

Setup SSL on Ingress

Ingress bisa dikatakan sebagai gateway untuk meng expose layanan yang ada di cluster kubernetes. Ingress memiliki beberapa macam controller, namun yang paling populer adalah ingress-nginx-controller. Kali ini kita akan mengkonfigurasi SSL pada ingress-nginx-controller agar bisa diakses melalui protokol HTTPS.

Sebelum kita melakukan setup untuk SSL nya, kita pastikan dulu ingress sudah bisa dijalankan melalui protokol HTTP. Apabila belum mengkonfigurasi platform pada ingress sama sekali, ikuti petunjuk ini dari awal. Apabila ingress sudah siap digunakan, silahkan [klik disini](#).

Setup Ingress

Untuk implementasinya, kita harus mengaktifkan ingress terlebih dahulu dengan menggunakan minikube.

```
minikube addons enable ingress
```

```
ex-dec@ex-pc:~/project/private-notes> minikube addons list
```

ADDON NAME	PROFILE	STATUS	MAINTAINER
ambassador	minikube	disabled	3rd party (Ambassador)
auto-pause	minikube	disabled	minikube
cloud-spanner	minikube	disabled	Google
csi-hostpath-driver	minikube	disabled	Kubernetes
dashboard	minikube	enabled ✓	Kubernetes
default-storageclass	minikube	enabled ✓	Kubernetes
efk	minikube	disabled	3rd party (Elastic)
freshpod	minikube	disabled	Google
gcp-auth	minikube	disabled	Google
gvisor	minikube	disabled	minikube
headlamp	minikube	disabled	3rd party (kinvolk.io)
helm-tiller	minikube	disabled	3rd party (Helm)
inaccel	minikube	disabled	3rd party (InAccel [info@inaccel.com])
ingress	minikube	disabled	Kubernetes
ingress-dns	minikube	disabled	minikube
inspektor-gadget	minikube	disabled	3rd party (inspektor-gadget.io)
istio	minikube	disabled	3rd party (Istio)
istio-provisioner	minikube	disabled	3rd party (Istio)
kong	minikube	disabled	3rd party (Kong HQ)
kubeflow	minikube	disabled	3rd party
kubevirt	minikube	disabled	3rd party (KubeVirt)
logviewer	minikube	disabled	3rd party (unknown)
metallb	minikube	disabled	3rd party (MetalLB)
metrics-server	minikube	enabled ✓	Kubernetes
nvidia-device-plugin	minikube	disabled	3rd party (NVIDIA)
nvidia-driver-installer	minikube	disabled	3rd party (Nvidia)
nvidia-gpu-device-plugin	minikube	disabled	3rd party (Nvidia)
olm	minikube	disabled	3rd party (Operator Framework)
pod-security-policy	minikube	disabled	3rd party (unknown)
portainer	minikube	disabled	3rd party (Portainer.io)
registry	minikube	disabled	minikube
registry-aliases	minikube	disabled	3rd party (unknown)
registry-creds	minikube	disabled	3rd party (UPMC Enterprises)
storage-provisioner	minikube	enabled ✓	minikube
storage-provisioner-gluster	minikube	disabled	3rd party (Gluster)
storage-provisioner-rancher	minikube	disabled	3rd party (Rancher)
volumesnapshots	minikube	disabled	Kubernetes

```
ex-dec@ex-pc:~/project/private-notes> minikube addons enable ingress
```

💡 ingress is an addon maintained by Kubernetes. For any concerns contact minikube on GitHub.

