Kubernetes Traefik Ingress LetsEncrypt – cert-manager, TLS

by FOSS TechNix

In this article we are going to cover Kubernetes Traefik Ingress LetsEncrypt – cert-manager, TLS.

Install Helm 3 on Kubernetes Cluster, Install Traefik Ingress Controller on Kubernetes using Helm 3.

Creating Deployment and service for nginx app, Creating Traefik Ingress Resources and Exposing the apps.

Pointing Traefik Ingress Loadbalancer in Domain Name provider

Here we are installing Traefik 2 on Kubernetes Cluster.

Table of Contents

Prerequisites:

- Kubernetes Cluster with v1.19.0+

netes Cluster

Install helm3 on Kubernetes Cluster on Kubernetes Cluster using below

```
curl -fsSL -o get_helm.sh https://raw.githubusercontent.com/helm/helm/master/scripts/get-helm-3
```

```
chmod 700 get_helm.sh
```

```
./get_helm.sh
```

To check helm3 version

```
helm version
```

Output:

```
version.BuildInfo{Version:"v3.6.0",
GitCommit:"7f2df6467771a75f5646b7f12afb408590ed1755", GitTreeState:"clean",
GoVersion:"go1.16.3"}
```

#2: Install Traefik Ingress Controller on Kubernetes using Helm 3

Add the Traefik ingress helm repo in Kubernetes kops cluster, follow this Traefik ingress official page to install latest Traefik ingress helm chart

```
helm repo add traefik https://helm.traefik.io/traefik
```

Update the helm repo

```
helm repo update
```

etes using Helm 3

[Optional Steps]:

To install Traefik in specific namespace use below commands

```
kubectl create ns traefik-v2
```

```
helm install --namespace=traefik-v2 \
    traefik traefik/traefik
```

To check Traefik ingress controller service

```
kubectl get svc
```

Output:

```
NAME         TYPE           CLUSTER-IP      EXTERNAL-IP
PORT(S)                      AGE
kubernetes   ClusterIP      100.64.0.1      <none>
443/TCP                      5m58s
traefik      LoadBalancer   100.68.145.32
a8f0f6c0290354e57a682620757e4271-937262111.ap-south-1.elb.amazonaws.com
80:30088/TCP,443:31429/TCP   55s
```

#3. Creating Deployment and service for nginx app

Lets deploy the sample nginx app on nginx ingress controller

Create the nginx app deployment

```
sudo nano nginx-deploy.yml
```

paste the below deployment

```
  name: nginx-app
```

```
  replicas: 1
  selector:
    matchLabels:
      app: nginx-app
  template:
    metadata:
      labels:
        app: nginx-app
    spec:
      containers:
      - name: nginx
        image: "nginx"
```

Create the nginx app service

```
sudo nano nginx-svc.yml
```

paste the below code

```
apiVersion: v1
kind: Service
metadata:
  name: nginx-app
  namespace: default
spec:
  selector:
    app: nginx-app
  ports:
  - name: http
    targetPort: 80
    port: 80
```

deploy the nginx app deployment and service on kubernetes

```
kubectl create -f nginx-deploy.yml
kubectl create -f nginx-svc.yml
```

Resources and Exposing the apps

Kubernetes to expose the apps

```
sudo nano traefik-ingress.yml
```

Paste the below nginx and nodejs app details, here service name should match with service.yml's

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: traefik-ingress
  namespace: default
  annotations:
    kubernetes.io/ingress.class: traefik
spec:
  rules:
  - host: nginxapp.fosstechnix.info
    http:
      paths:
      - backend:
          service:
            name: nginx-app
            port:
              number: 80
        path: /
        pathType: Prefix
```

deploy the traefik ingress resource on Kubernetes KOPS cluster

```
kubectl create -f traefik-ingress.yml
```

To check Kubernetes pods using kubectl

```
kubectl get pods
```

Output:

```
NAME                         READY   STATUS    RESTARTS   AGE
nginx-app-d6ff45774-tcslx    1/1     Running   0          93s
traefik-5d7c8ddd5d-7nqxt     1/1     Running   0          3m34s
```

kubectl

Output:

```
NAME        READY   UP-TO-DATE   AVAILABLE   AGE
nginx-app   1/1     1            1           2m5s
traefik     1/1     1            1           4m6s
```

To check Kubernetes service using kubectl

