Complex Kubernetes Configuration with Helm and Istio for a Three-Service Application

Grok

June 1, 2025

1 Introduction

This document provides Helm charts and Istio configurations for deploying a three-service application (React front-end, Node.js/Express backend, MongoDB database) on Kubernetes. The setup uses Helm for templated deployments and Istio for service mesh features like mTLS, traffic routing, and observability with Prometheus and Grafana. Comments explain adaptability to requirement changes.[](https://istio.io/latest/about/service-mesh/)[](https://ww to-install-and-use-istio-with-kubernetes)

2 Prerequisites

- A Kubernetes cluster (e.g., Minikube, GKE) with `kubectl` configured.
- Helm installed (`https://helm.sh/docs/intro/install/`).
- Istio installed (`https://istio.io/latest/docs/setup/install/helm/`).
- Docker images for front-end and backend pushed to a registry (e.g., Docker Hub).
- MongoDB credentials stored securely (e.g., in a `.env` file).

3 Istio Installation

Configuring Istio with Helm.

Listing 1: Install Istio

```
# Add Istio Helm repository
helm repo add istio https://istio-release.storage.googleapis.com/charts
helm repo update

# Create Istio namespace
kubectl create namespace istio-system

# Install Istio CRDs
helm install istio-base istio/base -n istio-system --wait

```

```
# Install Istiod (Istio control plane)
helm install istiod istio/istiod -n istio-system --wait \\
  --set profile=demo # Change to minimal or other profile if needed

# Install Istio ingress gateway
helm install istio-ingress istio/gateway -n istio-system --wait

# Enable sidecar injection for default namespace
kubectl label namespace default istio-injection=enabled # Change namespace if
    different
```

4 Observability Addons

Installing Prometheus and Grafana for monitoring.

Listing 2: Install Prometheus and Grafana

```
helm repo add prometheus-community https://prometheus-community.github.io/helm
    -charts
helm repo update
helm install prometheus prometheus-community/prometheus -n istio-system --wait
helm install grafana grafana/grafana -n istio-system --wait
```

5 Helm Chart Structure

Creating a Helm chart for the application.

Listing 3: Chart Directory Structure

```
mkdir -p my-app/templates
touch my-app/Chart.yaml my-app/values.yaml
touch my-app/templates/secret.yaml my-app/templates/frontend-deployment.yaml
touch my-app/templates/backend-deployment.yaml my-app/templates/database-
    deployment.yaml
touch my-app/templates/gateway.yaml my-app/templates/virtualservice.yaml
```

6 Chart.yaml

Defining the Helm chart metadata.

Listing 4: my-app/Chart.yaml

```
apiVersion: v2
name: my-app
description: Helm chart for three-service application
type: application
version: 0.1.0
appVersion: "1.0"
```

7 values.yaml

Defining configurable values for the Helm chart.

Listing 5: my-app/values.yaml

```
frontend:
  image: your_dockerhub_username/frontend:latest # Update with your Docker Hub image
  replicas: 2 # Adjust for scaling
  port: 3000 # Change if frontend framework changes (e.g., Angular: 4200)
backend:
  image: your_dockerhub_username/backend:latest # Update with your Docker Hub image
  replicas: 2 # Adjust for scaling
  port: 5000 # Change if backend framework changes (e.g., FastAPI: 8000)
mongodb:
  image: mongo:7.0 # Change to different DB if needed
  replicas: 1 # Typically single replica for MongoDB
  port: 27017
  storage: 10Gi # Adjust storage size
  username: mongo_user # Update for security
  password: mongo_password # Update for security
istio:
  gateway:
    host: app.example.com # Update with your domain
```

8 Kubernetes Secret

Defining sensitive data for MongoDB.

Listing 6: my-app/templates/secret.yaml

```
apiVersion: v1
kind: Secret
metadata:
  name: mongodb-credentials
type: Opaque
data:
  mongo-user: {{ .Values.mongodb.username | b64enc }} # Update if credentials change
  mongo-password: {{ .Values.mongodb.password | b64enc }}
```

9 Frontend Deployment and Service

Configuring the React front-end with Istio sidecar injection.

Listing 7: my-app/templates/frontend-deployment.yaml

```
apiVersion: apps/v1
kind: Deployment
```

```
metadata:
  name: frontend
  labels:
    app: frontend
spec:
  replicas: {{ .Values.frontend.replicas }}
  selector:
    matchLabels:
      app: frontend
  template:
    metadata:
      labels:
        app: frontend
    spec:
      containers:
      - name: frontend
        image: {{ .Values.frontend.image }}
        ports:
        - containerPort: {{ .Values.frontend.port }}
---
apiVersion: v1
kind: Service
metadata:
  name: frontend-service
spec:
  selector:
    app: frontend
  ports:
  - port: 80
    targetPort: {{ .Values.frontend.port }}
  type: ClusterIP # Change to LoadBalancer for external access without Istio
```

10 Backend Deployment and Service

Configuring the Node.js/Express backend with Istio sidecar injection.

