## Autoscaling in Kubernetes

## Step One: Run & expose php-apache server

To demonstrate Horizontal Pod Autoscaler we will use a custom docker image based on the php-apache image. The Dockerfile can be found [here](https://kubernetes.io/docs/user-guide/horizontal-pod-autoscaling/image/Dockerfile). It defines an [index.php](https://kubernetes.io/docs/user-guide/horizontal-pod-autoscaling/image/index.php) page which performs some CPU intensive computations.

First, we will start a deployment running the image and expose it as a service:

**$ kubectl run php-apache --image=gcr.io/google\_containers/hpa-example --requests=cpu=200m --expose --port=80**

**service "php-apache" created**

**deployment "php-apache" created**

## Step Two: Create Horizontal Pod Autoscaler

Now that the server is running, we will create the autoscaler using [kubectl autoscale](https://github.com/kubernetes/kubernetes/blob/master/docs/user-guide/kubectl/kubectl_autoscale.md). The following command will create a Horizontal Pod Autoscaler that maintains between 1 and 10 replicas of the Pods controlled by the php-apache deployment we created in the first step of these instructions. Roughly speaking, HPA will increase and decrease the number of replicas (via the deployment) to maintain an average CPU utilization across all Pods of 50% (since each pod requests 200 milli-cores by [kubectl run](https://github.com/kubernetes/kubernetes/blob/master/docs/user-guide/kubectl/kubectl_run.md), this means average CPU usage of 100 milli-cores). See [here](https://git.k8s.io/community/contributors/design-proposals/autoscaling/horizontal-pod-autoscaler.md#autoscaling-algorithm) for more details on the algorithm.

**$ kubectl autoscale deployment php-apache --cpu-percent=50 --min=1 --max=10**

**deployment "php-apache" autoscaled**

We may check the current status of autoscaler by running:

**$ kubectl get hpa**

**NAME REFERENCE TARGET MINPODS MAXPODS REPLICAS AGE**

**php-apache Deployment/php-apache/scale 0% / 50% 1 10 1 18s**

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).

## Step Three: 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 php-apache service (please run it in a different terminal):

**$ kubectl run -i --tty load-generator --image=busybox /bin/sh**

**Hit enter for command prompt**

**$ while true; do wget -q -O- http://php-apache; done**

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

**$ kubectl get hpa**

**NAME REFERENCE TARGET CURRENT MINPODS MAXPODS REPLICAS AGE**

**php-apache Deployment/php-apache/scale 305% / 50% 305% 1 10 1 3m**

Here, CPU consumption has increased to 305% of the request. As a result, the deployment was resized to 7 replicas:

**$ kubectl get deployment php-apache**

**NAME DESIRED CURRENT UP-TO-DATE AVAILABLE AGE**

**php-apache 7 7 7 7 19m**

**Note** Sometimes it may take a few minutes to stabilize the number of replicas. Since the amount of load is not controlled in any way it may happen that the final number of replicas will differ from this example.

## Step Four: Stop load

We will finish our example by stopping the user load.

In the terminal where we created the container with **busybox** image, terminate the load generation by typing **<Ctrl> + C**.

Then we will verify the result state (after a minute or so):

**$ kubectl get hpa**

**NAME REFERENCE TARGET MINPODS MAXPODS REPLICAS AGE**

**php-apache Deployment/php-apache/scale 0% / 50% 1 10 1 11m**

**$ kubectl get deployment php-apache**

**NAME DESIRED CURRENT UP-TO-DATE AVAILABLE AGE**

**php-apache 1 1 1 1 27m**

Here CPU utilization dropped to 0, and so HPA autoscaled the number of replicas back down to 1.

**Note** autoscaling the replicas may take a few minutes.

## Deleting a deployment

Delete the deployment by name:

**kubectl delete deployment nginx-deployment**