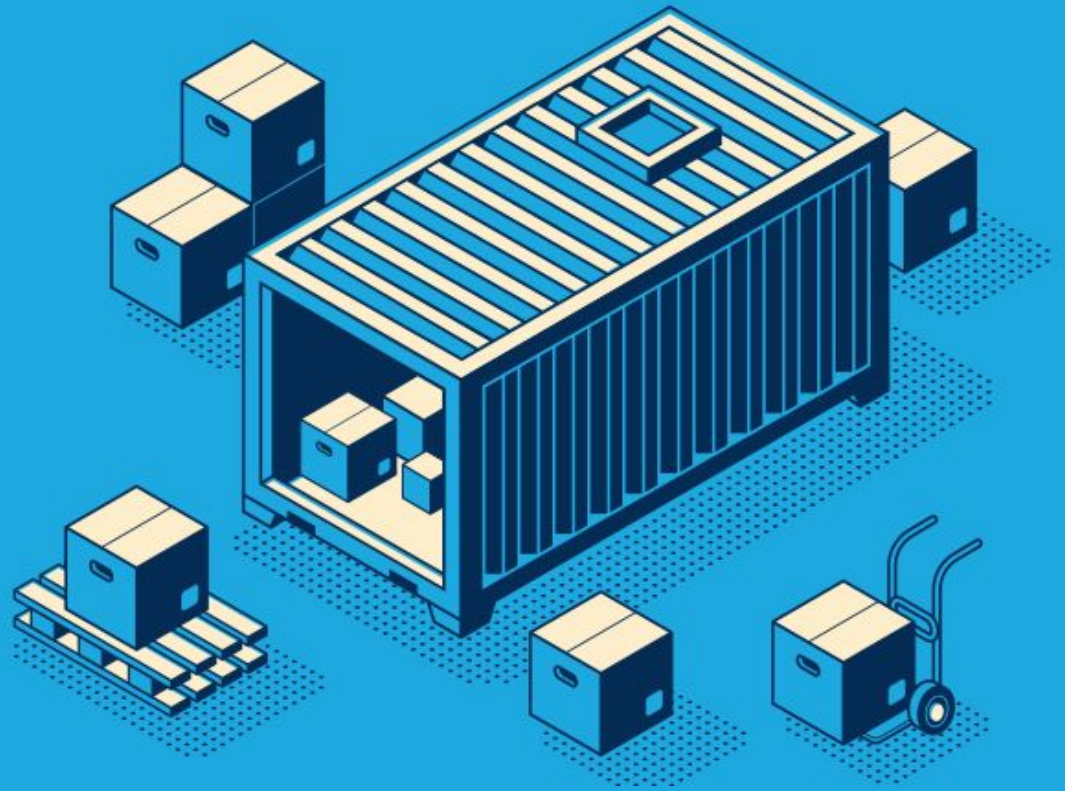
Kubernetes

Vertical Pod Autoscaler Operator



CS 6620 - Cloud Computing

The Team

Selen

Laxmi

Apoorva

Akshay

Sindhu

Mentors (Redhat)

Humair Khan & Anand Sanmukhani

What is Kubernetes ?

What is Kubernetes ?

- Containers are a good way to bundle and run your applications.
- In a production environment, you need to manage the containers that run the applications and ensure that there is no downtime.
- For example, if a container goes down, another container needs to start.
- Wouldn't it be easier if this behavior was handled by a system?

What is Kubernetes ?

- That's how Kubernetes comes to the rescue!
- Kubernetes provides you with a framework to run distributed systems resiliently.
- It takes care of scaling and failover for your application, provides deployment patterns, and more.
- For example, Kubernetes can easily manage a canary deployment for your system.

What is a POD?

What is a POD?

- Pods are the smallest, most basic deployable objects in Kubernetes.
- Pods contain one or more containers, such as Docker containers.
- When a Pod runs multiple containers, the containers are managed as a single entity and share the Pod's resources.
- Pods also contain shared networking and storage resources for their containers:

POD Resources

POD Resources

- When you create a Pod, you normally specify the storage and networking that containers share within that Pod.
- The Kubernetes scheduler will then look for nodes that have the resources required to run the Pod.

```
- name: cpu-request-limit-container
  image: images.example/app-image
  resources:
    requests:
      cpu: "500m"
    limits:
      cpu: "1500m"
```

Problem

Problem

- Now both forms of compute resources, CPU and Memory, are specified in similar ways, and are done so in a static manner.
- A human operator will first specify what compute resources this Pod will require to operate.
- Workloads demand more resources at peak times, while remaining mostly dormant during off peak hours.
- As one might expect, this results in either waste of compute resources or reduced pod performance.

Solution

Solution

- The solution here is to move away from manually specifying the compute resources for a pod that are static, and opt for a more automated scaling solution.

- The ideal solution would identify when a Pod requires additional resources, and adjust it’s requirements accordingly during the lifetime of the Pod.

Solution

- The Vertical Pod Autoscaler (VPA) was designed to specifically do this. The VPA will automatically scale the resources for a pod based on its usage trends.

- The goal of this project is to transform Operate First OCP clusters from manually specifying compute resource requirements for pods, and to instead use the VPA instead.

- Thus having all workloads automatically scale up/down their resources on a need basis.

Scope of the project

- Configure the VPA through GitOps in OpenShift cluster
- Monitor the CPU/memory utilization during fluctuating workload through grafana dashboard

Features of the project

- POC in MOC k8s namespace
  - Install VPA through OLM
  - Decide & Deploy an application on top of the cluster
- Deploy VPA in one live openshift cluster
  - Track CPU and memory utilization using grafana dashboard.
- Deploy VPA on all compute heavy openshift clusters.

