National College of Ireland

Configuration Manual

MSc Research Project
Programme Name

Harsh Nailesh Doshi
Student ID: X22207848

School of Computing
National College of Ireland

Supervisor: Eugene Mclaughlin

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Harsh Nailesh Doshi
Student ID:	X22207848
Programme:	Masters in Cybersecurity **Year:** 2024-25
Module:	Configuration Manual
Supervisor:	Eugene Mclaughlin
Submission Due Date:	12-12-2024
Project Title:	Adopting Zero Trust Architecture for Robust Supply Chain Protection

Word Count: 790 Page Count: 6

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:

Date: 12-12-2024

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Harsh Nailesh Doshi
Student ID: X22207848

1 Setting Up Minikube

Set up Minikube to run "kubectl" to interact with Kubernetes environment based on your environment.

1.1 Windows

1. **Check Powershell Version for PowerShell 5.1 or higher:**
   a. $PSVersionTable.PSVersion

2. **Install Chocolatey:**
   a. Set-ExecutionPolicy Bypass -Scope Process -Force; `
   [System.Net.ServicePointManager]::SecurityProtocol = `
   [System.Net.ServicePointManager]::SecurityProtocol -bor 3072; `
   iex ((New-Object
   System.Net.WebClient).DownloadString('https://chocolatey.org/install.ps1'))

3. **Install Docker:**
   a. choco install docker-desktop -y

4. **Install Minikube [1]:**
   a. choco install minikube -y

5. **Verify Installation:**
   a. minikube version

1.2 Linux

1. **Download Minikube:**
   a. curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64
   b. sudo install minikube-linux-amd64 /usr/local/bin/minikube

2. **Verifying the installation:**
   a. minikube version

3. **Start Minikube:**
   a. minikube start

4. **Verifying Status:**
   a. minikube status

2 Setting Up Kubernetes

2.1 Windows

1. **Install Kubectl [1]:**
    a. choco install kubernetes-cli -y

2. **Verifying the installation:**
    a. kubectl version –client

2.2 Linux

1. **Install kubectl:**
    a. sudo apt-get update
    b. sudo apt-get install -y kubectl

2. **Verifying the installation:**
    a. kubectl version –client

3. **Test Kubernetes Cluster:**
    a. kubectl get nodes

3 Building the Docker Image

1. **Ensuring having Dockerfile:**
   The Dockerfile should have the following content:

```
# Base image
FROM ubuntu:20.04

# Install network tools
RUN apt-get update && apt-get install -y \
    iputils-ping \
    curl \
    wget \
    net-tools \
    dnsutils \
    && apt-get clean \
    && rm -rf /var/lib/apt/lists/*

# Keep the container running
CMD ["bash"]
```

Fig 1. Dockerfile.

2. **Build the Image:**
   Use the docker build in the same location as the Dockerfile:
    a. docker build -t ubuntu-zt .

3. **Listing Images:**
    a. docker image ls

4. **Using Docker Daemon inside Minikube:**
    a. eval $(minikube docker-env)

4 Setting up OPA Certificate

1. **Communication between Kubernetes and OPA should be secured using TLS:**
    a. openssl genrsa -out ca.key 2048
    b. openssl req -x509 -new -nodes -sha256 -key ca.key -days 100000 -out ca.crt -subj "/CN=admission_ca"

2. **Make sure the server.conf file looks like this:**

```
cat >server.conf <<EOF
[ req ]
prompt = no
req_extensions = v3_ext
distinguished_name = dn

[ dn ]
CN = opa.opa.svc

[ v3_ext ]
basicConstraints = CA:FALSE
keyUsage = nonRepudiation, digitalSignature, keyEncipherment
extendedKeyUsage = clientAuth, serverAuth
subjectAltName = DNS:opa.opa.svc,DNS:opa.opa.svc.cluster,DNS:opa.opa.svc.cluster.local
EOF
```

Fig 2. Server.conf file [2].

3. **Run these command to generate CA Bundle key:**
    a. openssl genrsa -out server.key 2048
    b. openssl req -new -key server.key -sha256 -out server.csr -extensions v3_ext -config server.conf
    c. openssl x509 -req -in server.csr -sha256 -CA ca.crt -CAkey ca.key -CAcreateserial -out server.crt -days 100000 -extensions v3_ext -extfile server.conf

4. **Add the CA Bundle Key:**
    a. Convert the CA Bundle Key to base 64 and add it under the opa-webhook.yaml file

```
└─$ cat opa-webhook.yaml
apiVersion: admissionregistration.k8s.io/v1
kind: ValidatingWebhookConfiguration
metadata:
  name: opa-validating-webhook
webhooks:
  - name: validation.openpolicyagent.org
    clientConfig:
      service:
        name: opa
        namespace: opa
        path: "/v1/admit"
      caBundle: <OPA_CA_BUNDLE>  # Replace this with the base64 encoded CA bundle
    admissionReviewVersions: ["v1"]
    sideEffects: None
    rules:
      - operations: ["CREATE", "UPDATE"]
        apiGroups: [""]
        apiVersions: ["v1"]
        resources: ["pods"]
    failurePolicy: Ignore
```

Fig 3. Opa-webhook.yaml file.

