Kubernetes Interview Questions

1. Introduction to Kubernetes

Kubernetes is an open-source container orchestration platform that automates the deployment, scaling, and management of containerized applications. It abstracts the underlying infrastructure and provides a consistent API to deploy applications across different environments—on-premises or in the cloud. Kubernetes ensures high availability, fault tolerance, self-healing, and easy rollouts/rollbacks, which makes it the industry standard for managing container-based workloads.

1. What is Kubernetes and why is it used

Kubernetes (K8s) is used to manage large-scale containerized applications in a distributed system. It allows developers and operations teams to deploy applications in a declarative manner, automate scaling, monitor health, handle failovers, and manage configurations. It simplifies infrastructure complexity and helps teams adopt DevOps and GitOps practices efficiently.

1. What are Pods in Kubernetes

A Pod is the smallest and simplest Kubernetes object. It encapsulates one or more containers that share the same network IP, hostname, storage, and namespace. Containers in the same Pod can communicate with each other using localhost and share volumes. Pods are ephemeral; if a Pod dies, Kubernetes replaces it with a new one, not the same one.

1. What is a Kubernetes Node

A Node is a worker machine in Kubernetes where containers are scheduled and run. It can be a physical server or a virtual machine. Each Node runs at least:

Kubelet: the agent that communicates with the control plane.

Kube-proxy: handles networking and service routing.

Container runtime (e.g., containerd or Docker): executes containers.

1. What is a Deployment in Kubernetes

A Deployment is a controller that provides declarative updates for Pods and ReplicaSets. It defines the desired state of an application (e.g., number of replicas, container image) and automatically manages the creation and rollout of Pods to match that state. Deployments support rolling updates, rollbacks, and pause/resume operations.

1. How does Kubernetes perform load balancing

Kubernetes performs load balancing through Services. A Service is an abstraction that defines a policy to access a set of Pods. Kubernetes uses kube-proxy to distribute incoming network traffic to the appropriate Pods, usually using round-robin or IP-hashing. For external traffic, LoadBalancer and Ingress resources are used to distribute traffic to backend Pods.

1. A Pod keeps crashing. How do you troubleshoot it

To troubleshoot a crashing Pod:

Use kubectl describe pod <pod-name> to check events and reasons (e.g., image pull error, OOM, probe failures).

Use kubectl logs <pod-name> to view container logs.

Check exit codes and container statuses.

Use kubectl get events to find scheduling/resource errors.

Review resource limits and requests.

Check readiness/liveness probes.

1. What is a ConfigMap in Kubernetes

A ConfigMap is used to inject configuration data (non-sensitive) into Pods. It allows decoupling of configuration from container images, making apps more portable. ConfigMaps can be used to provide environment variables, command-line arguments, or configuration files to containers.

1. A service is not reaching the correct Pods. How do you debug it

To debug service-to-pod communication issues:

Check the Service selector and match it against Pod labels.

Use kubectl get endpoints <service-name> to verify if endpoints exist.

Confirm Pod readiness; only ready Pods are part of the Service.

Use kubectl describe service <service-name> to examine configuration.

Test networking with tools like nslookup, curl, or kubectl exec.

1. What is the purpose of Namespaces in Kubernetes

Namespaces are a way to divide cluster resources among multiple users or teams. They provide scope for names and resources like Pods, Services, and ConfigMaps. Namespaces are useful for:

Resource isolation.

Access control using RBAC.

Managing multiple environments (dev, test, prod).

1. How do you scale a Kubernetes deployment

Scaling can be done manually using:

Kubectl scale deployment <name> --replicas=5

Or automatically using the Horizontal Pod Autoscaler (HPA), which adjusts the number of Pod replicas based on CPU/memory usage or custom metrics.

1. What is a StatefulSet in Kubernetes

A StatefulSet is used to manage stateful applications. Unlike Deployments, each Pod in a StatefulSet has:

A persistent identity (stable hostname).

Dedicated persistent storage (via PVC).

Ordered, graceful deployment and scaling. Useful for databases, Kafka, and apps needing stable network identities.

1. How does Kubernetes handle rolling updates and rollbacks

Kubernetes supports zero-downtime rolling updates by gradually replacing old Pods with new ones. This is controlled via the Deployment strategy (maxUnavailable, maxSurge). If something goes wrong, you can rollback using:

Kubectl rollout undo deployment <name>

Rollouts can also be paused and resumed.

1. What is a DaemonSet

A DaemonSet ensures that a specific Pod runs on all (or selected) Nodes in the cluster. It’s commonly used for:

Log collection agents (Fluentd, Logstash)

Monitoring (Prometheus Node Exporter)

Networking (CNI plugins)

1. A Pod is stuck in Pending state. How do you debug it

Check:

Kubectl describe pod for events like scheduling issues.

Node resource availability (CPU, memory).

PVC binding—storage class might be missing or unavailable.

Taints and tolerations.