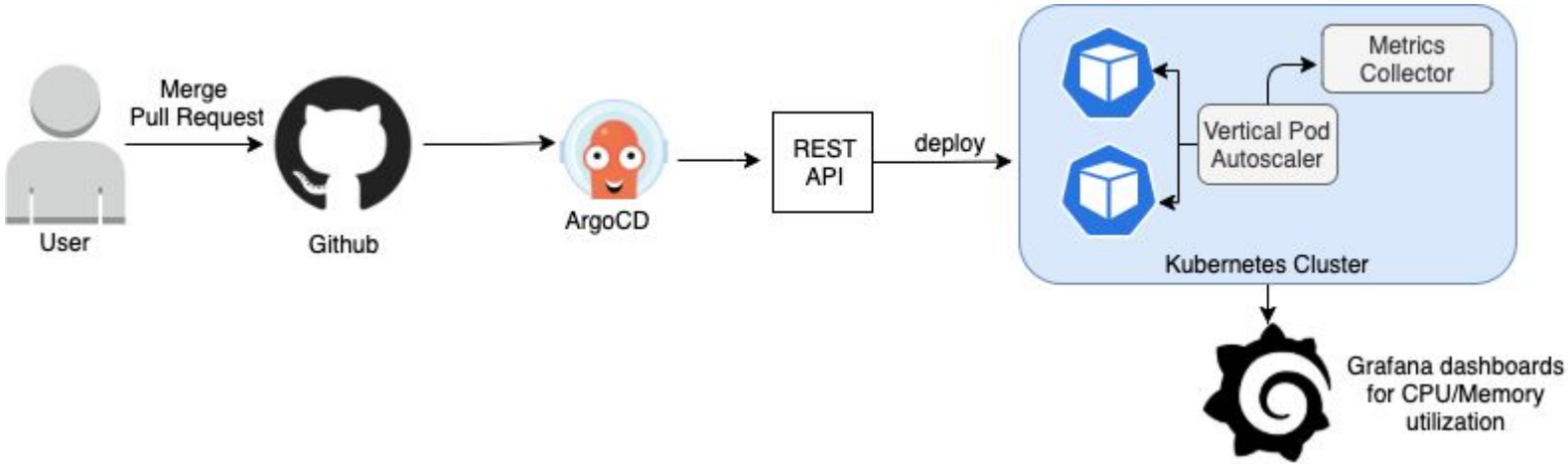
GitOps workflow

GitOps, uses Git repositories as single source of truth to deliver infrastructure as code.

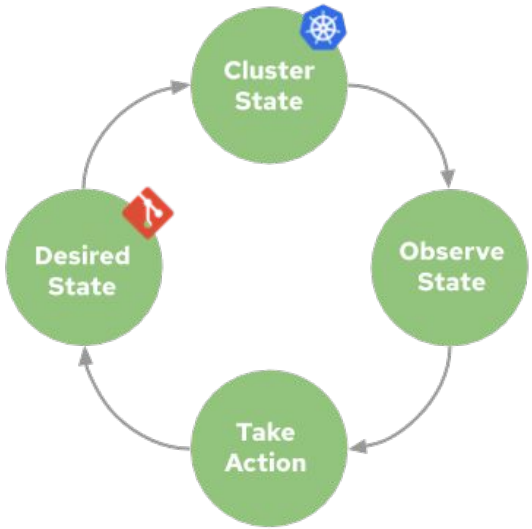
GitOps delivers:

- Improved reliability with visibility and version control through Git
- Consistency across any cluster, any cloud, and any on-prem environment
- Standard workflow for application development
- Increased security by setting application requirements up front

