
# 🔥 Kubernetes Interview Questions & Answers

## **1. What is Kubernetes, and why do we use it?**

👉 **Answer:**
Kubernetes is an **open-source container orchestration platform** that automates the deployment, scaling, and management of containerized applications.

* Without Kubernetes → you manually run containers, manage scaling, networking, etc.
* With Kubernetes → you get automated scheduling, self-healing, service discovery, storage, and scaling.

📌 Example: Instead of manually running 50 containers with Docker, Kubernetes can scale them automatically when CPU usage is high.

## **2. Explain Kubernetes architecture.**

👉 **Answer:**

* **Master Node (Control Plane):**

  * **API Server** – Entry point for all commands (`kubectl`).
  * **etcd** – Key-value store (cluster state).
  * **Controller Manager** – Ensures desired state.
  * **Scheduler** – Assigns Pods to nodes.

* **Worker Node:**

  * **Kubelet** – Talks to API server, runs pods.
  * **Kube-proxy** – Networking, service load balancing.
  * **Container runtime** – Runs containers (Docker/Containerd).

📌 Interview Trick: “Master maintains **desired state**, Worker enforces it.”

## **3. What is a Pod in Kubernetes?**

👉 **Answer:**

* Pod is the **smallest deployable unit** in Kubernetes.
* A Pod can contain **one or more containers** that share the same **network namespace** and **storage volumes**.

📌 Example: If you have an app + sidecar (logging container), you put both in the same Pod.

## **4. Difference between Deployment, ReplicaSet, StatefulSet, and DaemonSet?**

👉 **Answer:**

* **ReplicaSet** → Ensures a specified number of identical Pods are running.
* **Deployment** → Manages ReplicaSets, supports rolling updates & rollbacks.

* **StatefulSet** → For stateful apps (DBs), provides stable network IDs & persistent storage.
* **DaemonSet** → Runs **one Pod per Node** (e.g., logging, monitoring).

📌 Example:

* Nginx web app → Deployment.
* Kafka/Zookeeper → StatefulSet.
* Node logging agent → DaemonSet.

## **5. How does Kubernetes handle Service Discovery?**

👉 **Answer:**

* Kubernetes provides **Services** to expose Pods.
* Each Service gets a **ClusterIP** (internal) or **LoadBalancer/NodePort** (external).
* DNS (via CoreDNS) maps service names → IP.

📌 Example: Pod connects to `http://mysql-service:3306` instead of IP.

## **6. What is a Namespace in Kubernetes?**

👉 **Answer:**
Namespace is a **logical isolation** inside a cluster.

* Useful for multi-team or multi-environment setups (dev, prod).
* Default namespaces: `default`, `kube-system`, `kube-public`.

📌 Example: Deploy dev and prod versions of the same app in different namespaces.

## **7. How do you debug a Pod stuck in CrashLoopBackOff?**

👉 **Answer:**

1. `kubectl describe pod <pod>` → Check events.
2. `kubectl logs <pod>` → Check app logs.
3. `kubectl exec -it <pod> -- sh` → Debug inside container.
4. Check probes (liveness/readiness).
5. Check ConfigMaps/Secrets are mounted correctly.

## **8. Explain ConfigMap vs Secret.**

👉 **Answer:**

* **ConfigMap** → Stores non-sensitive configuration (key-value).
* **Secret** → Stores **sensitive data** (passwords, tokens), base64 encoded.

📌 Example:

* ConfigMap → `APP_ENV=production`
* Secret → `DB_PASSWORD=mysecret123`

## **9. What is Ingress in Kubernetes?**

👉 **Answer:**
Ingress exposes HTTP/HTTPS routes from outside cluster → services inside cluster.

* Needs an **Ingress Controller** (nginx/traefik).
* Supports load balancing, SSL termination, path-based routing.

📌 Example:

* `/app1` → service1
* `/app2` → service2

## **10. How does Kubernetes handle storage?**

👉 **Answer:**

* **PV (Persistent Volume):** Actual storage (NFS, EBS, etc.).
* **PVC (Persistent Volume Claim):** Request for storage by Pod.
* **StorageClass:** Defines dynamic storage provisioning.

📌 Example:
Pod claims 10Gi PVC → PVC bound to 10Gi PV/EBS volume.

## **11. What is a Kubernetes Operator?**

👉 **Answer:**
Operator = Custom controller that automates management of complex apps (like DBs).

* Uses **CRDs (Custom Resource Definitions)**.
* Example: MongoDB Operator can handle backup, scaling, failover.

