LINUXCON containercon CHINA 中国 CLOUDOPEN

THINK OPEN
开放性思维

Kubernetes Autoscaling on Azure

Pengfei Ni
Microsoft Azure

Abstract

- Why autoscaling
- Autoscaling in Kubernetes
- Practice on Azure
- Q&A

Why autoscaling

- Autoscaling
  - Adjust computational resources automatically
- Benefits
  - Reduce cost
  - Increase service availability
  - Increase elasticity

Cloud provider autoscaling

- Horizontal
  - Scale number of virtual machines
- Vertical
  - Scale resources of virtual machines
- Drawbacks
  - Not aware of Kubernetes scheduler (e.g. multiple node groups)
  - May remove nodes with critical Pods
  - Hard to conform kubernetes evictions

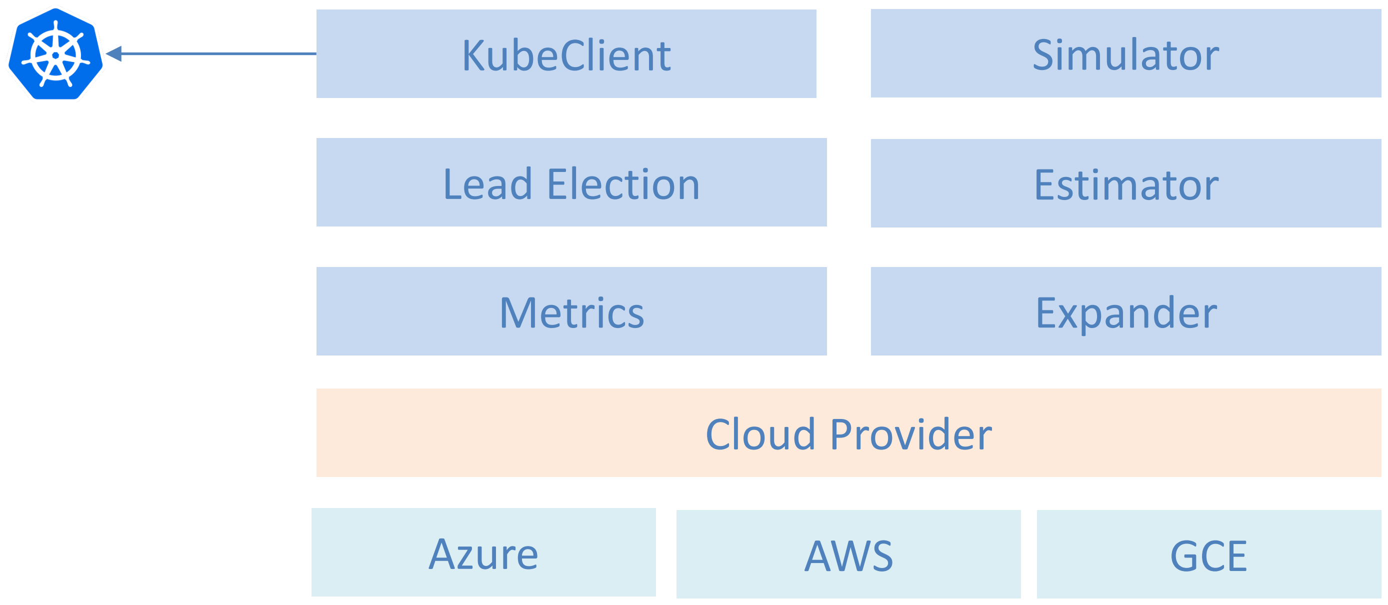
Autoscaling in Kubernetes

- Horizontal Pod autoscaler (HPA)
  - Scale number of Pods
- Vertical Pod autoscaler (VPA)
  - Scale resources of Pods
- Cluster proportional autoscaler (CPA)
  - Scale replicas of Pods based on number of nodes
- Cluster autoscaler (CA)
  - Scale number of nodes