GitOps workflow



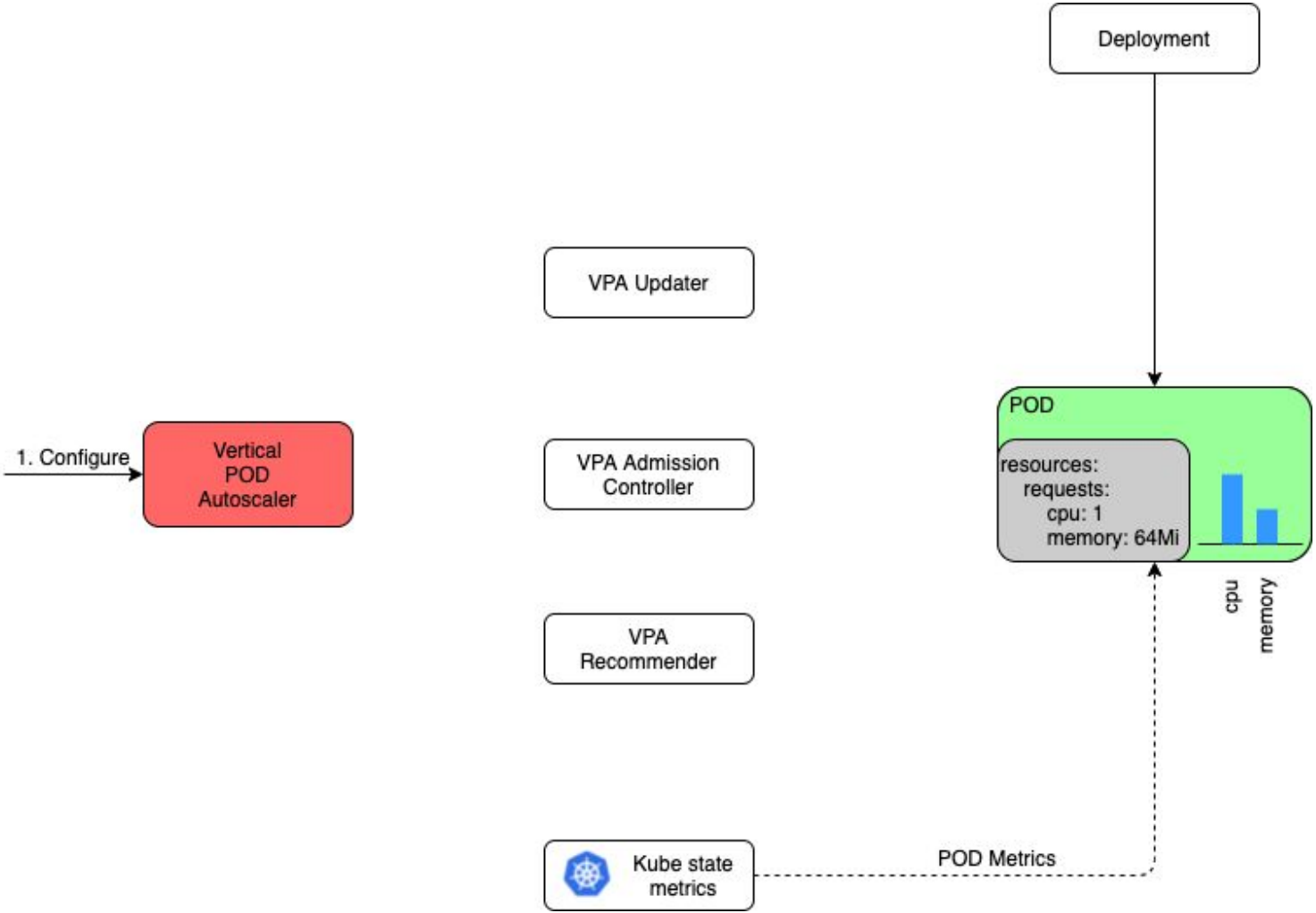
GitOps workflow

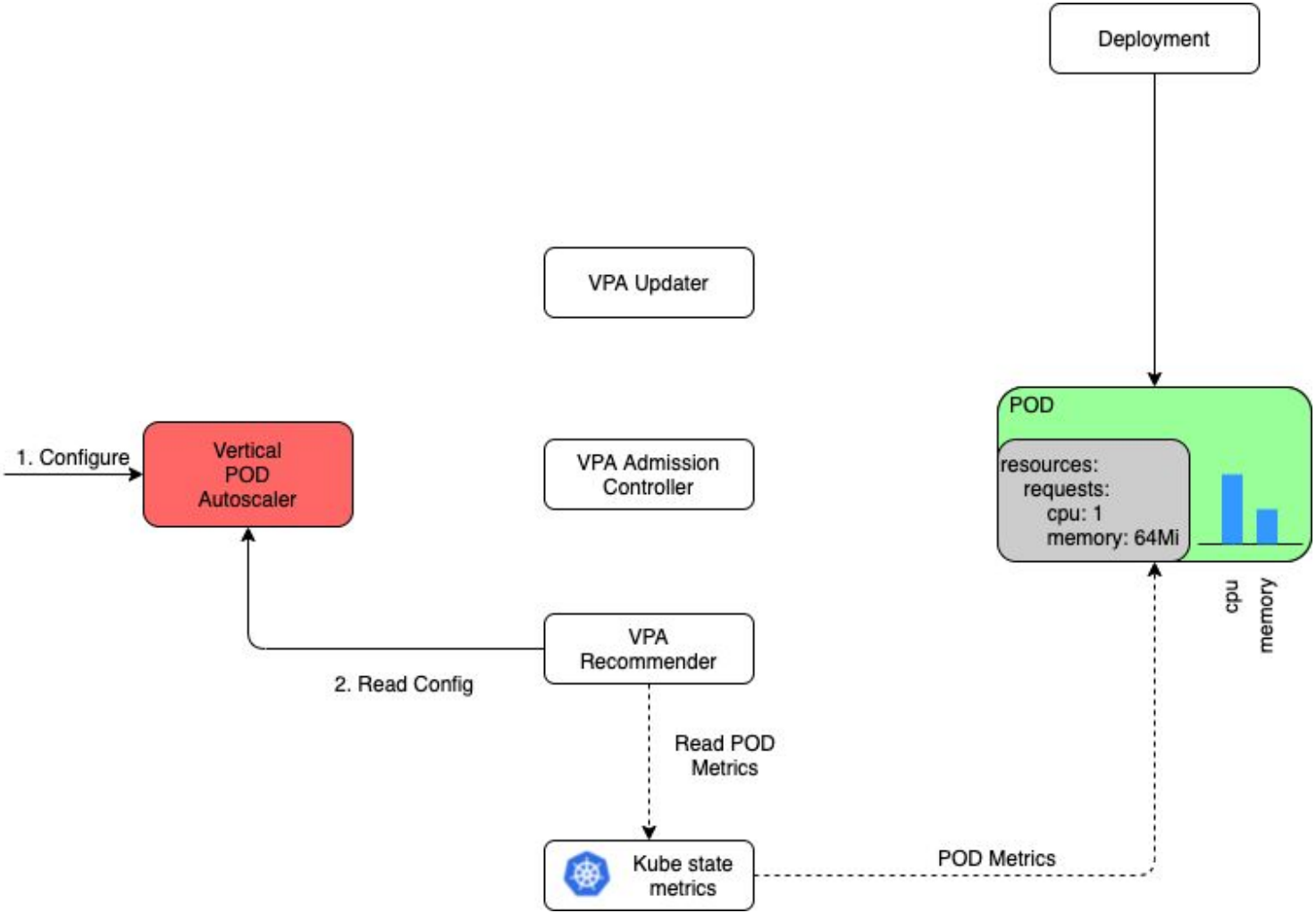


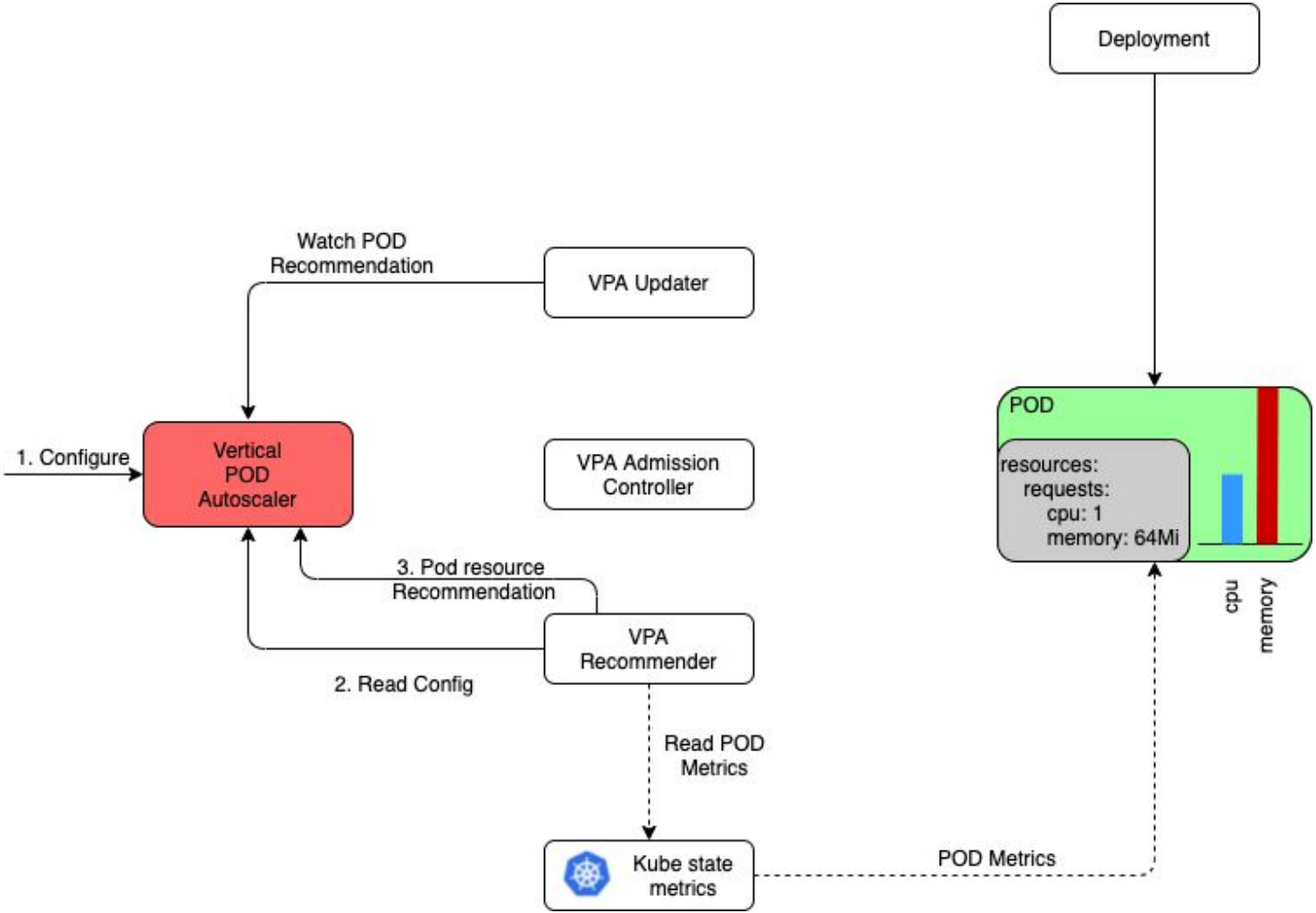
GitOps approach to Continuous Delivery on Kubernetes

