Service服务发现

前言

- 本章节主要介绍了Kubernetes服务发现相关知识，包括Service的概念与定义，集群内DNS，如何对外暴露服务，什么是headless服务等。

目标

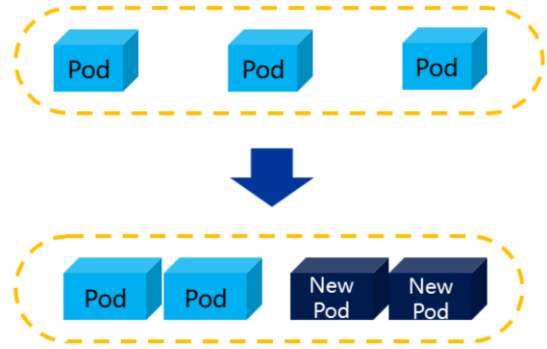
- 学完本课程后，您将能够：
  - 使用Kubernetes中Service对象
  - 使用core-dns的基本功能
  - 使用headless服务

目录

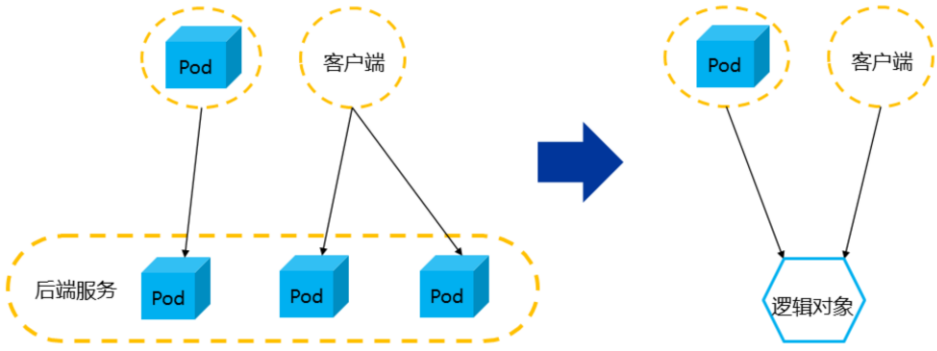
1. **Service基本概念**
2. 服务发现
3. 集群中的DNS
4. Headless Service

Pod的特征

- 在前面章节中我们介绍了pod等资源概念。
  - Pod有自己独立的IP
  - Pod可以被创建，销毁
  - 当扩缩容时，pod的数量会发生变更
  - 当pod故障时，replicaset会创建新的pod
- 如何保证在pod进行如此多变化时，业务都能被访问?



一种解决方案



HUAWEI

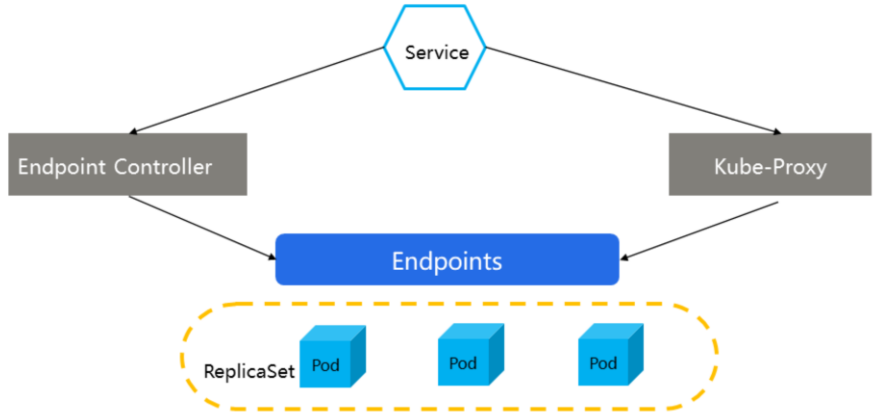
- Kubernetes如何实现这个逻辑对象?

Service

- Kubernetes Service定义了这样一种抽象：逻辑上的一组Pod，一种可以访问它们的策略 —— 通常称为微服务。 这一组Pod能够被Service访问到，通常是通过Label Selector实现的。
- Service的实现类型
  - ClusterIP：提供一个集群内部的虚拟IP地址以供Pod访问（默认模式）
  - NodePort：在Node上打开一个端口以供外部访问
  - LoadBalancer：通过外部的负载均衡器来访问