You can view the list of minikube maintainers at: <https://github.com/kubernetes/minikube/blob/master/OWNERS>

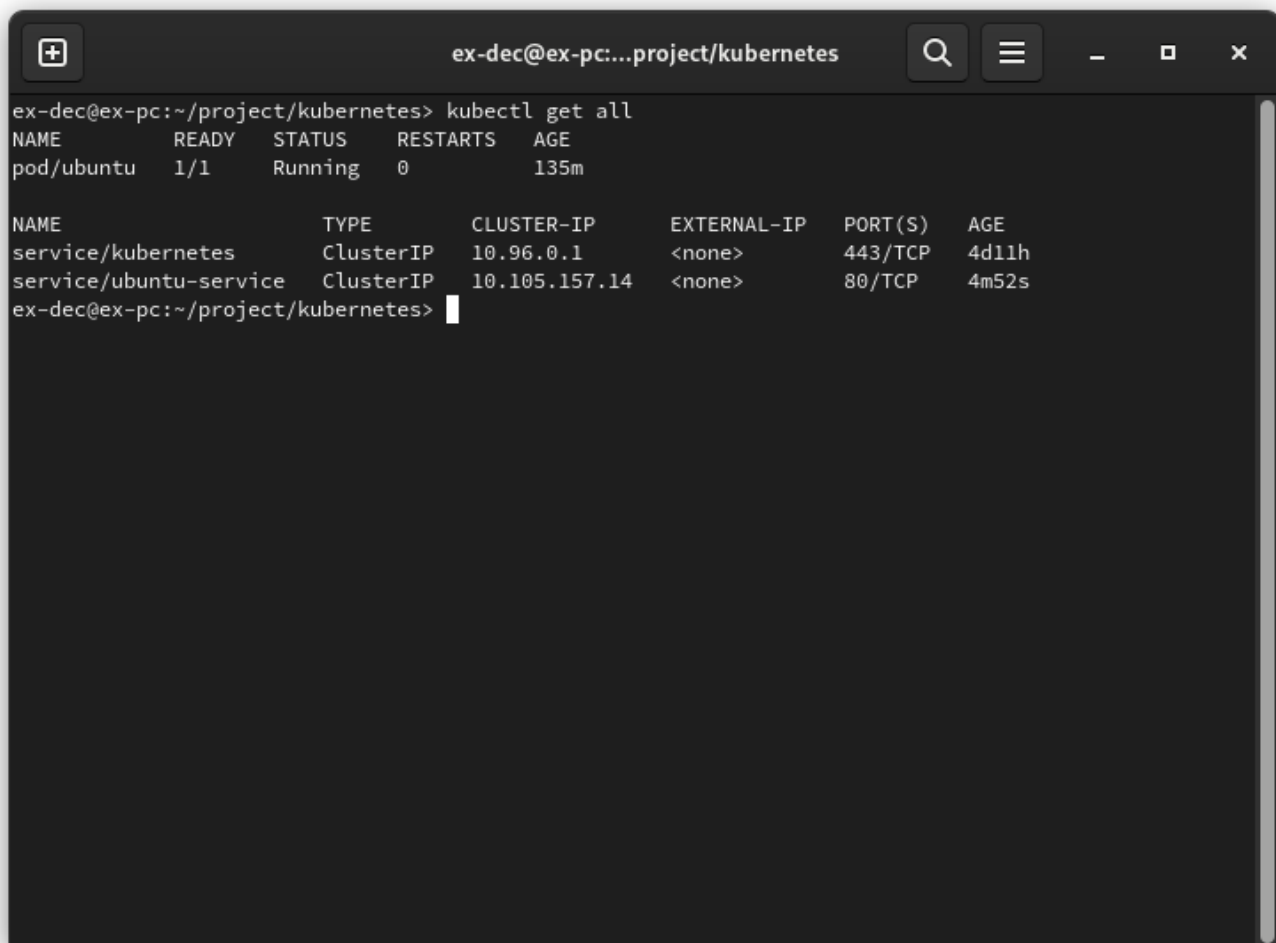
- Using image registry.k8s.io/ingress-nginx/controller:v1.9.4
- Using image registry.k8s.io/ingress-nginx/kube-webhook-certgen:v20231011-8b53cabe0
- Using image registry.k8s.io/ingress-nginx/kube-webhook-certgen:v20231011-8b53cabe0

🔍 Verifying ingress addon...