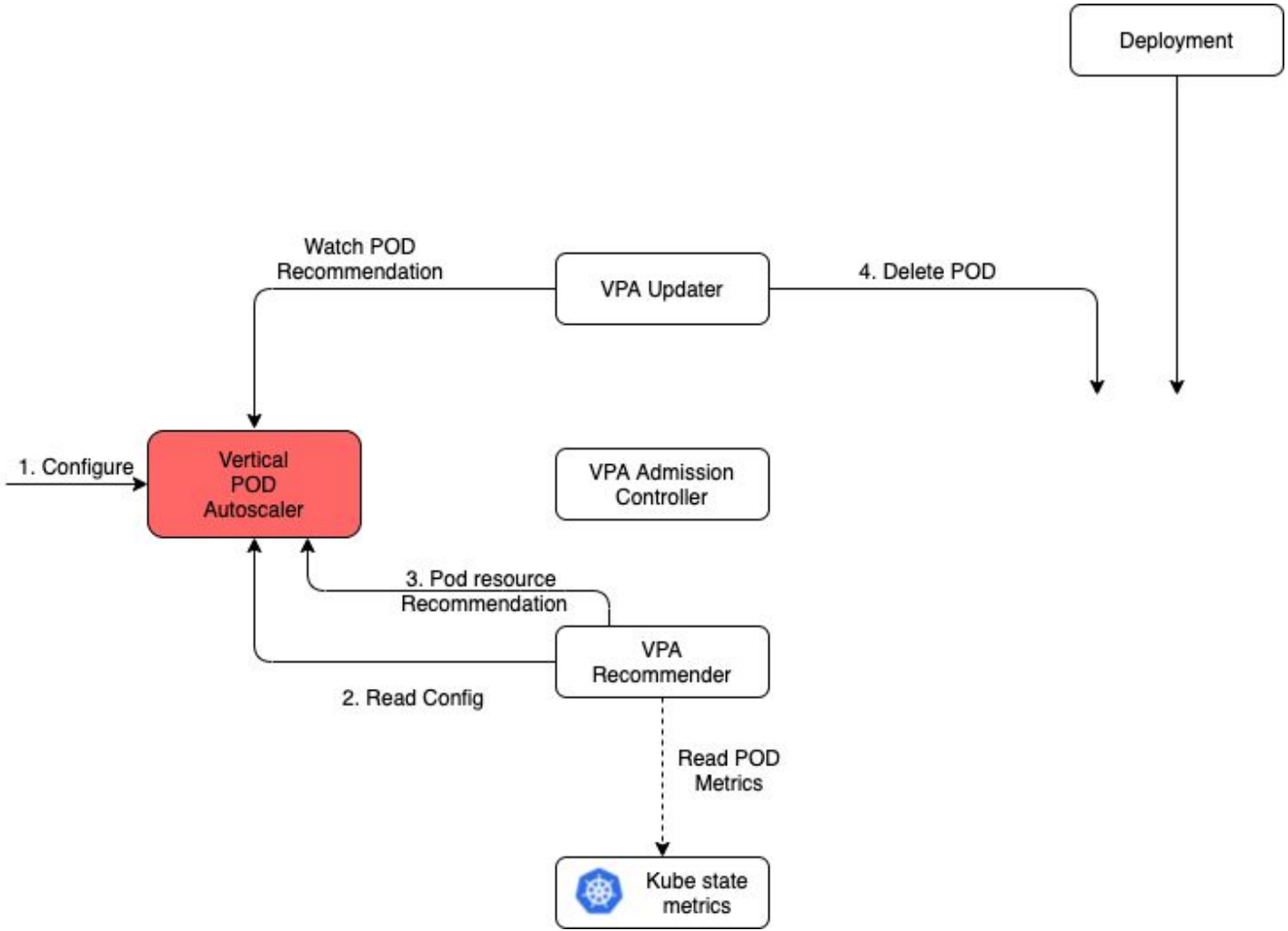
(Image credits: https://cloud.redhat.com/learn/topics/gitops/)