Node selectors or affinity/anti-affinity rules.

1. How do you secure Kubernetes Secrets

Store secrets using the Secret resource.

Enable encryption at rest for secrets in etcd.

Use RBAC to restrict access to secrets.

Consider external secret managers (Vault, AWS Secrets Manager) for improved security.

1. A deployment update caused downtime. How do you prevent this

To avoid downtime:

Use proper liveness/readiness probes.

Configure rolling update strategy (maxUnavailable, maxSurge).

Use canary or blue-green deployments for safer rollouts.

Monitor app behavior during the rollout.

1. What is Persistent Volume (PV) and Persistent Volume Claim (PVC)

PV: A cluster-managed storage resource.

PVC: A user request for storage. PVC binds to a PV that matches its requirements. This abstraction decouples storage provisioning from usage.

1. How do you set up autoscaling in Kubernetes

Use the Horizontal Pod Autoscaler (HPA) by enabling the metrics server:

Kubectl autoscale deployment <name> --cpu-percent=80 –min=1 –max=10

For Node-level scaling, use the Cluster Autoscaler which adjusts the number of nodes in response to pending Pods.

1. What is a Network Policy in Kubernetes

Network Policies define rules for network traffic between Pods. By default, all traffic is allowed. With policies, you can restrict access to/from specific Pods, namespaces, or IP blocks. They help enforce the principle of least privilege at the network layer.

1. How do you expose a Kubernetes application externally

You can expose applications using:

NodePort: Exposes the service on a port on each node.

LoadBalancer: Provisions an external load balancer (cloud providers).

Ingress: Manages external HTTP/HTTPS traffic via a single entry point, supporting path and host-based routing.

1. How does Kubernetes handle node failures

Kubelet regularly sends heartbeats to the control plane.

If a Node fails, Pods on it are marked “Unknown” or “NotReady.”

The scheduler reschedules those Pods on healthy nodes.

DaemonSets and StatefulSets have special behavior that may require manual intervention for recovery.

1. What are Custom Resource Definitions (CRDs)

CRDs allow users to extend Kubernetes by defining custom resources. Paired with custom controllers or Operators, they allow you to manage domain-specific logic and automate complex workflows. For example, you can create a CRD for a “Database” and manage its lifecycle declaratively.

1. How do you implement GitOps in Kubernetes

GitOps is the practice of using Git as the source of truth for infrastructure and application configuration. Tools like Argo CD or Flux sync changes from Git repositories to Kubernetes clusters. This ensures auditability, version control, and consistency across environments.

1. How do you troubleshoot a memory leak in a Kubernetes application

Use kubectl top or Prometheus to monitor memory usage.

Check container logs for out-of-memory (OOM) errors.

Add memory limits to enforce boundaries and trigger restarts.

Use profiling tools (e.g., heap dumps, Valgrind, pprof).

Run the app locally with monitoring tools to replicate and analyze leaks.

1. What is a Kubernetes Operator

An Operator is a method of packaging, deploying, and managing a Kubernetes application. It uses CRDs and custom controllers to encode operational knowledge, automating tasks like backup, scaling, upgrades, and failover for complex, stateful applications.

1. How does Kubernetes RBAC work

RBAC (Role-Based Access Control) defines what actions users can perform on which resources. It uses:

Roles/ClusterRoles: Define permissions.

RoleBindings/ClusterRoleBindings: Assign those roles to users, groups, or service accounts. This provides fine-grained access control across the cluster or within specific namespaces.

1. How do you secure Kubernetes clusters

Enable RBAC and limit access using least privilege.

Encrypt etcd and enable audit logging.

Use network policies to isolate workloads.

Scan images for vulnerabilities.

Use tools like OPA Gatekeeper for policy enforcement.

Rotate credentials regularly and keep the cluster updated.

1. Your cluster is running slow. How do you optimize it

Monitor resource usage (CPU, memory, disk, network).

Use kubectl top or Prometheus + Grafana.

Optimize workload placement with affinities and taints.

Tune autoscaling parameters.

Check API server and etcd performance.

Limit excessive logging or metric collection.

Right-size Pods and nodes.

1. How do you handle multi-cluster Kubernetes deployments

Multi-cluster setups can improve availability and isolation. Handle them using:

Cluster Federation: Manage multiple clusters from a single control plane.

Service Mesh (Istio, Linkerd): For cross-cluster service discovery and security.

GitOps tools: Use ArgoCD’s ApplicationSets or Flux for managing apps in multiple clusters.

DNS and load balancers: Route traffic across clusters efficiently.

1. What are Kubernetes Admission Controllers

Admission Controllers intercept requests to the Kubernetes API server before they’re persisted. They can validate (e.g., deny privileged containers), mutate (e.g., inject sidecars), or enforce policies. Common ones include:

NamespaceLifecycle

LimitRanger

PodSecurityPolicy

ValidatingAdmissionWebhook

MutatingAdmissionWebhook