```
kubectl get svc
```

Output:

```
NAME         TYPE           CLUSTER-IP       EXTERNAL-IP
PORT(S)                      AGE
kubernetes   ClusterIP      100.64.0.1       <none>
443/TCP                      10m
nginx-app    ClusterIP      100.68.179.217   <none>
80/TCP                       2m58s
traefik      LoadBalancer   100.68.145.32
a8f0f6c0290354e57a682620757e4271-937262111.ap-south-1.elb.amazonaws.com
80:30088/TCP,443:31429/TCP   5m4s
```

To check Kubernetes ingress using kubectl

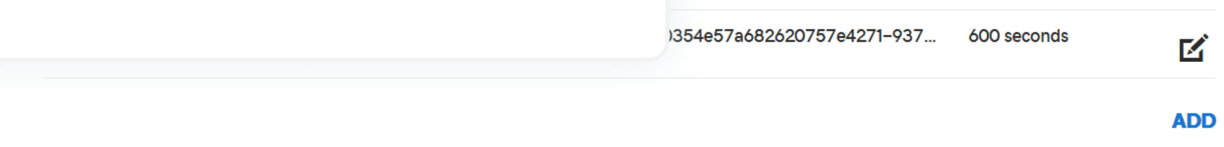
```
kubectl get ingress
```

Output:

```
NAME              CLASS    HOSTS                         ADDRESS   PORTS   AGE
traefik-ingress   <none>   nginxapp.fosstechnix.info               80      2m2s
```

#5. Pointing Traefik Ingress Loadbalancer in Domain Name provider

we have pointed loadbalancer URL in Domain name provider as CNAME.



354e57a682620757e4271-937… 600 seconds

Now access nginx app using domain name

#6: Configure cert manager for Traefik Ingress

once treafik ingress controller setup is done on your Kubernetes cluster, Lets install and configure cert manager using below kubectl command for **Kubernetes version 1.16+**

```
kubectl apply --validate=false -f https://github.com/jetstack/cert-manager
/releases/download/v1.0.1/cert-manager.yaml
```

Sample Output:

```
service/cert-manager unchanged
service/cert-manager-webhook unchanged
deployment.apps/cert-manager-cainjector unchanged
deployment.apps/cert-manager unchanged
deployment.apps/cert-manager-webhook unchanged
mutatingwebhookconfiguration.admissionregistration.k8s.io/cert-manager-webhook
configured
validatingwebhookconfiguration.admissionregistration.k8s.io/cert-manager-webhook
configured
```

for **Kubernetes <1.16 version**

```
kubectl apply --validate=false -f https://github.com/jetstack/cert-manager
/releases/download/v1.0.1/cert-manager-legacy.yaml
```

it will install cert manager packages on your k8s cluster

ss LetsEncrypt

To configure Kubernetes Traefik Ingress Controller LetsEncrypt , navigate to cert
ure Let’s Encrypt Issuer, copy the let’s
below.

```
sudo nano  letsencrypt-issuer.yml
```

```
apiVersion: cert-manager.io/v1
kind: ClusterIssuer
metadata:
  name: letsencrypt-prod
  namespace: default
spec:
  acme:
    # The ACME server URL
    server: https://acme-v02.api.letsencrypt.org/directory
    # Email address used for ACME registration
    email: fosstechnixinfo@gmail.com
    # Name of a secret used to store the ACME account private key
    privateKeySecretRef:
      name: letsencrypt-prod
    # Enable the HTTP-01 challenge provider
    solvers:
    - http01:
        ingress:
          class: traefik
```

```
kubectl apply -f letsencrypt-issuer.yml
```

We have deployed let’s encrypt issuer which issues certificates,

#8: Creating Traefik Ingress Let’s Encrypt TLS

Certificate

ypt TLS certificate for your microservice.