## **12. How does Kubernetes achieve High Availability?**

👉 **Answer:**

* Control Plane → Multiple masters + etcd cluster.
* Worker Nodes → Apps spread across nodes.
* Self-healing → Restarts failed Pods.
* Load balancer distributes traffic across nodes.

## **13. How do you upgrade a Kubernetes cluster?**

👉 **Answer:**

1. Upgrade **control plane components** (API Server, etcd).
2. Upgrade **kubelet + kube-proxy** on worker nodes.
3. Upgrade deployments with rolling updates.

📌 Tool: `kubeadm upgrade apply <version>`

## **14. What is RBAC in Kubernetes?**

👉 **Answer:**

* RBAC = Role-Based Access Control.
* Defines **who** can do **what** on **which resources**.
* Uses: Role, ClusterRole, RoleBinding, ClusterRoleBinding.

📌 Example: Allow dev team to only `get/list` Pods in `dev` namespace.

## **15. Difference between Helm and Kustomize?**

👉 **Answer:**

* **Helm** → Package manager for Kubernetes (charts). Good for complex apps.
* **Kustomize** → Native k8s templating (overlays). Simpler customization.

# 🎯 Pro Tip for Interviews

When asked **“How do you debug…?”** or **“What happens if…?”**,
👉 Always answer in a **step-by-step approach** (use commands like `kubectl get`, `describe`, `logs`, `exec`).

Perfect 🚀 Let’s build you a **Kubernetes Interview Cheat Sheet** —
👉 Around **50+ commonly asked questions with short crisp answers** that you can **revise quickly before interviews**.

# 📝 Kubernetes Interview Q\&A Cheat Sheet (50+)

## **Basics**

1. **What is Kubernetes?**
   Container orchestration platform for automating deployment, scaling, and management of containerized apps.

2. **What is a Pod?**
   Smallest deployable unit, can contain one or more containers sharing network & storage.

3. **What is a Node?**
   Worker machine (VM/physical) that runs Pods.

4. **Difference between Master & Worker Node?**

   * Master: Control plane (API Server, etcd, Scheduler, Controller Manager).
   * Worker: Runs workloads (Pods).

5. **What is kubelet?**
   Agent on each node that ensures Pods are running.

6. **What is kube-proxy?**
   Handles networking, load balancing for Services.

7. **What is etcd?**
   Distributed key-value store holding cluster state.

## **Controllers**

8. **What is a Deployment?**
   Manages ReplicaSets, provides rolling updates & rollback.

9. **What is a ReplicaSet?**
   Ensures a fixed number of Pods are running.

10. **What is a StatefulSet?**
    Manages stateful apps, provides stable IDs & storage.

11. **What is a DaemonSet?**
    Ensures one Pod per node (e.g., logging/monitoring).

12. **What is a Job vs CronJob?**

* Job: Runs once until completion.
* CronJob: Runs periodically on schedule.

## **Networking**

13. **What is a Service in Kubernetes?**
    Abstract way to expose Pods.

14. **Types of Services?**

* ClusterIP (default, internal).
* NodePort (exposes via node IP\:port).
* LoadBalancer (cloud LB).
* ExternalName (DNS alias).

15. **What is Ingress?**
    HTTP/HTTPS routing to Services (supports SSL, path-based routing).

16. **What is CoreDNS?**
    DNS server inside cluster, provides service discovery.

## **Storage**

17. **What is a PV (Persistent Volume)?**
    Actual storage resource (EBS, NFS, etc.).

18. **What is a PVC (Persistent Volume Claim)?**
    Request for storage by a Pod.

19. **What is StorageClass?**
    Defines dynamic provisioning of storage.

20. **StatefulSet vs PVC?**
    StatefulSets automatically map Pods to PVCs.

## **Configuration**

21. **What is a ConfigMap?**
    Stores non-sensitive config (key-value).

22. **What is a Secret?**
    Stores sensitive data (passwords, tokens). Base64 encoded.

23. **How to inject configs into Pods?**

* Env variables.
* Mounted volumes.

## **Scheduling**

24. **What is a Scheduler?**
    Assigns Pods to nodes.

25. **What are Taints & Tolerations?**

* Taint: Repels Pods from a node.
* Toleration: Allows Pod to run on tainted nodes.

26. **What are Node Affinity & Anti-Affinity?**
    Rules to co-locate or separate Pods on nodes.

27. **What is Pod Priority & Preemption?**
    Higher-priority Pods can evict lower-priority ones during resource shortage.

