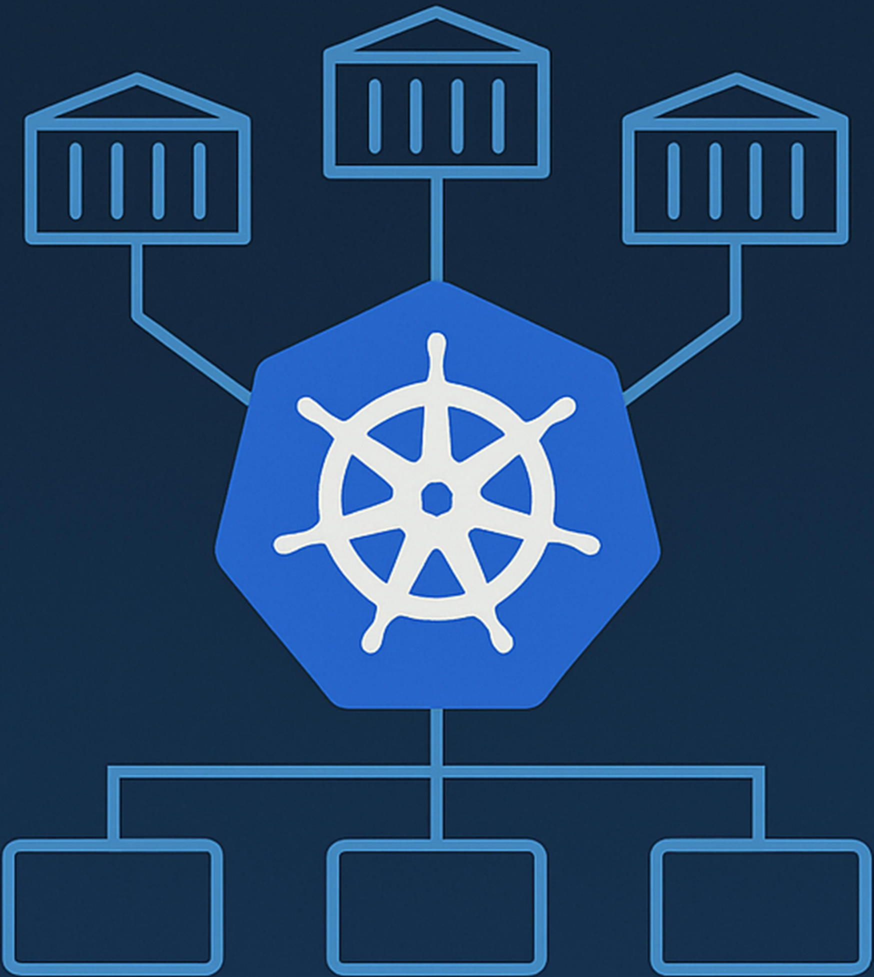
5 Real-Time Kubernetes Interview Questions & Answers

Sharpen Your Skills for Your Next DevOps Interview



Venkatesh Jilakarra

1) What is the Difference Between Ingress and Load Balancer Services?

Answer:

- **LoadBalancer Service**: Exposes a service externally via a public IP (cloud provider dependent).
- **Ingress**: Routes HTTP/S traffic to internal services based on host/path rules and is managed by an Ingress Controller (e.g., NGINX, Traefik, AWS ALB).

Example: Ingress YAML

yaml
CopyEdit

```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: my-ingress
spec:
  rules:
  - host: myapp.example.com
    http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          service:
            name: my-service
            port:
              number: 80
```

Use Case:

Use **Ingress** to route multiple applications through a single entry point instead of creating multiple LoadBalancer services.

2) How Does Kubernetes Handle Node Failures?

Answer:
Kubernetes handles node failure in the following steps:

1. The **kubelet** stops sending heartbeats, and the node is marked as **Not Ready**.
2. Pods on the failed node are marked as being in the **Unknown** state.
3. The **Controller Manager** attempts to reschedule the affected pods on healthy nodes.
4. If **Pod Disruption Budgets (PDBs)** are configured, Kubernetes ensures availability before evicting the pods.
5. The **Cluster Autoscaler**, if enabled, may replace the failed node.

Best Practice:

Use **Node Affinity** and **Taints/Tolerations** to fine-tune scheduling behavior.

3) What Are Kubernetes Resource Quotas and Limit Ranges?

Answer:
These features help control and limit resource consumption at the namespace level.

- **Resource Quotas**: Define limits on the total resources that can be used in a namespace (e.g., CPU, memory, number of pods).
- **Limit Ranges**: Define minimum/maximum/default resource usage per pod or container.

Example: ResourceQuota YAML

yaml
CopyEdit

```
apiVersion: v1
kind: ResourceQuota
metadata:
  name: dev-quota
  namespace: dev
spec:
  hard:
    pods: "10"
    requests.cpu: "4"
    limits.cpu: "10"
```

4) How Do You Perform Zero-Downtime Deployments in Kubernetes?

Answer:
Use **Rolling Updates** with appropriate configuration to avoid disruption during deployments.

Best Practices:

- Set maxUnavailable: 0 to ensure no pods are terminated before a new one is ready.
- Configure **readiness probes** to only serve traffic from healthy pods.
- Use **preStop hooks** for graceful termination of existing pods.

Example: Rolling Update Strategy

yaml
CopyEdit

```yaml
strategy:
  type: RollingUpdate
  rollingUpdate:
    maxUnavailable: 0
    maxSurge: 1
```

For Mission-Critical Workloads:

Adopt **Canary** or **Blue-Green** deployment strategies using tools like **ArgoCD**, **Flagger**, or **Istio**.

5) What is a Kubernetes DaemonSet, and When Should You Use It?

Answer:
A **DaemonSet** ensures a specific pod is scheduled on all (or selected) nodes in the cluster. This is particularly useful for running background services or system-level daemons.

Common Use Cases:

- Deploying **Fluentd** for log aggregation.
- Running **Node Exporter** for Prometheus-based monitoring.
- Installing **CNI plugins** like Calico or Cilium on each node.