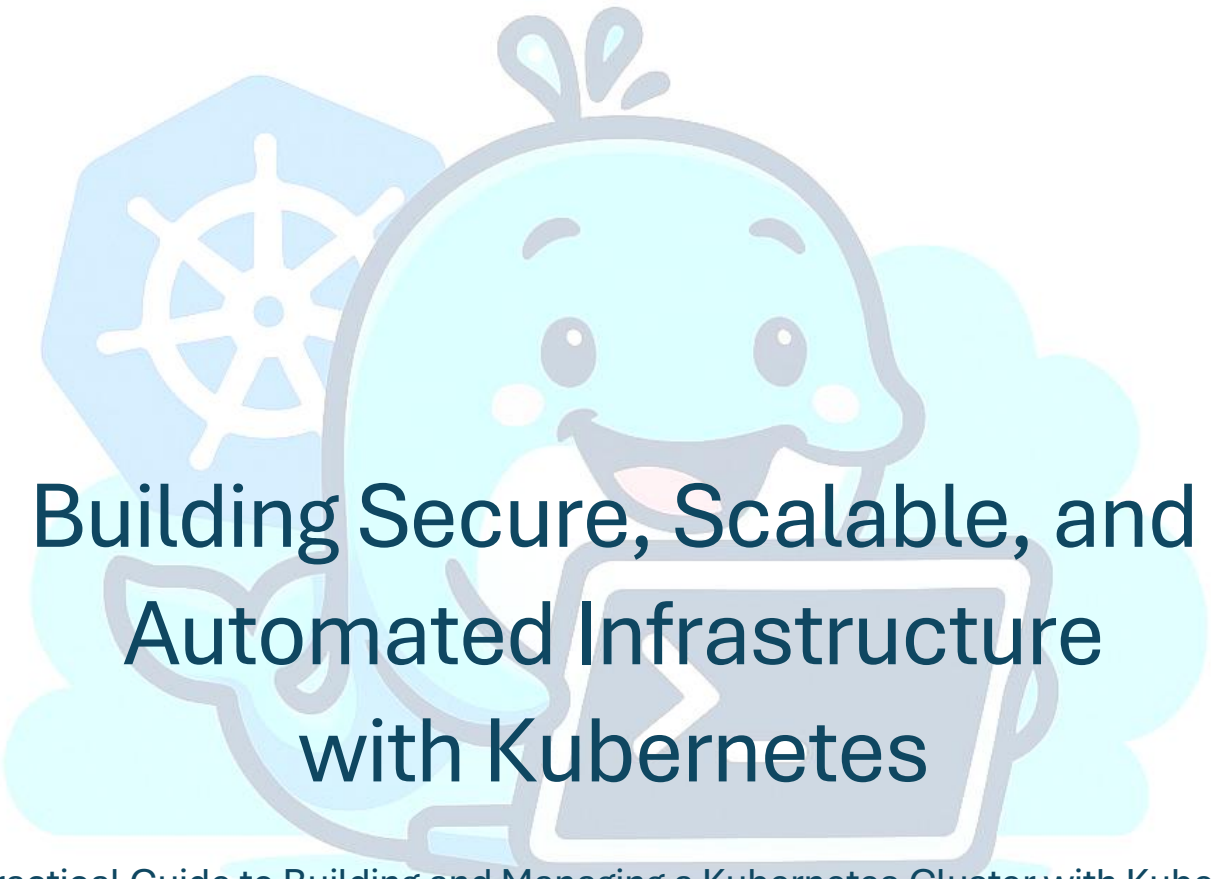


Building Secure, Scalable, and Automated Infrastructure with Kubernetes

A Practical Guide to Building and Managing a Kubernetes Cluster with Kubeadm

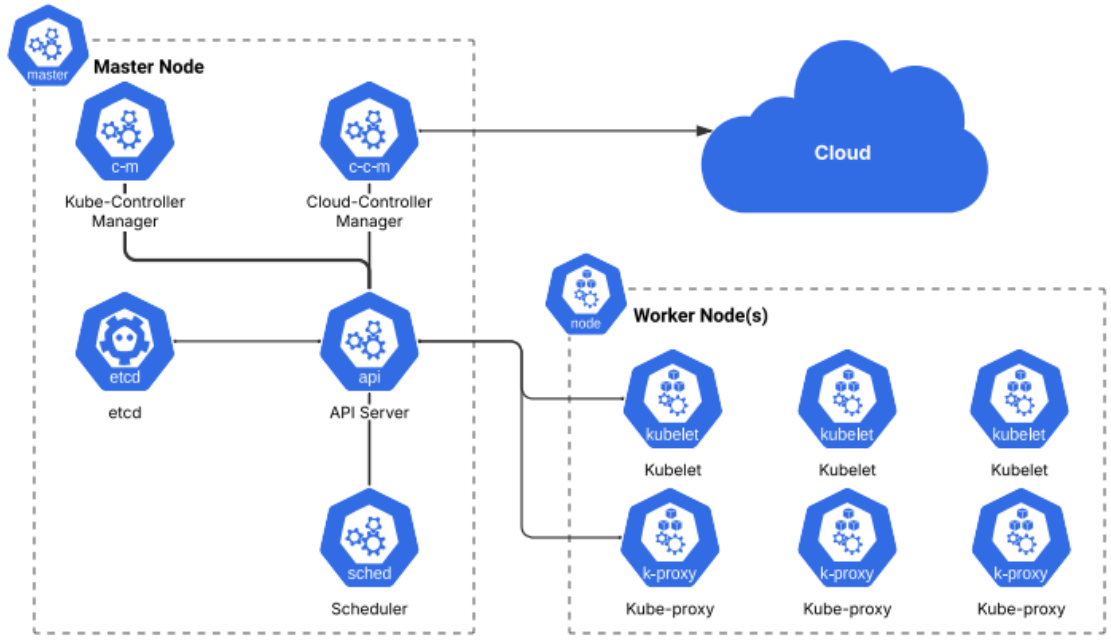
K&Dctl

Introduction

This book covers the deployment of a microservices-based application architecture using Kubernetes set up via **Kubeadm**. You'll learn how to deploy, monitor, and secure your workloads on a production-grade cluster with persistent storage and autoscaling.

