HPA

Autoscaling is a method that dynamically scales up / down the number of computing resources that are being allocated to your application based on its needs. For example, you might have a situation where the load on your website increases only at the end of every month. You might need additional web servers to handle the load at the end of every month. But what about the rest of the days, your servers sit idle and your monthly cloud bill also increases. With Autoscaling enabled the number of servers can be increased/decreased based on the load and the number of users. While the world moving towards the Kubernetes era, autoscaling plays a major role in the scalability of containerized applications. In this article, we will try to understand the various types of autoscalers available in Kubernetes and try to understand the best suitable autoscaler for application in Kubernetes.

Installing metrics server

For Horizontal Pod Autoscaler ( HPA ) and Vertical Pod Autoscaler ( VPA ) to work it requires the metrics to be exported from the kubelet. Metrics Server collects resource metrics from Kubelets and exposes them in Kubernetes apiserver through Metrics API for use by Horizontal Pod Autoscaler and Vertical Pod Autoscaler.

git clone <https://github.com/pavan-kumar-99/medium-manifests.git> \  
-b autoscaling

$ cd medium-manifests/

$ kubectl apply -f metrics-server.yaml

$ kubectl top po

kubectl apply -f deploy.yaml $ kubectl get svc k8s-autoscaler -o jsonpath='{.status.loadBalancer.ingress[0].ip}'

$ kubectl apply -f hpa.yaml

$ external\_ip=$(kubectl get svc k8s-autoscaler -o jsonpath='{.status.loadBalancer.ingress[0].ip}')$ while true; do curl $external\_ip ; done

kubectl desribe hpa k8s-autoscaler