VPA

VPA automatically adjusts the **CPU and memory requests** of your pods based on actual usage over time. Unlike HPA (which scales the number of pods), VPA **resizes the pods themselves** to right-size resource allocation.

It’s perfect for:

* Long-running workloads (like your mysql pod)
* Jobs or CronJobs with unpredictable memory spikes (like your S3 backup job)
* Avoiding over-provisioning and wasted resources

Step-by-Step: Add VPA to Your Cluster

1. Clone and Deploy VPA

***git clone https://github.com/kubernetes/autoscaler.git cd autoscaler/vertical-pod-autoscaler/ ./hack/vpa-up.sh***

This installs:

* CRDs for VerticalPodAutoscaler
* The three core components
* RBAC and webhook configs

2. Verify It’s Running

***kubectl get pods -n kube-system | grep vpa***

You should see:

* vpa-admission-controller
* vpa-recommender
* vpa-updater

3. Create a VPA Object

Here’s an example for your mysql deployment:

***apiVersion: autoscaling.k8s.io/v1***

***kind: VerticalPodAutoscaler***

***metadata:***

***name: mysql-vpa***

***spec:***

***targetRef:***

***apiVersion: "apps/v1"***

***kind: Deployment***

***name: mysql***

***updatePolicy:***

***updateMode: "Auto"***

Apply it:

***kubectl apply -f mysql-vpa.yaml***

4. View Recommendations

After a few minutes of usage:

kubectl describe vpa mysql-vpa

You’ll see:

* **Target**: what VPA recommends now
* **Lower/Upper Bound**: safe min/max ranges
* **Uncapped Target**: raw recommendation without constraints

Modes You Can Use

| **Mode** | **Behavior** |
| --- | --- |
| Auto | VPA evicts and reschedules pods with new resources (default) |
| Recreate | Same as Auto, but more explicit — pods are restarted to apply changes |
| Initial | Only sets resources at pod creation, never updates existing pods |
| Off | Just gives recommendations, doesn’t apply them |

Want me to generate a VPA manifest for your mysql or backup-upload CronJob next? Or show how to combine VPA with PodDisruptionBudget for safe evictions? Let’s fine-tune this cluster like a pro