Cluster Architecture

- **Master Node**: 1 (Cloud VM)
- **Worker Node**: 1 (Cloud VM)
- **Kubernetes Install**: Kubeadm



Application Microservice Overview

➢ Backend Microservice

- 2 replicas
- HPA (min 2, max 5)
- ClusterIP Service
- ConfigMap for environment variables

➢ Database

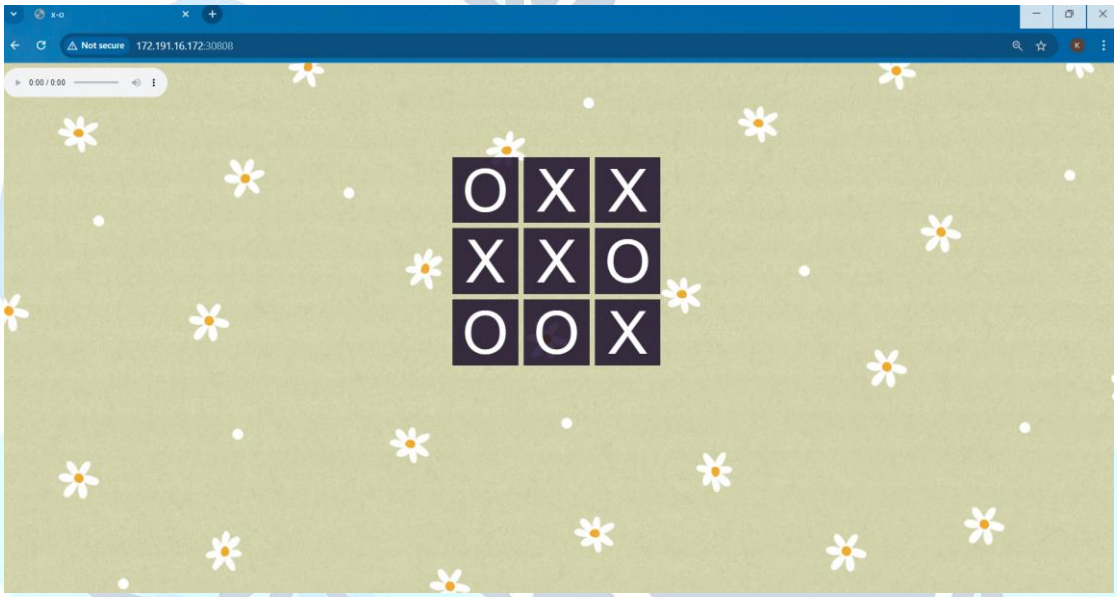
- 2 replicas
- PersistentVolume (PV) + PersistentVolumeClaim (PVC)
- ClusterIP Service
- Restricted access via NetworkPolicy

