Ingress

Ingress

- Ensemble de règles pour la connection aux services du cluster depuis internet
- Nécessite qu'un **Ingress Controller** soit déployé
  - container Docker avec un processus de contrôle
  - load balancer managé par Kubernetes
  - exemples : GCE, nginx, Traefik, HAProxy
- Différents cas d'usage
  - routage HTTP, ex: hôtes virtuels basés sur le nom
  - terminaison TLS
  - Load balancing
- Add-ons pour Minikube: "*$ minikube addons enable ingress*"

Ingress

D’autres options pour exposer les services à l’extérieur

- Service de type NodePort
  - statique
  - load balancer externe pour le dispatch entre plusieurs nodes
- Service de type Load Balancer
  - nécessite l’utilisation d’un cloud provider (AWS, GCE, Azure, ...)

Ingress : routage par nom de domaine

```
$ cat www-ingress-domain.yaml
apiVersion: extensions/v1beta1                  Objet présent dans l'API beta
kind: Ingress
metadata:
  name: www-domain
spec:
  rules:
    - host: www.example.com
      http:
        paths:
          - backend:
              serviceName: www
              servicePort: 80
```

Ingress : routage par nom de domaine

```
$ cat www-ingress-domain.yaml
apiVersion: extensions/v1beta1
kind: Ingress                                    Ressource de type Ingress
metadata:
  name: www-domain
spec:
  rules:
    - host: www.example.com
      http:
        paths:
          - backend:
              serviceName: www
              servicePort: 80
```

Ingress : routage par nom de domaine

```
$ cat www-ingress-domain.yaml
apiVersion: extensions/v1beta1
kind: Ingress
metadata:                                    Définition du nom de la ressource
  name: www-domain
spec:
  rules:
    - host: www.example.com
      http:
        paths:
          - backend:
              serviceName: www
              servicePort: 80
```

Ingress : routage par nom de domaine

```
$ cat www-ingress-domain.yaml
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: www-domain
spec:
  rules:
    - host: www.example.com
      http:
        paths:
          - backend:
              serviceName: www
              servicePort: 80
```

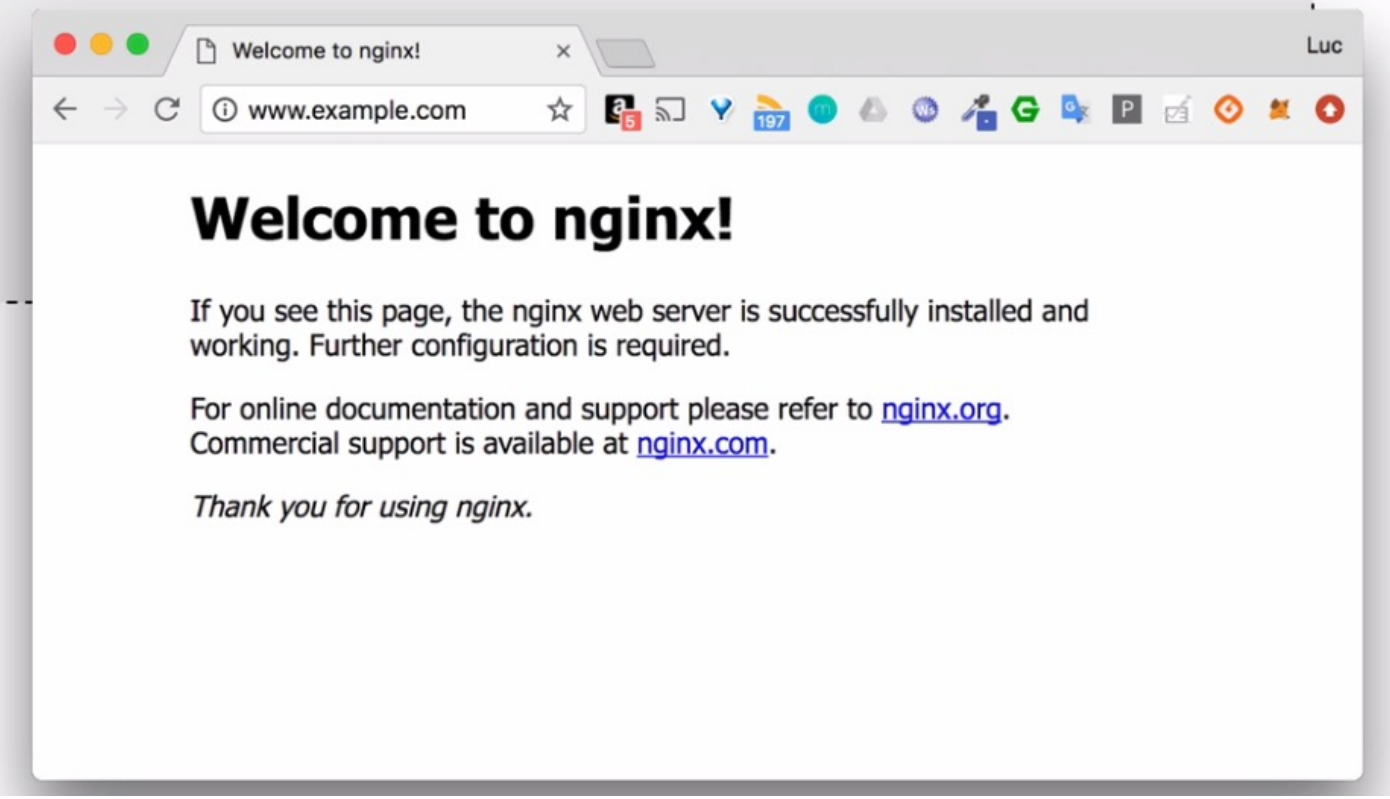
Les requètes envoyées sur www.example.com seront forwardées sur le service www

Ingress : routage par nom de domaine

```
# Création d'un Deployment basé sur nginx
$ kubectl run www --image=nginx:1.12.2
deployment "www" created

# Exposition du Deployment via un Service
$ kubectl expose deployment www --port=80 --target-port=80
service "www" exposed

# Création de l'objet Ingress
$ kubectl create -f www-ingress-domain.yaml
ingress "www-domain" created
```



Ingress : routage via le path de la requête

```
$ cat www-ingress-path.yaml
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  annotations:
    ingress.kubernetes.io/rewrite-target: /
  name: www-path
spec:
  rules:
  - host: example.com
    http:
      paths:
      - path: /www                                  Les requètes envoyés sur example.com/www sont forwardées
        backend:                                    sur le service www
          serviceName: w3
          servicePort: 80
```

Ingress : routage via le path de la requête

```
$ cat www-ingress-path.yaml
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  annotations:
    ingress.kubernetes.io/rewrite-target: /        Annotation utilisée pour la ré-écriture des requêtes
  name: www-path
spec:
  rules:
  - host: example.com
    http:
      paths:
      - path: /www
        backend:
          serviceName: w3
          servicePort: 80
```

Ingress : routage via le path de la requête

```
# Création d'un Deployment basé sur nginx
$ kubectl run w3 --image=nginx:1.12.2
deployment "w3" created

# Exposition du Deployment via un Service
$ kubectl expose deployment w3 --port=80 --target-port=80
service "w3" exposed

# Création de l'objet Ingress
$ kubectl create -f www-ingress-domain.yaml
ingress "www-domain" created
```