Comment créer un service dans Kubernetes

Un service est une couche d'abstraction sur les Pods. Il définit un ensemble logique de Pods. Il fournit une adresse IP et un nom DNS uniques grâce auxquels les pods sont accessibles. Il est utilisé pour exposer les gousses.

Il existe 3 types de services différents dans Kubernetes :

1. ClusterIP : il
   expose le service au sein du cluster Kubernetes. Ce service n'est accessible qu'à partir du cluster. Il n'est pas accessible depuis l'extérieur du cluster.
2. NodePort : il
   exposera le service sur un port statique du nœud déployé. Ce service est accessible depuis l'extérieur du cluster à l'aide de NodeIP:Nodeport.
3. Équilibreur de charge :
   Expose le service en externe à l'aide de l'équilibreur de charge d'un fournisseur de cloud, cela crée une adresse IP publique sur le fournisseur de cloud 4.

ExternalName :

Il mappe le service au contenu du champ externalName en renvoyant un enregistrement CNAME

Cliquez ici pour en savoir plus sur le service Kubernetes.

Dans cet article, nous verrons les étapes pour créer un Serice de type NodePort.

Conditions préalables

1. Cluster Kubernetes avec au moins 1 nœud de travail.
   Si vous souhaitez apprendre à créer un cluster Kubernetes, cliquez ici. Ce guide vous aidera à créer un cluster Kubernetes avec 1 maître et 2 nœuds sur les instances AWS Ubuntu EC2.

Qu'allons nous faire

1. Créer un service

Créer un service

Tout d'abord, nous allons créer un déploiement en utilisant la définition suivante vers lequel le service redirigera toutes les requêtes qui lui parviennent.

Créez un nouveau fichier et ajoutez-y le contenu suivant, cela créera un déploiement pour Nginx.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx
spec:
  strategy:
    type: Recreate
  selector:
    matchLabels:
      app: nginx
  replicas: 3
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx
        ports:
        - containerPort: 80
```

```
master $ vim my-deployment.yml
master $ cat my-deployment.yml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx
spec:
  strategy:
    type: Recreate
  selector:
    matchLabels:
      app: nginx
  replicas: 3
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx
        ports:
        - containerPort: 80
master $
```

Pour créer un déploiement, exécutez la commande suivante.

```
kubectl create -f my-deployment.yml
```

```
master $ kubectl create -f my-deployment.yml
deployment.apps/nginx created
```

Cela créera un déploiement pour Nginx avec 3 réplicas.

Vous pouvez obtenir les détails du déploiement, du jeu de réplicas et du pod à l'aide des commandes suivantes.

```
kubectl get deployment | grep nginx
```

```
kubectl get replicaset | grep nginx
```

```
kubectl get pod | grep nginx
```

```
master $ kubectl create -f my-deployment.yml
deployment.apps/nginx created
master $ kubectl get deployment | grep nginx
nginx   3/3     3            3           14s
master $ kubectl get replicaset | grep nginx
nginx-d46f5678b   3         3         3       40s
master $ kubectl get pod | grep nginx
nginx-d46f5678b-2d9r7   1/1     Running   0          51s
nginx-d46f5678b-ddqhc   1/1     Running   0          51s
nginx-d46f5678b-kj99p   1/1     Running   0          51s
master $
```

Dans la capture d'écran ci-dessus, vous pouvez voir que 3 répliques de Nginx ont été créées.

Maintenant, créez une définition de service en utilisant le contenu suivant.

```
vim my-service.yml
```

```
apiVersion: v1
kind: Service
metadata:
  name: nginx
  namespace: default
  labels:
```

```
    app: nginx
spec:
  externalTrafficPolicy: Local
  ports:
  - name: http
    port: 80
    protocol: TCP
    targetPort: 80
  selector:
    app: nginx
  type: NodePort
```

```
master $ vim my-service.yml
master $ cat my-service.yml
apiVersion: v1
kind: Service
metadata:
  name: nginx
  namespace: default
  labels:
    app: nginx
spec:
  externalTrafficPolicy: Local
  ports:
  - name: http
    port: 80
    protocol: TCP
    targetPort: 80
  selector:
    app: nginx
  type: NodePort

master $
```

La définition de service ci-dessus créera un service de type NodePort en utilisant l'espace de noms par défaut et rediriger les demandes vers Pod correspondant à l'étiquette nginx, c'est-à-dire les pods que nous avons créés à l'aide de l'étape de création de déploiement précédente.

Exécutez la commande suivante pour créer un service.

```
kubectl create -f my-service.yml
```

```
master $ kubectl create -f my-service.yml
service/nginx created
master $
```

Obtenez les détails du service et vérifiez le NodePort sur lequel le service sera disponible.