5 Setting Up Istio

5.1 Download and Install Istio CLI

1. **Download Istio:**
    a. curl -L https://istio.io/downloadIstio | sh -
    b. cd istio-<version>
    c. export PATH=$PWD/bin:$PATH

2. **Install Istio in the cluster:**
    a. istioctl install --set profile=demo -y

3. **Verify Istio installation:**
    a. kubectl get pods -n istio-system

5.2 Enable Sidecar Injection

1. **Label namespaces for sidecar injection:**
    a. kubectl label namespace procurement istio-injection=enabled
    b. kubectl label namespace logistics istio-injection=enabled
    c. kubectl label namespace inventory istio-injection=enabled
    d. kubectl label namespace distribution istio-injection=enabled

2. **Verify the labels:**
    a. kubectl get namespaces --show-labels

6 Setting Up Hashicorp Vault

6.1 Deploy Vault Using Helm

1. **Add the HashiCorp Helm repository:**
    a. helm repo add hashicorp https://helm.releases.hashicorp.com
    b. helm repo update

2. **Install Vault:**
    a. helm install vault hashicorp/vault --namespace vault --create-namespace

3. **Unseal Vault:**
    a. vault operator init
    b. vault operator unseal

4. **Login to Vault Using the Root Token:**
    a. vault login

6.2 Configure Vault for Kubernetes

1. **Enable the Kubernetes Authentication Method:**
    a. vault auth enable Kubernetes

2. **Configure the Kubernetes Authentication Backend:**
    a. TOKEN=$(kubectl get secret $(kubectl get serviceaccount vault -n vault -o jsonpath='{.secrets[0].name}') -n vault -o jsonpath='{.data.token}' | base64 --decode)

3. **Configure the backend:**
    a. vault write auth/kubernetes/config \
    token_reviewer_jwt="$TOKEN" \
    kubernetes_host="https://$(kubectl get svc kubernetes -o jsonpath='{.spec.clusterIP}')"

6.3 Store Secrets in Vault

1. **Enable KV secrets engine:**
    a. vault secrets enable -path=secret kv

2. **Add a secret for procurement:**
    a. vault kv put secret/procurement/db username=admin password=securepassword

7 Setting Up Monitoring using Prometheus and Grafana

7.1 Deploy Prometheus

1. **Install Prometheus using Helm:**
    a. helm repo add prometheus-community https://prometheus-community.github.io/helm-charts
    b. helm repo add grafana https://grafana.github.io/helm-charts
    c. helm repo update
    d. helm install prometheus prometheus-community/prometheus --namespace monitoring --create-namespace

7.2 Deploy Grafana

1. **Install Grafana using Helm:**
    a. helm install grafana grafana/grafana --namespace monitoring

2. **Expose Grafana with a NodePort:**
    a. kubectl patch svc grafana -n monitoring -p '{"spec": {"type": "NodePort"}}'

3. **Get Grafana's Access Details:**
    a. kubectl get svc grafana -n monitoring

4. **Get the Admin Password:**
    a. kubectl get secret -n monitoring grafana -o jsonpath="{.data.admin-password}" | base64 --decode

7.3 Configure Monitoring

1. **Access Grafana via port-forward:**
    a. kubectl port-forward svc/grafana -n monitoring 3000:80

2. Add Prometheus as a data source in Grafana.

8 Deploying the Supply chain Microservices

1. **Run the Makefile to deploy the services:**
    a. make build

2. **Clean the services:**
    a. make clean

References

1. Sanjay Shivanna (2021*). Installing minikube,kubectl with chocolatey - windows.* [online] DEV Community. Available at: https://dev.to/sanjaybsm/installing-minikube-kubectl-with-chocolatey-windows-5gc7

2. Guest Expert (2022). *Open Policy Agent with Kubernetes - Tutorial (Pt. 1).* [online] GitGuardian Blog - Take Control of Your Secrets Security. Available at: https://blog.gitguardian.com/open-policy-agent-with-kubernetes-tutorial-pt-1/.