- ClusterIP是默认模式，LoadBalancer需要额外的模组来提供负载均衡。

Service模型

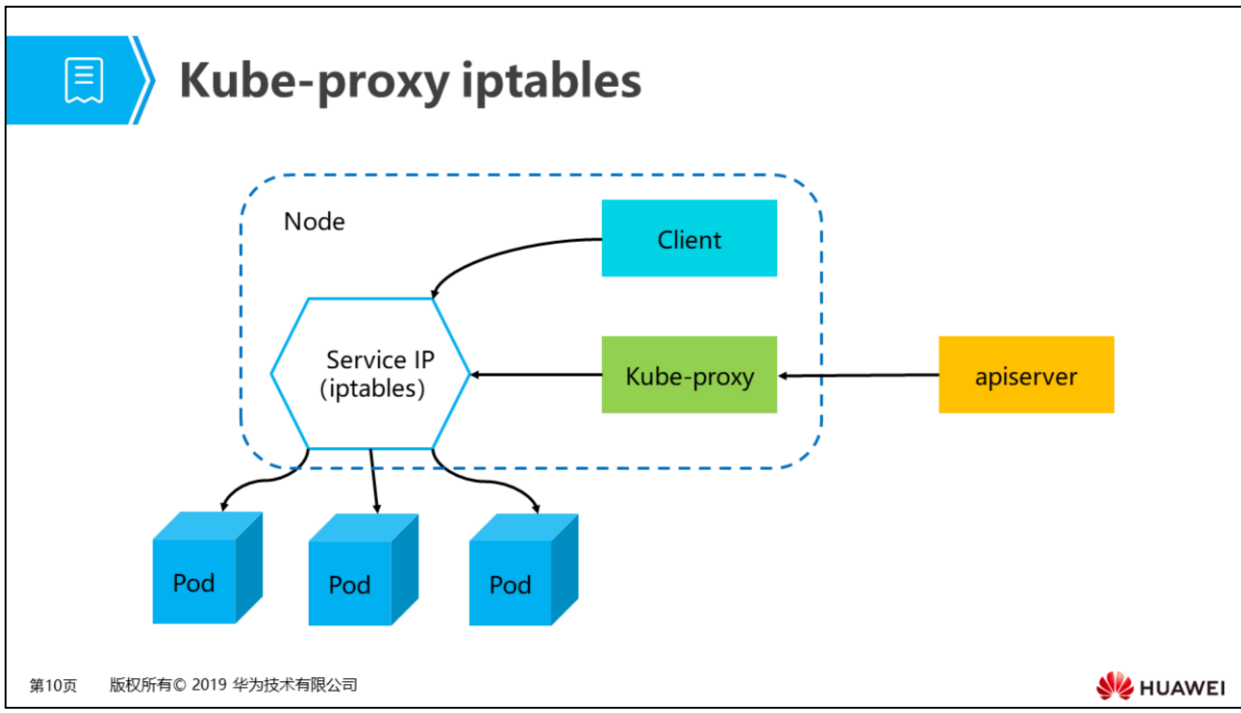


HUAWEI

- Endpoint Controller负责维护Service和Pod的对应关系
- kube-proxy负责service的实现，即实现了Kubernetes内部从pod到service和外部从node port到service的访问。

Endpoint Controller

- 负责生成和维护所有endpoint对象的控制器
- 负责监听service和对应pod的变化
- 监听到service被删除，则删除和该service同名的endpoint对象
- 监听到新的service被创建，则根据新建service信息获取相关pod列表，然后创建对应endpoint对象
- 监听到service被更新，则根据更新后的service信息获取相关pod列表，然后更新对应endpoint对象
- 监听到pod事件，则更新对应的service的endpoint对象，将pod IP记录到endpoint中



- Iptables是默认模式

Kube-proxy IPVS

- 从Kubernetes的1.8版本开始，kube-proxy引入了IPVS模式。IPVS模式与iptables实现方式类似，但是采用的hash表。当Service数量达到一定规模时，hash查表的速度优势就会显现出来，从而提高Service的服务性能。

Service基数	1	5,000	20,000
Rules基数	8	40,000	160,000
增加1条iptables规则	50us	11min	5hour
增加1条ipvs规则	30us	50us	70us

