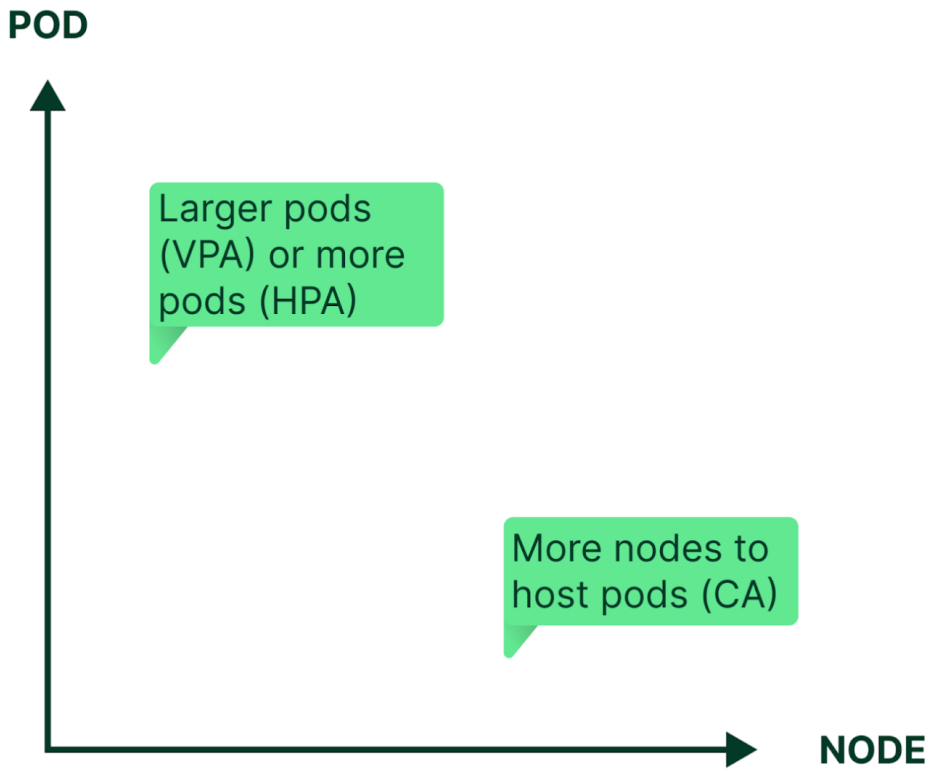
CS-552/452 Introduction to Cloud Computing

Orchestration - AutoScaling

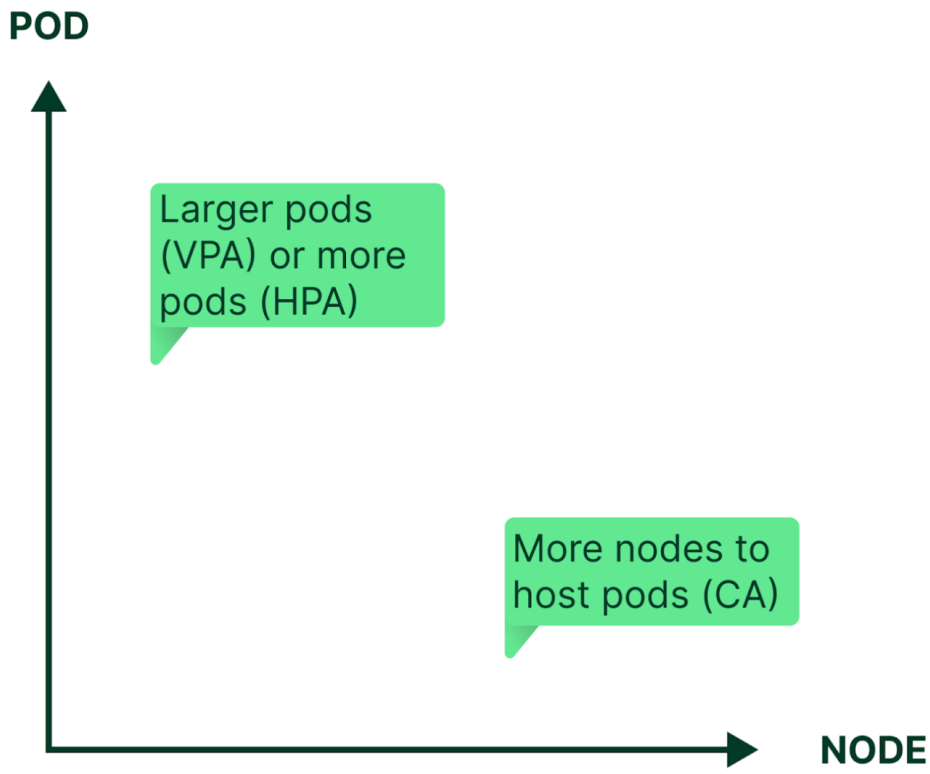
What is Autoscaling?