```
kubectl get service | grep nginx
```

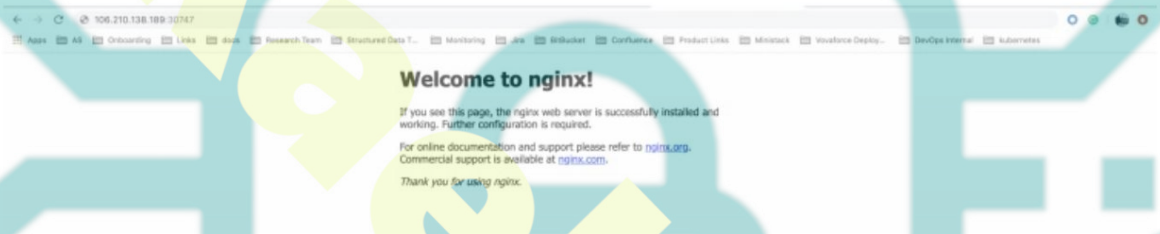
```
kubectl describe service nginx
```

```
master $ kubectl get service | grep nginx
nginx        NodePort    10.99.79.56   <none>        80:30747/TCP   56s
master $ kubectl describe service nginx
Name:                     nginx
Namespace:                default
Labels:                   app=nginx
Annotations:              <none>
Selector:                 app=nginx
Type:                     NodePort
IP:                       10.99.79.56
Port:                     http  80/TCP
TargetPort:               80/TCP
NodePort:                 http  30747/TCP
Endpoints:                10.244.1.3:80,10.244.1.4:80,10.244.1.5:80
Session Affinity:         None
External Traffic Policy:  Local
Events:                   <none>
master $
```

Dans la capture d'écran ci-dessus, on peut voir que le service est disponible sur le port 30747. Cela peut être différent pour vous car le port est attribué de manière aléatoire dans la plage disponible.

Désormais, l'application nginx est accessible via ce service sur NodeIp:NodePort

Ici, c'est : 106.210.138.189:30747



Foire aux questions sur la création d'un service dans Kubernetes

Qu'est-ce qu'un service Kubernetes et pourquoi est-il important ?

Un service Kubernetes est une abstraction qui définit un ensemble logique de pods et une politique pour y accéder. Les services permettent la communication entre différents composants à l'intérieur et à l'extérieur de votre cluster Kubernetes, fournissant ainsi une interface stable aux pods qui peut évoluer au fil du temps.

Comment créer un service dans Kubernetes ?

Pour créer un service, vous pouvez utiliser un fichier YAML pour définir la configuration du service. Une fois défini, appliquez-le avec kubectl apply -f [filename.yaml]. Cette commande crée un service basé sur les spécifications de votre fichier YAML.

Quels sont les différents types de services dans Kubernetes ?

Les types principaux sont ClusterIP (par défaut, accessible au sein du cluster), NodePort (expose le service sur l'adresse IP de chaque nœud sur un port statique), LoadBalancer (s'intègre aux équilibreurs de charge basés sur le cloud) et ExternalName (mappe un service sur un nom DNS.).

Comment exposer mon service en dehors du cluster Kubernetes ?

Utilisez un service de type NodePort ou LoadBalancer. NodePort expose le service sur un port statique sur l'adresse IP du nœud. LoadBalancer utilise l'équilibreur de charge du fournisseur de cloud pour exposer le service.

Puis-je créer un service sans fichier YAML ?

Oui, vous pouvez utiliser kubectl expose pour créer un service à partir d'une ligne de commande. Par exemple, kubectl expose le déploiement [deployment-name] --type=LoadBalancer --name=[service-name].

Comment puis-je déterminer l'adresse IP et le port de mon service Kubernetes ?

Utilisez kubectl get services pour répertorier tous les services. Cela vous montrera l'adresse IP interne et le port attribué à chaque service.

Comment un service découvre-t-il les pods dans Kubernetes ?

Les services utilisent des étiquettes pour sélectionner les pods. Lorsque vous définissez un service, vous spécifiez un sélecteur qui correspond aux étiquettes des pods que vous souhaitez cibler.

Un service Kubernetes peut-il équilibrer la charge du trafic vers les pods ?

Oui, les services équilibrent automatiquement la charge du trafic vers tous les pods qui correspondent au sélecteur du service.

Comment mettre à jour un service dans Kubernetes ?

Mettez à jour le fichier de configuration YAML du service, puis appliquez les modifications à l'aide de kubectl apply -f [filename.yaml].
Kubernetes mettra à jour le Service selon les spécifications modifiées.

Qu'arrive-t-il au Service si les Pods qu'il cible sont supprimés ?

Le service continue d'exister mais ne transmettra pas le trafic tant que de nouveaux pods correspondant à son sélecteur ne seront pas créés.

Comment puis-je supprimer un service dans Kubernetes ?

Utilisez kubectl delete service [service-name] pour supprimer un service. Cela n'affectera pas les Pods, mais ils ne seront plus accessibles via ce service.

Un service dans Kubernetes peut-il s'étendre sur plusieurs espaces de noms ?

Non, les services sont spécifiques à l'espace de noms. Pour communiquer entre les espaces de noms, vous devez utiliser l'espace de noms dans le cadre du nom DNS du service.

Conclusion

Dans cet article, nous avons créé un déploiement pour Nginx avec 3 réplicas et créé un service de type NodePort. Nous avons vu comment l'application Nginx créée à l'aide du déploiement est accessible sur NodeIP:Port.
