KUBERNETES (K8s) – COMPLETE SYSTEM ADMINISTRATION NOTES

1. Introduction

Kubernetes (K8s) is an open-source container orchestration platform originally developed by Google and now maintained by the Cloud Native Computing Foundation (CNCF). It is used to automate deployment, scaling, management, and operations of containerized applications.

2. Why Kubernetes?

Problems without Kubernetes: - Managing many containers manually is difficult. - No automatic scaling. - No self-healing if containers crash. - Downtime during updates. Kubernetes solves these by providing automation, scalability, and reliability.

3. Kubernetes Architecture

A Kubernetes Cluster consists of: Control Plane (Master Node): - API Server - Scheduler - Controller Manager - etcd (cluster database) Worker Nodes: - kubelet - kube-proxy - Container runtime (containerd / Docker)

4. Core Concepts

Pod: Smallest deployable unit. Contains one or more containers. Node: A physical or virtual machine that runs Pods. Cluster: A group of nodes managed by Kubernetes. Deployment: Manages replica sets, rolling updates, and ensures desired state. Service: Exposes Pods to internal or external traffic. Types of Services: - ClusterIP - NodePort - LoadBalancer

5. Important Objects

ReplicaSet: Ensures a specified number of pod replicas are running. ConfigMap: Stores configuration data. Secret: Stores sensitive data like passwords and tokens. Namespace: Logical separation inside a cluster. Ingress: Manages external HTTP/HTTPS access to services.

6. Scaling and Self-Healing

Horizontal Scaling: kubectl scale deployment app-name --replicas=5 Auto Scaling: Using Horizontal Pod Autoscaler (HPA). Self-Healing: If a Pod crashes, Kubernetes automatically recreates it.

7. Rolling Updates and Rollbacks

Rolling Update: Updates application without downtime. Rollback: kubectl rollout undo deployment app-name

8. Networking Basics

Every Pod gets its own IP address. Services provide stable IP and DNS. kube-proxy manages networking rules.

9. Storage

Volumes: Temporary storage attached to Pods. Persistent Volume (PV): Cluster-wide storage resource. Persistent Volume Claim (PVC): Request for storage by a Pod.

10. Useful kubectl Commands

kubectl get pods kubectl get nodes kubectl get services kubectl describe pod pod-name kubectl apply -f file.yaml kubectl delete pod pod-name

11. Production Best Practices

- Use namespaces for environments (dev, staging, prod). - Use resource limits (CPU & memory). - Use readiness and liveness probes. - Use RBAC for security. - Monitor with Prometheus & Grafana.

12. Kubernetes in Cloud

Managed Kubernetes Services: - Amazon EKS - Google GKE - Azure AKS These reduce operational overhead.

13. Learning Path

1. Learn Docker basics. 2. Understand YAML configuration. 3. Practice with Minikube or Kind. 4. Deploy real applications. 5. Study CKA certification topics.