298      1/1     Running   0          7m53s
pod/myapp-747784665b-fksp8      1/1     Running   0          7m38s
pod/myapp-747784665b-lhphb      1/1     Running   0          7m23s
pod/myapp-747784665b-mz2jd      1/1     Running   0          7m38s
pod/myapp-747784665b-tcvqq      1/1     Running   0          7m38s
pod/myapp-747784665b-vggtd      1/1     Running   0          7m53s
pod/myapp-747784665b-zdhv9      1/1     Running   0          7m53s

NAME                 TYPE        CLUSTER-IP    EXTERNAL-IP   PORT(S)   AGE
service/kubernetes   ClusterIP   10.96.0.1     <none>        443/TCP   49m

NAME                     READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/myapp    10/10   10           10          13m

NAME                                DESIRED   CURRENT   READY   AGE
replicaset.apps/myapp-747784665b    10        10        10      13m

NAME                                                                  REFERENCE
TARGETS     MINPODS   MAXPODS   REPLICAS   AGE
horizontalpodautoscaler.autoscaling/keda-hpa-cron-scaledobject   Deployment/myapp
1/1 (avg)   1         100       10         10m

press ctrl + c to exit
```

The Cron scaler allows you to define a time range in which you want to scale your workloads up and down. When the time window starts, it will scale from the minimum number of replicas to the desired number of replicas based on your configuration.

In this lab exercise, we made use of a simple Cron scaler, which scales the applications based on a specific schedule. You can further explore different scalers available.