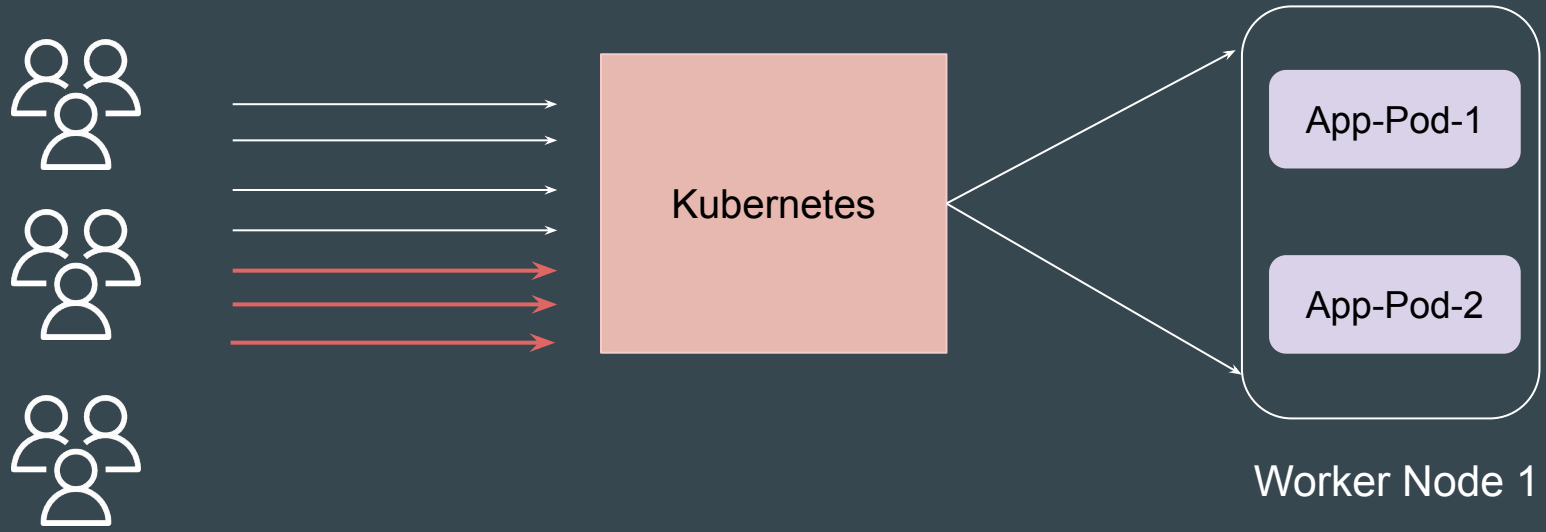
Horizontal Pod Autoscaling

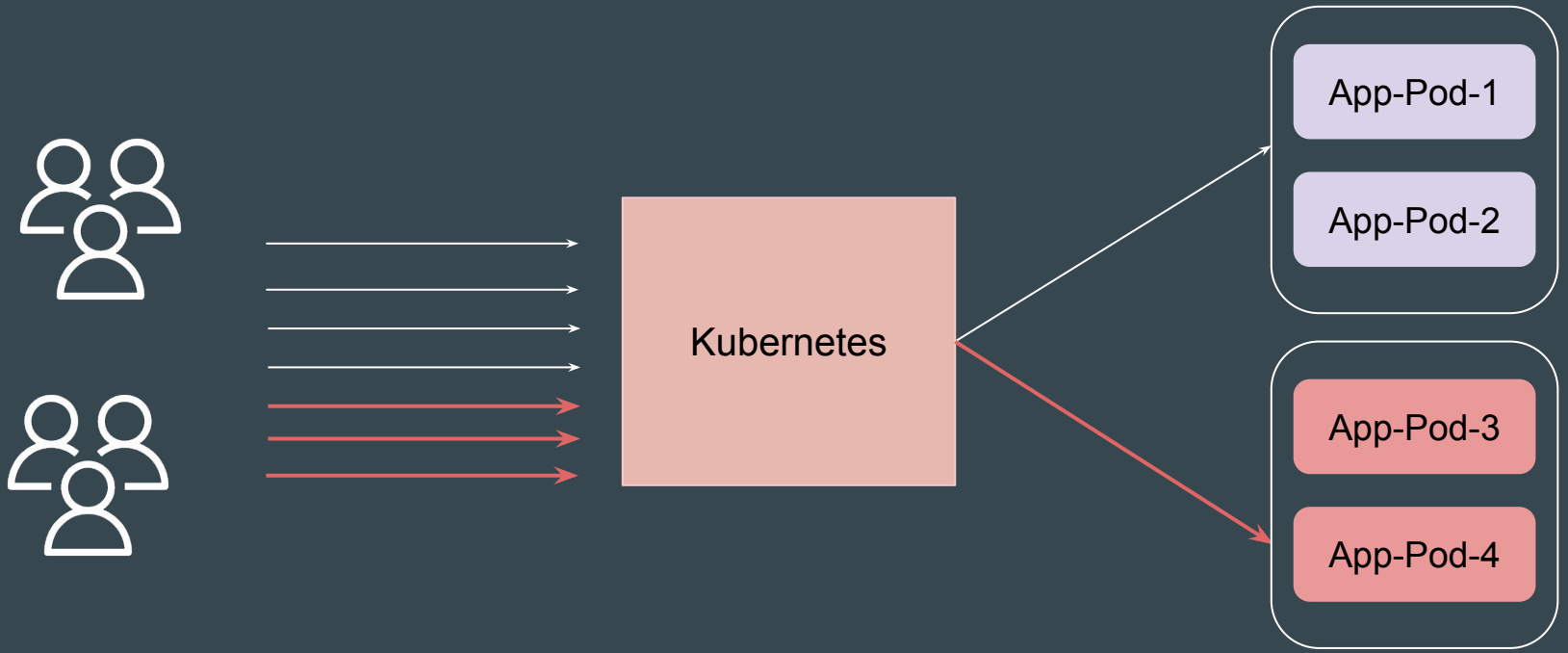
Setting the Base

Workloads in a Kubernetes cluster are not always static; they fluctuate based on user demand, traffic spikes, or computational requirements.



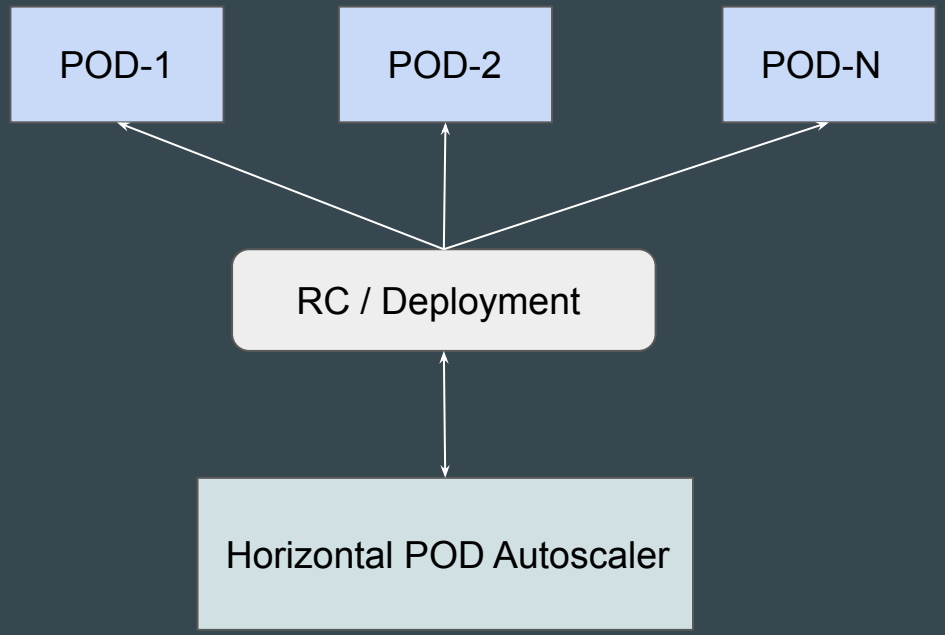
Introducing HPA

Horizontal Pod AutoScaler (HPA) automatically adjusts the number of running pods in a deployment based on observed CPU utilization, memory usage, or custom metrics.



The Problem that HPA Solves

1. Automatically scaling out (adding pods) when resource usage increases
2. Automatically scaling in (removing pods) when resource usage decreases



HPA Workflow

```
root@k8s:~# kubectl get hpa php-apache --watch
NAME         REFERENCE                  TARGETS        MINPODS   MAXPODS   REPLICAS   AGE
php-apache   Deployment/php-apache   cpu: 2%/50%    1         3         1          59s
php-apache   Deployment/php-apache   cpu: 380%/50%    1         3         1          90s
php-apache   Deployment/php-apache   cpu: 380%/50%    1         3         3          106s
php-apache   Deployment/php-apache   cpu: 902%/50%    1         3         3          2m1s
php-apache   Deployment/php-apache   cpu: 305%/50%    1         3         3          2m16s
php-apache   Deployment/php-apache   cpu: 323%/50%    1         3         3          2m31s
php-apache   Deployment/php-apache   cpu: 7%/50%      1         3         3          2m46s
php-apache   Deployment/php-apache   cpu: 2%/50%      1         3         3          3m1s
php-apache   Deployment/php-apache   cpu: 2%/50%      1         3         3          7m31s
php-apache   Deployment/php-apache   cpu: 2%/50%      1         3         1          7m46s
```

Points to Note

HPA does not apply to objects that cannot be scaled, like Daemonsets.

HPA - Stabilaization Window

Understanding the Challenge

Temporary metric drops can create rapid scale down of Pods which might cause service instability.

