**Lesson 6 Demo 4**

**Guidelines for Understanding Transport Layer Security**

**Objective:** To understand transport layer security on Kubernetes cluster

**Tools required:** kubeadm, kubectl, kubelet, and etcd

**Prerequisites:** A Kubernetes cluster must be set up (follow steps of Lesson 2 Demo 1)

Steps to be followed:

1. Getting started with Ingress
2. Deploying Httpd and openshift
3. Generating the self-signed SSL certificate
4. Creating a tls-certificate
5. Working on Ingress

**Step 1: Getting started with Ingress**

* + 1. Deploy Ingress by using the following command:

**kubectl apply -f** [**https://raw.githubusercontent.com/kubernetes/ingress-nginx/controller-v1.1.0/deploy/static/provider/cloud/deploy.yaml**](https://raw.githubusercontent.com/kubernetes/ingress-nginx/controller-v1.1.0/deploy/static/provider/cloud/deploy.yaml)

Text

Description automatically generated

## Verify all Ingress Deployments, Pods, and Services by using the following command:

**kubectl get all -n ingress-nginx**

A screenshot of a computer

Description automatically generated

## Verify Pods created in Ingress using the following command:

**kubectl get pod -n ingress-nginx**

A screenshot of a computer

Description automatically generated

**Step 2: Deploying Httpd and openshift**

* 1. Deploy Httpd and openshift deployment and expose service of it by using the following command:

**kubectl create deployment myapp1 --image=docker.io/httpd**

**kubectl create deployment myapp2 --image=docker.io/openshift/hello-openshift**

**kubectl expose deployment myapp1 --port=80**

**kubectl expose deployment myapp2 --port=8080**

**kubectl get svc**

A screenshot of a computer

Description automatically generated

* 1. Create a directory for Ingress tls:

**mkdir ingress1**

**cd ingress1/**

Graphical user interface, text

Description automatically generated

**Step 3: Generating a self-signed SSL certificate**

* 1. Generate a self-signed SSL certificate using OpenSSL:

**openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout ingress.key -out ingress.crt -subj "/CN=master.example.com/O=security"**

A screenshot of a computer

Description automatically generated

**Step 4: Creating a tls-certificate**

* 1. Create a tls-cert certificate by using the following command:

**kubectl create secret tls tls-cert --key ingress.key --cert ingress.crt**

Text

Description automatically generated

**Step 5:** **Working on Ingress**

* 1. Apply the Ingress rule for myapp1:

**vi rule.yaml**

**apiVersion: networking.k8s.io/v1**

**kind: Ingress**

**metadata:**

**annotations:**

**nginx.ingress.kubernetes.io/rewrite-target: /$2**

**name: rewrite**

**spec:**

**tls:**

**- hosts:**

**- master.example.com**

**secretName: tls-cert**

**ingressClassName: nginx**

**rules:**

**- host: master.example.com**

**http:**

**paths:**

**- path: /**

**pathType: Prefix**

**backend:**

**service:**

**name: myapp1**

**port:**

**number: 80**

A screenshot of a computer

Description automatically generated

* 1. Create and verify the Ingress rule by using the following command:

**kubectl create -f rule.yaml**

**kubectl get ingress**

A screenshot of a computer

Description automatically generated

* 1. Paste Internal IP and hostname in the following file below the localhost line as highlighted:

**ip a | grep ens**

A screenshot of a computer

Description automatically generated

## To print IP hostnames and addresses for the localhost using the following command:

## **sudo vi /etc/hosts**

Graphical user interface, text

Description automatically generated

# Verify the Ingress pod running on which node takes Node IP and NodePort of service from the following command:

**kubectl get svc -n ingress-nginx**

**kubectl get pod -n ingress-nginx -o wide**

**kubectl get nodes -o wide**

A screenshot of a computer

Description automatically generated

# Verify the generated certificate:

**curl -kv https://master.example.com:31909/test**

Graphical user interface, text

Description automatically generated

# As shown in the above screenshot, the certificate is generated successfully.