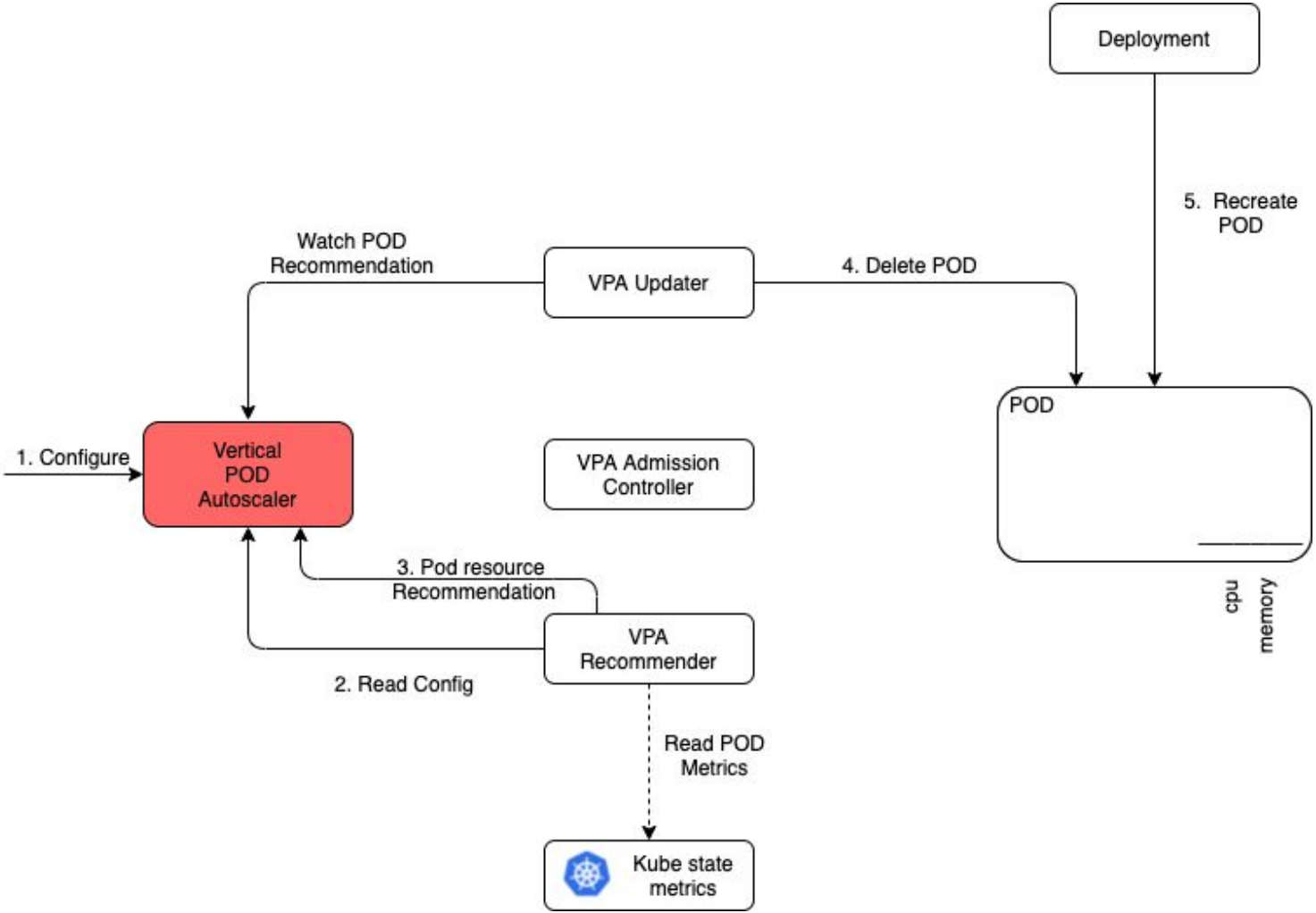
How VPA works?