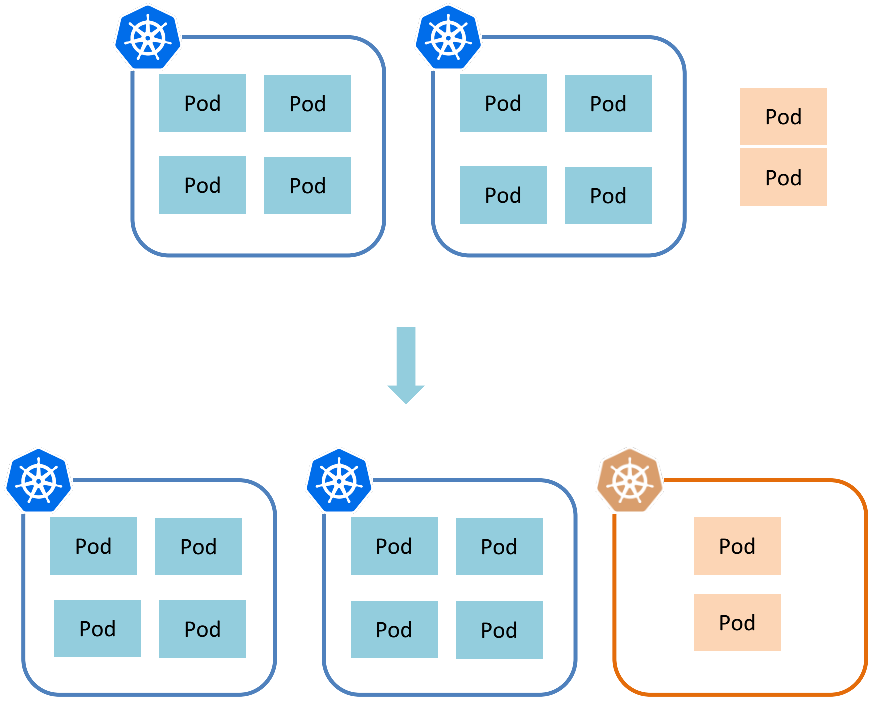
Cluster Autoscaler

- Adjust number of Nodes automatically
  - Add nodes when there’re Pods failed to schedule
  - Remove nodes when they are underutilized for an extended period
- Supported Cloud providers
  - Azure (VMAS/VMSS/AKS/ACS)
  - AWS
  - GCE/GKE

How CA works


KubeClient
Simulator
Lead Election
Estimator
Metrics
Expander
Cloud Provider
Azure
AWS
GCE