➢ Frontend App

- 2 replicas
- HPA (min 2, max 5)
- NodePort Service

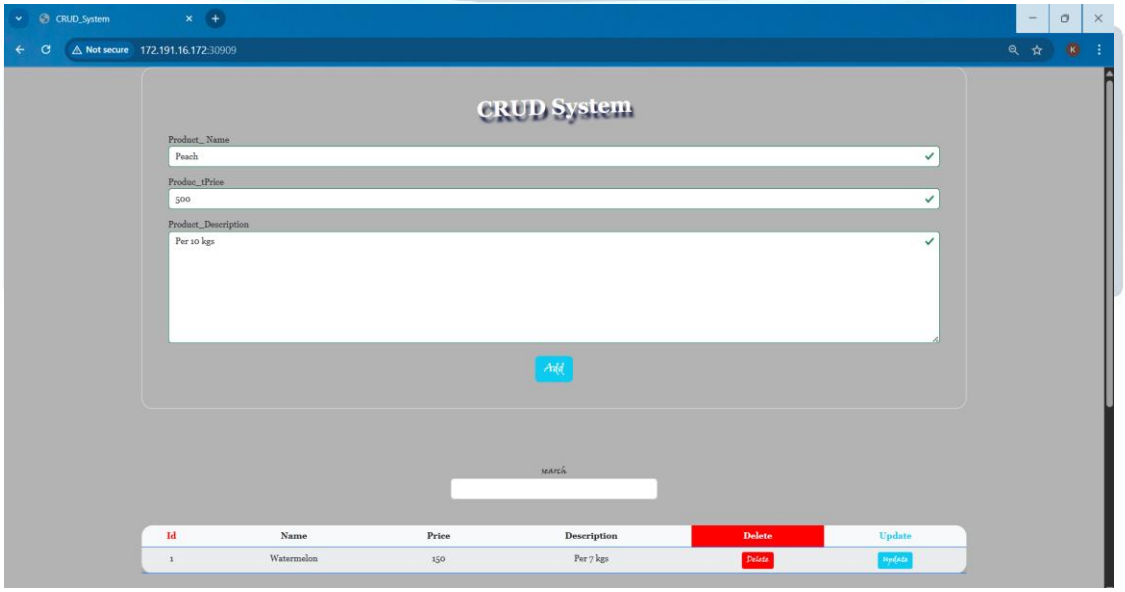
➢ XO Game App

- 2 replicas
- NodePort Service



➢ CRUD App

- 2 replicas
- NodePort Service



Deployment Process

➢ Prerequisites

- Kubernetes Cluster (via Kubeadm)
- `kubectl`

➢ Deployment Steps

ConfigMaps

```
kubectl apply -f configmap-backend.yaml
```

```
[diaa@master chat-app]$ kubectl get secret
NAME         TYPE     DATA   AGE
env-secret   Opaque   4      5d13h
```