- Autoscaling is one of the most compelling features of the orchestration platforms.
- Autoscaling saves administrators time, prevents performance bottlenecks, and helps avoid financial waste.
- Three dimensions of Kubernetes autoscaling
  - *Cluster Autoscaler* (CA) adds or removes nodes dedicated to the cluster to provide the appropriate amount of computing resources needed to host the desired workloads.



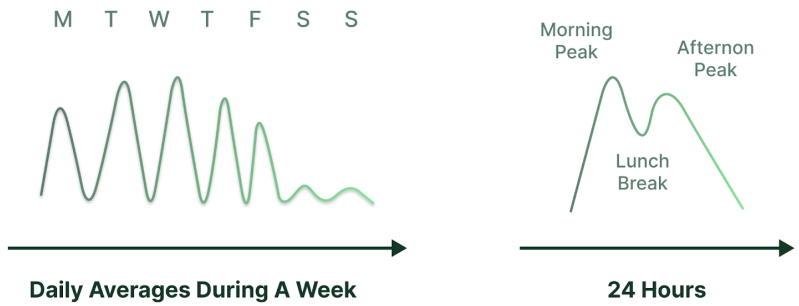
What is Autoscaling?

- Autoscaling is one of the most compelling features of the orchestration platforms.
- Autoscaling saves administrators time, prevents performance bottlenecks, and helps avoid financial waste.
- Three dimensions of Kubernetes autoscaling
  - *Vertical Pod Autoscaler* (VPA) can either increase or decrease the CPU and memory allocated to each pod
  - *Horizontal Pod Autoscaler* (HPA) can replicate or terminate pods, thus affecting the total pod count.



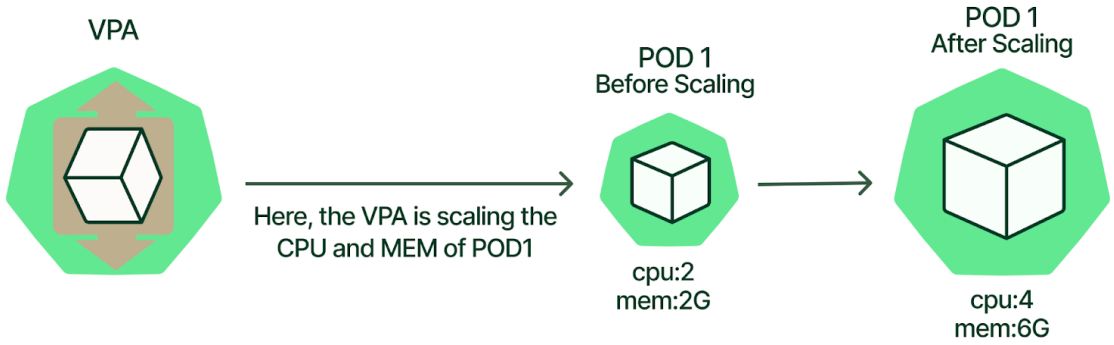
Why Autoscaling?

- Most application workloads have daily, weekly, and seasonal rhythms driven by user activity.
  - Application performance to degrade due to resource constraints, or
  - Unnecessary spending due to over-provisioning.
- Administrators must provision capacity within seconds and remove the capacity when it's no longer needed.
- Autoscaling automates the process of adding and removing capacity.