## **Health & Scaling**

28. **What is a Liveness Probe?**
    Checks if container is alive, restarts if fails.

29. **What is a Readiness Probe?**
    Checks if container is ready to serve traffic.

30. **What is a Startup Probe?**
    Special probe for slow-start apps.

31. **What is HPA (Horizontal Pod Autoscaler)?**
    Scales Pods based on metrics (CPU, memory, custom).

32. **What is VPA (Vertical Pod Autoscaler)?**
    Adjusts resource requests/limits of Pods.

33. **What is Cluster Autoscaler?**
    Adds/removes nodes based on workload demand.

## **Security**

34. **What is RBAC?**
    Role-Based Access Control, defines permissions.

35. **What are Roles & ClusterRoles?**

* Role: Permissions in a namespace.
* ClusterRole: Permissions cluster-wide.

36. **What are Service Accounts?**
    Provide identity to Pods for API access.

37. **What is Pod Security Policy (PSP)?**
    Restricts security settings (deprecated, now replaced by OPA/Gatekeeper/PSA).

38. **How to secure Secrets?**
    Use KMS (cloud key management) or SealedSecrets.

## **Debugging**

39. **How to debug a Pod in CrashLoopBackOff?**

* `kubectl describe pod` (events)
* `kubectl logs`
* `kubectl exec`
* Check probes/configs.

40. **How to check cluster health?**

* `kubectl get componentstatus`
* `kubectl get nodes`
* Metrics-server & monitoring tools.

41. **How to debug networking issues?**

* `kubectl exec` → ping/curl.
* Check `kube-proxy`, CNI plugin.
* `kubectl describe svc/pod`.

## **Advanced**

42. **What is a Custom Resource Definition (CRD)?**
    Extend Kubernetes API with custom objects.

43. **What is an Operator?**
    Controller + CRD that automates app lifecycle.

44. **What is Helm?**
    Package manager for Kubernetes (Helm charts).

45. **Helm vs Kustomize?**

* Helm → Templates + package manager.
* Kustomize → Native YAML overlay system.

46. **What is a Sidecar container?**
    Helper container in a Pod (logging, proxy).

47. **What is a Service Mesh?**
    Layer for service-to-service communication (Istio/Linkerd).

48. **What is Istio used for?**
    Traffic management, security, observability.

## **Cluster Management**

49. **How to upgrade Kubernetes cluster?**

* Upgrade control plane (`kubeadm upgrade`).
* Upgrade kubelet on nodes.
* Rolling restart of workloads.

50. **How to achieve High Availability in Kubernetes?**

* Multi-master setup.
* etcd cluster.
* Pod replicas across nodes.
* Load balancer.

51. **How to backup etcd?**

* `etcdctl snapshot save <file>`
* Restore with `etcdctl snapshot restore`.

52. **Difference between Kubernetes & Docker Swarm?**

* Swarm → Simple, less features.
* Kubernetes → Complex, production-grade.

✅ This gives you **50+ questions with crisp answers**.
👉 For practice: Try answering in **2–3 sentences max** (interviews prefer concise answers).

Do you want me to also create a **separate "scenario-based questions" list** (like “What if a pod keeps restarting?”, “How do you scale a service under high traffic?”), since those are very common in **senior-level interviews**?

Perfect 👍 Scenario-based questions are where most **interviewers test real-world Kubernetes knowledge**.
Here’s a **Kubernetes Scenario-Based Q\&A Cheat Sheet** 🚀

# 📝 Kubernetes Scenario-Based Interview Questions & Answers

## **Pod / Deployment Issues**

### 1. **A Pod is stuck in `CrashLoopBackOff`. How do you troubleshoot?**

* Check logs: `kubectl logs <pod> -p` (previous container logs).
* Check events: `kubectl describe pod <pod>`.
* Common causes: wrong image, bad config/env, failing probes.
* Fix the root cause → redeploy.

### 2. **A Pod is in `Pending` state. Why?**

* Not enough resources on nodes (CPU/memory).
* No matching node selector/affinity.

* PVC claim not bound.
* Check with:

  ```bash
  kubectl describe pod <pod>
  kubectl get events --sort-by=.metadata.creationTimestamp
  ```

### 3. **How to debug a Pod with no logs (container not starting)?**

* Run:

  ```bash
  kubectl describe pod <pod>
  kubectl get nodes
  ```
* If image not found → check imagePullSecrets.
* If initContainer failing → check logs of initContainer.

