FROM ZERO TO KUBERNETES: CORE CONCEPTS EVERY FRESHER MUST KNOW

✅ Kubernetes Basics (Must Know)

1) What is Kubernetes?

Kubernetes (K8s) is a tool that deploys, manages, scales, and heals containers automatically.
It helps run applications reliably in many servers.

✅ Example: If a container crashes, Kubernetes creates a new one automatically.

---

2) What is a Cluster?

A cluster is a group of machines (nodes) managed together by Kubernetes.
It has:

- **Control Plane (Master)** → manages
- **Worker Nodes** → runs apps

✅ Cluster = Full Kubernetes environment.

---

3) What is a Node?

A node is a machine (VM/Server) where Kubernetes runs your applications.
It can be:

- Control plane node (master)
- Worker node

✅ Node = one server inside the cluster.

---

4) What is a Pod?

A pod is the smallest unit in Kubernetes that runs your container.
One pod can contain:

- 1 container (mostly)
- or multiple containers (rare)

✅ Pod = container running inside Kubernetes.

📌 Pod is **ephemeral** (temporary). It can be deleted and created again.

✅ Control Plane (Master) Components

5) What are the components of Control Plane?

Control plane is the **brain of Kubernetes**.

✅ (1) kube-apiserver

Main entry point of Kubernetes.
All commands go through it.

✅ Example: kubectl get pods → goes to API server.

✅ (2) etcd

Database of Kubernetes.
Stores complete cluster state.

✅ Stores: pods, deployments, services, secrets, configmaps, nodes, etc.

✅ (3) kube-scheduler

Chooses which worker node will run a pod.
It selects the best node based on CPU/RAM availability.

✅ (4) kube-controller-manager

Runs controllers to maintain desired state.
If something fails, it corrects it.

✅ Example: If pod dies → it creates new pod.

✅ (Optional) cloud-controller-manager

Connects Kubernetes to cloud providers
(AWS/Azure/GCP).

✅ Example: Creates cloud load balancers, volumes.

✅ Worker Node Components

6) What are the components of Worker Node?

Worker node is where your app runs.

✅ (1) kubelet

Agent running on every node.
It ensures pods/containers are running properly.

✅ Talks to API server.

✅ (2) Container Runtime

Runs the container inside pod.
Example:

- containerd ✅
- CRI-O

- Docker (old)

✅ (3) kube-proxy

Handles networking rules and service traffic routing.

✅ Example: Service IP → forwards traffic to pod.

✅ (4) Pods

Your application runs here.
Example: 2048 pod / Nginx pod.

✅ Namespace & Resource Concepts

7) What is a Namespace?

Namespace is like a folder inside a Kubernetes cluster.
It helps separate applications/environments.

✅ Example:

- dev
- test
- prod
- game-2048

---

8) What is Metadata in Kubernetes?

Metadata is information about a Kubernetes resource.
Used for identification.

✅ Includes:

- name
- namespace
- labels
- annotations

Example:

```
metadata:
  name: myapp
  namespace: dev
```

---

9) What are Labels?

Labels are key-value tags used to identify and group resources.

✅ Example:

```
labels:
  app: nginx
```

Services use labels to find pods.

---

10) What are Annotations?

Annotations are extra information used by tools/controllers.
Not mainly for selecting pods.

✅ Example: Ingress annotations for ALB / NGINX settings.

✅ **Service & Ingress**

11) What is a Service in Kubernetes?

Service gives a stable IP/Name to access pods.
Because pods can restart and change IP.

✅ Service = stable network endpoint.

---

12) Types of Services

✅ **ClusterIP**

Works only inside cluster (internal access)

✅ **NodePort**

Exposes service through node port
Example: NodeIP:30080

✅ **LoadBalancer**

Creates external load balancer (in cloud)
Example: AWS ELB/ALB

---

13) What is Ingress?