Kubernetes Vertical Pod Autoscaler (VPA)

- Increases and decreases container's CPU and memory to align resource allotment with actual usage.
- Two types of resource configurations:
  - Requests
    - *Requests:* minimum amount of resources that containers need.
  - Limits
    - *Limits:* maximum amount of resources that a container can consume.



Components of VPA

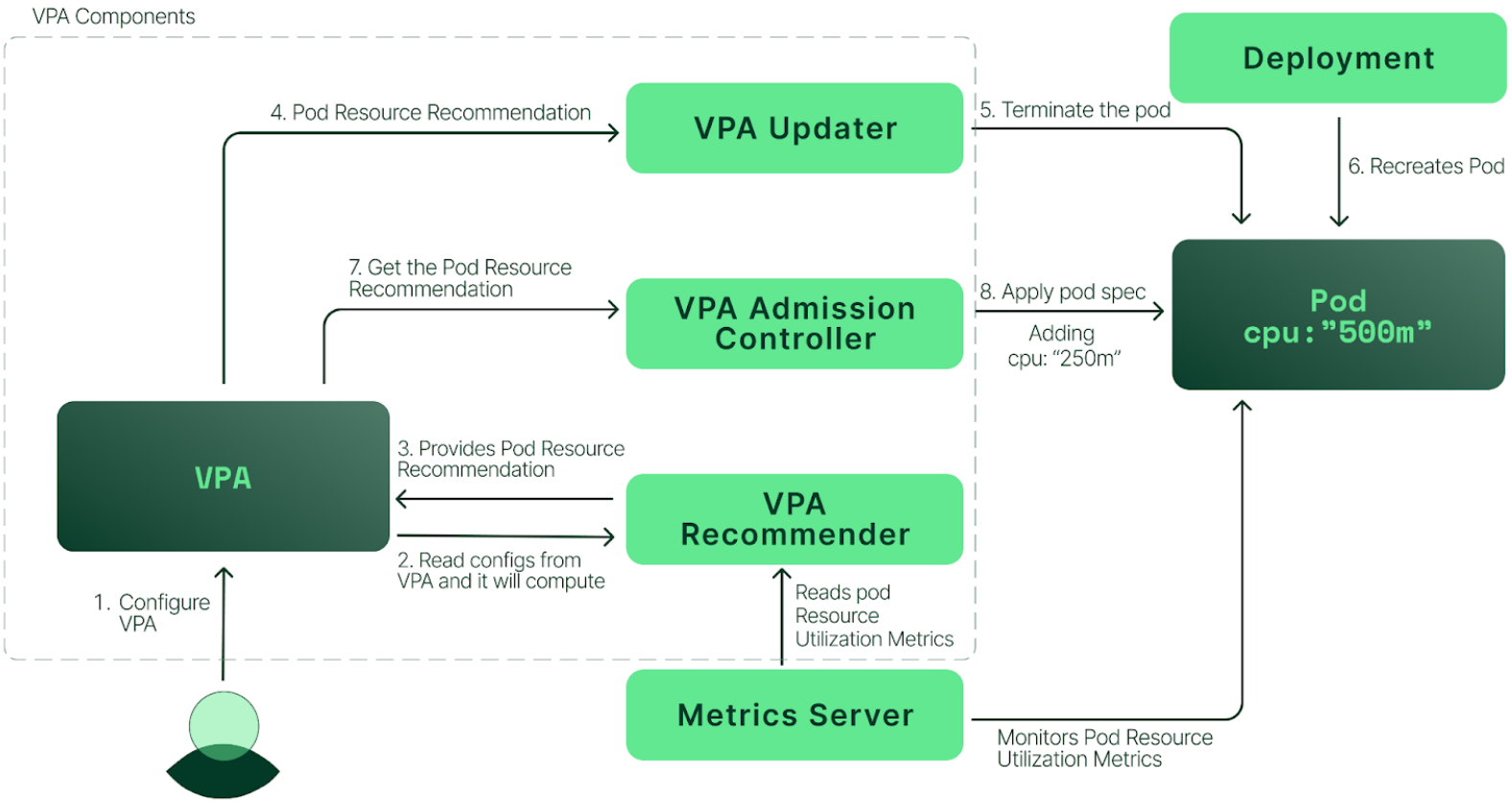
- The VPA Recommender:
  - *Monitors* resource utilization and computes target values.
  - Looks at the metric history, OOM events, and the VPA deployment spec and suggests fair requests. The limits are raised/lowered based on the *limits-requests* proportion defined.