- IPVS需要手动开启方可使用

目录

1. Service基本概念
2. **服务发现**
3. 集群中的DNS
4. Headless Service

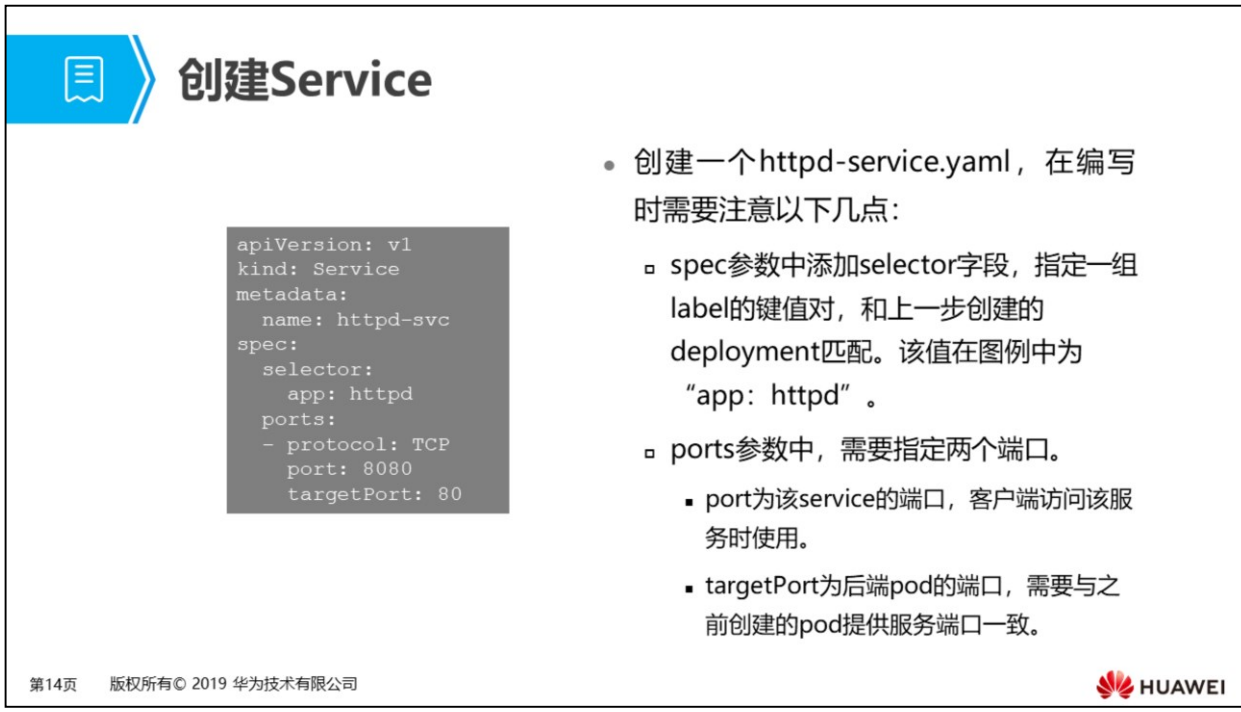
HUAWEI

创建后端Deployment

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

- 创建一个deployment，特别注意其中的几个选项要和service匹配
  - template选项必须配置labels，该配置和service匹配。图例中配置参数为“app: httpd”。
  - Pod的属性中ports选项指定pod对外提供服务的容器端口，该端口需要和service匹配。图例中为“containerPort：80”。

- 详细yaml文件说明参考pod章节



- targetPort端口和前面的deployment端口一致。

查看Service

- 查看Service简明信息，可以获取Service提供服务的ip地址和端口。

```
[root@k8s-master runfile]# kubectl get service
NAME         TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)    AGE
httpd-svc    ClusterIP   10.103.76.128   <none>        8080/TCP   41m
kubernetes   ClusterIP   10.96.0.1       <none>        443/TCP    134m
```

- 测试Service是否正常提供服务。

```
[root@k8s-master runfile]# curl 10.103.76.128:8080
<html><body><h1>It works!</h1></body></html>
```

- 使用describe命令可以查看Service详细信息。如Endpoints信息，显示Service关联pod的地址和服务端口。

```
……
Endpoints:         10.244.1.7:80,10.244.2.10:80,10.244.2.9:80
```


- 其中名为Kubernetes的Service是系统默认创建的，切记不要进行操作。
- 在本次试验中，我们使用的Service实现类型是什么？

创建可供外部访问的Service

```
apiVersion: v1
kind: Service
metadata:
  name: httpd-svc
spec:
  type: NodePort
  selector:
    app: httpd
  ports:
  - protocol: TCP
    port: 8080
    targetPort: 80
    nodePort: 30144
```

- 如果需要Service可供外部进行访问，可以使用Nodeport的方式。
- 编辑Yaml文件时，添加type参数。
- 可以在使用nodeport字段指定对外服务端口，如果不进行指定，系统会自动分配空闲端口。
- 访问时通过访问“节点IP地址：端口”进行服务使用。

目录

1. Service基本概念
2. 服务发现
3. **集群中的DNS**
4. Headless Service

CoreDNS

- 在前文查看Kubernetes集群组件的时候，可以看到名为“CoreDNS”的Pod。

```
NAME                                   READY   STATUS    RESTARTS   AGE
coredns-fb8b8dccf-8k49v                1/1     Running   0          15d
coredns-fb8b8dccf-h7rp7                1/1     Running   0          15d
```

