PROJECT: STREAMLINING SECURITY ACROSS ENVIRONMENTS WITH DEVSECOPS

PHASE 4 – Final Review

College Name: Vemana Institute of Technology
Name: Monika P
CAN ID Number: CAN_33387138

Streamlining Security Across Environments with DevSecOps

Overview

This project focuses on integrating **security into DevOps (DevSecOps)** by automating security scans, ensuring container security, and deploying a secured application on **IBM Cloud Kubernetes Service (IKS)**.

Key Components

- **Containerization**: Securing applications inside Docker containers.
- **Security Scanning**: Using **Trivy/Snyk** to scan images for vulnerabilities.
- **CI/CD Pipeline**: Automated **security scans, builds, and deployments**.
- **IBM Cloud Kubernetes Service (IKS)**: Secure orchestration of containers.
- **Monitoring & Logging**: Integrating **Prometheus + Grafana**.

---

1. Setting Up DevSecOps Pipeline

Step 1: Install Required Tools

Ensure you have the following installed:

- **Docker** (for containerization)
- **Kubectl** (for managing Kubernetes)
- **IBM Cloud CLI** (for interacting with IBM Cloud)
- **Trivy** (for security scanning)

sh

CopyEdit

```
# Install IBM Cloud CLI
curl -fsSL https://clis.cloud.ibm.com/install/linux | sh
ibmcloud login --apikey <YOUR_API_KEY>

# Install Kubernetes CLI
ibmcloud ks cluster config --cluster <cluster_name>

# Install Trivy (Security Scanner)
brew install aquasecurity/trivy/trivy
```

Step 2: Secure Containerization

Create a **secure Dockerfile** with **non-root user** and **least privileges**.

Dockerfile (Backend)

dockerfile

CopyEdit

```
FROM node:16-alpine

# Create a non-root user
RUN addgroup -S appgroup && adduser -S appuser -G appgroup
USER appuser

WORKDIR /app
COPY . .
RUN npm install

EXPOSE 5000
CMD ["node", "server.js"]
```

Building and Scanning Image

sh

CopyEdit

```
# Build Docker Image
docker build -t backend-app:1.0 .

# Run Trivy Security Scan
trivy image backend-app:1.0
```

Push Image to IBM Cloud Registry

sh

CopyEdit

```
# Authenticate & Push to IBM Cloud
ibmcloud cr login
docker tag backend-app:1.0 <region>.icr.io/<namespace>/backend-app:1.0
docker push <region>.icr.io/<namespace>/backend-app:1.0
```

---

2. Deploy Securely on Kubernetes

Create Kubernetes deployment and apply security policies.

backend-deployment.yaml

yaml

CopyEdit

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: backend-deployment
spec:
  replicas: 3
  selector:
    matchLabels:
      app: backend
  template:
    metadata:
      labels:
        app: backend
    spec:
      securityContext:
        runAsNonRoot: true
        seccompProfile:
          type: RuntimeDefault
      containers:
      - name: backend
        image: <region>.icr.io/<namespace>/backend-app:1.0
        ports:
        - containerPort: 5000
```

Deploying to Kubernetes

sh

CopyEdit

```
kubectl apply -f backend-deployment.yaml
kubectl get pods
```

---

3. Setting Up CI/CD with Security Checks

GitHub Actions Workflow (.github/workflows/devsecops.yml)

yaml

CopyEdit

```
name: DevSecOps Pipeline
on:
  push:
    branches:
      - main
jobs:
  security_scan:
    runs-on: ubuntu-latest
    steps:
    - name: Checkout Repository
      uses: actions/checkout@v3

    - name: Run Trivy Security Scan
      run: trivy image backend-app:1.0

  deploy:
    needs: security_scan
    runs-on: ubuntu-latest
    steps:
    - name: Deploy to IBM Kubernetes
      run: |
        ibmcloud login --apikey ${{ secrets.IBM_API_KEY }}
        kubectl apply -f backend-deployment.yaml
```

---

4. HTML-Based Monitoring Dashboard

Create an **HTML UI** to monitor deployments.

index.html

html

CopyEdit

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>DevSecOps Monitoring Dashboard</title>
    <script>
        async function fetchStatus() {
            const response = await fetch('/status');
            const data = await response.json();
            document.getElementById("status").innerText = data.status;
        }
        setInterval(fetchStatus, 5000);
    </script>
</head>
<body>
    <h1>DevSecOps Monitoring Dashboard</h1>
    <p>Application Status: <span id="status">Checking...</span></p>
</body>
</html>
```

server.js (Backend API for Dashboard)

js

CopyEdit

```js
const express = require('express');
const app = express();
```

```
app.get('/status', (req, res) => {
   res.json({ status: "Running Securely" });
});

app.listen(3000, () => console.log("Monitoring API running on port 3000"));
```

Run it with:

sh

CopyEdit

```
node server.js
```

---

5. Enhancements & Next Steps

- **Add Web Application Firewall (WAF)**
- **Integrate Advanced Logging (ELK Stack)**
- **Enable Kubernetes Network Policies**
- **Automate Security Policies with OPA/Gatekeeper**

Final Steps to Run in VS Code

1. Clone the repo:

sh

CopyEdit

```
git clone <your-github-repo>
cd devsecops-project
```

2. Run backend:

sh

CopyEdit

```
node server.js
```

3. Open index.html in a browser.
4. Deploy containers:

sh

CopyEdit

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

5. Monitor logs:

sh

CopyEdit

```
kubectl logs -f deployment/backend-deployment
```

This project ensures **end-to-end security integration** across development, deployment, and monitoring using **DevSecOps best practices**.