- The VPA Updater:
  - *Evicts* those pods that exceed the new resource limits.
  - Implements whatever the Recommender recommends

- The VPA Admission Controller:
  - *Changes* the CPU and memory settings before a new pod starts whenever the VPA Updater evicts and restarts a pod.
  - Due to the design of Kubernetes, the only way to modify the resource requests of a running pod is to recreate the pod.

How does Kubernetes VPA work?

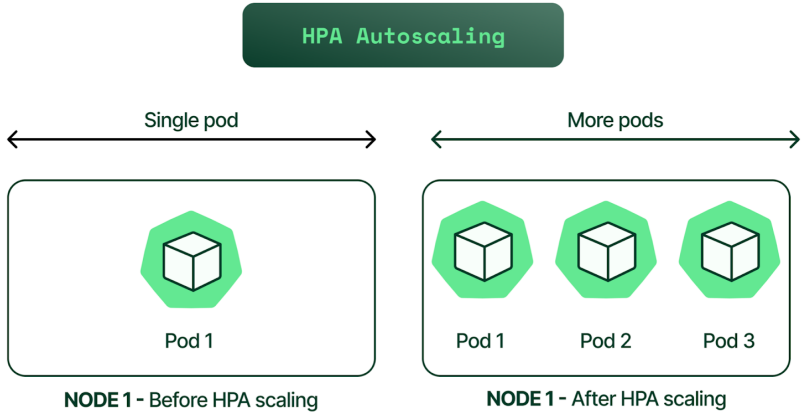
observe resource usage → recommend resource requests → update resources



https://www.kubecost.com/kubernetes-autoscaling/kubernetes-vpa

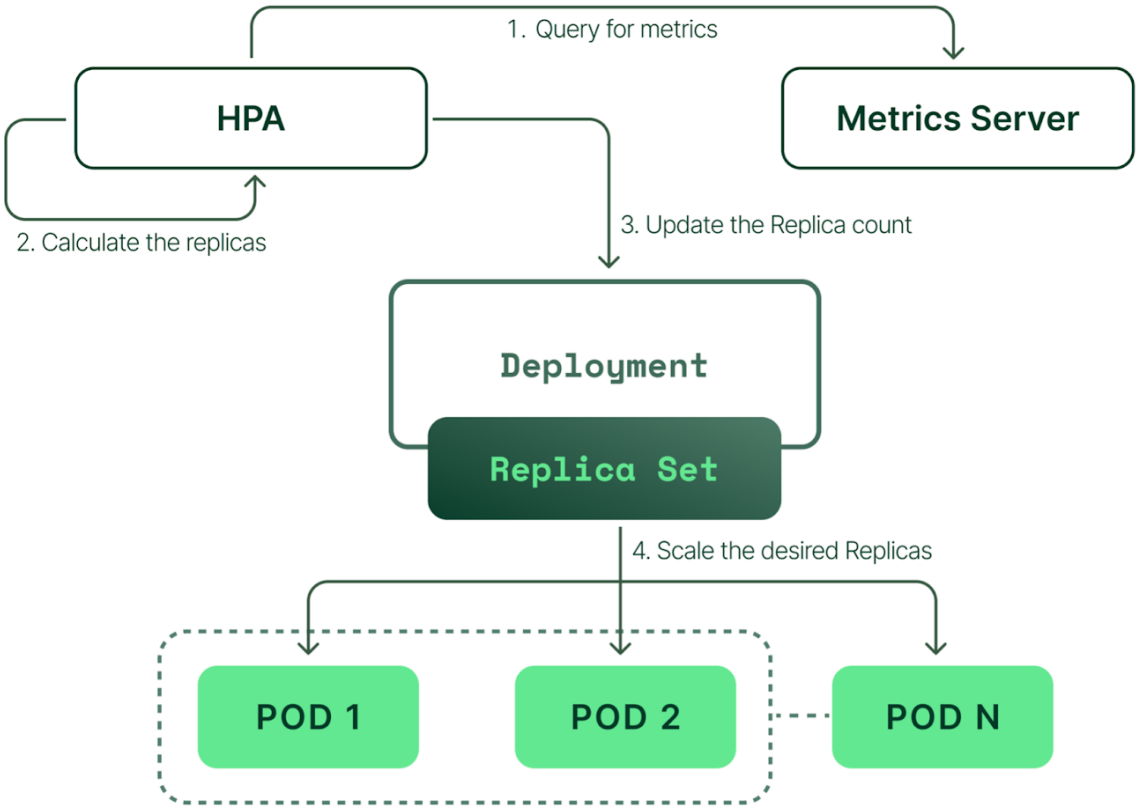
Kubernetes Horizontal Pod Autoscaler (HPA)