Backend Deployment

```
kubectl apply -f backend-deployment.yaml
kubectl apply -f backend-hpa.yaml
kubectl apply -f backend-service.yaml
```

```
[diaa@master]$ kubectl get po
NAME                                    READY   STATUS    RESTARTS       AGE
chat-app-deployment-849499776b-rtvk4    1/1     Running   3 (111m ago)   4d20h
chat-app-deployment-849499776b-rzfjb    1/1     Running   3 (111m ago)   4d20h
crud-app-deployment-6f59d99b6d-l7wc9    1/1     Running   1 (111m ago)   19h
crud-app-deployment-6f59d99b6d-qtmf7    1/1     Running   1 (111m ago)   19h
```

```
[diaa@master]$ kubectl get hpa
NAME          REFERENCE                          TARGETS              MINPODS   MAXPODS   REPLICAS   AGE
my-app-hpa    Deployment/chat-app-deployment     cpu: 20/60%   2      5         2          5d13h
```

```
[diaa@master]$ kubectl get svc
NAME                    TYPE       CLUSTER-IP      EXTERNAL-IP   PORT(S)                         AGE
chat-app-backend-svc    NodePort   10.111.47.114   <none>        5000:30000/TCP,5001:30001/TCP   5d10h
```

```
[diaa@master chat-app]$ kubectl get pv
NAME                CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS   CLAIM                          STORAGECLASS    VOLUMEATTRIBUTESCLASS   REASON   AGE
mongodb-chat-app-pv   2Gi        RWO            Retain           Bound    default/mongodb-chat-app-pvc   local-storage   <unset>                          5d1h
[diaa@master chat-app]$ kubectl get pvc
NAME                  STATUS   VOLUME                CAPACITY   ACCESS MODES   STORAGECLASS    VOLUMEATTRIBUTESCLASS   AGE
mongodb-chat-app-pvc   Bound    mongodb-chat-app-pv   2Gi        RWO            local-storage   <unset>                 5d1h
```

Database Deployment

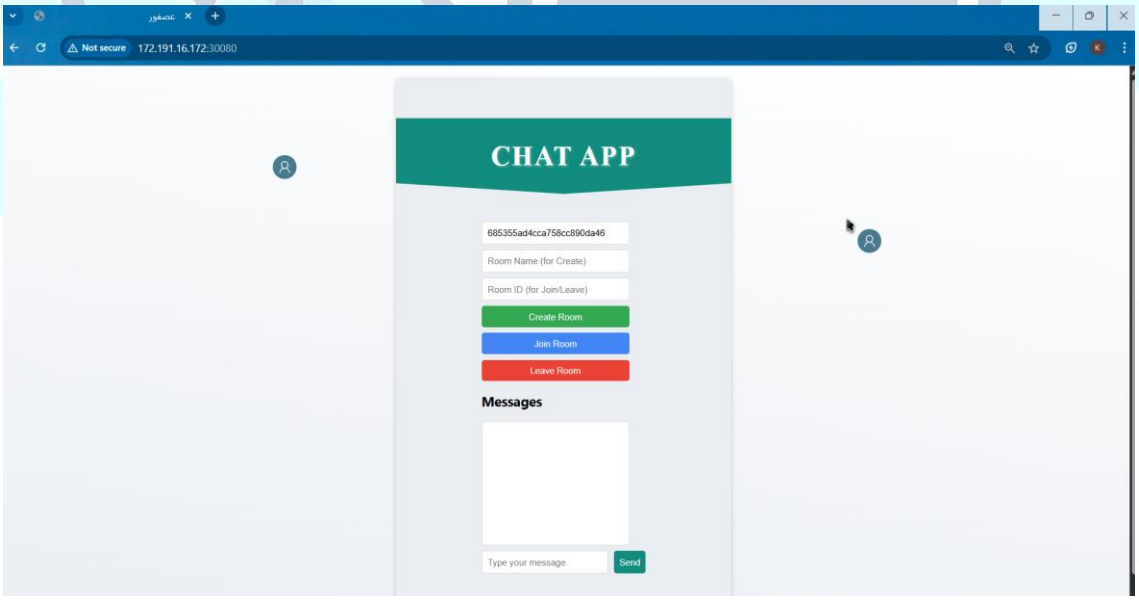
```
kubectl apply -f db-pv.yaml
kubectl apply -f db-pvc.yaml
kubectl apply -f db-deployment.yaml
kubectl apply -f db-service.yaml
```

```
[diaa@master]$ kubectl get po
NAME                                  READY   STATUS    RESTARTS       AGE
mongodb-deployment-5d66fd56f5-b6vjw   1/1     Running   3 (111m ago)   5d
mongodb-deployment-5d66fd56f3-dsjkw   1/1     Running   3 (112m ago)   5d
```

```
[diaa@master ]$ kubectl get svc
NAME                                TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)           AGE
mongodb-chat-app-service-cluster-ip ClusterIP   10.100.250.230   <none>        27017/TCP         5d14h
```