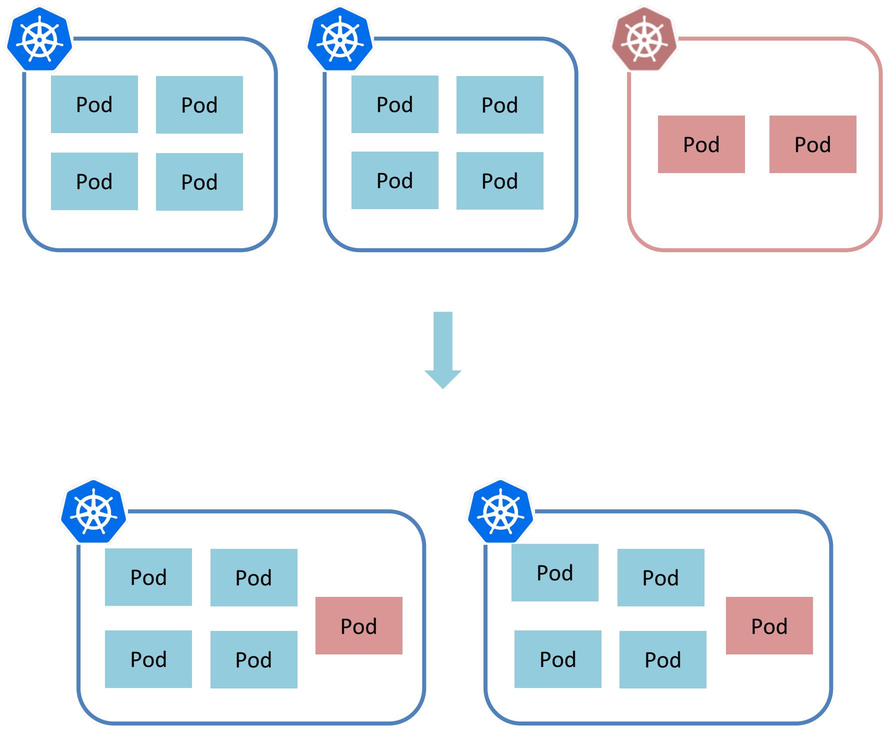
Scale Up


Pod
Pod
Pod
Pod
Pod
Pod
Pod
Pod
Pod
Pod
Pod
Pod
Pod
Pod
Pod
Pod
Pod
Pod
Pod
Pod

Scale Up

- Get node groups from cloud provider
- Build template nodes for each node groups
- Check Pods with unschedulable condition
- Check which template node fit the pending Pods
  - If more than one node groups, select by expander
    - random, most-pods, least-waste, price
- Create Node by cloud provider
- Wait for node ready

Scale Down


Pod
Pod
Pod
Pod
Pod
Pod
Pod
Pod
Pod
Pod
Pod
Pod
Pod
Pod
Pod
Pod
Pod
Pod
Pod
Pod

Scale Down

- Check unneeded nodes
  - Sum of CPU/Memory requests is less than 50%
  - All Pods on the node could be evicted
    - Managed by controllers
    - No restrictive PodDisruptionBudget
    - No constraints (e.g. node selector) preventing node moving
  - No scale down annotation
- Wait a while (e.g. 10 min)
- Evict, taint and then remove the node from cloud provider

Avoid abrupt scaling

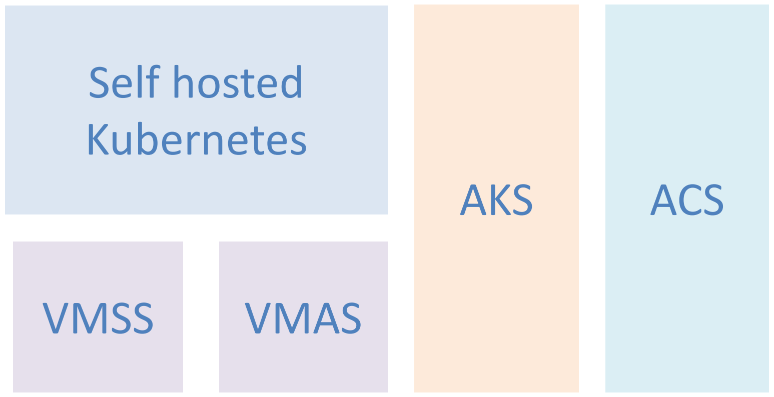
- Mark node as unneeded and wait for a while (10 min) before removal
- Evict and taint node first before removal
- Stop scaling down for a while (10 min) after scaling up
- Stop operating when unready nodes are more than 45% or 3
- Use PodDisruptionBudget

Limitations

- Up to 1000 nodes are supported
- Up to 10 min graceful termination period
- Not compatible with cpu-metrics based autoscalers