🌟 The 'ingress' addon is enabled

```
ex-dec@ex-pc:~/project/private-notes> 
```

Setelah ingress aktif, siapkan pod yang akan dibuka akses ke luar. Pastikan pod sudah berjalan dan sudah dijalankan dengan servicenya. Kita cukup menggunakan service dengan ClusterIP untuk melakukan expose port dari pod nya.

A terminal window titled 'ex-dec@ex-pc:...project/kubernetes' showing the output of 'kubectl get all'. The output lists a pod and two services.

```
ex-dec@ex-pc:~/project/kubernetes> kubectl get all
NAME                READY   STATUS    RESTARTS   AGE
pod/ubuntu           1/1     Running   0           135m

NAME                TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)    AGE
service/kubernetes  ClusterIP   10.96.0.1    <none>        443/TCP    4d11h
service/ubuntu-service ClusterIP   10.105.157.14 <none>        80/TCP     4m52s
ex-dec@ex-pc:~/project/kubernetes>
```

Setelah kedua hal tersebut sudah berjalan, sekarang kita siapkan ingress nya.

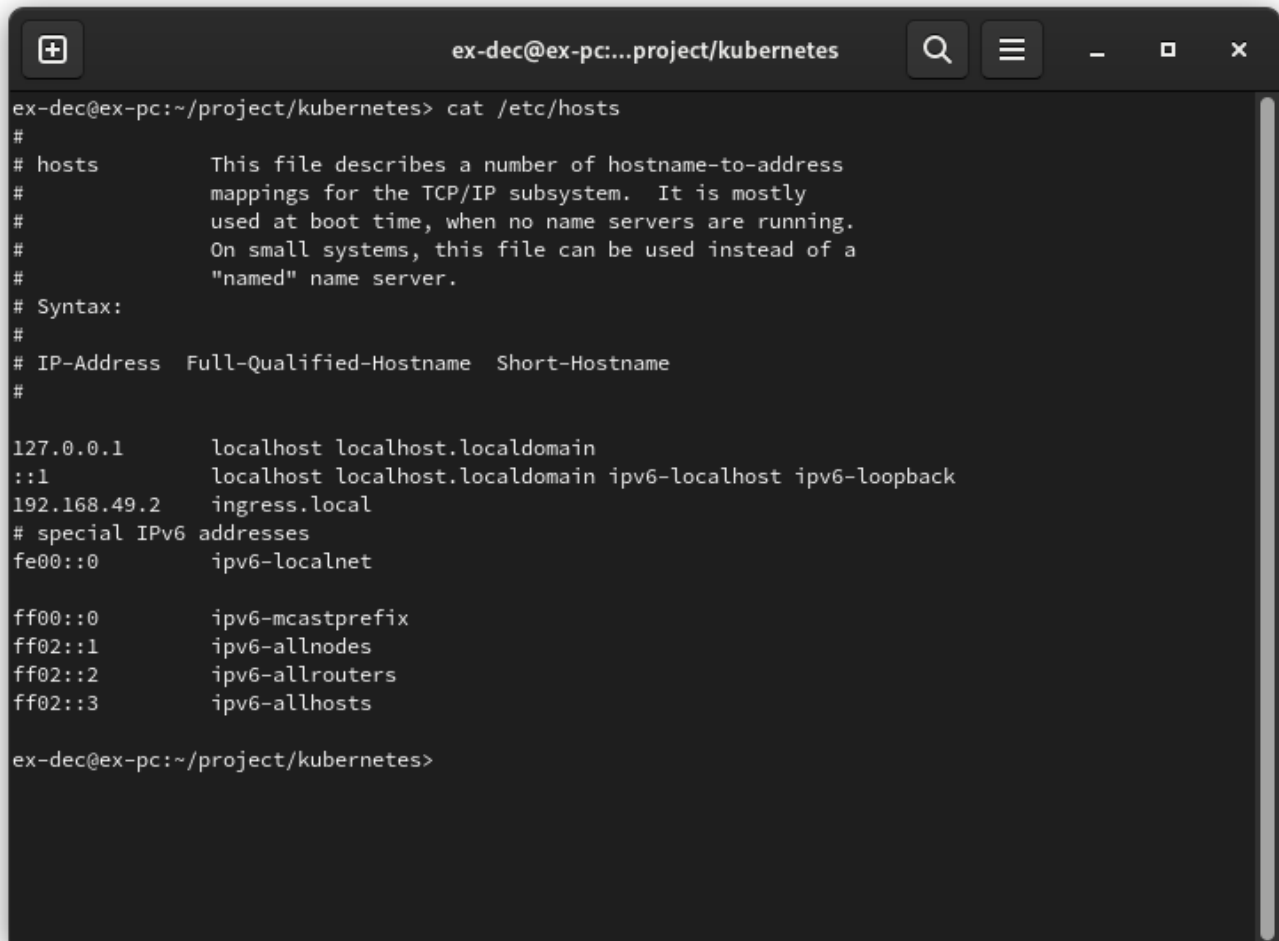
ubuntu-ingress.yaml

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

Terapkan konfigurasi tersebut dan konfigurasi hostname agar bisa terbaca di host kita. Untuk host saya sendiri perlu mengkonfigurasi pada file `/etc/hosts`.

Disini saya akan menggunakan host 'ingress.local' sesuai dengan konfigurasi diatas.