- It increases or decreases the number of pods in a deployment
- The scaling is horizontal because it affects the number of instances rather than the resources allocated to a single container.
- HPA can make scaling decisions based on custom (or externally provided) metrics and works automatically after initial configuration
- Once configured, the Horizontal Pod Autoscaler controller is in charge of checking the metrics and then scaling your replicas up or down accordingly



How does Kubernetes HPA work?

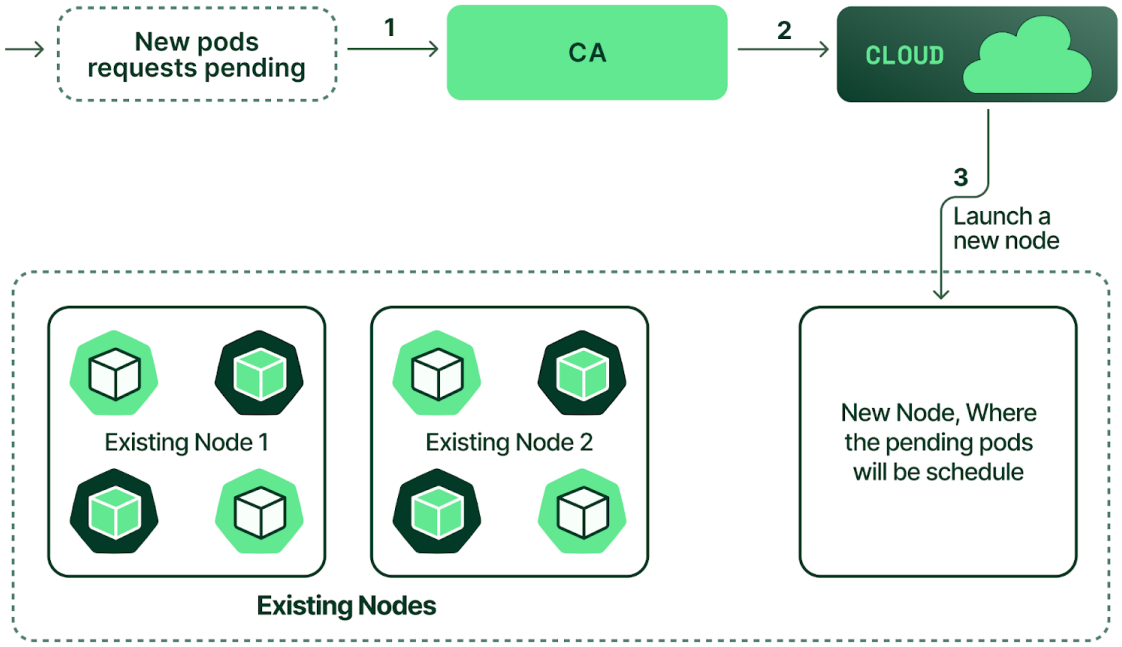
Check-> update→ check again



https://www.kubecost.com/kubernetes-autoscaling/kubernetes-vpa

Kubernetes Cluster Autoscaler (CA)

- The Cluster Autoscaler automatically adds or removes nodes in a cluster based on resource requests from pods.
  - It checks every 10 seconds to detect any pods in a pending state, suggesting that the scheduler could not assign them to a node due to insufficient cluster capacity



Sources

- Kubernetes Autoscaling: https://www.kubecost.com/kubernetes-autoscaling