```
sudo nano letsencrypt-cert.yml
```

LS certificate as per your microservice/domain name

```
apiVersion: cert-manager.io/v1
kind: Certificate
metadata:
  name: nginxapp.fosstechnix.info
  namespace: default
spec:
  secretName: nginxapp.fosstechnix.info-tls
  issuerRef:
    name: letsencrypt-prod
    kind: ClusterIssuer
  commonName: nginxapp.fosstechnix.info
  dnsNames:
  - nginxapp.fosstechnix.info
```

```
kubectl apply -f letsencrypt-cert.yml
```

once done, it will create a Traefik ingress letsencrypt TLS certificate for domain nginxapp.fosstechnix.info and injects into Kubernetes secrets.

Lets check the certificate is created

```
kubectl get certificates nginxapp.fosstechnix.info
```

Output:

```
kubectl get certificates nginxapp.fosstechnix.info
NAME                        READY   SECRET                          AGE
nginxapp.fosstechnix.info   True    nginxapp.fosstechnix.info-tls   32s
```

Let’s check secrets to check Traefik Ingress letsencrypt TLS

```
kubectl get secrets nginxapp.fosstechnix.info-tls
```

```
                                                            nfo-tls
                                                                       DATA   AGE
                                                            .io/tls    2      36s
```

We have covered Kubernetes Traefik Ingress Controller LetsEncrypt [cert-manager, TLS]

#9: Point Traefik Ingress Let’s Encrypt Certificate in Traefik Ingress Resource

Now point/refer the generated Nginx Ingress Let’s Encrypt in your Kubernetes Traefik Ingress resource as shown below.

Add the **highlighted lines** in **Traefik ingress resource**.

```
kubectl edit ingress traefik-ingress
```

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  annotations:
    cert-manager.io/cluster-issuer: letsencrypt-prod
    kubernetes.io/ingress.class: traefik
  creationTimestamp: "2021-04-22T03:20:24Z"
  generation: 2
  name: traefik-ingress
  namespace: default
  resourceVersion: "5902"
  uid: 62300582-7b91-4f56-a229-75f9664f9334
spec:
  rules:
  - host: nginxapp.fosstechnix.info
    http:
      paths:
      - backend:
          service:
            name: nginx-app
            port:
              number: 80
        path: /
        pathType: Prefix
  tls:
```

```
  - hosts:
    - nginxapp.fosstechnix.info
    secretName: nginxapp.fosstechnix.info-tls
```

```
,81784d-1960968212.ap-south-
1.elb.amazonaws.com
```

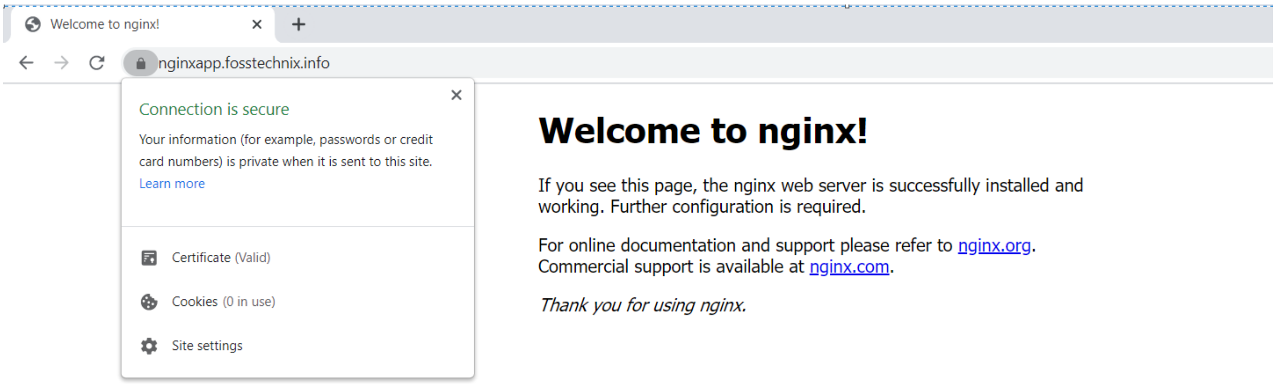
.fosstechnix.info-tls and added annotation **cert-manager.io/cluster-issuer: letsencrypt-prod.**

Note: secret and certificates should be in same namespace as ingress.