Ingress provides external HTTP/HTTPS access to services.
It routes traffic to correct service using rules.

✅ Example:

- /app1 → service1
- /app2 → service2

📌 Ingress needs **Ingress Controller** to work.

---

14) What is an Ingress Controller?

Ingress Controller is the component that actually creates routing/load balancer.

Examples:

- NGINX Ingress Controller
- AWS Load Balancer Controller (ALB)
- Traefik

✅ Ingress = rules
✅ Controller = does the actual work

✅ Helm (Very Important)

15) What is Helm?

Helm is a package manager for Kubernetes.
It helps install applications easily.

✅ Example:
Install controller using 1 command instead of 10 YAML files.

Helm uses **Charts** (packages).

✅ OIDC & IRSA (Cloud DevOps Key)

16) What is OIDC?

OIDC (OpenID Connect) is an identity system used to allow Kubernetes to connect with IAM securely.

✅ In EKS, OIDC helps Kubernetes service accounts assume IAM roles.

---

17) What is IRSA?

IRSA = IAM Roles for Service Accounts.
It allows a Kubernetes pod to get AWS permissions without storing AWS keys.

✅ Safe way to give AWS access to pods.

Example:
AWS Load Balancer Controller pod needs permission to create ALB.

✅ YAML Common Keywords

18) What is apiVersion?

Defines which Kubernetes API version is used.

Example:

- apps/v1
- v1

---

19) What is kind?

Tells what type of resource it is.

Examples:

- Pod
- Deployment
- Service
- Ingress
- ConfigMap

---

20) What is spec?

spec means desired configuration.
It tells Kubernetes what you want.

✅ Example: replicas = 2

✅ **ConfigMap & Secret**

21) What is ConfigMap?

ConfigMap stores normal configuration data.
Example: app URL, environment name.

✅ Not for passwords.

---

22) What is Secret?

Secret stores sensitive data like password, token, key.

✅ Used for DB password, API key

📌 Secrets are base64 encoded (not fully encrypted by default).

✅ **Scaling & Self Healing**

23) What is Scaling in Kubernetes?

Scaling means increasing or decreasing the number of pods.

✅ Example:
replicas 1 → 3

---

24) What is Self-Healing?

If a pod crashes, Kubernetes automatically recreates it.

✅ This is done mainly by **Deployment + ReplicaSet controllers**.

✅ **Rolling Updates**

25) What is Rolling Update?

Kubernetes updates application without downtime by gradually replacing old pods with new pods.

✅ Used in production deployments.

✅ **Ephemeral Meaning (Important word)**

26) What is Ephemeral?

Ephemeral means temporary and not permanent.
Pods and containers can be deleted/recreated anytime.

✅ For permanent data we use volumes (PVC).

✅ Most Common Interview “Difference” Questions

27) Pod vs Deployment

- ✅ Pod: runs container
- ✅ Deployment: manages pods, scaling, self-healing

---

28) Service vs Ingress

- ✅ Service: access pods inside cluster
- ✅ Ingress: external HTTP routing to services

---

29) Node vs Pod

- ✅ Node: machine/server
- ✅ Pod: runs inside node

---

30) Namespace vs Cluster

- ✅ Cluster: full Kubernetes setup
- ✅ Namespace: logical separation inside cluster

✅ Quick One-Line Summary (Best for Interview)

✅ Kubernetes = manages containers
✅ Cluster = group of machines
✅ Node = machine/server
✅ Pod = runs container
✅ Control plane = API server, etcd, scheduler, controller manager
✅ Worker node = kubelet, container runtime, kube-proxy
✅ Service = stable access to pods
✅ Ingress = external routing
✅ OIDC = identity connection
✅ IRSA = IAM role for pod (secure AWS access)
✅ Namespace = separation
✅ Helm = install apps easily
✅ Metadata = name/labels/annotations info

Prepared by:

M. Pandian | Cloud & DevOps Engineer