Frontend Deployment

```
kubectl apply -f frontend-deployment.yaml
kubectl apply -f frontend-hpa.yaml
kubectl apply -f frontend-service.yaml
```




```
[diaa@master front]$ kubectl get svc
NAME                TYPE       CLUSTER-IP       EXTERNAL-IP   PORT(S)          AGE
react-app-service   NodePort   10.101.231.225   <none>        80:30080/TCP     5d10h

[diaa@master front]$ kubectl get po
NAME                                    READY   STATUS    RESTARTS       AGE
react-chat-app-deployment-677dd4f7f-xg266   1/1     Running   3 (111m ago)   4d20h
```

XO Game Deployment

```
kubectl apply -f xo-deployment.yaml
kubectl apply -f xo-service.yaml
```

```
[diaa@master chat-app]$ kubectl get deploy
NAME                    READY   UP-TO-DATE   AVAILABLE   AGE
crud-app-deployment     2/2     2            2           20h
xo-app-deployment       2/2     2            2           30h
```

CRUD App Deployment

```
kubectl apply -f crud-deployment.yaml
kubectl apply -f crud-service.yaml
```

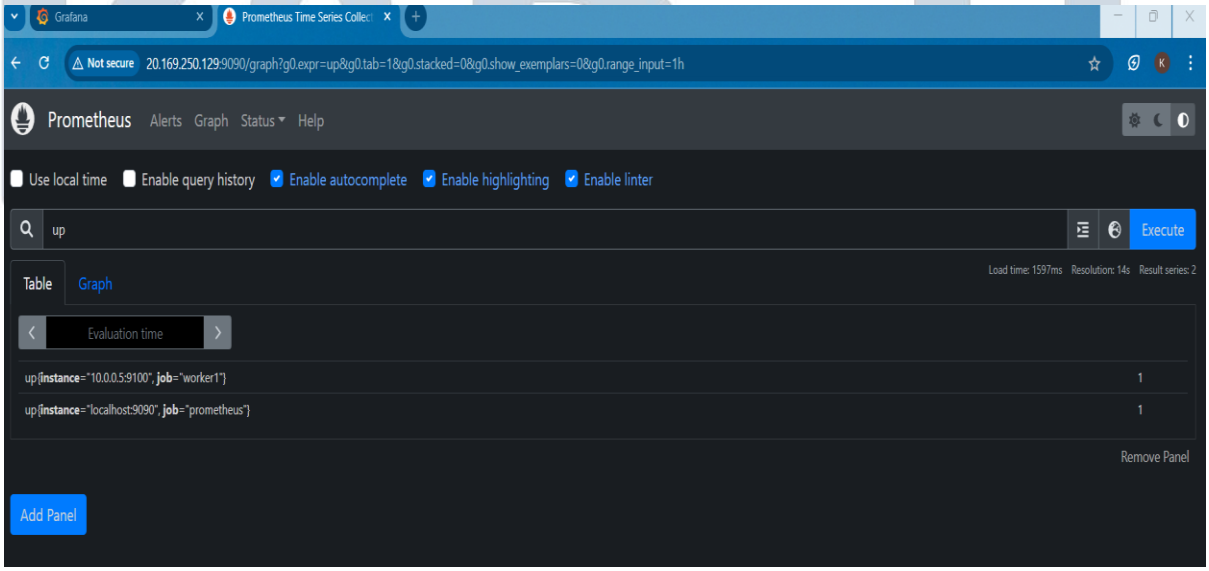
```
[diaa@master chat-app]$ kubectl get deploy
NAME                    READY   UP-TO-DATE   AVAILABLE   AGE
crud-app-deployment     2/2     2            2           20h
```