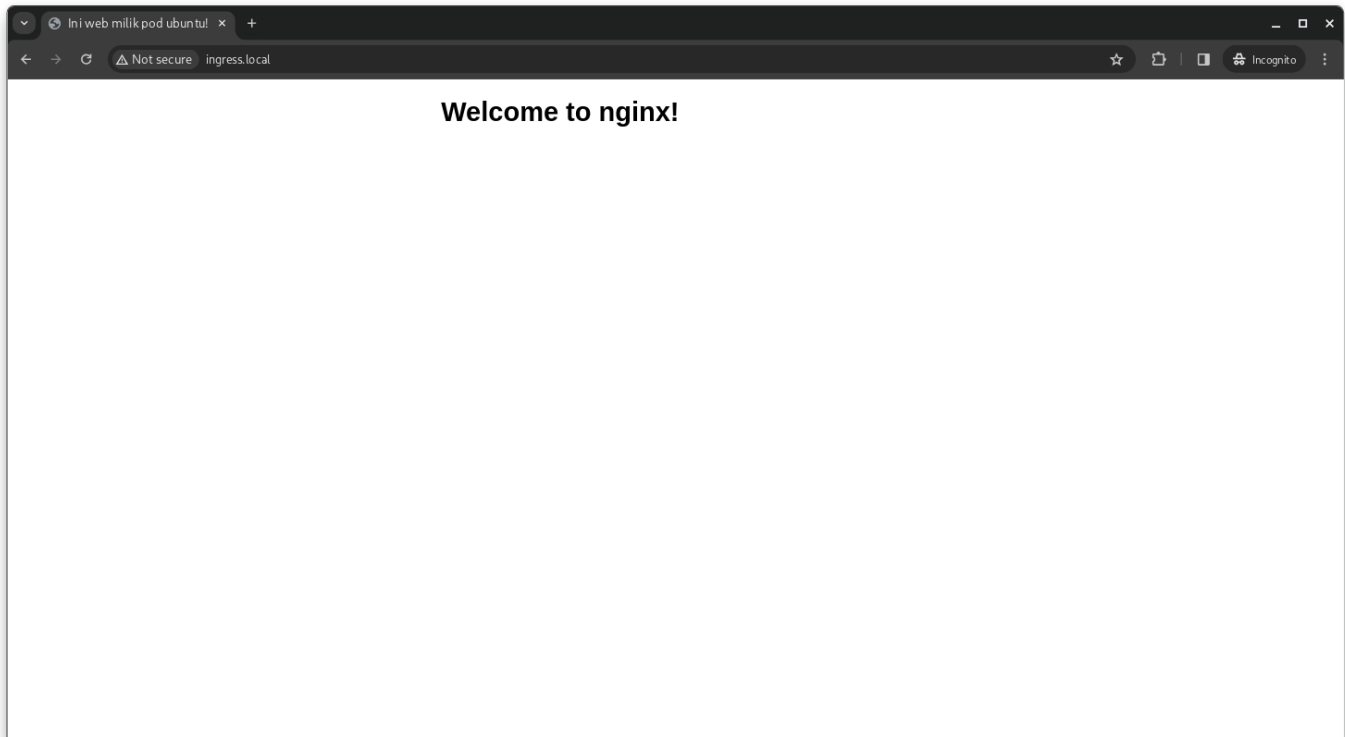
`/etc/hosts`

A terminal window titled 'ex-dec@ex-pc:...project/kubernetes' showing the output of the command 'cat /etc/hosts'. The output lists various IP addresses and their corresponding hostnames, including localhost, ingress.local, and several IPv6 addresses.

```
ex-dec@ex-pc:~/project/kubernetes> cat /etc/hosts
#
# hosts        This file describes a number of hostname-to-address
#               mappings for the TCP/IP subsystem.  It is mostly
#               used at boot time, when no name servers are running.
#               On small systems, this file can be used instead of a
#               "named" name server.
#
# Syntax:
#
# IP-Address   Full-Qualified-Hostname  Short-Hostname
#
127.0.0.1      localhost localhost.localdomain
::1           localhost localhost.localdomain ipv6-localhost ipv6-loopback
192.168.49.2   ingress.local
# special IPv6 addresses
fe00::0       ipv6-localnet
ff00::0       ipv6-mcastprefix
ff02::1       ipv6-allnodes
ff02::2       ipv6-allrouters
ff02::3       ipv6-allhosts

ex-dec@ex-pc:~/project/kubernetes>
```

Setelah itu kita coba akses melalui browser.



SSL

Berikut adalah langkah-langkah yang bisa dilakukan untuk menggunakan ssl.

1. Generate SSL

Untuk melakukan generate kita perlu memastikan bahwa kita sudah menginstall openssl.

```
ex-dec@ex-pc:~  
ex-dec@ex-pc:~> openssl version  
OpenSSL 3.1.4 24 Oct 2023 (Library: OpenSSL 3.1.4 24 Oct 2023)  
ex-dec@ex-pc:~>