### 4. **How do you roll back a failed Deployment update?**

```bash
kubectl rollout undo deployment <deployment-name>
kubectl rollout status deployment <deployment-name>
```

## **Networking Issues**

### 5. **A Service is not reachable. What do you check?**

* Check if Pods are running & Ready.
* Verify Service selectors match Pod labels.
* Test from inside cluster:

  ```bash
  kubectl exec -it <pod> -- curl <svc-name>:<port>
  ```
* If external: check Ingress/LoadBalancer rules.

### 6. **Pod A cannot talk to Pod B. What do you check?**

* Check both Pods are running.
* Validate DNS: `kubectl exec <pod> -- nslookup <service-name>`.
* Check `NetworkPolicy` (may block traffic).
* Check kube-proxy & CNI plugin logs.

## **Storage Issues**

### 7. **Pod using PVC is stuck in Pending. Why?**

* PVC not bound to PV.
* No matching StorageClass.
* Check:

```bash
  kubectl get pvc
  kubectl describe pvc <pvc-name>
  ```

### 8. **How do you resize a PVC?**

* Update PVC YAML with new storage size.
* Only works if `allowVolumeExpansion: true` in StorageClass.
* Example:

  ```yaml
  resources:
    requests:
      storage: 10Gi
  ```

## **Scaling & Performance**

### 9. **Your app is experiencing high traffic. How do you scale it?**

* **Manual scaling:**

  ```bash
  kubectl scale deployment myapp --replicas=10
  ```
* **Auto scaling:**

  ```bash
  kubectl autoscale deployment myapp --cpu-percent=70 --min=3 --max=10
  ```
* Also check cluster autoscaler for node scaling.

### 10. **How do you debug an OOMKilled Pod?**

* Check Pod status: `kubectl describe pod <pod>`.
* If OOMKilled → container exceeded memory limit.
* Fix by:

  * Increasing memory limits.
  * Optimizing app memory usage.

## **Cluster Management**

### 11. **A node is `NotReady`. How do you fix it?**

* Check node status: `kubectl describe node <node>`.
* Common causes: kubelet stopped, disk full, network issue.
* Restart kubelet:

  ```bash
  systemctl status kubelet
  systemctl restart kubelet
  ```

### 12. **How to drain a node for maintenance?**

```bash
kubectl drain <node> --ignore-daemonsets --delete-emptydir-data
```

Bring it back:

```bash
kubectl uncordon <node>
```

### 13. **How to recover if etcd is corrupted?**

* Take backup: `etcdctl snapshot save backup.db`.
* Restore cluster:

  ```bash
  etcdctl snapshot restore backup.db
  ```

## **Security & Access**

### 14. **How do you restrict a user to only one namespace?**

* Create a Role & RoleBinding. Example:

  ```bash
  kubectl create role dev-role --verb=get,list,watch --resource=pods --namespace=dev
  kubectl create rolebinding dev-bind --role=dev-role --user=dev-user --namespace=dev
  ```

### 15. **How do you secure Secrets in Kubernetes?**

* Use SealedSecrets / External Secrets.
* Enable encryption at rest in `kube-apiserver`.
* Use cloud KMS integration (AWS KMS, GCP KMS).

## **Advanced / Real-World**

### 16. **How do you debug high pod-to-pod latency?**

* Check CNI plugin performance.
* Use `kubectl exec` → `ping`, `iperf`.
* Check network policies.
* Monitor with tools (Cilium, Istio).

### 17. **How do you perform a rolling deployment without downtime?**

* Use Deployment with `maxUnavailable=0`, `maxSurge=1`.
* Example:

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

### 18. **How do you upgrade Kubernetes with zero downtime?**

* Upgrade control plane nodes one by one.
* Use `kubectl drain` before upgrade.
* Upgrade worker nodes gradually.
* Rely on Deployment replicas for HA.

### 19. **How do you handle app secrets across environments (dev, staging, prod)?**

* Use External Secrets Operator with cloud secret stores.
* Use Kustomize overlays for environment-specific configs.
* Never hardcode in YAML.

### 20. **What do you do if an entire namespace crashes?**

* Check `kubectl get events -n <namespace>`.
* Check if resource quota limits were exceeded.
* Verify etcd consistency.
* Restore namespace objects from backup (Velero).

✅ With these **20+ scenario-based answers** you can confidently handle **practical Kubernetes interview questions**.

👉 Do you want me to now prepare a **mock interview script** (where I ask you questions like an interviewer and you answer), so you can **practice live interview flow**?