Network Policies

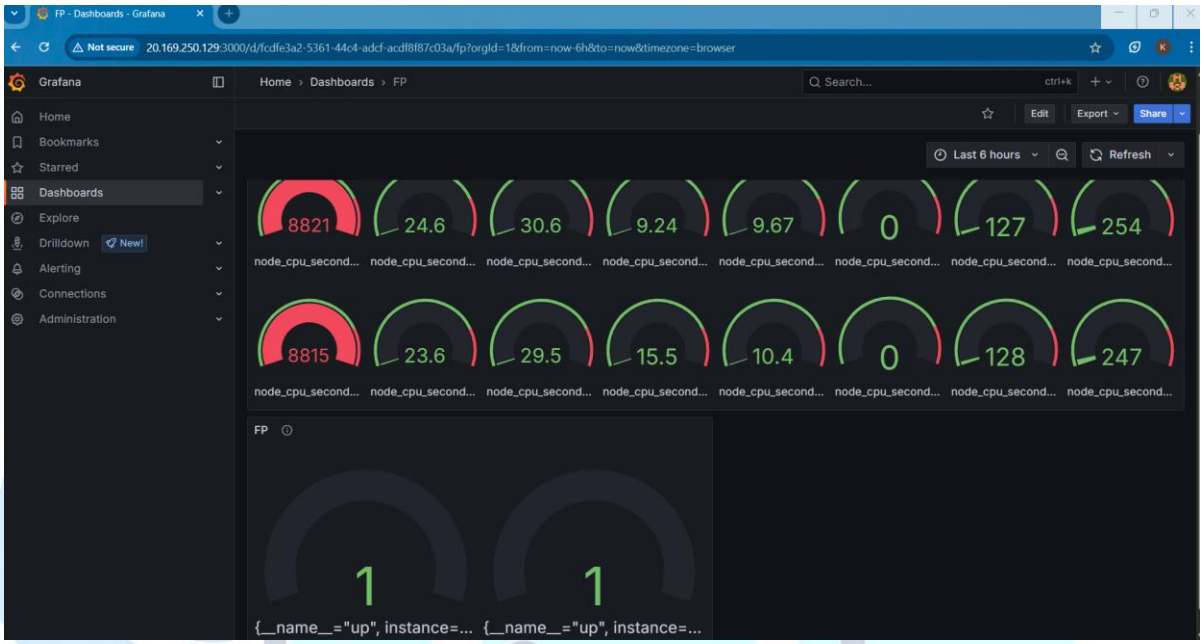
```
kubectl apply -f networkpolicy-backend.yaml
kubectl apply -f networkpolicy-frontend.yaml
kubectl apply -f networkpolicy-db.yaml
```

Monitoring and Observability

- **Prometheus** gathers metrics for workloads and cluster components.



- **Grafana** visualizes metrics via customizable dashboards.



Security and Networking

- **NetworkPolicies** restrict service access:
  - Backend ↔ Frontend
  - Backend ↔ DB
- Enforced service isolation.

Storage Configuration

- **PV/PVC** configured for Database persistence.
- Storage class configured according to cloud provider settings.

Operational Guide

- **Service Access**

Application	Access Method
Frontend	NodePort
XO Game	NodePort
CRUD App	NodePort
Backend	ClusterIP
DB	ClusterIP
Grafana	NodePort or LoadBalancer

Conclusion

This project delivers a modular, scalable, and observable microservices deployment on a Kubernetes cluster using **Kubeadm**. By following this guide, developers and DevOps teams can manage production-ready services with monitoring, persistence, and network isolation.