```

Apabila sudah terinstall dengan baik, lakukan generate dengan perintah berikut.

```
openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout  
${KEY_FILE} -out ${CERT_FILE} -subj "/CN=${HOST}/O=${HOST}" -addext  
"subjectAltName = DNS:${HOST}"
```

Kalau disesuaikan dengan kebutuhan saat ini, maka perintahnya menjadi seperti berikut

```
openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout secret.key  
-out secret.crt -subj "/CN=ingress.local/O=ingress.local" -addext  
"subjectAltName = DNS:ingress.local"
```

Hasilnya akan menjadi seperti berikut

[illegible]

2. Menambahkan key ke kubernetes

Kubernetes memiliki layanan untuk menyimpan key yaitu secret. Pada secret, semua key disimpan dan bisa digunakan oleh berbagai macam layanan yang ada di kubernetes, salah satunya adalah ingress. Untuk menambahkan key tersebut, kita bisa menggunakan perintah berikut

```
kubectl create secret tls ${CERT_NAME} --key ${KEY_FILE} --cert  
${CERT_FILE}
```

Apabila disesuaikan dengan kebutuhan diatas, maka perintahnya akan menjadi seperti berikut

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

```
ex-dec@ex-pc:~/project/kubernetes/secret> kubectl create secret tls tls-ingress --key secret.key --cert secret.crt
secret/tls-ingress created
ex-dec@ex-pc:~/project/kubernetes/secret> kubectl get secret
NAME          TYPE          DATA   AGE
tls-ingress   kubernetes.io/tls  2       6s
ex-dec@ex-pc:~/project/kubernetes/secret>
```

