Complete CI/CD Pipeline with Security & GitOps

Table of Contents

CI/CD Pipeline Overview

Step	Description
Code Commit	Developers push code to GitHub.
SAST - SonarQube	Static security analysis of the code.
Build & Test (Maven)	Compiling, running tests, and packaging the application.
Dependency Scanning	Checking for vulnerable dependencies.
DAST - OWASP ZAP	Dynamic security testing of the application.
Build Docker Image	Creating a containerized version of the application.
Push to Docker Hub	Storing the image for deployment.
Create Kubernetes Manifests	Defining Deployment, Service, and Ingress for Kubernetes.
Update Kubernetes Manifests with GitOps	Modifying YAML for ArgoCD-based deployments.
Deploy with ArgoCD	Using GitOps to deploy applications in Kubernetes.
Ingress Controller (NGINX)	Handling external traffic to the application.
HTTPS with Cert-Manager	Enforcing SSL/TLS security.

Static Code Analysis (SAST - SonarQube)

```
sonar:
```

```
  host:
    url: http://sonarqube:9000
  login: ${SONARQUBE_TOKEN}
```

Build & Test (Maven)

```
stages:
  - build
build:
  stage: build
  script:
    - mvn clean install
```

Dependency Scanning

```
dependency_scanning:
  stage: test
  script:
    - mvn dependency-check:check
```

DAST - OWASP ZAP

```
dast_scan:
  stage: test
  script:
    - zap-baseline.py -t http://my-app.example.com
```

Build Docker Image

```
docker_build:
  stage: build
  script:
    - docker build -t my-dockerhub-user/my-app:${CI_COMMIT_SHA} .
```

Push to Docker Hub

```
docker_push:
  stage: deploy
  script:
    - echo ${DOCKER_HUB_PASSWORD} | docker login -u ${DOCKER_HUB_USERNAME} --password-stdin
```

```
        - docker push my-dockerhub-user/my-app:${CI_COMMIT_SHA}
```

Create Kubernetes Manifests

```
    apiVersion: apps/v1
    kind: Deployment
    metadata:
      name: my-app
    spec:
      replicas: 2
      selector:
        matchLabels:
          app: my-app
      template:
        metadata:
          labels:
            app: my-app
        spec:
          containers:
            - name: my-app
              image: my-dockerhub-user/my-app:${{ github.sha }}
              ports:
                - containerPort: 8080
```

Update Kubernetes Manifests with GitOps (ArgoCD)

```
    - name: Update Kubernetes Manifests in Git
      run: |
                sed  -i  's|image:  my-dockerhub-user/my-app:.*|image:  my-dockerhub-user/my-app:${{  github.sha  }}|'
k8s-app/manifests/deployment.yaml
        git config --global user.name "GitHub Actions"
        git config --global user.email "actions@github.com"
        git add k8s-app/manifests/
        git commit -m "Update Kubernetes manifests for ArgoCD deployment"
        git push
```

Deploy with ArgoCD

```
    apiVersion: argoproj.io/v1alpha1
    kind: Application
    metadata:
      name: my-app
```

```
spec:
  destination:
    namespace: my-namespace
    server: https://kubernetes.default.svc
  source:
    repoURL: 'https://github.com/my-org/my-app.git'
    path: k8s-app/manifests
    targetRevision: main
  syncPolicy:
    automated:
      prune: true
      selfHeal: true
```

Ingress Controller (NGINX)

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: my-app-ingress
  annotations:
    nginx.ingress.kubernetes.io/rewrite-target: /
spec:
  ingressClassName: nginx
  rules:
    - host: my-app.example.com
      http:
        paths:
          - path: /
            pathType: Prefix
            backend:
              service:
                name: my-app-service
                port:
                  number: 80
```

HTTPS with Cert-Manager

```
apiVersion: cert-manager.io/v1
kind: Certificate
metadata:
  name: my-app-cert
spec:
  secretName: my-app-tls
```

```
issuerRef:
  name: letsencrypt-prod
  kind: ClusterIssuer
dnsNames:
  - my-app.example.com
```