How does the Horizontal Pod Autoscaler work?

The Horizontal Pod Autoscaler automatically scales the number of pods in a replication controller, deployment or replica set based on observed CPU utilization (or, with custom metrics support, on some other application-provided metrics).

a . Run & expose the Application

Create a new deployment with the below command and replace `<your-deployment-name>` with unique deployment name.

1. $ kubectl run <your-deployment-name> --image=asyed755/dell-load-testing:latest --requests=cpu=200m

2. $ kubectl expose deployment <your-deployment-name> --type=NodePort --port=80

Please use asyed755/dell-load-testing:latest **Image for the LAB.**

b . Create Horizontal Pod Autoscaler

Now that the server is running, we will create the autoscaler using kubectl autoscale.
The following command will create a Horizontal Pod Autoscaler that maintains between 1 and 10 replicas of the Pods.

HPA will increase and decrease the number of replicas (via the deployment) to maintain an average CPU utilization across all Pods of 50%

1. $ kubectl autoscale deployment <your-deployment-name> --cpu-percent=20 --min=1 --max=10

Participant need to wait for 1-2 minutes before running the below command

2. $ kubectl get hpa

```
arshad@arshad-Latitude-E6330:~$ kubectl get hpa
NAME              REFERENCE                    TARGETS   MINPODS   MAXPODS   REPLICAS   AGE
arshad            Deployment/arshad            0%/20%    1         10        1          40s
rajni-deployment  Deployment/rajni-deployment  0%/20%    1         20        1          28m
```

Please note that the current CPU consumption is 0% as we are not sending any requests to the server (the CURRENT column shows the average across all the pods controlled by the corresponding deployment).

c . Increase load

Now, we will see how the autoscaler reacts to increased load. We will start a container, and send an infinite loop of queries to the ***<your-service-name>*** service (please run it in a different terminal):

1. $ kubectl run -i --tty load-generator-<your-name> --image=busybox /bin/sh

{

Note: If you want to login to an already deployed load generated.

Run the below command to get the load-generator pod name
$ kubecetl get pods --all-namespces # to get the load-generator pod name

$ kubectl attach load-generator-7bbbb4fdd4-g697m -c load-generator-your-name -i -t

Where **load-generator-7bbbb4fdd4-g697m** is the pod name and ***load-generator-your-name*** *is the load generator deployment name.*

}

Hit enter **for** command prompt

2. $ while true; do wget -q -O- http://<node-public-ip>:NodePort/ **; done**

To get the Public IP of the Worker-Node where the POD has been deployed
3. $ kubectl get services <your-service-name>

```
NAME     TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S     )       AGE
arshad   NodePort  100.66.70.157   <none>        80:32587/TCP   3m
```

Check the Node details where the POD has been deployed
4. $ kubectl get pod -o wide

```
NAME                      READY   STATUS    RESTARTS   AGE   IP              NODE
arshad-5fdff48b48-7r4pg         1/1     Running   0          2m    100.96.4.43   ip-172-20-55-125.ec2.internal
```

```
arshad-5fdff48b48-gg7j4   1/1   Running   0   2m   100.96.4.44   ip-172-20-55-125.ec2.internal
rajni-deployment-d667     1/1   Running   0   3h   100.96.4.13   ip-172-20-55-125.ec2.internal
```

Now run the below command (replace the load-balancer url)

On the load generator terminal
Within a minute or so, we should see the higher CPU load by executing:

Meantime open **two more terminals** and run the below commands
On terminal one run the below command

4. $ watch -n 1 kubectl get hpa

On terminal two run the below command

5. $ watch -n 1 kubectl get pods

It should look like below

6. $ kubectl get hpa

```
NAME   REFERENCE                  TARGET          CURRENT   MINPODS   MAXPODS   REPLICAS   AGE
arshad   Deployment/arshad/scale   305% / 20%      305%      1         10        1          3m
```

7. $ kubectl get deployment <your-deployment

```
NAME      DESIRED   CURRENT   UP-TO-DATE   AVAILABLE   AGE
arshad    7         7         7            7           19m
```

Login to the Dashboard and you will be able to observe that the application has started scaling horizontally.

Note. Once the cpu (or any other metric) goes below the defined threshold, the HPA will automatically scale down the application to minimum.