3. Menambahkan secret pada ingress

Secara default, ingress akan menggunakan koneksi tanpa tls. Maka dari itu, kita tambahkan secret yang sebelumnya sudah ditambahkan ke cluster kita. File konfigurasi ingress yang baru akan tampak seperti berikut

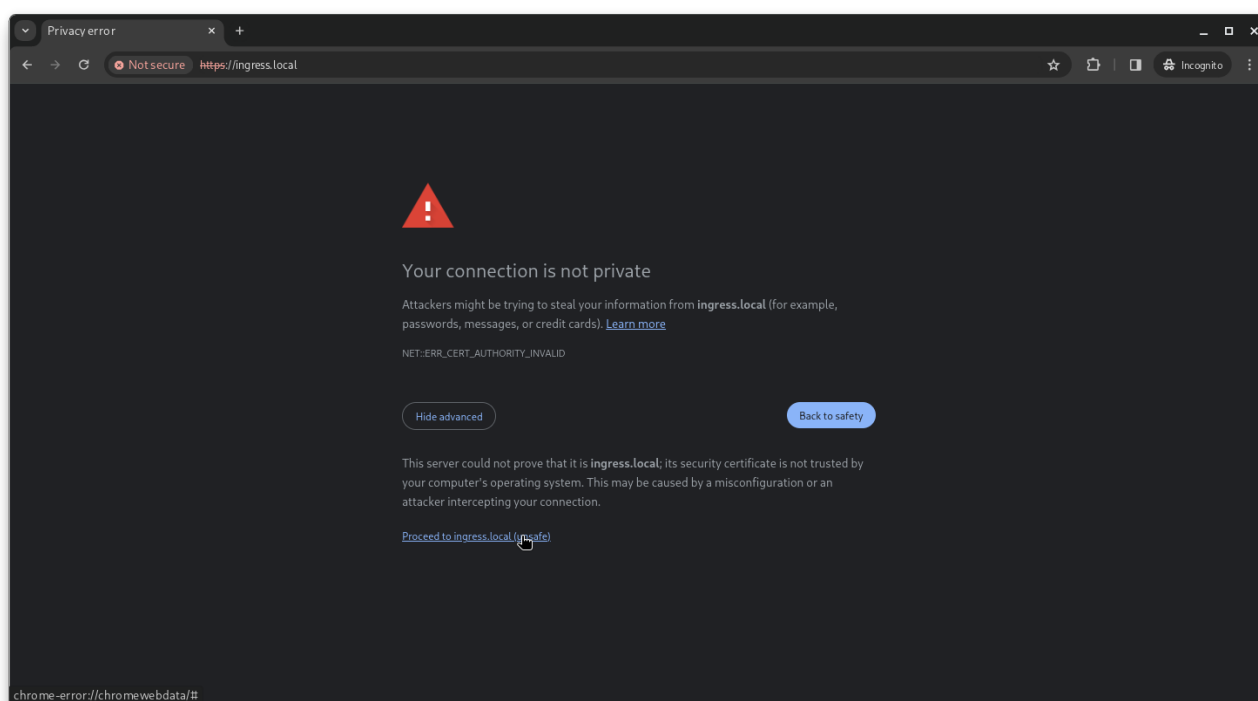
ubuntu-ingress-ssl.yaml

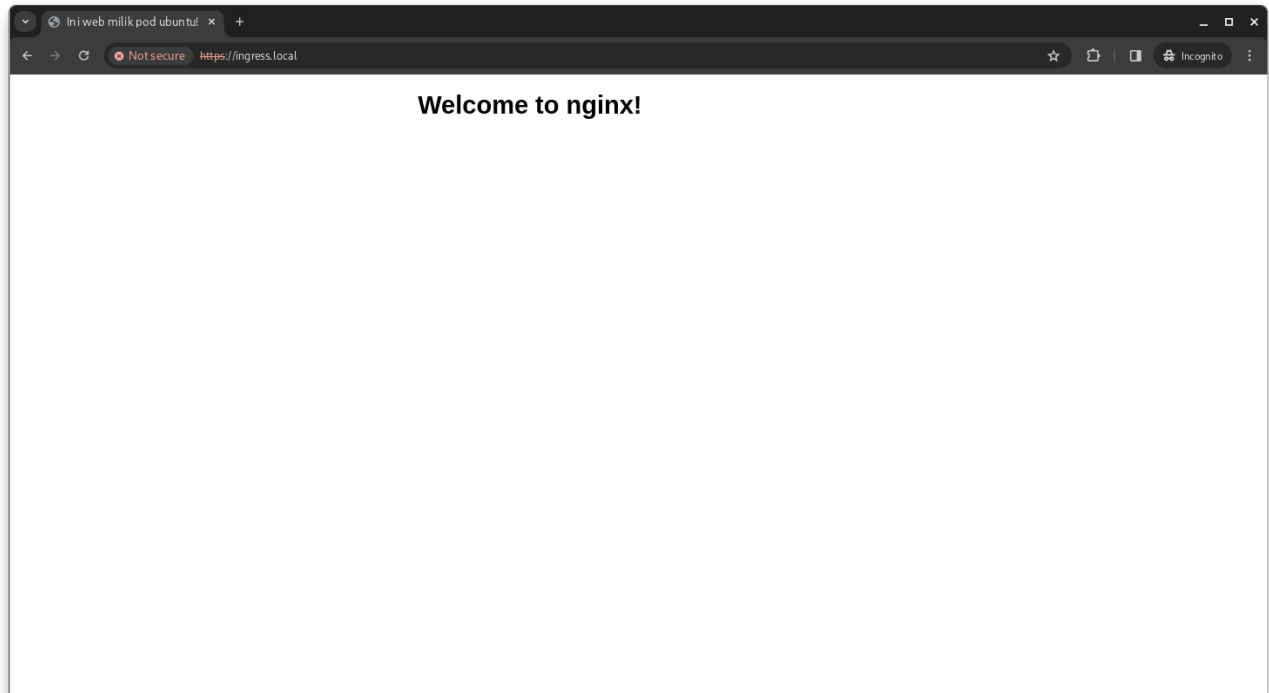
```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: ubuntu-ingress
  annotations:
    nginx.ingress.kubernetes.io/rewrite-target: /
spec:
  ingressClassName: nginx
  tls:
  - hosts:
    - ingress.local
    secretName: tls-ingress
  rules:
  - host: ingress.local
    http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          service:
            name: ubuntu-service
            port:
              number: 80
```

Terapkan file tersebut dan hasilnya akan menjadi seperti berikut

```
ex-dec@ex-pc:~/project/kubernetes> kubectl apply -f ubuntu-ingress-ssl.yaml
ingress.networking.k8s.io/ubuntu-ingress configured
ex-dec@ex-pc:~/project/kubernetes> kubectl describe ingress ubuntu-ingress
Name:          ubuntu-ingress
Labels:        <none>
Namespace:     default
Address:       192.168.49.2
Ingress Class: nginx
Default backend: <default>
TLS:
  tls-ingress terminates ingress.local
Rules:
  Host      Path  Backends
  ----      -
  ingress.local  /    ubuntu-service:80 (10.244.0.91:80)
Annotations:  nginx.ingress.kubernetes.io/rewrite-target: /
Events:
  Type    Reason    Age          From                    Message
  ----    -
  Normal  Sync      9s (x6 over 128m)  nginx-ingress-controller  Scheduled for sync
ex-dec@ex-pc:~/project/kubernetes>
```

Ketika dilakukan percobaan pada browser, akan muncul warning. Hal tersebut tidak menjadi masalah karena kita sedang menggunakan platform ini di local dan sertifikat yang kita buat sebelumnya merupakan hasil generate secara private.





Selamat!!! Ingress yang sebelumnya hanya berjalan pada protokol http, sudah bisa berjalan menggunakan protokol https dan kita bisa menggunakan protokol tersebut.