Listing 8: my-app/templates/backend-deployment.yaml

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: backend
  labels:
    app: backend
spec:
  replicas: {{ .Values.backend.replicas }}
  selector:
    matchLabels:
      app: backend
  template:
    metadata:
```

```
      labels:
        app: backend
    spec:
      containers:
      - name: backend
        image: {{ .Values.backend.image }}
        ports:
        - containerPort: {{ .Values.backend.port }}
        env:
        - name: MONGO_URI
          value: "mongodb://{{ .Values.mongodb.username }}:{{ .Values.mongodb.
              password }}@mongodb-service:{{ .Values.mongodb.port }}/mydb"
---
apiVersion: v1
kind: Service
metadata:
  name: backend-service
spec:
  selector:
    app: backend
  ports:
  - port: 80
    targetPort: {{ .Values.backend.port }}
  type: ClusterIP
```

11 Database Deployment and Service

Configuring the MongoDB database.

Listing 9: my-app/templates/database-deployment.yaml

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: mongodb
  labels:
    app: mongodb
spec:
  replicas: {{ .Values.mongodb.replicas }}
  selector:
    matchLabels:
      app: mongodb
  template:
    metadata:
      labels:
        app: mongodb
    spec:
      containers:
      - name: mongodb
        image: {{ .Values.mongodb.image }}
        ports:
```

```
        - containerPort: {{ .Values.mongodb.port }}
        env:
        - name: MONGO_INITDB_ROOT_USERNAME
          valueFrom:
            secretKeyRef:
              name: mongodb-credentials
              key: mongo-user
        - name: MONGO_INITDB_ROOT_PASSWORD
          valueFrom:
            secretKeyRef:
              name: mongodb-credentials
              key: mongo-password
        volumeMounts:
        - name: mongo-data
          mountPath: /data/db
      volumes:
      - name: mongo-data
        persistentVolumeClaim:
          claimName: mongo-pvc
---
apiVersion: v1
kind: Service
metadata:
  name: mongodb-service
spec:
  selector:
    app: mongodb
  ports:
  - port: {{ .Values.mongodb.port }}
    targetPort: {{ .Values.mongodb.port }}
  type: ClusterIP
---
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: mongo-pvc
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: {{ .Values.mongodb.storage }}
```

12 Istio Gateway and VirtualService

Configuring external access and traffic routing.

Listing 10: my-app/templates/gateway.yaml

```
apiVersion: networking.istio.io/v1alpha3
kind: Gateway
```

```
metadata:
  name: app-gateway
spec:
  selector:
    istio: ingressgateway # Change if gateway name differs
  servers:
  - port:
      number: 80
      name: http
      protocol: HTTP
    hosts:
    - "{{ .Values.istio.gateway.host }}" # Update with your domain
---
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: app-virtualservice
spec:
  hosts:
  - "{{ .Values.istio.gateway.host }}"
  gateways:
  - app-gateway
  http:
  - route:
    - destination:
        host: frontend-service
        port:
          number: 80
```

13 How to Run

1. Ensure Docker images are pushed to a registry.
2. Install Istio: Run commands from the Istio Installation section.
3. Install observability addons: Run Prometheus and Grafana installation commands.
4. Create Helm chart directory and files as shown in Chart Structure.
5. Deploy the application: `helm install my-app ./my-app -n default`.
6. Get the ingress gateway IP: `kubectl get svc istio-ingressgateway -n istio-system`.
7. Access the application at `http://<ingress-gateway-ip>`.
8. Access Grafana for monitoring: `kubectl port-forward svc/grafana -n istio-system 3000:3000`, then visit `http://localhost:3000`.
9. Uninstall: `helm uninstall my-app -n default; helm uninstall istio-base istiod istio-ingress prometheus grafana -n istio-system`.

14 Adapting to Requirement Changes

- **Change Frontend Framework**: Update `values.yaml` and `frontend-deployment.yaml` for image and port.
- **Change Backend Framework**: Update `values.yaml` and `backend-deployment.yaml` for image and port.
- **Change Database**: Update `values.yaml` and `database-deployment.yaml` for image and volume paths.
- **Adjust Scaling**: Modify `replicas` in `values.yaml`.
- **Change Domain**: Update `istio.gateway.host` in `values.yaml`.
- **Add Istio Features**: Extend `virtualservice.yaml` for canary deployments or traffic mirroring.