**StreamHub Cloud Deployment with Jenkins – A DevSecOps Project**

**Overview**

This guide walks you through deploying a media streaming application, **StreamHub**, on the cloud using DevSecOps principles. It covers infrastructure provisioning, application containerization, security scanning, CI/CD integration, and monitoring.

**Phase 1: Environment Setup & App Deployment**

**Launch Ubuntu Server (EC2):**

* Deploy an Ubuntu 22.04 instance on AWS EC2.
* Connect to your instance via SSH.

**Get the Source Code:**

Update system packages and pull the repository:

bash

CopyEdit

sudo apt update && sudo apt upgrade

git clone https://github.com/N4si/DevSecOps-Project.git

**Docker Installation & App Containerization:**

Install Docker:

bash

CopyEdit

sudo apt install docker.io -y

sudo usermod -aG docker $USER

newgrp docker

sudo chmod 777 /var/run/docker.sock

Build and run the container:

bash

CopyEdit

docker build -t streamhub .

docker run -d --name streamhub -p 8081:80 streamhub:latest

🛑 Note: Application requires an API key to run properly.

**Get TMDB API Key:**

1. Register/login to [TMDB](https://www.themoviedb.org/).
2. Navigate to settings → API.
3. Generate a new API key.

Rebuild the Docker image using your key:

bash

CopyEdit

docker build --build-arg TMDB\_V3\_API\_KEY=<your-api-key> -t streamhub .

**Phase 2: Security Integration**

**Static & Container Security Tools**

**Run SonarQube:**

bash

CopyEdit

docker run -d --name sonar -p 9000:9000 sonarqube:lts-community

Visit: http://<your-ip>:9000 (login: admin/admin)

**Install Trivy:**

bash

CopyEdit

sudo apt install wget apt-transport-https gnupg lsb-release

wget -qO - https://aquasecurity.github.io/trivy-repo/deb/public.key | sudo apt-key add -

echo deb https://aquasecurity.github.io/trivy-repo/deb $(lsb\_release -sc) main | sudo tee -a /etc/apt/sources.list.d/trivy.list

sudo apt update

sudo apt install trivy

Scan the container:

bash

CopyEdit

trivy image streamhub:latest

**Phase 3: CI/CD Pipeline with Jenkins**

**Install Jenkins & Dependencies:**

bash

CopyEdit

sudo apt update

sudo apt install fontconfig openjdk-17-jre

Install Jenkins:

bash

CopyEdit

wget -O /usr/share/keyrings/jenkins-keyring.asc https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key

echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] https://pkg.jenkins.io/debian-stable binary/ | sudo tee /etc/apt/sources.list.d/jenkins.list > /dev/null

sudo apt update && sudo apt install jenkins

sudo systemctl start jenkins

sudo systemctl enable jenkins

**Configure Jenkins:**

Install these plugins:

* SonarQube Scanner
* NodeJS
* OWASP Dependency-Check
* Docker Pipeline

Configure tools via: **Manage Jenkins → Global Tool Configuration**

Set up secrets for DockerHub and SonarQube in **Manage Credentials**.

**Sample Jenkins Pipeline:**

groovy

CopyEdit

pipeline {

agent any

tools {

jdk 'jdk17'

nodejs 'node16'

}

environment {

SCANNER\_HOME = tool 'sonar-scanner'

}

stages {

stage('Clean') {

steps { cleanWs() }

}

stage('Checkout') {

steps { git branch: 'main', url: 'https://github.com/N4si/DevSecOps-Project.git' }

}

stage('Code Analysis') {

steps {

withSonarQubeEnv('sonar-server') {

sh '''$SCANNER\_HOME/bin/sonar-scanner -Dsonar.projectName=StreamHub -Dsonar.projectKey=StreamHub'''

}

}

}

stage('Install Dependencies') {

steps { sh 'npm install' }

}

stage('Vulnerability Scan') {

steps {

dependencyCheck additionalArguments: '--scan ./', odcInstallation: 'DP-Check'

dependencyCheckPublisher pattern: '\*\*/dependency-check-report.xml'

}

}

stage('Build & Push') {

steps {

script {

withDockerRegistry(credentialsId: 'docker', toolName: 'docker') {

sh 'docker build --build-arg TMDB\_V3\_API\_KEY=<yourapikey> -t streamhub .'

sh 'docker tag streamhub yourdockerhub/streamhub:latest'

sh 'docker push yourdockerhub/streamhub:latest'

}

}

}

}

stage('Deploy') {

steps {

sh 'docker run -d --name streamhub -p 8081:80 yourdockerhub/streamhub:latest'

}

}

}

}

**Phase 4: Monitoring with Prometheus & Grafana**

**Prometheus:**

* Follow official instructions to install Prometheus.
* Expose port 9090.
* Create systemd service as needed.
* Update prometheus.yml to include Jenkins and Node Exporter jobs.

**Grafana:**

* Install Grafana via APT.
* Start service and access via port 3000.
* Add Prometheus as data source.
* Import dashboard ID (e.g., 1860) for visualization.

**Phase 5: Notification System**

Enable Jenkins email notifications or connect Slack for build alerts.

`**Phase 6: Kubernetes Deployment (Optional)**

* Provision EKS cluster.
* Install Prometheus Node Exporter with Helm:

bash

CopyEdit

helm repo add prometheus-community https://prometheus-community.github.io/helm-charts

kubectl create namespace prometheus-node-exporter

helm install prometheus-node-exporter prometheus-community/prometheus-node-exporter --namespace prometheus-node-exporter

* Deploy with ArgoCD if desired.

**Phase 7: Cleanup**

Don't forget to:

* Remove unused EC2 resources.
* Clear containers, volumes, and unused images.