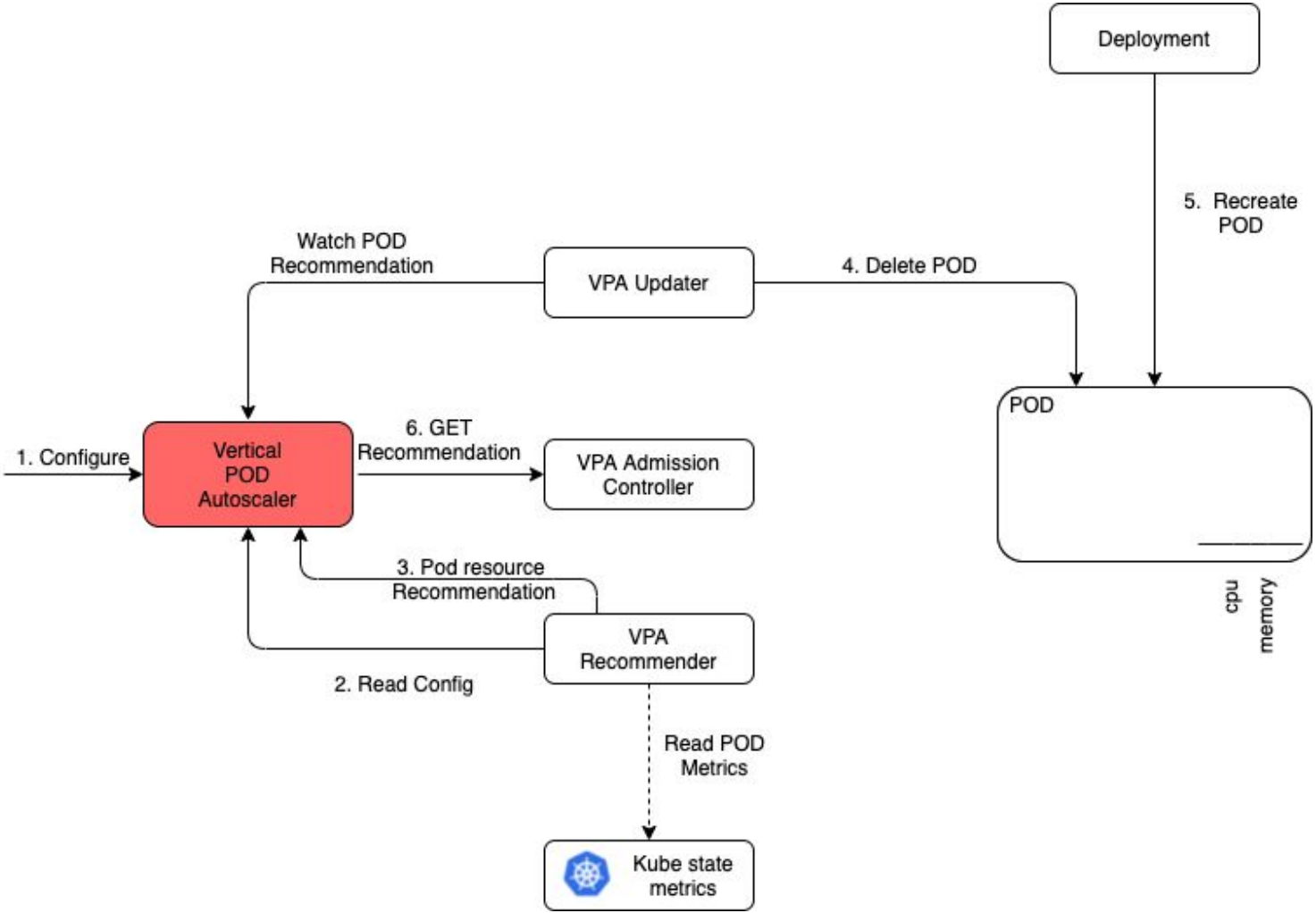

Deployment
VPA Updater
POD
1. Configure
Vertical
POD
Autoscaler
VPA Admission
Controller
resources:
requests:
cpu: 1
memory: 64Mi
cpu
memory
VPA
Recommender
Kube state
metrics
POD Metrics

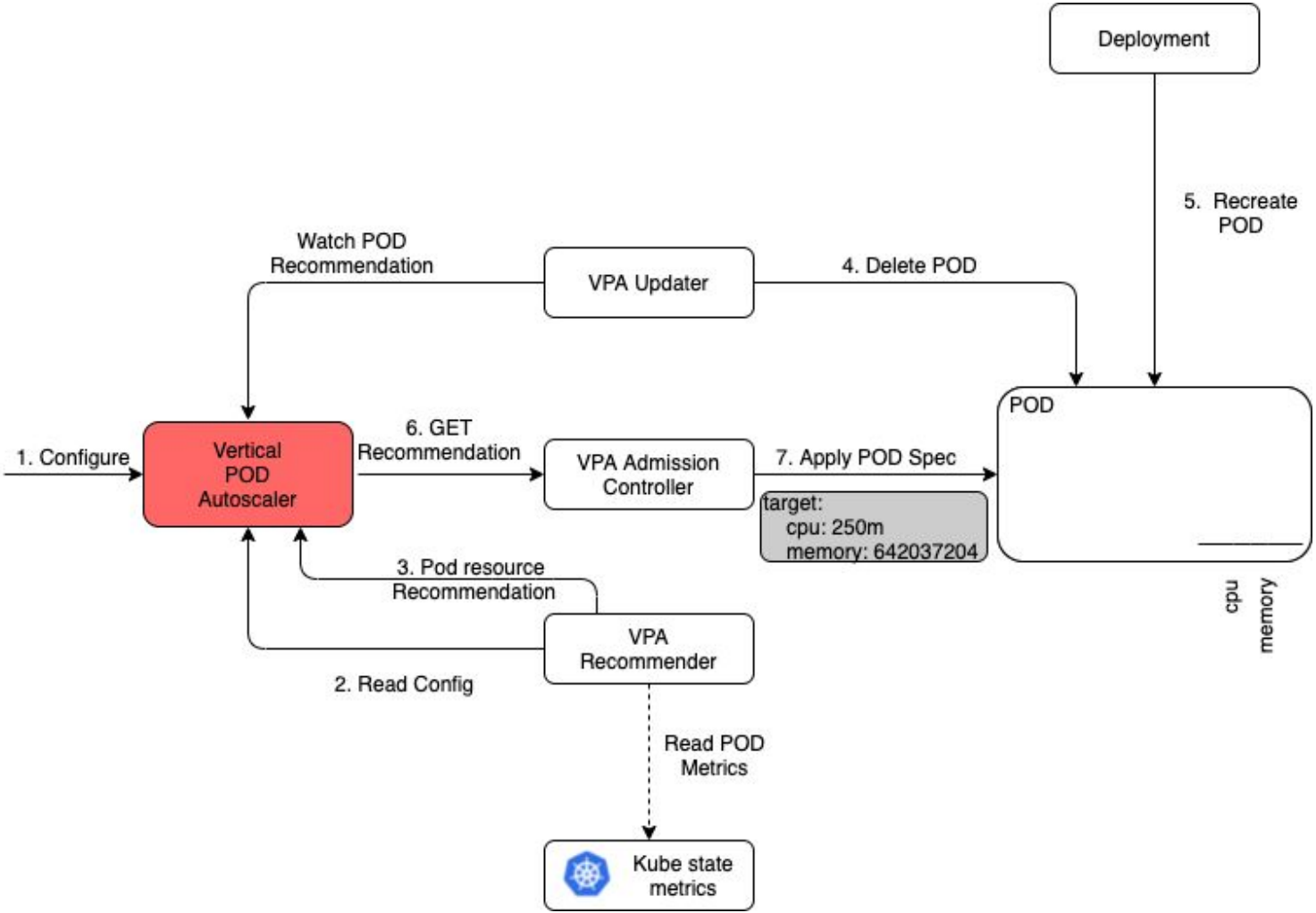

Deployment
VPA Updater
POD
1. Configure
Vertical
POD
Autoscaler
VPA Admission
Controller
resources:
requests:
cpu: 1
memory: 64Mi
cpu
memory
VPA
Recommender
2. Read Config
Read POD
Metrics
Kube state
metrics
POD Metrics