Stabilization Window can adds delay before Pods are downscaled.

```
root@k8s:~# kubectl get hpa php-apache --watch
NAME         REFERENCE                  TARGETS       MINPODS   MAXPODS   REPLICAS   AGE
php-apache   Deployment/php-apache   cpu: 2%/50%   1         3         1          59s
php-apache   Deployment/php-apache   cpu: 380%/50%   1         3         1          90s
php-apache   Deployment/php-apache   cpu: 380%/50%   1         3         3          106s
php-apache   Deployment/php-apache   cpu: 902%/50%   1         3         3          2m1s
php-apache   Deployment/php-apache   cpu: 305%/50%   1         3         3          2m16s
php-apache   Deployment/php-apache   cpu: 323%/50%   1         3         3          2m31s
php-apache   Deployment/php-apache   cpu: 7%/50%     1         3         3          2m46s
php-apache   Deployment/php-apache   cpu: 2%/50%     1         3         3          3m1s
php-apache   Deployment/php-apache   cpu: 2%/50%     1         3         3          7m31s
php-apache   Deployment/php-apache   cpu: 2%/50%     1         3         1          7m46s
```

Setting the Base

Stabilization Window configuration tells HPA to wait for a certain period before scaling down.

Default Stabilization Window	Description
Scale Up	0 (no-delay)
Scale Down	300 seconds (5 minutes delay)

```yaml
behavior:
  scaleDown:
    stabilizationWindowSeconds: 300
  scaleUp:
    stabilizationWindowSeconds: 0
```

After Stabilization Windows of 60 seconds (scaleDown)

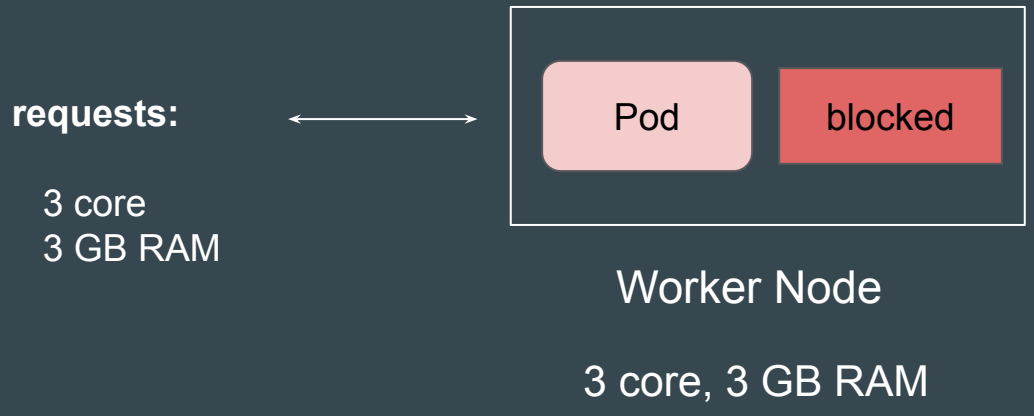
```
C:\kplabs-k8s>kubectl get hpa cpu-hpa --watch
NAME      REFERENCE                  TARGETS        MINPODS   MAXPODS   REPLICAS   AGE
cpu-hpa   Deployment/php-apache   cpu: 2%/50%    1         5         1          38s
cpu-hpa   Deployment/php-apache   cpu: 840%/50%   1         5         1          75s
cpu-hpa   Deployment/php-apache   cpu: 1700%/50%   1         5         5          90s
cpu-hpa   Deployment/php-apache   cpu: 1698%/50%   1         5         5          105s
cpu-hpa   Deployment/php-apache   cpu: 712%/50%    1         5         5          2m
cpu-hpa   Deployment/php-apache   cpu: 2%/50%      1         5         5          2m30s
cpu-hpa   Deployment/php-apache   cpu: 2%/50%      1         5         5          3m15s
cpu-hpa   Deployment/php-apache   cpu: 2%/50%      1         5         1          3m30s
```

Vertical Pod Auto-Scaler

Understanding the Challenge

If the Kubernetes Requests and Limits are set too low, your app might crash; if too high, you waste resources.

Even if your application is idle and using very little CPU or memory, the requested amount is blocked off and cannot be used to schedule other pods onto that node



Setting the Base

The Vertical Pod Autoscaler (VPA) is a Kubernetes component that automatically adjusts the CPU and memory requests for containers running within your pods.

```
  Recommendation:
    Container Recommendations:
      Container Name:  nginx
      Lower Bound:
        Cpu:     25m
        Memory:  262144k
      Target:
        Cpu:     25m
        Memory:  262144k
      Uncapped Target:
        Cpu:     25m
        Memory:  262144k
      Upper Bound:
        Cpu:     25m
        Memory:  262144k
```

Update Modes for VPA

updateMode	Description
Off	VPA only provides recommendations; no changes are applied.
Initial	VPA only assigns resource requests on pod creation and never changes them later.
Auto	VPA actively applies recommendations by evicting and restarting pods.