- CoreDNS是一个轻量级的DNS服务器，通过插件的形式在Kubernetes集群内实现，提供服务发现功能，使得用户除了可以用IP访问服务外，也可用域名来访问服务。
- 从1.13版本的Kubernetes开始CoreDNS取代了原有的kubeDNS，成为了Kubernetes集群内部的默认DNS组件。

- kubeDNS是前期版本kubernetes中默认的dns组件

查看服务的完整域名

- 创建一个clientpod，用于查看httpd服务的完整名字。
- 在记录中可以看到，服务的IP地址对应的名称是httpd-svc.default.svc.cluster.local

```
apiVersion: v1
kind: Pod
metadata:
  name: clientpod
spec:
  containers:
    - name: clientpod
      image: busybox:1.28.3
      args:
      - /bin/sh
      - -c
      - sleep 30000
```

```
/ # nslookup 10.105.253.35
Server:    10.96.0.10
Address 1: 10.96.0.10 kube-dns.kube-system.svc.cluster.local
Name:      10.105.253.35
Address 1: 10.105.253.35 httpd-svc.default.svc.cluster.local
```

  HUAWEI

- nslookup可以指定查询的类型，可以查到DNS记录的生存时间还可以指定使用哪个DNS服务器进行解释。

DNS记录

- 服务的DNS记录名称为：

  <服务名称>.<namespace>.svc.cluster.local

- 服务后端的deployment中Pod的DNS记录名称为：

  <PodIP>.<服务名称>.<namespace>.svc.cluster.local

- ClientPod访问服务时，可以使用<服务名称>.<namespace>便捷抵达服务，甚至在ClientPod与服务在同一namespace时，直接用<服务名称>进行访问。

```
/ # wget httpd-svc
Connecting to httpd-svc (10.105.253.35:80)
index.html           100%
|**********************************|    45   0:00:00 ETA
```


- 访问一个服务完整的路径为”httpd-svc.default.svc.cluster.local”

目录

1. Service基本概念
2. 服务发现
3. 集群中的DNS
4. **Headless Service**

HUAWEI

Headless Service

- 有的时候不需要或者不想要负载均衡，以及单独的Service IP，可以通过指定Cluster IP的值为“None”来创建Headless Service。
- 对这类Service并不会分配Cluster IP，kube-proxy不会处理他们，并且平台也不会为他们进行负载均衡和路由。
- 对定义了selector的Headless Service，意味着后端有一些提供业务的Pod，Endpoint控制器在API中创建了Endpoints记录，当通过域名访问服务时，流量会被直接转发到对应的Pod上。

- Headless Service主要用于StatefulSet。

创建Headless Service

- 创建一个简单的deployment和Headless Service。区别为多了一条clusterIP：None的参数。

```
apiVersion: v1
kind: Service
metadata:
  name: headless-svc
spec:
  selector:
    app: httpd
  ports:
    - protocol: TCP
      port: 80
      targetPort: 80
  clusterIP: None
```

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


HUAWEI

- Service的selector需要与pod的label对应，注意不是deployment的label。

使用Headless Service

- 查看Headless Service的信息，可以看到没有IP地址。

```
[root@k8s-master servicefile]# kubectl get svc
NAME           TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
headless-svc   ClusterIP   None         <none>        80/TCP    3s
```

- 使用的时候利用DNS功能，通过访问“headless-svc”或“headless-svc.default”来访问服务。可以看到，域名解析到的IP地址其实是Pod的IP地址。

```
/ # nslookup headless-svc
Server:    10.96.0.10
Address 1: 10.96.0.10 kube-dns.kube-system.svc.cluster.local

Name:      headless-svc
Address 1: 10.244.1.95 10-244-1-95.headless-svc.default.svc.cluster.local
Address 2: 10.244.2.115 10-244-2-115.headless-svc.default.svc.cluster.local
Address 3: 10.244.0.19 10-244-0-19.headless-svc.default.svc.cluster.local
```


- 通过域名访问需要在pod中进行操作。如果在外部操作dns不生效无法访问。

实验&实训任务

- 实验任务
  - 请按照实验手册2.6章节完成Service相关实验，包括：
    - 创建和使用Service
    - 使用Core-DNS
    - 使用Headless Service
- 实训任务
  - 请灵活使用本章节课程及实验手册中学到的知识，按照实验手册2.6.4章节完成Service实训任务。

本章总结

- 本章节主要介绍了：
  - Service的基本概念
  - 如何使用Service进行服务发现
  - CoreDNS
  - Headless Service


谢 谢
www.huawei.com