Practice on Azure

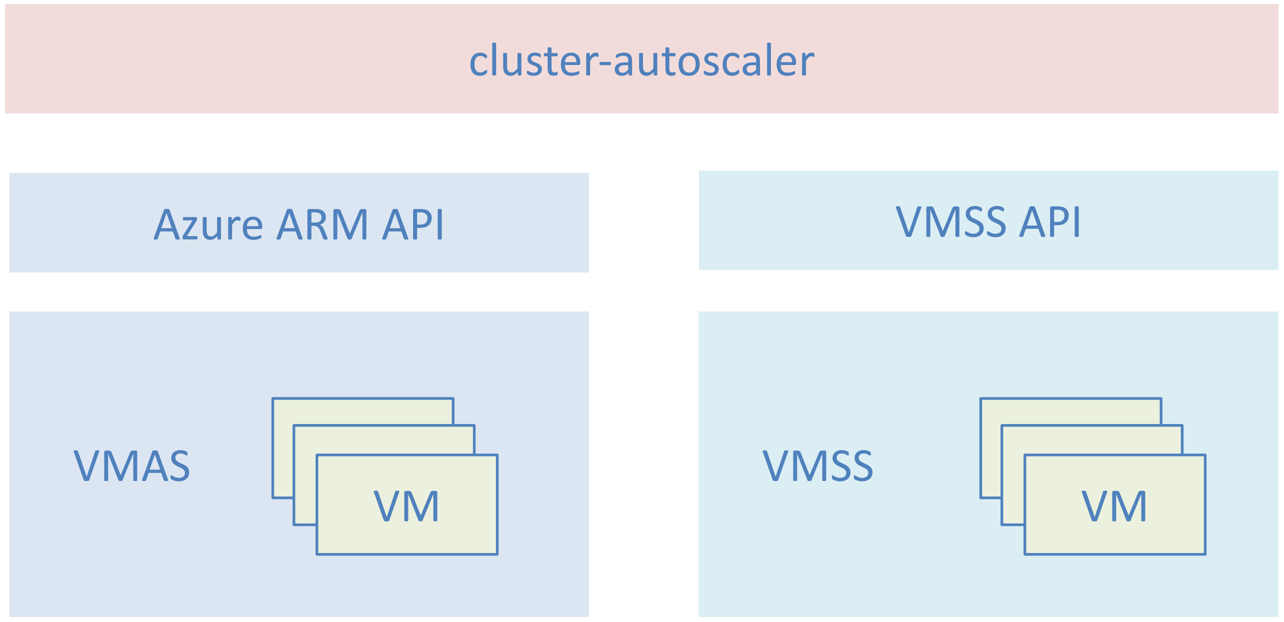
- Container services on Azure
  - AKS
  - ACS
  - Self hosted Kubernetes
- VM Type
  - Availability Set (VMAS)
  - Scale Set (VMSS)



VMAS/VMSS

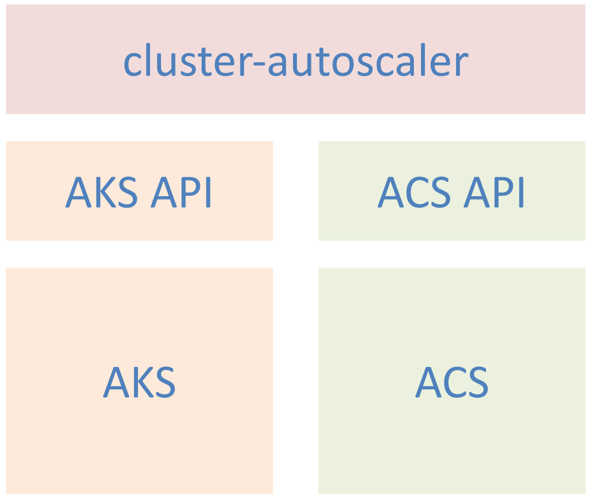
	VMSS	VMAS
VM configuration	Identical	Usually different
VM creation	Automated	Manually
Load Balancer	Automated with ALB	Manually
Scaling	Automated	Manually
High Availability	Auto distribution across availability zones or availability sets	Isolated hardware, manually setup availability zones

VMAS/VMSS



AKS/ACS

- Managed Kubernetes cluster
- No charge of controller plane
- ACS will be deprecated
- AKS is highly recommended
- Easily setup
  - az aks create
  - az aks upgrade
  - az aks scale



Best Practices

- Run cluster-autoscaler with matched k8s version
- Run containers with multiple replicas
- Setup resource requests for containers
- Use PodDisruptionBudgets to prevent Pods being removed abruptly
- Do not manage node manually
- Disable other virtual machine autoscalers (e.g. those from cloud provider)
- Setup min/max nodes and ensure quota sufficient

LINUXCON containercon CHINA 中国 CLOUDOPEN

THINK OPEN
开放性思维

Thanks

Q&A

LF ASIA, LLC