#10: Accessing Traefik Ingress Resources using Let's Encrypt

Finally we can see your application site **https://nginxapp.fosstechnix.info** using Lets's Encrypt SSL (Kubernetes Traefik Ingress Controller LetsEncrypt-cert-manager,TLS).

```
https://nginxapp.fosstechnix.info
```



Connection is secure

Your information (for example, passwords or credit card numbers) is private when it is sent to this site.
Learn more

Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

#6: Accessing Traefik Dashboard

By default traefik dashboard is not exposed when we install traefik using helm chart for security reason,

There are multiple ways to access traefik dashboard, lets access traefik dashboard by forwarding traefik pod to any address using below command.

```
kubectl port-forward $(kubectl get pods --selector
"app.kubernetes.io/name=traefik" --output=name) --address 0.0.0.0 9000:9000
```

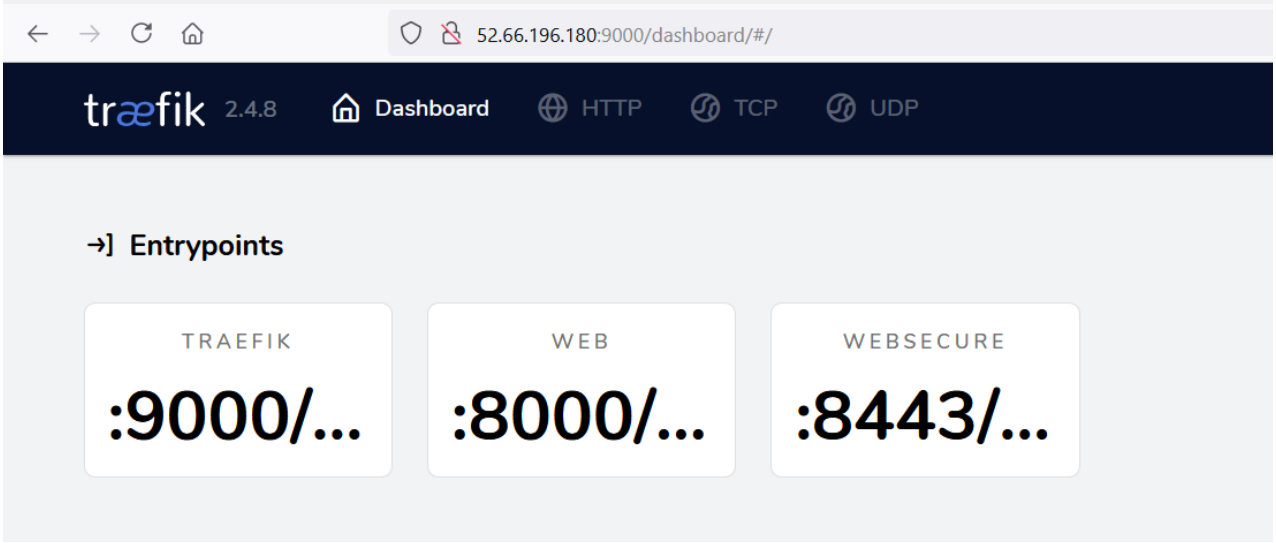
Now you can access traefik with IP address of instance from instance.

OR

Cluster Node IP with port number

```
http://65.2.81.244:9000/dashboard/#/
```

Output:



52.66.196.180:9000/dashboard/#/

Conclusion:

We have covered Kubernetes Traefik Ingress LetsEncrypt – cert-manager, TLS, Install Helm 3 on Kubernetes Cluster, Install Traefik Ingress Controller on Kubernetes using Helm 3, Creating Deployment and service for nginx app, Creating Traefik Ingress Resources and Exposing the apps, Pointing Traefik Ingress Loadbalancer in Domain Name provider

Related Articles:

- Configure Traefik Ingress Controller on Kubernetes [5 Steps]

MENU

sitory on Ubuntu 20.04 LTS

Date and Time in PHP with Examples [2 Steps]

FOSS TechNix

FOSS TechNix (Free,Open Source Software's and Technology Nix*) founded in 2019 is a community platform where you can find How-to Guides, articles for DevOps Tools,Linux and Databases.

Leave a Comment

Name *

Email *

Website

Search ...

RECENT POSTS

How to Configure Postfix with Gmail on Ubuntu

04 LTS

How to Install Elasticsearch 8 on Ubuntu 20.04 LTS

Ansible Variables with Examples

Ansible Playbook Example

ABOUT FOSS TECHNIX

FOSS TechNix (Free ,Open Source Softwares and Technology Nix*) is a community site where you can find How-To Guides, Articles, Tips and Tricks for DevOps Tools, Linux, Databases, Clouds and Automation.

DEVOPS TOOLS & TIPS

What does DevOps Engineer do ?

Kubernetes Tutorial for Beginners