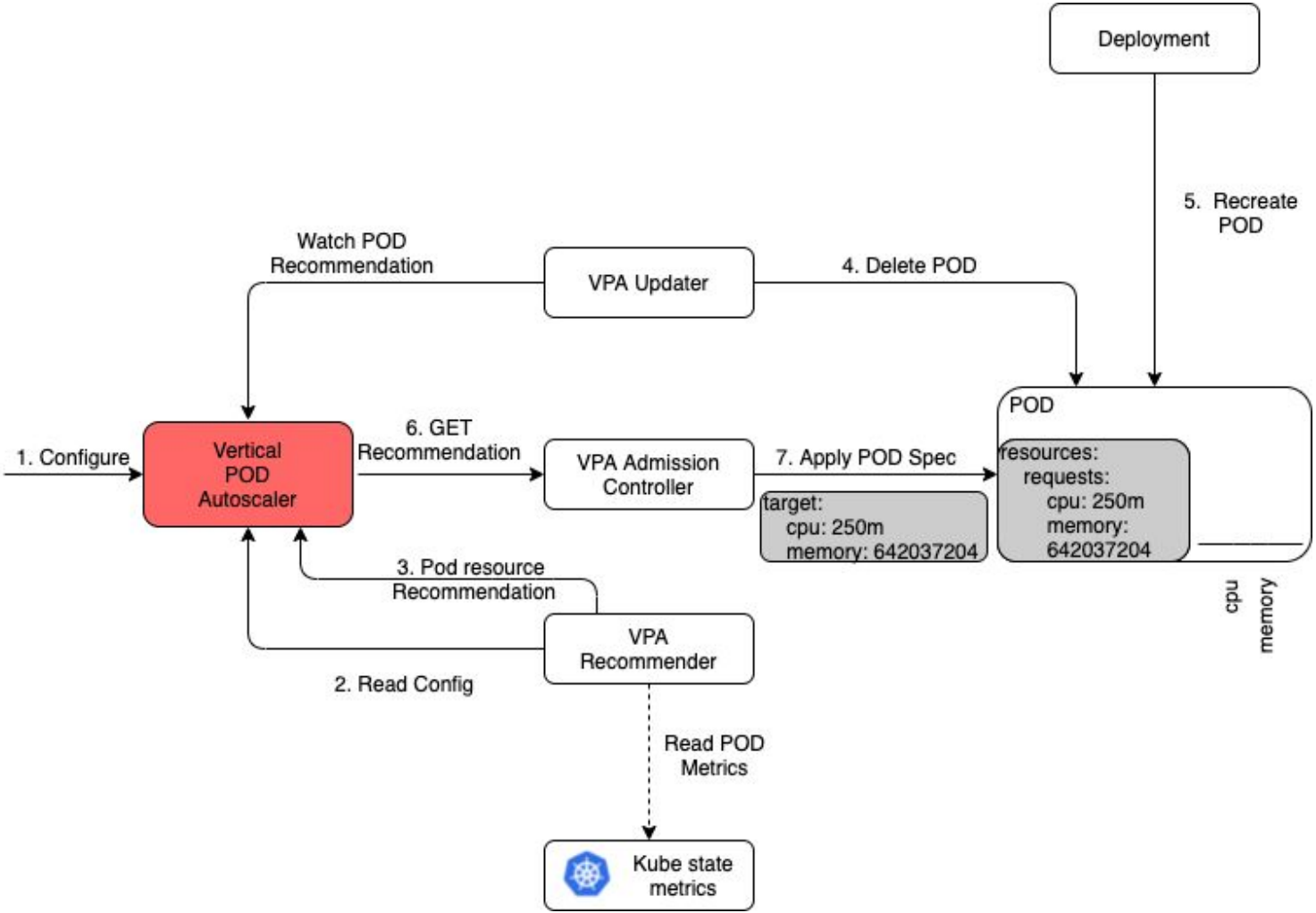

Deployment
Watch POD
Recommendation
VPA Updater
POD
resources:
requests:
cpu: 1
memory: 64Mi
cpu
memory
1. Configure
Vertical
POD
Autoscaler
VPA Admission
Controller
3. Pod resource
Recommendation
VPA
Recommender
2. Read Config
Read POD
Metrics
Kube state
metrics
POD Metrics

