AKS - Scaling

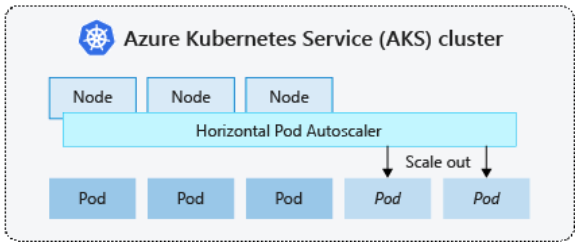
13 July 2023

18:39

**Scaling options for applications in AKS**

**Horizontal Pod Autoscaler (HPA)**

Kubernetes uses the horizontal pod autoscaler (HPA) to monitor the resource demand and automatically scale the number of pods. By default, the HPA checks the Metrics API every 15 seconds for any required changes in replica count, and the Metrics API retrieves data from the Kubelet every 60 seconds. When changes are required, the number of replicas is increased or decreased accordingly.



**Note:**

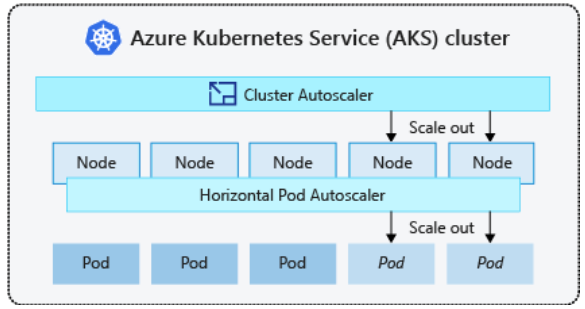
* HPA works with AKS clusters that have deployed the Metrics Server for Kubernetes 1.8+.
* HPA allows you to define the minimum and maximum number of replicas that can run, when you configure the HPA for a given deployment.
* HPA allows you to define the metric to monitor and base any scaling decisions on, such as CPU usage and memory usage.

**Cooldown of scaling events**

* To restrict the flapping of replica count, HPA allows you to configure the delay on scale-up and scale-down events.
* This value defines how long the HPA must wait after a scale event before another scale event can be triggered.
* This behavior allows the new replica count to take effect and the Metrics API to reflect the distributed workload.
* There's no delay for scale-up events as of Kubernetes 1.12, however, the default delay on scale down events is 5 minutes.

**Cluster Autoscalar**

Kubernetes cluster autoscaler adjusts the number of nodes based on the requested compute resources in the node pool. By default, the cluster autoscaler checks the Metrics API server every 10 seconds for any required changes in node count. If the cluster autoscaler determines that a change is required, the number of nodes in your AKS cluster is increased or decreased accordingly.



**Note:**

* The cluster autoscaler works with Kubernetes RBAC-enabled AKS clusters that run Kubernetes 1.10.x or higher.
* The cluster autoscaler is typically used alongside the horizontal pod autoscaler.
* When combined, the horizontal pod autoscaler increases or decreases the number of pods based on application demand, and the cluster autoscaler adjusts the number of nodes to run additional pods.

**Scale out event**

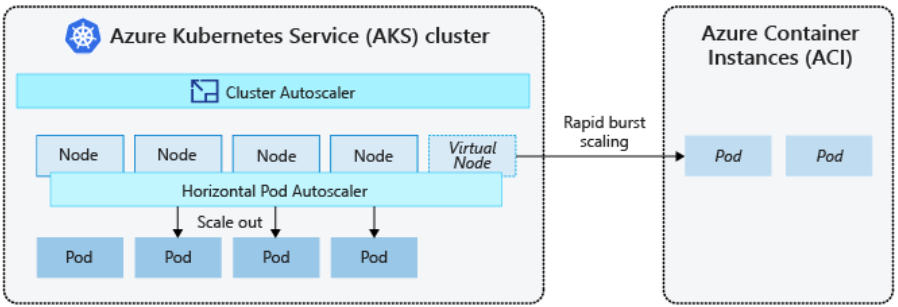
* When cluster autoscaler notices pods that can't be scheduled because of node pool resource constraints, the number of nodes within the node pool is increased to provide the additional compute resources.
* When Additional nodes are successfully deployed and available for use within the node pool, the pods are then scheduled to run on them.
* If your application needs to scale rapidly, some pods may remain in a state waiting to be scheduled until the additional nodes deployed by the cluster autoscaler can accept the scheduled pods.

**Scale in event**

* The cluster autoscaler also monitors the pod scheduling status for nodes that haven't recently received new scheduling requests. This scenario indicates the node pool has more compute resources than required, and the number of nodes can be decreased.
* By default, nodes that pass a threshold for no longer being needed for 10 minutes is scheduled for deletion. When this situation occurs, pods are scheduled to run on other nodes within the node pool, and the cluster autoscaler decreases the number of nodes.
* Your applications may experience some disruption as pods are scheduled on different nodes when the cluster autoscaler decreases the number of nodes. To minimize disruption, avoid applications that use a single pod instance.

**Burst to Azure Container Instances (ACI)**

ACI lets you quickly deploy container instances without additional infrastructure overhead. When you connect with AKS, ACI becomes a secured, logical extension of your AKS cluster. The virtual nodes component, which is based on virtual Kubelet, is installed in your AKS cluster that presents ACI as a virtual Kubernetes node. Kubernetes can then schedule pods that run as ACI instances through virtual nodes, not as pods on VM nodes directly in your AKS cluster.



**Note:**

* Your application requires no modifications to use virtual nodes.
  + Your deployments can scale across AKS and ACI and with no delay as the cluster autoscaler deploys new nodes in your AKS cluster.
* Virtual nodes are deployed to an additional subnet in the same virtual network as your AKS cluster.
  + This virtual network configuration secures the traffic between ACI and AKS.
  + Like an AKS cluster, an ACI instance is a secure, logical compute resource isolated from other users.