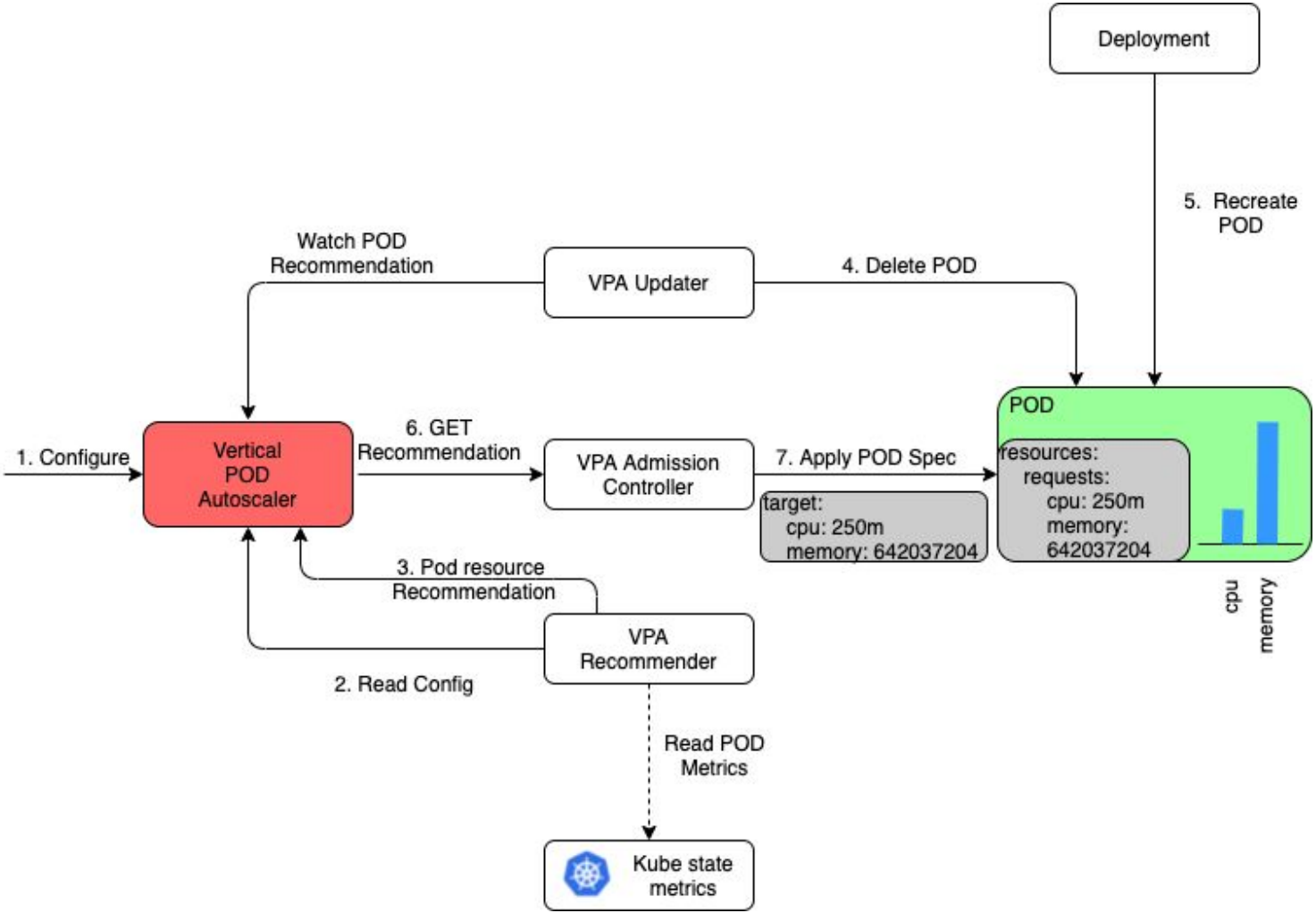

Deployment
Watch POD
Recommendation
VPA Updater
4. Delete POD
1. Configure
Vertical
POD
Autoscaler
VPA Admission
Controller
3. Pod resource
Recommendation
VPA
Recommender
2. Read Config
Read POD
Metrics
Kube state
metrics


Deployment
5. Recreate POD
Watch POD Recommendation
VPA Updater
4. Delete POD
POD
1. Configure
Vertical POD Autoscaler
VPA Admission Controller
3. Pod resource Recommendation
VPA Recommender
2. Read Config
cpu
memory
Read POD Metrics
Kube state metrics


Deployment
5. Recreate POD
Watch POD Recommendation
VPA Updater
4. Delete POD
POD
1. Configure
Vertical POD Autoscaler
6. GET Recommendation
VPA Admission Controller
3. Pod resource Recommendation
VPA Recommender
2. Read Config
cpu
memory
Read POD Metrics
Kube state metrics


Deployment
5. Recreate POD
Watch POD Recommendation
VPA Updater
4. Delete POD
POD
1. Configure
Vertical POD Autoscaler
6. GET Recommendation
VPA Admission Controller
7. Apply POD Spec
target:
cpu: 250m
memory: 642037204
3. Pod resource Recommendation
VPA Recommender
2. Read Config
cpu
memory
Read POD Metrics
Kube state metrics


Deployment
5. Recreate POD
Watch POD Recommendation
VPA Updater
4. Delete POD
POD
1. Configure
Vertical POD Autoscaler
6. GET Recommendation
VPA Admission Controller
7. Apply POD Spec
target:
cpu: 250m
memory: 642037204
resources:
requests:
cpu: 250m
memory: 642037204
cpu
memory
3. Pod resource Recommendation
VPA Recommender
2. Read Config
Read POD Metrics
Kube state metrics


Deployment
5. Recreate POD
Watch POD Recommendation
VPA Updater
4. Delete POD
POD
1. Configure
Vertical POD Autoscaler
6. GET Recommendation
VPA Admission Controller
7. Apply POD Spec
target:
cpu: 250m
memory: 642037204
resources:
requests:
cpu: 250m
memory: 642037204
cpu
memory
3. Pod resource Recommendation
VPA Recommender
2. Read Config
Read POD Metrics
Kube state metrics

Sprint Planning & Taiga

Taiga Setup

Modules:

- Epics
  - High level Project achievements
  - Great way to define broader value oriented goals that can be later split into proper user stories, and can be assigned to sprints.
  - Optional but Powerful.

Artifacts:

- Backlog
- User Stories
- Sprints

Sprint 1 - Sprint Amsterdam (DONE)

(13 Sep 2021 to 26 Sep 2021)

Goal:

- To gain conceptual understanding of how containers, k8s/OpenShift works.
- To finish the onboarding setup.

Tasks:

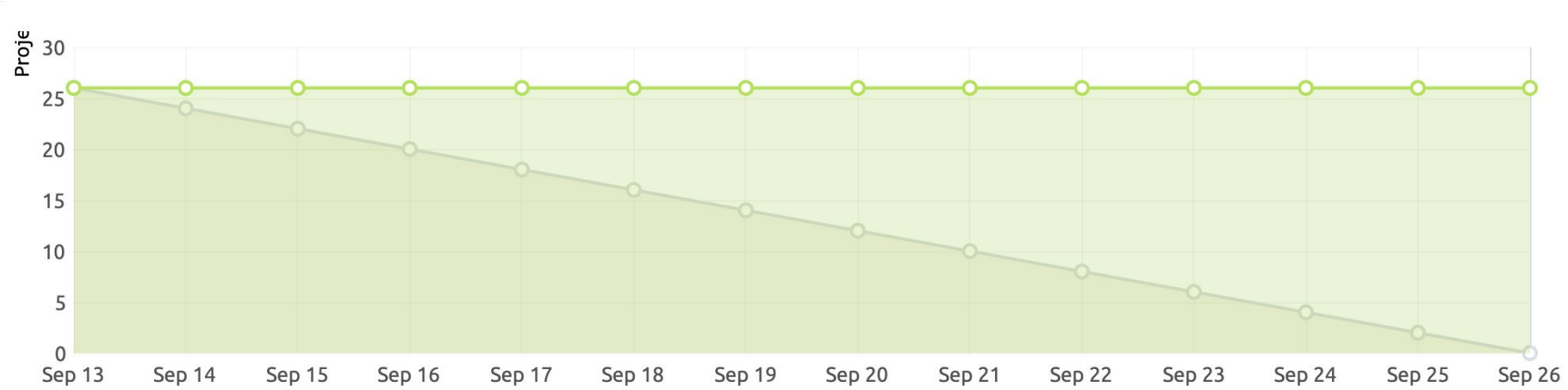
1. Revisit containers and openshift basics.
2. Write a project proposal.
3. Create a PR to access Operate First Github Repo.
4. Understand GitOps workflow.
5. Understand VPA architecture.

Outcome:

- All the team members are familiar with the fundamental concepts of containers, k8s/openshift, VPA, and GitOps.
- Successfully merged the PR and gained access to the Operate First Repo.
- Finished Project Proposal.

Sprint Amsterdam - Burndown

Dedicated 26 hours/points for this sprint.



Sprint 2 - Sprint Boston (IN PROGRESS)

(27 Sep 2021 to 11 Oct 2021)

Tasks:

1. Gain access to create an OCP cluster in MOC (In Progress).
2. SPIKE: Investigate various options to simulate fluctuating workload (New).
3. Configure and create a Deployment with a POC app with default/manual resource requests. (New).
4. Create a demo video for Sprint 1 (Ready for test).